ATSAMB11XR/ZR

Ultra-Low Power Bluetooth® Low Energy SiP/Module

Introduction

The ATSAMB11-XR2100A is an ultra-low power Bluetooth Low Energy (BLE) 5.0 System in a Package (SiP) with Integrated MCU, transceiver, modem, MAC, PA, Transmit/Receive (T/R) switch, and Power Management Unit (PMU). It is a standalone Cortex® -M0 applications processor with embedded Flash memory and BLE connectivity.

The Bluetooth SIG-qualified Bluetooth Low Energy protocol stack is stored in a dedicated ROM. The firmware includes L2CAP service layer protocols, Security Manager, Attribute protocol (ATT), Generic Attribute Profile (GATT), and the Generic Access Profile (GAP). Additionally, example applications are available for application profiles such as proximity, thermometer, heart rate and blood pressure, and many others.

The ATSAMB11-XR2100A provides a compact footprint and various embedded features, such as a 26 MHz crystal oscillator.

The ATSAMB11-ZR210CA is a fully certified module that contains the ATSAMB11-XR2100A and all external RF circuitry required, including a ceramic high-gain antenna. The user simply places the module into their PCB and provides power with a 32.768 kHz Real-Time Clock (RTC) or crystal, and an I/O path.

Microchip BluSDK Smart offers a comprehensive set of tools and reference applications for several Bluetooth SIG defined profiles and a custom profile. The BluSDK Smart will help the user quickly evaluate, design and develop BLE products with the ATSAMB11-XR2100A and ATSAMB11-ZR210CA.

The ATSAMB11-XR2100A and associated ATSAMB11-ZR210CA module have passed the Bluetooth SIG certification for interoperability with the Bluetooth Low Energy 5.0 specification QDID: 117593.

Features

- 2.4 GHz Transceiver and Modem:
  - -92.5 dBm receiver sensitivity
  - -55 dBm to +3.5 dBm programmable TX output power
  - Integrated T/R switch
  - Single wire antenna connection (ATSAMB11-XR2100A)
  - Incorporated chip antenna (ATSAMB11-ZR210CA)
- Processor Features:
  - ARM® Cortex®-M0 32-bit processor
  - Serial Wire Debug (SWD) interface
  - Four-channel Direct Memory Access (DMA) controller
  - Watchdog Timer
- Memory:

  - 128 KB embedded Random Access Memory (RAM)
  - 128 KB embedded ROM
  - 256 KB stacked Flash memory
- Hardware Security Accelerators:
  - Advanced Encryption Standard (AES)-128
  - Secure Hash Algorithm (SHA)-256
- Peripherals:
  - 23 digital and 4 mixed-signal General Purpose Input Outputs (GPIOs) with 96 kOhm internal programmable pull-up or down resistors and retention capability, and one wake-up GPIO with 96 kOhm internal pull-up resistor
  - Two Serial Peripheral Interface (SPI) Master/Slave
  - Two Inter-Integrated Circuit (I^2C) Master/Slave
  - Two UART
  - One SPI Flash interface (used for accessing the internal stacked Flash)
  - Three-axis quadrature decoder
  - Four Pulse Width Modulation (PWM) channels
  - Three General Purpose Timers and one Always-On (AON) sleep Timer
  - 4-channel, 11-bit Analog-to-Digital Converter (ADC)
- Clock:
  - Integrated 26 MHz RC oscillator
  - Integrated 2 MHz RC oscillator
  - 26 MHz crystal oscillator (XO)
  - 32.768 kHz Real Time Clock crystal oscillator (RTC XO)
- Ultra-Low Power:
  - 2.03 µA sleep current
  - 4.18 mA peak TX current (1)
  - 5.26 mA peak RX current
  - 16.4 µA average advertisement current (three channels, 1s interval) (2)
- Integrated Power Management:
  - 2.3V to 4.3V battery voltage range
  - 2.3V to 3.6V input range for I/O (limited by Flash memory)
  - Fully integrated Buck DC/DC converter
- Temperature Range:
  - -40°C to 85°C
- Package:
  - 49-pin FLGA SiP package 5.50 mm x 4.50 mm
  - 35-pin module package 10.541 mm x 7.503 mm
  - BT SIG QDID: 117593

Note:

1. TX output power - 0 dBm.
2. Advertisement channels - 3 ; Advertising interval - 1 second ; Advertising event type - Connectable undirected; Advertisement data payload size - 31 octets.

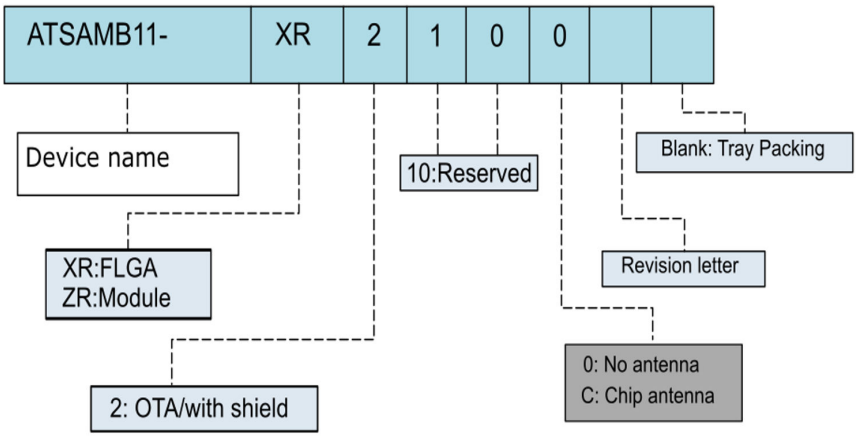
Table of Contents

1. Ordering Information

Table 1-1. Ordering Details

Model Number	Ordering Code	Package	Description	Regulatory Information
ATSAMB11-XR2100A	ATSAMB11-XR2100A	5.5 mm x 4.5 mm	ATSAMB11 SiP tray	N/A
ATSAMB11-ZR210CA	ATSAMB11-ZR210CA	7.5 mm X 10.5 mm	ATSAMB11 module with chip antenna	FCC, ISED, CE, MIC, KCC, NCC

Figure 1-1. Marking information



2. Package Information

Table 2-1. ATSAMB11-XR2100A SiP 49 Package Information

Parameter	Value	Units	Tolerance
Package size	5.50 x 4.50	mm	±0.05 mm
Pad count	49		
Total thickness	1.40	mm	Max
Tolerance (maximum pad pitch)	0.40	mm	±0.05 mm
Pad width	0.21		
Exposed pad size	0.50 x 0.50		

Note: For drawing details, see Figure 26-1.

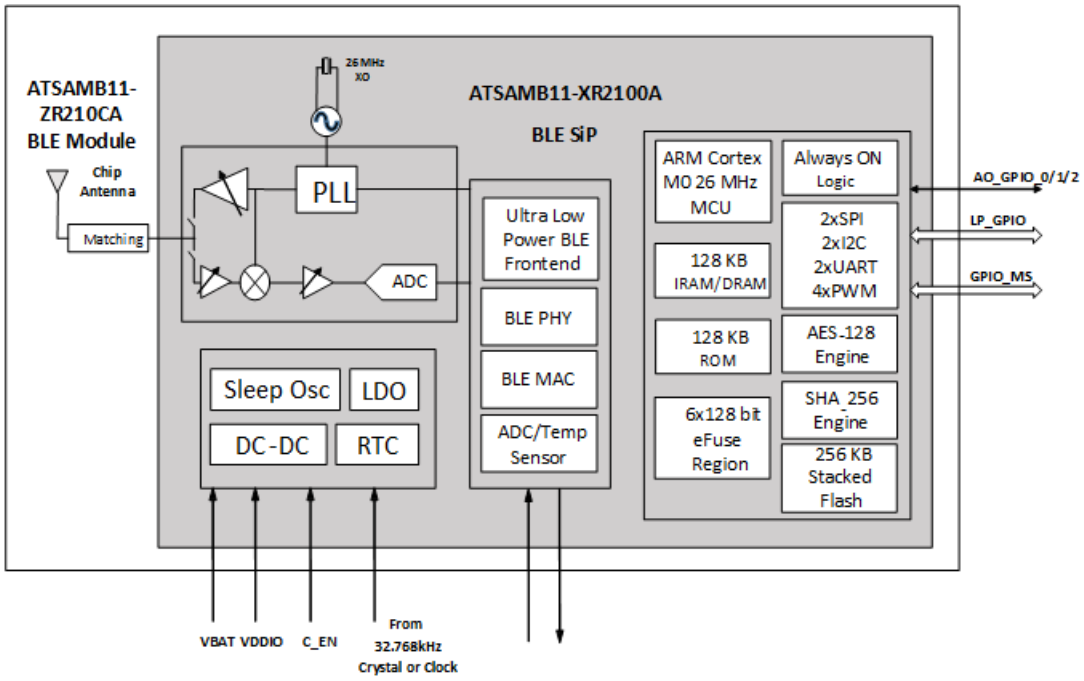
Table 2-2. ATSAMB11-ZR210CA Module Information

Parameter	Value	Units	Tolerance
Package size	7.503 x 10.541	mm	Untoleranced dimension
Pad count	35		
Total thickness	1.868	mm	Untoleranced dimensions
Pad pitch	0.61		
Pad width	0.406		
Exposed pad size	2.705 x 2.705		

Note: For drawing details, see Figure 26-2.

3. Block Diagram

Figure 3-1. Block Diagram



4. Pinout Information

The ATSAMB11-XR2100A is offered in an exposed pad 49-pin SiP package. This package has an exposed paddle that must be connected to the system board ground. The SiP package pin assignment is shown in the figure below. The colored shading is used to indicate the pin type as follows:

- Red – analog
- Green – digital I/O (switchable power domain)
- Blue – digital I/O (always-on power domain)
- Yellow – power
- Purple – PMU
- Shaded green/red – configurable mixed-signal GPIO (digital/analog)

The ATSAMB11-ZR210CA module is a castellated PCB with the ATSAMB11-XR2100A integrated with a matched chip antenna. The pins are identified in the pin description table. The ATSAMB11-XR2100A also contains a paddle pad on the bottom of the PCB, that must be soldered to the system ground.

Figure 4-1. ATSAMB11-XR2100A Pin Assignment

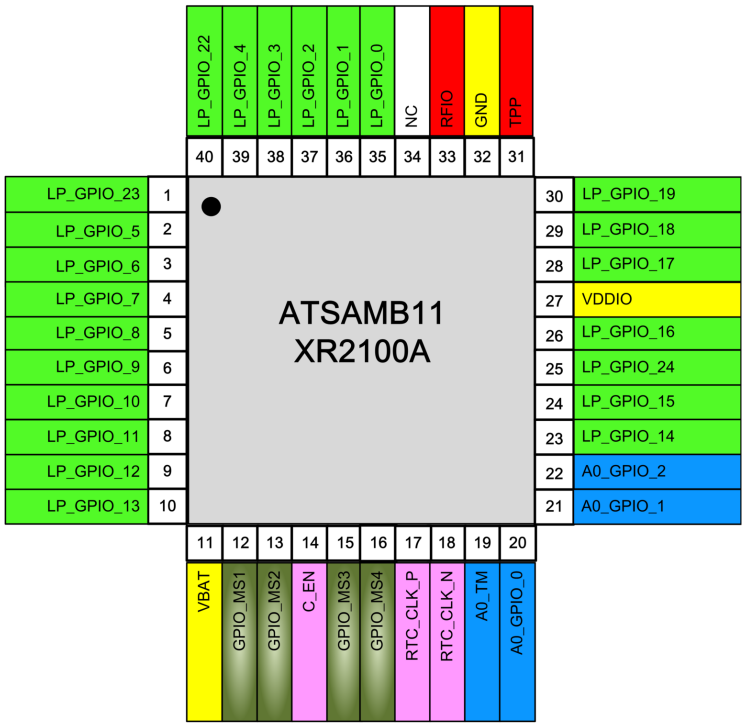
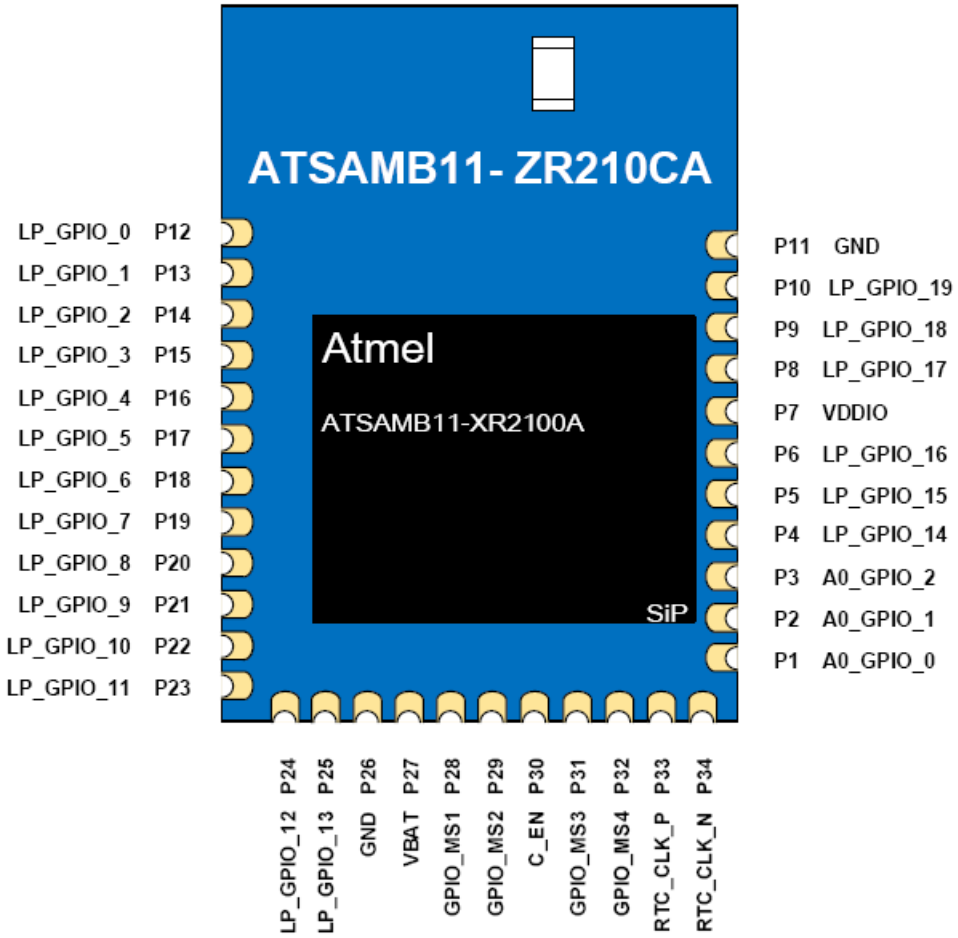


Figure 4-2. ATSAMB11-ZR210CA Pin Descriptions



The following table lists the pin assignments for both the ATSAMB11-XR2100A and the ATSAMB11-ZR210CA.

Table 4-1. ATSAMB11-XR2100A and ATSAMB11-ZR210CA Pin Description

ATSAMB11-XR2100A Pin #	ATSAMB11-ZR210CA Pin #	Pin Name	Pin Type	Description / Default Function
1	-	LP_GPIO_23	Digital I/O	GPIO with Programmable Pull Up/Down
2	17	LP_GPIO_5	Digital I/O	GPIO with Programmable Pull Up/Down
3	18	LP_GPIO_6	Digital I/O	GPIO with Programmable Pull Up/Down
4	19	LP_GPIO_7	Digital I/O	GPIO with Programmable Pull Up/Down
5	20	LP_GPIO_8(1)	Digital I/O	GPIO with Programmable Pull Up/Down

.....continued

ATSAMB11-XR2100A Pin #	ATSAMB11-ZR210CA Pin #	Pin Name	Pin Type	Description / Default Function
6	21	LP_GPIO_9(1)	Digital I/O	GPIO with Programmable Pull Up/Down
7	22	LP_GPIO_10	Digital I/O	GPIO with Programmable Pull Up/Down
8	23	LP_GPIO_11	Digital I/O	GPIO with Programmable Pull Up/Down
9	24	LP_GPIO_12	Digital I/O	GPIO with Programmable Pull Up/Down
10	25	LP_GPIO_13	Digital I/O	GPIO with Programmable Pull Up/Down
11	27	VBAT	Power supply	Power supply pin for the DC/DC convertor
12	28	GPIO_MS1	Mixed Signal I/O	Configurable to be a GPIO digital and analog signal. Only analog input for ADC interface.
13	29	GPIO_MS2	Mixed Signal I/O	Configurable to be a GPIO digital and analog signal. Only analog input for ADC interface.
14	30	C_EN	Digital Input	Can be used to control the state of PMU. High level enables the module; low-level places module in Power-Down mode.
15	31	GPIO_MS3	Mixed Signal I/O	Configurable to be a GPIO digital and analog signal. Only analog input for ADC interface.
16	32	GPIO_MS4	Mixed Signal I/O	Configurable to be a GPIO digital and analog signal. Only analog input for ADC interface.
17	33	RTC_CLK_P	Analog	Crystal pin or External clock supply, see 9.3 32.768 kHz RTC Crystal Oscillator (RTC XO)
18	34	RTC_CLK_N	Analog	Crystal pin or External clock supply, see 9.3 32.768 kHz RTC Crystal Oscillator (RTC XO)

.....continued

ATSAMB11-XR2100A Pin #	ATSAMB11-ZR210CA Pin #	Pin Name	Pin Type	Description / Default Function
19	-	AO_TM	Digital Input	Always-On Test Mode. Connect to GND
20	1	AO_GPIO_0	Always On Digital I/O, Programmable Pull-Up	To be held in logic '0' GND to allow the device to enter Ultra_Low_Power mode Can be used to Wake-up the device from Ultra_Low_Power mode.
21	2	AO_GPIO_1	Always On. Digital I/O, Programmable Pull- Up	GPIO with Programmable Pull Up
22	3	AO_GPIO_2	Always On. Digital I/O, Programmable Pull- Up	GPIO with Programmable Pull Up
23	4	LP_GPIO_14	Digital I/O	GPIO with Programmable Pull Up/Down
24	5	LP_GPIO_15	Digital I/O	GPIO with Programmable Pull Up/Down
25	-	LP_GPIO_24	Digital I/O	GPIO with Programmable Pull Up/Down
26	6	LP_GPIO_16	Digital I/O	GPIO with Programmable Pull Up/Down
27	7	VDDIO	Power supply	Power supply pin for the I/O pins. Can be less than or equal to voltage supplied at VBAT
28	8	LP_GPIO_17	Digital I/O	GPIO with Programmable Pull Up/Down
29	9	LP_GPIO_18	Digital I/O	GPIO with Programmable Pull Up/Down
30	10	LP_GPIO_19	Digital I/O	GPIO with Programmable Pull Up/Down
31	-	TPP		Do not connect
32	11, 26	GND	Ground	
33	-	RFIO	Analog I/O	RX input and TX output. Single-ended RF I/O; To be connected to antenna

.....continued

ATSAMB11-XR2100A Pin #	ATSAMB11-ZR210CA Pin #	Pin Name	Pin Type	Description / Default Function
34	-	NC		Do not connect
35	12	LP_GPIO_0	Digital I/O	SWD clock
36	13	LP_GPIO_1	Digital I/O	SWD I/O
37	14	LP_GPIO_2	Digital I/O	GPIO with Programmable Pull Up/Down
38	15	LP_GPIO_3	Digital I/O	GPIO with Programmable Pull Up/Down
39	16	LP_GPIO_4	Digital I/O	GPIO with Programmable Pull Up/Down
40	-	LP_GPIO_22	Digital I/O	GPIO with Programmable Pull Up/Down
41 - 49	35	Paddle	Ground	Exposed paddle must be soldered to system ground

Note:

1. These GPIO pads are high-drive pads. Refer Table 25-3.

5. Device States

This section includes details on the description and controlling of the Device states.

5.1 Description of Device States

The ATSAMB11-XR2100A and the ATSAMB11-ZR210CA have multiple device states, depending on the state of the ARM processor and BLE subsystem.

Note: The ARM is required to be powered on, if the BLE subsystem is active.

- BLE_On_Transmit – Device is actively transmitting a BLE signal.
- BLE_On_Receive – Device is in active receive state.
- MCU_Only – Device has ARM processor powered-on and BLE subsystem powered-down.
- Ultra_Low_Power – BLE subsystem and ARM processor are powered-down.
- Power_Down – Device core supply off.

5.1.1 Controlling the Device States

The following pins are used to switch between the main device states:

- C_EN – used to enable PMU
- VDDIO – I/O supply voltage from an external power supply
- AO_GPIO_0 - can be used to control the device from entering/exiting Ultra_Low_Power mode

To be in the Power_Down state, the VDDIO supply must be turned on and the C_EN must be maintained at logic low (at GND level). To switch between the Power_Down state and the MCU_Only state, C_EN is to be maintained at logic high (VDDIO voltage level). Once the device is in the MCU_Only state, all other state transitions are controlled entirely by software. When VDDIO supply is turned off and C_EN is in logic low, the chip is powered off with no leakage.

When VDDIO supply is turned off, voltage cannot be applied to the ATSAMB11-XR2100A pins, as each pin contains an ESD diode from the pin to supply. This diode turns on when a voltage higher than one diode-drop is supplied to the pin.

If voltage is to be applied to the signal pads while the chip is in a low-power state, the VDDIO supply must be on, so that the Power_Down state is used. Similarly, to prevent the pin-to-ground diode from turning on, do not apply a voltage to any pin that is more than one diode-drop below ground.

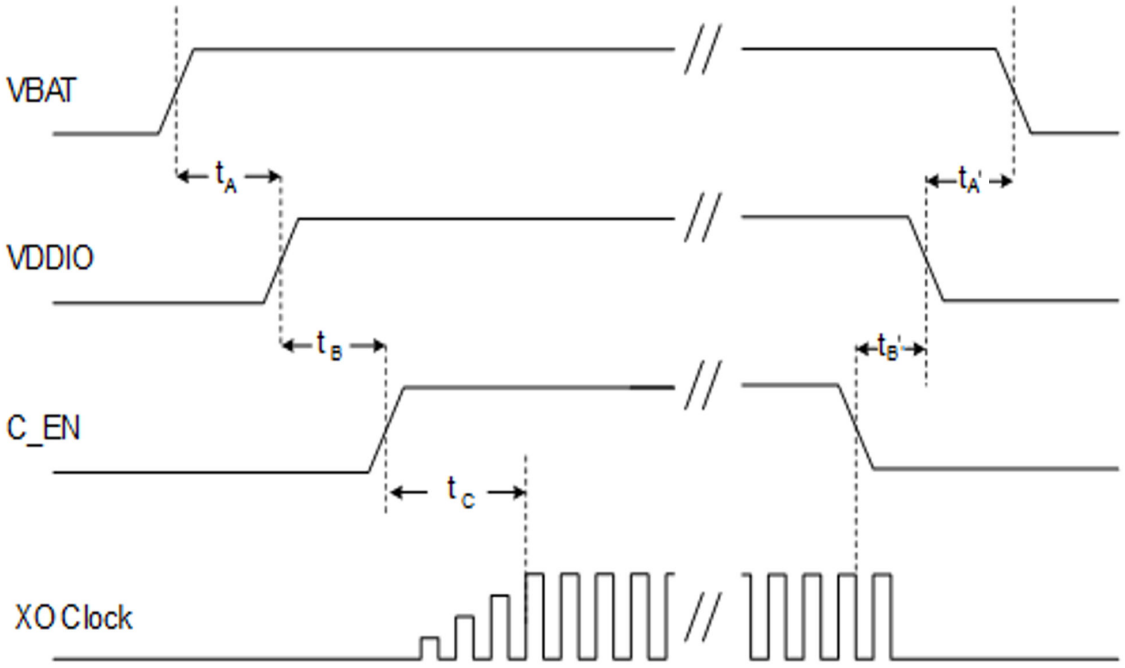
The AO_GPIO_0 pin can be used to control the device from entering and exiting Ultra_Low_Power mode. When AO_GPIO_0 is maintained in logic high state, the device will not enter Ultra_Low_Power mode. When the AO_GPIO_0 is maintained in logic low, the device will enter Ultra_Low_Power mode provided there are no BLE events to be handled.

For more details on how sleep and wake-up are handled, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide.

5.2 Power Sequences

The power sequences and timing parameters for the ATSAMB11-XR2100A and ATSAMB11-ZR210CA, are illustrated below.

Figure 5-1. Power-up/Power-down Sequence



The timing parameters are provided in following table.

Table 5-1. Power-up/Power-down Sequence Timing

Parameter	Min.	Max.	Units	Description	Notes
t_A	0		ms	VBAT rise to VDDIO rise	VBAT and VDDIO can rise simultaneously or can be tied together
t_B	0			VDDIO rise to C_EN rise	C_EN must not rise before VDDIO. C_EN must be driven high or low, not left floating.
t_C	10		µs	C_EN rise to 31.25 kHz (2 MHz/64) oscillator stabilizing	
$t_{B'}$	0		ms	C_EN fall to VDDIO fall	C_EN must fall before VDDIO. C_EN must be driven high or low, not left floating.
$t_{A'}$	0			VDDIO fall to VBAT fall	VBAT and VDDIO can fall simultaneously or be tied together

5.3 Digital and Mixed-Signal I/O Pin Behavior during Power-Up Sequences

The following table represents I/O pin states corresponding to device power modes.

Table 5-2. I/O Pin Behavior in the Different Device States (1)

Device State	VDDIO	CHIP_EN	Output Driver	Input Driver	Pull Up/Down Resistor (2)
Power_Down: core supply off	High	Low	Disabled (Hi-Z)	Disabled	Disabled
Power-on Reset: core supply on, POR hard reset pulse on	High	High	Disabled (Hi-Z)	Disabled	Disabled (3)
Power-on Default: core supply on, device out of reset but not programmed yet	High	High	Disabled (Hi-Z)	Enabled(4)	Enabled Pull-Up (4)
MCU_Only, BLE_On: core supply on, device programmed by firmware	High	High	Programmed by firmware for each pin: Enabled or Disabled (Hi-Z) (5) ,when Enabled driving 0 or 1	Opposite of Output Driver state: Disabled or Enabled (5)	Programmed by firmware for each pin: Enabled or Disabled, Pull-Up or Pull- Down(5)
Ultra_Low_Power: core supply on for always-on domain, core supply off for switchable domains	High	High	Retains previous state(6) for each pin: Enabled or Disabled (Hi-Z), when Enabled driving 0 or 1	Opposite of Output Driver state: Disabled or Enabled(6)	Retains previous state (6) for each pin: Enabled or Disabled, Pull-Up or Pull-Down

Note:

1. This table applies to all three types of I/O pins (digital switchable domain GPIOs, digital always-on/wake-up GPIO, and mixed-signal GPIOs) unless otherwise noted.
2. Pull-up/down resistor value is 96 kOhm ±10%.
3. In Power-on Reset state, the pull-up resistor is enabled in the always-on/wake-up GPIO only.
4. In Power-on Default state, the input drivers and pull-up/down resistors are disabled in the mixed-signal GPIOs only (mixed-signal GPIOs are defaulted to analog mode, see the note below).
5. Mixed-signal GPIOs can be programmed to be in analog or digital mode for each pin: when programmed to analog mode (default), the output driver, input driver, and pull-up/down resistors are all disabled.
6. In Ultra_Low_Power state, the always-on/wake-up GPIO does not have retention capability and behaves same as in MCU_Only or BLE_On states, also for mixed-signal GPIOs programming analog mode overrides retention functionality for each pin.

6. Processor Architecture

6.1 ARM Subsystem

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have an ARM Cortex-M0 32-bit processor. It is responsible for controlling the BLE Subsystem and handling all application features.

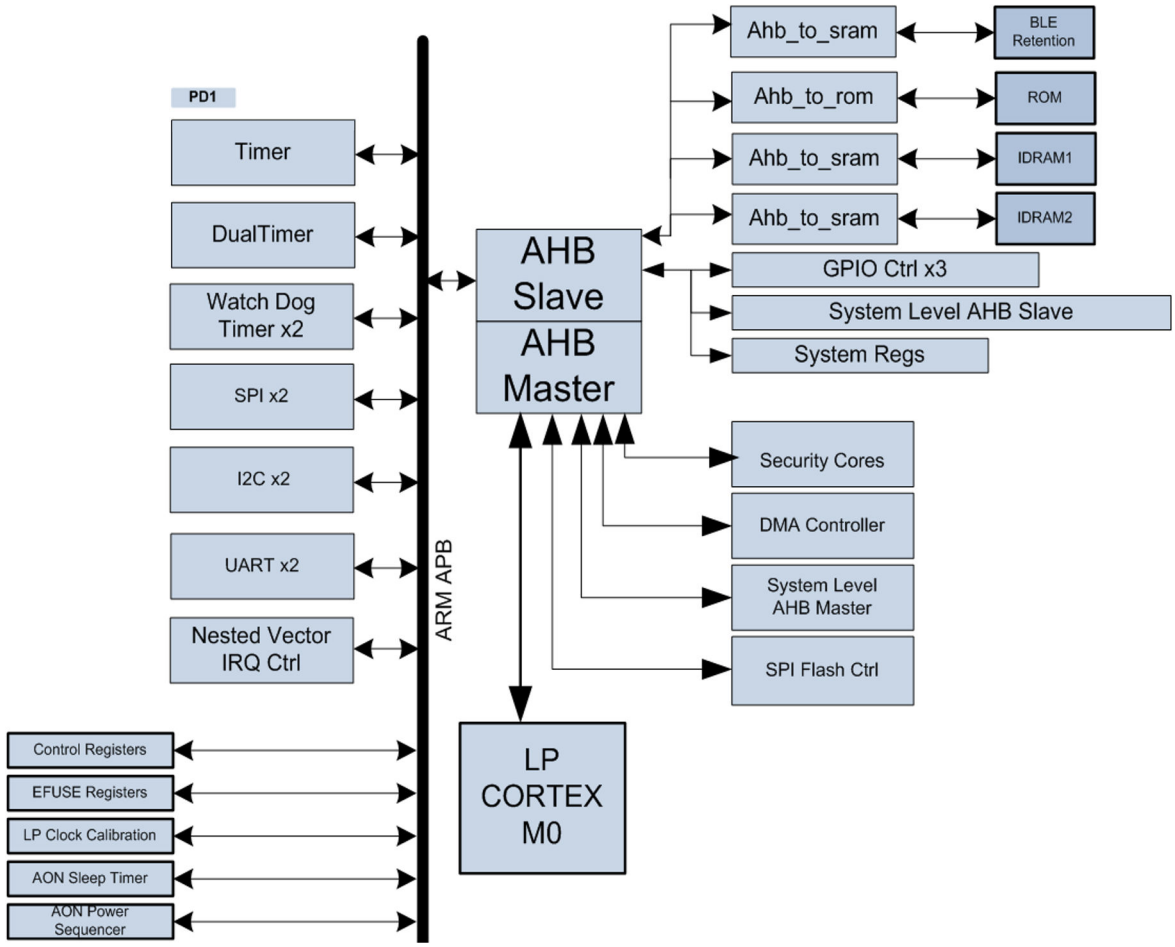
The Cortex-M0 Microcontroller consists of a full 32-bit processor capable of addressing 4GB of memory. It has a RISC like load/store instruction set and internal 3-stage Pipeline Von Neumann architecture.

The Cortex-M0 processor provides a single system-level interface using AMBA technology to provide high speed, low latency memory accesses.

The Cortex-M0 processor implements a complete hardware debug solution with four hardware breakpoint and two watchpoint options. This provides high system visibility of the processor, memory, and peripherals through a 2-pin Serial Wire Debug (SWD) port that is ideal for microcontrollers and other small package devices.

ATSAMB11 is running a proprietary RTOS tightly coupled with FW in the ROM and the user can not override it. SysTick timer is being used by the stack and will not be available for usage by the application.

Figure 6-1. ARM Cortex-M0 Subsystem



6.1.1 Features

The processor features and benefits are:

- Tight integration with the system peripherals to reduce area and development costs
- Thumb instruction set combines high code density with 32-bit performance
- Integrated sleep modes using a Wakeup Interrupt Controller for low power consumption
- Deterministic and high-performance interrupt handling via Nested Vector Interrupt Controller for time-critical applications
- Serial Wire Debug reduces the number of pins required for debugging
- DMA engine for Peripheral-to-Memory, Memory-to-Memory, and Memory-to-Peripheral operation

6.1.2 Wakeup Sources

Ultra_Low_Power is the lowest possible power state for the system. In Ultra_Low_Power state, ARM Cortex-M0, BLE core, GPIO's, and all other peripheral cores are powered-down. Only AON-GPIO_0 and AON-Sleep timer are functional in this state.

ATSAMB11 contains the following wake-up sources that wake up the system from Ultra_Low_Power mode:

- BLE events
- AON-GPIO_0
- AON-Sleep timer

6.2 Cortex M0 Peripherals

- System Control Space (SCS)
  The processor provides debug through registers in the SCS. For more details, refer to the Cortex-M0 Devices Generic User Guide (http://www.arm.com).
- Nested Vectored Interrupt Controller (NVIC)
  External interrupt signals connect to the NVIC, and the NVIC prioritizes the interrupts. Software can set the priority of each interrupt. The NVIC and the Cortex-M0 processor core are closely coupled, providing low latency interrupt processing and efficient processing of late arriving interrupts. For more details, refer to the Cortex-M0 Technical Reference Manual (http://www.arm.com).
- System Timer (SysTick)
  The System Timer is a 24-bit timer clocked by CLK_CPU that extends the functionality of both the processor and the NVIC. For more details, refer to the Cortex-M0 Technical Reference Manual (http://www.arm.com).
- System Control Block (SCB)
  The System Control Block provides system implementation information, and system control. This includes configuration, control, and reporting of the system exceptions. For more details, refer to the Cortex-M0 Devices Generic User Guide (http://www.arm.com).

6.2.1 Cortex M0 Peripheral Memory Map

- 0xE000E000 System Control Space (SCS)
- 0xE000E010 System Timer (SysTick)
- 0xE000E100 Nested Vectored Interrupt Controller (NVIC)
- 0xE000ED00 System Control Block (SCB)

6.3 Nested Vector Interrupt Controller

External interrupt signals are connected to the NVIC, and the NVIC prioritizes the interrupts. Software can set the priority of each interrupt. The NVIC and the Cortex-M0 processor core are closely coupled to provide low-latency interrupt processing and efficient processing of late arriving interrupts.

All NVIC registers are accessible via word transfers and are little endian. Any attempt to read or write a half-word or byte individually is unpredictable.

The NVIC allows the CPU to be able to individually enable or disable each interrupt source, and hold each interrupt until it is serviced and cleared by the CPU.

Table 6-1. NVIC Register Summary

Name	Description
ISER	Interrupt Set-Enable Register
ICER	Interrupt Clear-Enable Register
ISPR	Interrupt Set-Pending Register
ICPR	Interrupt Clear-Pending Register
IPR0-IPR7	Interrupt Priority Registers

Note: For a description of each register, see the Cortex-M0 documentation from ARM (http://www.arm.com).

6.3.1 Functional Description

The Cortex-M0 NVIC is connected to 32 IRQ sources. The following table lists the interrupts that are available in ATSAMB11. Also, some of the interrupts are marked as RESERVED as they are used by the BLE stack and are used for firmware in general. Applications must refrain from registering an ISR for those interrupts as it affects the chip functionality.

Perform the following steps to enable an interrupt:

- Configure and enable peripheral interrupt using peripheral-specific registers. Refer to the intended peripheral chapter for configuring interrupt.
- The ISRs are mapped in RAM memory called interrupt vector table. 0x10000000 is the start address of first ISR index 0 and 4 bytes are allocated for each ISR index in incrementing order. The specific peripheral ISR handler to be registered by assigning the handler address to this interrupt vector table.
- Set the NVIC priority of the interrupt if required. IPR0-IPR7 ARM NVIC registers are used to set the priority level for individual interrupt sources. IRQ number of the specific interrupt source as per the following table is used. Only two bits are allocated for each interrupt source. Therefore, four priority levels (0, 1, 2, and 3) are possible. The priority value is zero by default, and is also the highest priority.
  Note: The BLE interrupts are higher priority, and the Application peripherals must be set to lower priority other than '0' either as 1, 2, 3. Therefore, the higher priority interrupts (BLE) can preempt the lower priority interrupts like Application peripheral interrupts to meet the BLE timing requirements. When the processor is executing an exception handler, an exception can preempt the exception handler if its priority is higher than the priority of the exception being handled.
- Enable NVIC interrupt of specific IRQ numbers using ISER register.

Table 6-2. ATSAMB11 Interrupt Vector Table

IRQ Number	ISR Index	Interrupt Source	Muxability
-15	1	Reset	Non-muxable
-14	2	NMI (Watchdog 0, Watchdog 1)	Non-muxable
-13	3	Hard Fault	Non-muxable
-5	11	SVC	Non-muxable
-2	14	Pending SV	Non-muxable
-1	15	SysTick	Non-muxable
0	16	UART0 RX	Muxable
1	17	UART0 TX	Muxable
2	18	UART1 RX	Muxable
3	19	UART1 TX	Muxable
4	20	SPI0 RX	Muxable
5	21	SPI0 TX	Muxable
6	22	SPI1 RX	Muxable
7	23	SPI1 TX	Muxable
8	24	I^2C0 RX	Muxable
9	25	I^2C0 TX	Muxable
10	26	I^2C1 RX	Muxable
11	27	I^2C1 TX	Muxable
12	28	Watchdog 0	Muxable
13	29	Watchdog 1	Muxable
14	30	ARM® Dual Timer	Muxable
15	31	BLE Peripheral Register	Muxable
16	32	EFuse Out of Reset	Muxable(1)
17	33	BLE Security	Muxable(1)
18	34	SPI Flash	Muxable
19	35	Calibration Done	Muxable(1)
20	36	Brown Out Detected(2)	Muxable
21	37	BLE specific	Non-Muxable(1)
22	38	BLE specific	Non-Muxable(1)
23	39	GPIO 0 Combined	Non-Muxable

.....continued

IRQ Number	ISR Index	Interrupt Source	Muxability
24	40	GPIO 1 Combined	Non-Muxable
25	41	GPIO 2 combined	Non-Muxable
26	42	ARM timer	Non-Muxable(1)
27	43	AON sleep timer	Non-Muxable
28	44	BLE specific	Non-Muxable(1)
29	45	BLE specific	Non-Muxable(1)
30	46	BLE specific	Non-Muxable(1)
31	47	BLE specific	Non-Muxable(1)

Note:

1. This ISR index is used by the BLE stack. Applications must refrain from registering an ISR as it affects the chip functionality.
2. Brown out detection feature is not supported for ATSAMB11XR/ZR variant. By default the ISR is mapped to brown-out detection interrupt. Since the feature is not supported in ATSAMB11XR/ZR variant, this ISR can be used for any user-specific ISR using muxable interrupt configuration.

For more details on configuration options for muxable interrupts, see Muxable Interrupt.

6.3.2 Example

The following is the sample code to register and enable the AON Sleep Timer interrupt:

```
/* ISR Index for AON Sleep Timer from above table = 43 */
/* ISR Index base address 0x10000000 (start
address of first ISR index 0) */
/* Assign user define AON_Sleep_Timer_Handler function */
*((uint32_t *)(43 * 4 + 0x10000000))= AON_Sleep_Timer_Handler;
/* IRQ Number for AON Sleep Timer is 27 from above table. Enable AON Sleep Timer Interrupt */
  NVIC->ISER[0] = (1 << ((uint32_t)(27) & 0x1F));
```

7. Memory Subsystem

The Cortex-M0 core uses a 128 KB instruction/boot ROM along with a 128 KB shared instruction and data RAM.

Table 7-1. SAMB11 Physical Memory Map

Memory	Start Address	End Address	Size
Internal ROM	0x00000	0x1FFFF	128 KB
IDRAM	0x10000000	0x1001FFFF	128 KB
BLE Retention RAM	0x10040000	0x10041FFF	8 KB
SPI Flash	0x00000	0x3FFFF	256 KB

7.1 Shared Instruction and Data Memory

The Instruction and Data Memory (IDRAM1 and IDRAM2) contains instructions and data used by the ARM. The 128 KB size of IDRAM1 and IDRAM2 is used for the BLE subsystem and also for the user application. IDRAM1 contains three 32 KB memories and IDRAM2 contains two 16 KB memories that are accessible to the ARM and used for instruction/data storage.

The RAM memory is retained when the processor either goes into the Sleep mode or Power-Down mode.

RAM memory is used by the user application as well as ROM firmware for data storage. The memory split-up between application and firmware, and available RAM space for user application may change when there is a BluSDK SMART release. Refer to BluSDK SMART Example Profiles Application User Guide for the memory map of this memory section.

7.2 ROM

The ROM is used to store the boot code and BLE firmware, stack, and selected profiles. The ROM contains the 128 KB memory that is accessible to the ARM. The boot loader code stored in ROM loads the application firmware from the SPI Flash to RAM.

7.3 BLE Retention Memory

The BLE functionality requires 8 KB state, instruction, and data to be retained in memory, when the processor either goes into Sleep mode or Power down mode. The RAM is separated into specific power domains to allow tradeoff in power consumption with retention memory size.

7.4 Flash Memory

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have 256 kB of Flash memory, stacked on top of the MCU and BLE SoC (Part Number: W25X20CL). It is accessed through the SPI Flash controller.

Flash memory features are:

- 256 bytes per programmable page
- Uniform 4 kB Sectors, 32 kB & 64 kB Blocks

- Sector Erase (4 Kbyte)
- Block Erase (32 K or 64 Kbyte)
- Page program up to 256 bytes <1 ms
- More than 100,000 erase/write cycles and more than 20-year data retention
- 2.3 V to 3.6 V supply range

Dedicated LP_SIP_x pins TXD, RXD, SCK, SSN, WP (Write-protect) are connected to stacked Flash. This stacked Flash stores the firmware patch, user application images. This image is bootloaded on RAM on every power-on. As ATSAMB11 has 128 KB of RAM memory and the application and firmware patch is limited to 128 KB. The remaining 128 KB of SPI Flash can be used for OTA image storage or user data storage. For more details on SPI Flash memory split up if application uses OTA, see the ATSAMB11 BluSDK SMART OTAU Profile Getting Started Guide. If the user application does not use OTA, then SPI Flash address from 0x20000 to 0x3FFFF can be used for user data storage.

7.5 Non-Volatile Memory

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have 768 bits of non-volatile eFuse memory that can be read by the CPU after device reset. This memory region is One-Time-Programmable (OTP). It is partitioned into six 128-bit banks. Each bank is divided into four blocks with each block containing 32 bits of memory locations. This non-volatile, OTP memory is used to store customer specific parameters as listed below.

- 26 MHz XO Calibration information
- BT address

The bit map for the block containing the above parameters is detailed in the following figures.

Figure 7-1. Bank 5 Block 0

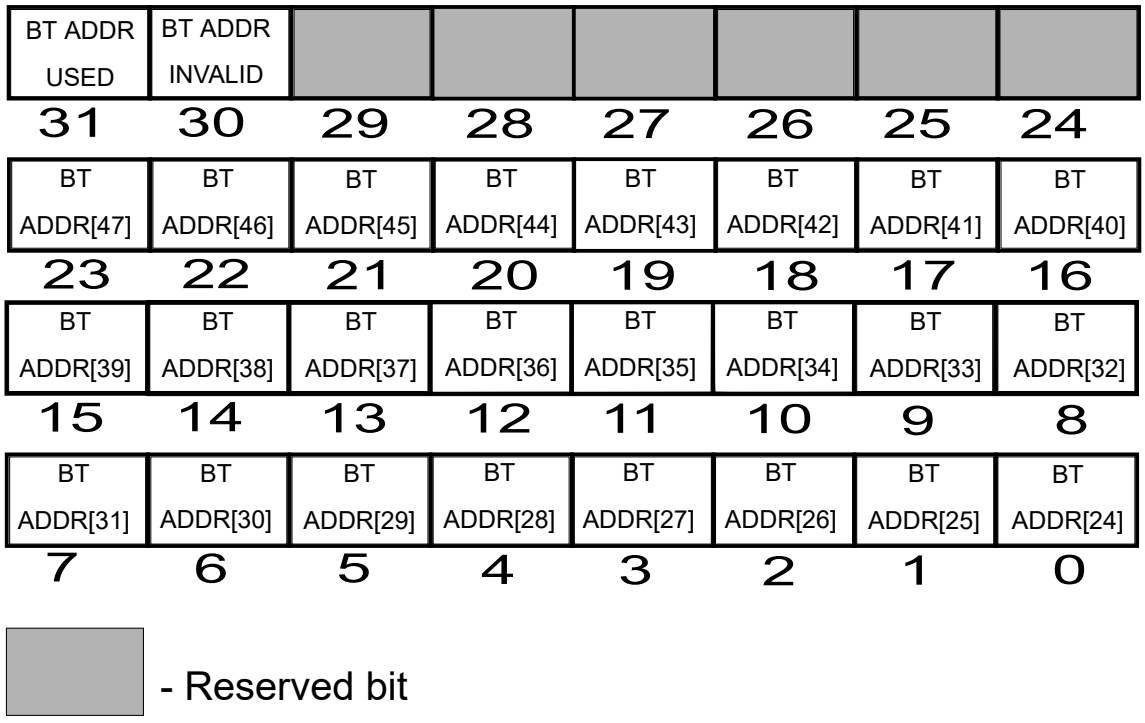


Figure 7-2. Bank 5 Block 1

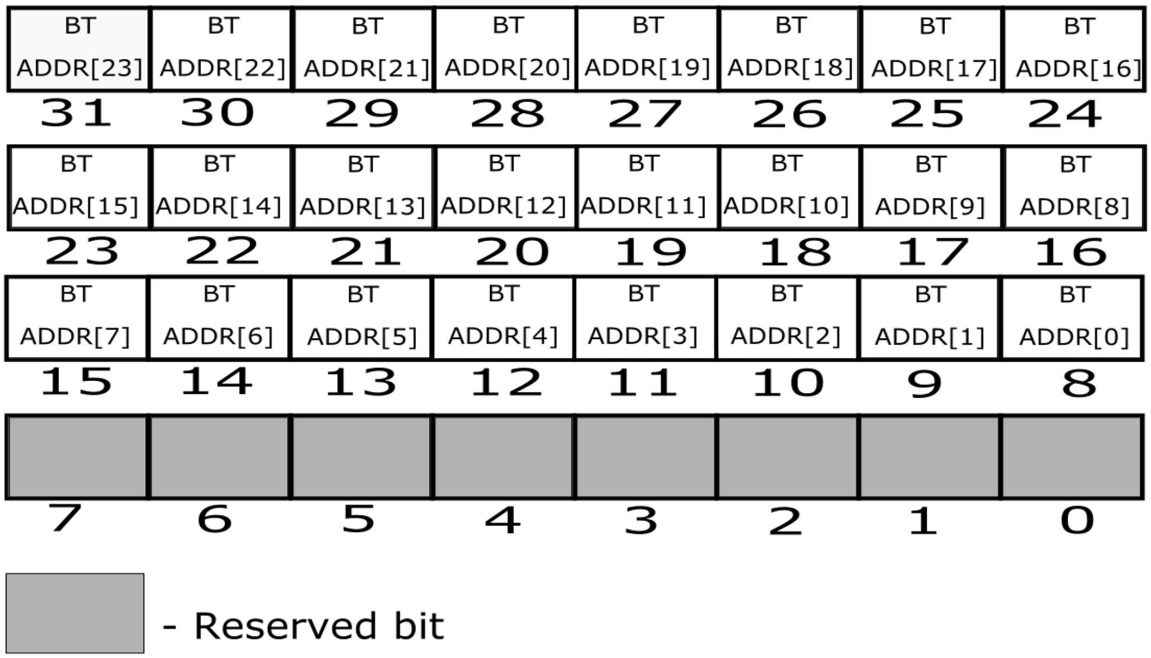
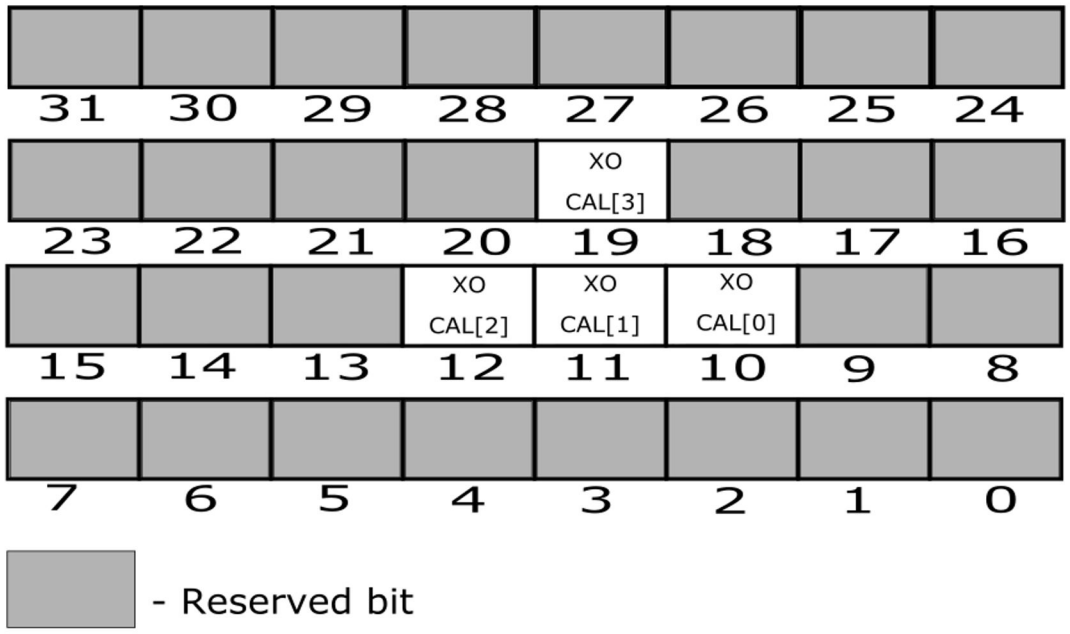


Figure 7-3. Bank 5 Block 3



The bits that are not depicted in the above register description are all reserved for future use.

7.5.1 26 MHz XO Calibration information

Information for both ATSAMB11-XR2100A and ATSAMB11-ZR210CA will be pre-programmed.

7.5.2 BT Address

These bits contain the BT address used by the user application. For ATSAMB11-ZR210CA modules, the BT address is pre-programmed. For ATSAMB11-XR2100A, the user must purchase the MAC address from IEEE and program it to the designated bit locations of the non-volatile memory.

Programming Bit 31 of Bank 5 Block 0 (BT_ADDR_USED) with a value of 1 indicates that the BT address in the eFuse memory location is intended to be used.

Programming Bit 30 of Bank 5 Block 0 (BT_ADDR_INVALID) with a value of 1 indicates that the BT address in the eFuse memory location is invalid.

7.5.3 Non-Volatile eFuse Memory Programming

7.5.3.1 Clock Configuration

The clock must be enabled before accessing the eFuse NVM register. This is done by setting the EFUSEn_CLK_EN (n=0,1.. 5) bit in the LPMCU_CLOCK_ENABLES_1 register. Each eFuse bank clock is gated by an individual CLK_EN bit. For more details on configuration, see Peripheral Clock Configuration.

7.5.3.2 Reading eFuse Memory

Perform the following steps to read the efuse memory:

1. Clear EFUSE_LDO_EN and PMU_BGR_EN bits of the AON_PMU_CTRL register
2. Write 0x007C082D to the intended eFuse Bank Control register (EFUSE_n_CONTROL). The completion of write operation is indicated by setting ‘1’ to EFUSE_LOAD_DONE bit of the EFUSE_n_CONTROL register.
3. The 128 bits of each bank is divided into four blocks with 32 bits each. The eFuse can be read as a block of 32 bits. Read the specific eFuse block from the EFUSE_n_STATUS_b register (n (Bank)=1, 2, to 6, b(Block)=0, 1, 2, and 3)

7.5.3.3 Writing eFuse Memory

Perform the following steps to write the efuse memory:

1. Clear EFUSE_LDO_EN and PMU_BGR_EN bits of the AON_PMU_CTRL register.
2. Load 0x007C082D to the intended eFuse bank control register (EFUSE_n_CONTROL). The completion of write operation for the control register is indicated by setting ‘1’ to EFUSE_LOAD_DONE bit of the EFUSE_n_CONTROL register.
3. The eFuse can be written as a block of 32 bits. Load the intended value to be written into specific bank and block of the program register, EFUSE_n_PROG_b (n (bank)=1, 2, to 5, b (block)=0, 1, 2, and 3).
4. Set EFUSE_LDO_EN and PMU_BGR_EN bits of the AON_PMU_CTRL register to ‘1’.
5. Write 0x007C081E to the intended eFuse bank control register (EFUSE_n_CONTROL). The completion of write operation for the control register is indicated by setting ‘1’ to EFUSE_LOAD_DONE bit of the EFUSE_n_CONTROL register. The eFuse block is programmed at the end of this process.
6. Write 0x007C081C to the intended eFuse bank control register (EFUSE_n_CONTROL).
7. Clear EFUSE_LDO_EN and PMU_BGR_EN bits of the AON_PMU_CTRL register

7.5.3.4 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000A000	EFUSE_MISC_REGS0	EFUSE_GLOBAL_RESET	7:0		EFUSE_6_RSTN	EFUSE_5_RSTN	EFUSE_4_RSTN	EFUSE_3_RSTN	EFUSE_2_RSTN	EFUSE_1_RSTN	GLOBAL_RSTN
0x4000A004 (EFUSE_1), 0x4000A008 (EFUSE_2), 0x4000A00C (EFUSE_3), 0x4000A010 (EFUSE_4), 0x4000A014 (EFUSE_5), 0x4000A018 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_CONTROL (n=1,2..6)	7:0	FIRST_READ_COUNT[5:0]						START_PROGRAM	FORCE_LOAD
			15:8	PROG_CLK_H_COUNT[0:0]	FIRST_PROG_COUNT[3:0]				FIRST_READ_COUNT[8:6]		
			23:16	PROG_CLK_H_COUNT[8:1]							
			31:24	EFUSE_LOAD_DONE					DEBUG_BUS_SEL[2:0]		
0x4000A01C (EFUSE_1), 0x4000A03C (EFUSE_2), 0x4000A05C (EFUSE_3), 0x4000A07C (EFUSE_4), 0x4000A09C (EFUSE_5), 0x4000A0BC (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_PROG_0 (n=1,2..6)	7:0	EFUSE_n_PROG_0[7:0]							
			15:8	EFUSE_n_PROG_0[15:8]							
			23:16	EFUSE_n_PROG_0[23:16]							
			31:24	EFUSE_n_PROG_0[31:24]							
0x4000A020 (EFUSE_1), 0x4000A040 (EFUSE_2), 0x4000A060 (EFUSE_3), 0x4000A080 (EFUSE_4), 0x4000A0A0 (EFUSE_5), 0x4000A0C0 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_PROG_1 (n=1,2..6)	7:0	EFUSE_n_PROG_1[7:0]							
			15:8	EFUSE_n_PROG_1[15:8]							
			23:16	EFUSE_n_PROG_1[23:16]							
			31:24	EFUSE_n_PROG_1[31:24]							
0x4000A024 (EFUSE_1), 0x4000A044 (EFUSE_2), 0x4000A064 (EFUSE_3), 0x4000A084 (EFUSE_4), 0x4000A0A4 (EFUSE_5), 0x4000A0C4 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_PROG_2 (n=1,2..6)	7:0	EFUSE_n_PROG_2[7:0]							
			15:8	EFUSE_n_PROG_2[15:8]							
			23:16	EFUSE_n_PROG_2[23:16]							
			31:24	EFUSE_n_PROG_2[31:24]							
0x4000A028 (EFUSE_1), 0x4000A048 (EFUSE_2), 0x4000A068 (EFUSE_3), 0x4000A088 (EFUSE_4), 0x4000A0A8 (EFUSE_5), 0x4000A0C8 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_PROG_3 (n=1,2..6)	7:0	EFUSE_n_PROG_3[7:0]							
			15:8	EFUSE_n_PROG_3[15:8]							
			23:16	EFUSE_n_PROG_3[23:16]							
			31:24	EFUSE_n_PROG_3[31:24]							
0x4000A02C (EFUSE_1), 0x4000A04C (EFUSE_2), 0x4000A06C (EFUSE_3), 0x4000A08C (EFUSE_4), 0x4000A0AC (EFUSE_5), 0x4000A0CC (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_STATUS_0 (n=1,2..6)	7:0	EFUSE_n_STATUS_0[7:0]							
			15:8	EFUSE_n_STATUS_0[15:8]							
			23:16	EFUSE_n_STATUS_0[23:16]							
			31:24	EFUSE_n_STATUS_0[31:24]							
0x4000A030 (EFUSE_1), 0x4000A050 (EFUSE_2), 0x4000A070 (EFUSE_3), 0x4000A090 (EFUSE_4), 0x4000A0B0 (EFUSE_5), 0x4000A0D0 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_STATUS_1 (n=1,2..6)	7:0	EFUSE_n_STATUS_1[7:0]							
			15:8	EFUSE_n_STATUS_1[15:8]							
			23:16	EFUSE_n_STATUS_1[23:16]							
			31:24	EFUSE_n_STATUS_1[31:24]							

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000A034 (EFUSE_1), 0x4000A054 (EFUSE_2), 0x4000A074 (EFUSE_3), 0x4000A094 (EFUSE_4), 0x4000A0B4 (EFUSE_5), 0x4000A0D4 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_STATUS_2 (n=1,2..6)	7:0	EFUSE_n_STATUS_2[7:0]							
			15:8	EFUSE_n_STATUS_2[15:8]							
			23:16	EFUSE_n_STATUS_2[23:16]							
			31:24	EFUSE_n_STATUS_2[31:24]							
0x4000A038 (EFUSE_1), 0x4000A058 (EFUSE_2), 0x4000A078 (EFUSE_3), 0x4000A098 (EFUSE_4), 0x4000A0B8 (EFUSE_5), 0x4000A0D8 (EFUSE_6)	EFUSE_MISC_REGS0	EFUSE_n_STATUS_3 (n=1,2..6)	7:0	EFUSE_n_STATUS_3[31:24]							
			15:8	EFUSE_n_STATUS_3[23:16]							
			23:16	EFUSE_n_STATUS_3[15:8]							
			31:24	EFUSE_n_STATUS_3[7:0]							
0x4000A0DC	EFUSE_MISC_REGS0	EFUSE_MISC_CTRL	7:0								OUT_OF_RESET

7.5.3.5 Register Description

7.5.3.5.1 eFuse Global Reset

Name: EFUSE_GLOBAL_RESET
Reset: 0x7F

Absolute Address: 0x4000A000

This register is a part of EFUSE_MISC_REGS0 registers. This register allows the user to reset the eFuse registers to default values.
Note: This register does not reset the eFuse program memory, as eFuse is one time programmable memory.

Bit	7	6	5	4	3	2	1	0
		EFUSE_6_RSTN	EFUSE_5_RSTN	EFUSE_4_RSTN	EFUSE_3_RSTN	EFUSE_2_RSTN	EFUSE_1_RSTN	GLOBAL_RSTN
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		1	1	1	1	1	1	1

Bit 6 – EFUSE_6_RSTN
Writing '0' to this bit resets EFUSE bank 6
Writing '1' to this bit allows normal operations

Bit 5 – EFUSE_5_RSTN
Writing '0' to this bit resets EFUSE bank 5
Writing '1' to this bit allows normal operations

Bit 4 – EFUSE_4_RSTN
Writing '0' to this bit resets EFUSE bank 4
Writing '1' to this bit allows normal operations

Bit 3 – EFUSE_3_RSTN
Writing '0' to this bit resets EFUSE bank 3
Writing '1' to this bit allows normal operations

Bit 2 – EFUSE_2_RSTN
Writing '0' to this bit resets EFUSE bank 2
Writing '1' to this bit allows normal operations

Bit 1 – EFUSE_1_RSTN
Writing '0' to this bit resets EFUSE bank 1
Writing '1' to this bit allows normal operations

Bit 0 – GLOBAL_RSTN
Writing '0' to this bit resets all six EFUSE banks
Writing '1' to this bit allows normal operations

7.5.3.5.2 eFuse Global Reset

Name: EFUSE_n_CONTROL (n=1,2..6)
Reset: 0x000007FC

Absolute Address: 0x4000A004 (EFUSE_1), 0x4000A008 (EFUSE_2), 0x4000A00C (EFUSE_3), 0x4000A010 (EFUSE_4), 0x4000A014 (EFUSE_5), 0x4000A018 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. The eFuse control register has six eFuse banks and hence there are six EFUSE_n_CONTROL(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_LOAD_DONE					DEBUG_BUS_SEL[2:0]		
Access	R					R/W	R/W	R/W
Reset	0					0	0	0

Bit	23	22	21	20	19	18	17	16
	PROG_CLK_H_COUNT[8:1]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PROG_CLK_H_COUNT[0:0]	FIRST_PROG_COUNT[3:0]				FIRST_READ_COUNT[8:6]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	1

Bit	7	6	5	4	3	2	1	0
	FIRST_READ_COUNT[5:0]						START_PROGRAM	FORCE_LOAD
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	0	0

Bit 31 – EFUSE_LOAD_DONE
Reading '1' to this bit indicates the loading of value in EFUSE_n_CONTROL register is complete
It is required to wait for this bit to set when the Control register is written with the value

Bits 26:24 – DEBUG_BUS_SEL[2:0]
These bits are 'INTERNAL' and recommended to use 0x00 as value for these bits

Bits 23:15 – PROG_CLK_H_COUNT[8:0]
These bits set the first eFuse clock count. The recommended value for these bits is 0xF8.

Bits 14:11 – FIRST_PROG_COUNT[3:0]
These bits set the first eFuse program count. The recommended value for these bits is 0x01.

Bits 10:2 – FIRST_READ_COUNT[8:0]
These bits set the first eFuse read count. The recommended value for read operation is 0x0B and the recommended value for write operation is 0x07.

Bit 1 – START_PROGRAM
Writing '1' to this bit writes the eFuse data in eFuse memory

Bit 0 – FORCE_LOAD
Writing '1' to this bit drives the loading of EFUSE_CONTROL_n register with a new value

7.5.3.5.3 eFuse Program Block 0

Name: EFUSE_n_PROG_0 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A01C (EFUSE_1), 0x4000A03C (EFUSE_2), 0x4000A05C (EFUSE_3), 0x4000A07C (EFUSE_4), 0x4000A09C (EFUSE_5), 0x4000A0BC (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. The eFuse Block 0 data which must be written into eFuse memory must be placed in this register. There are six eFuse banks and hence there are six EFUSE_n_PROG_0(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_PROG_0[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_PROG_0[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_PROG_0[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_PROG_0[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_PROG_0[31:0]
These bits hold the data to be written into Block 0 of EFUSE_n bank (n=1,2,..6)
At the end of eFuse programming, the data is loaded into eFuse bank n block 0 memory.

7.5.3.5.4 eFuse Program Block 1

Name: EFUSE_n_PROG_1 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A020 (EFUSE_1), 0x4000A040 (EFUSE_2), 0x4000A060 (EFUSE_3), 0x4000A080 (EFUSE_4), 0x4000A0A0 (EFUSE_5), 0x4000A0C0 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. The eFuse Block 1 data which must be written into eFuse memory must be placed in this register. There are six eFuse banks and hence there are six EFUSE_n_PROG_1(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_PROG_1[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_PROG_1[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_PROG_1[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_PROG_1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_PROG_1[31:0]
These bits hold the data to be written into Block 1 of EFUSE_n bank (n=1,2,..6)
At the end of eFuse programming, the data is loaded into eFuse bank n block 1 memory.

7.5.3.5.5 eFuse Program Block 2

Name: EFUSE_n_PROG_2 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A024 (EFUSE_1), 0x4000A044 (EFUSE_2), 0x4000A064 (EFUSE_3), 0x4000A084 (EFUSE_4), 0x4000A0A4 (EFUSE_5), 0x4000A0C4 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. The eFuse Block 2 data which must be written into eFuse memory must be placed in this register. There are six eFuse banks and hence there are six EFUSE_n_PROG_2(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_PROG_2[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_PROG_2[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_PROG_2[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_PROG_2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_PROG_2[31:0]
These bits hold the data to be written into Block 2 of EFUSE_n bank (n=1,2,..6)
At the end of eFuse programming, the data is loaded into eFuse bank n block 2 memory.

7.5.3.5.6 eFuse Program Block 3

Name: EFUSE_n_PROG_3 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A028 (EFUSE_1), 0x4000A048 (EFUSE_2), 0x4000A068 (EFUSE_3), 0x4000A088 (EFUSE_4), 0x4000A0A8 (EFUSE_5), 0x4000A0C8 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. The eFuse Block 3 data which must be written into eFuse memory must be placed in this register. There are six eFuse banks and hence there are six EFUSE_n_PROG_3(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_PROG_3[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_PROG_3[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_PROG_3[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_PROG_3[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_PROG_3[31:0]
These bits hold the data to be written into Block 3 of EFUSE_n bank (n=1,2,..6)
At the end of eFuse programming, the data is loaded into eFuse bank n block 3 memory.

7.5.3.5.7 eFuse Block 0 Read Data

Name: EFUSE_n_STATUS_0 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A02C (EFUSE_1), 0x4000A04C (EFUSE_2), 0x4000A06C (EFUSE_3), 0x4000A08C (EFUSE_4), 0x4000A0AC (EFUSE_5), 0x4000A0CC (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. This register holds the eFuse Block 0 data. There are six eFuse banks and hence there are six EFUSE_n_STATUS_0(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_STATUS_0[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_STATUS_0[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_STATUS_0[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_STATUS_0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_STATUS_0[31:0]
These bits hold the Block 0 of EFUSE_n bank (n=1,2,..6) data

7.5.3.5.8 eFuse Block 1 Read Data

Name: EFUSE_n_STATUS_1 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A030 (EFUSE_1), 0x4000A050 (EFUSE_2), 0x4000A070 (EFUSE_3), 0x4000A090 (EFUSE_4), 0x4000A0B0 (EFUSE_5), 0x4000A0D0 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. This register holds the eFuse Block 1 data. There are six eFuse banks and hence there are six EFUSE_n_STATUS_1(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_STATUS_1[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_STATUS_1[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_STATUS_1[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_STATUS_1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_STATUS_1[31:0]
These bits hold the Block 1 of EFUSE_n bank (n=1,2,..6) data

7.5.3.5.9 eFuse Block 2 Read Data

Name: EFUSE_n_STATUS_2 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A034 (EFUSE_1), 0x4000A054 (EFUSE_2), 0x4000A074 (EFUSE_3), 0x4000A094 (EFUSE_4), 0x4000A0B4 (EFUSE_5), 0x4000A0D4 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. This register holds the eFuse Block 2 data. There are six eFuse banks and hence there are six EFUSE_n_STATUS_2(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_STATUS_2[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_STATUS_2[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_STATUS_2[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_STATUS_2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_STATUS_2[31:0]
These bits hold the Block 2 of EFUSE_n bank (n=1,2,..6) data

7.5.3.5.10 eFuse Block 3 Read Data

Name: EFUSE_n_STATUS_3 (n=1,2..6)
Reset: 0x00000000

Absolute Address: 0x4000A038 (EFUSE_1), 0x4000A058 (EFUSE_2), 0x4000A078 (EFUSE_3), 0x4000A098 (EFUSE_4), 0x4000A0B8 (EFUSE_5), 0x4000A0D8 (EFUSE_6)

This register is a part of EFUSE_MISC_REGS0 registers. This register holds the eFuse Block 3 data. There are six eFuse banks and hence there are six EFUSE_n_STATUS_3(n=1,2..6) registers.

Bit	31	30	29	28	27	26	25	24
	EFUSE_n_STATUS_3[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EFUSE_n_STATUS_3[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFUSE_n_STATUS_3[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_n_STATUS_3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – EFUSE_n_STATUS_3[31:0]
These bits hold the Block 3 of EFUSE_n bank (n=1,2,..6) data

7.5.3.5.11 eFuse Miscellaneous

Name: EFUSE_MISC_CTRL
Reset: 0x01

Absolute Address: 0x4000A0DC

This register is a part of EFUSE_MISC_REGS0 registers and provides the Reset status of eFuse.

Bit	7	6	5	4	3	2	1	0
								OUT_OF_RESET
Access								R
Reset								1

Bit 0 – OUT_OF_RESET
Reading '1' to this bit indicates eFuse is out of Reset

8. Bluetooth Low Energy Subsystem

The Bluetooth Low Energy (BLE) subsystem implements all the critical real-time functions required for full compliance with specification of the Bluetooth System v5.0, Bluetooth SIG. It consists of a Bluetooth baseband controller (core), radio transceiver and the Microchip Bluetooth Smart Stack, and the BLE Software Platform.

8.1 BLE Core

The baseband controller consists of a modem and a Medium Access Controller (MAC). It constructs baseband data packages, schedules frames, and manages and monitors connection status, slot usage, data flow, routing, segmentation and buffer control.

The core performs link control layer management supporting the main BLE states, including advertising and connection.

8.2 Features

- Broadcaster, Central, Observer, Peripheral
- Simultaneous Master and Slave operation, connect up to eight connections
- Frequency Hopping
- Advertising/Data/Control packet types
- Encryption (AES-128, SHA-256)
- Bitstream processing (CRC, whitening)
- Operating clock 52 MHz

8.3 BLE Radio

The radio consists of a fully integrated transceiver, low noise amplifier, Receive (RX) down converter, analog baseband processing, Phase Locked Loop (PLL), Transmit (TX) Power Amplifier, and Transmit/Receive switch. At the RF front end, no external RF components on the PCB are required other than the antenna and a matching component.

Table 8-1. ATSAMB11 BLE Radio Features and Properties

Feature	Description
Part Number	ATSAMB11-XR2100A and ATSAMB11-ZR210CA
BLE standard	Bluetooth V5.0 – Bluetooth Low Energy
Frequency range	2402 MHz to 2480 MHz
Number of channels	40
Modulation	GFSK
PHY Data rate	1 Mbps

8.4 Microchip BluSDK Smart

The BluSDK Smart offers a comprehensive set of tools including reference applications for several Bluetooth SIG defined profiles and custom profile. This will help the user quickly evaluate, design and develop BLE products with ATSAMB11-XR2100A and ATSAMB11-ZR210CA.

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have a completely integrated Bluetooth Low Energy stack on chip that is fully qualified, mature, and Bluetooth V5.0 compliant.

Customer applications interface with the BLE protocol stack through the Microchip BLE API, which supports direct access to the GAP, SMP, ATT, GATT client / server, and L2CAP service layer protocols in the embedded firmware.

The stack includes numerous BLE profiles for applications such as:

- Smart Energy
- Consumer Wellness
- Home Automation
- Security
- Proximity Detection
- Entertainment
- Sports and Fitness
- Automotive

Together with the Atmel Studio Software Development environment, the additional customer profiles can be easily developed.

In addition to the protocol stack, the drivers for each peripheral hardware block are provided as part of the Advanced Software Framework (ASF).
Note: The SysTick Timer, ARM Timer, and WDT0 peripherals are used by BLE stack for its operation. It is not allowed by the user application to use those peripherals. For recommended clock setting by the BLE stack, see 9.5 Clock Settings for Critical Sections.

8.4.1 Direct Test Mode (DTM) Example Application

A DTM example application is among the reference applications offered in BluSDK Smart. Using this application, the user will be able to configure the device in the different test modes as defined in the Bluetooth Low Energy Core 5.0 specification (Vol6,Part F Direct Test Mode). Please refer the example *Getting Started Guide* available in the BluSDK Smart release package.

9. Clocking

9.1 Overview

Figure 9-1. Clock Architecture

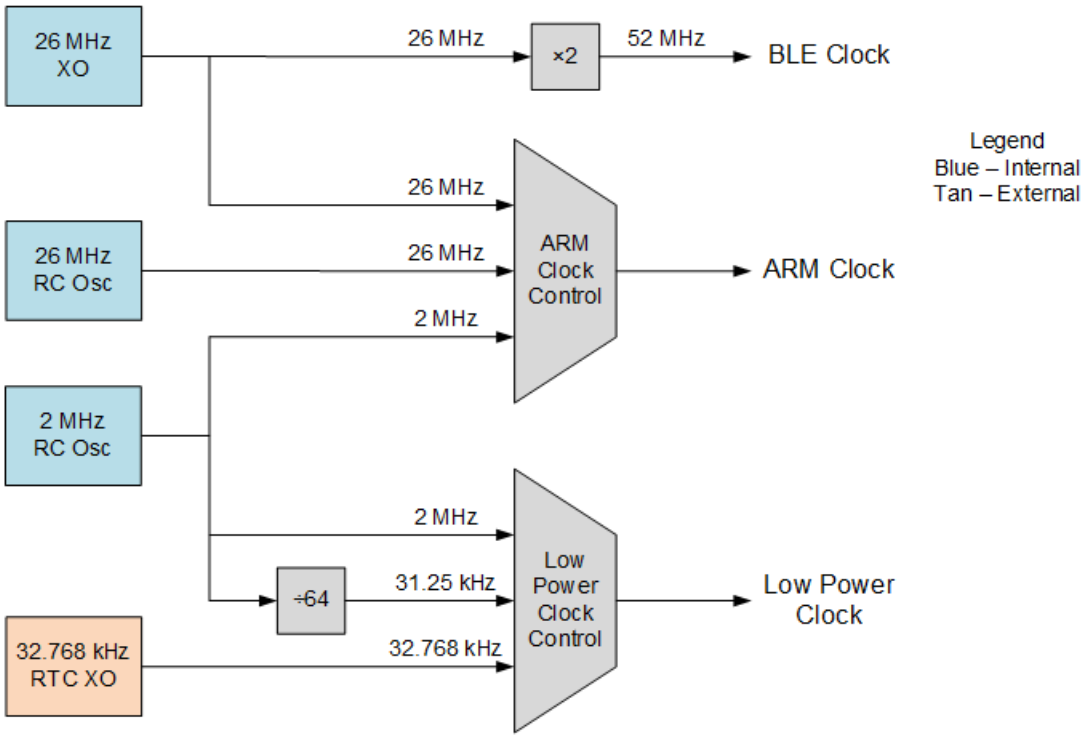


Figure 9-1 provides an overview of the clock tree and clock management blocks.

The BLE Clock is used to drive the BLE subsystem. The ARM clock is used to drive the Cortex-M0 MCU and its interfaces (UART, SPI, and I^2C); the recommended MCU clock speed is 26 MHz. The Low Power Clock is used to drive all the low-power applications like the BLE sleep timer, always-on power sequencer, always-on timer, and others.

The 26 MHz integrated RC Oscillator is used for most general purpose operations on the MCU and its peripherals. In cases when the BLE subsystem is not used, the RC oscillator can be used for lower power consumption. The frequency variation of this RC oscillator is up to ±50% over process, voltage, and temperature.

The frequency variation of 2 MHz integrated RC Oscillator is up to ±50% over process, voltage, and temperature.

The 32.768 kHz RTC Crystal Oscillator (RTC XO) is used for BLE operations as it will reduce power consumption by providing the best timing for wake-up precision, allowing circuits to be in low-power sleep mode for as long as possible until they need to wake-up and connect during the BLE connection event.

9.2 26 MHz Crystal Oscillator (XO)

A 26 MHz crystal oscillator is integrated into the ATSAMB11-XR2100A and ATSAMB11-ZR210CA to provide the precision clock for the BLE operations.

9.3 32.768 kHz RTC Crystal Oscillator (RTC XO)

9.3.1 General Information

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA contain a 32.768 kHz RTC oscillator that is used for BLE activities involving connection events. To be compliant with the BLE specifications for connection events, the frequency accuracy of this clock has to be within ±500 ppm. Because of the high accuracy of the 32.768 kHz crystal oscillator clock, the power consumption can be minimized by leaving radio circuits in low-power Sleep mode for as long as possible, until they need to wake up for the next connection timed event.

The block diagram in Figure 9-2 below shows how the internal low-frequency Crystal Oscillator (XO) is connected to the external crystal.

The RTC XO has a programmable internal capacitance with a maximum of 15 pF on each terminal, RTC_CLK_P, and RTC_CLK_N. When bypassing the crystal oscillator with an external signal, the user can program down the internal capacitance to its minimum value (~1 pF) for easier driving capability. The driving signal can be applied to the RTC_CLK_P terminal, as illustrated in Figure 9-2 below.

The need for external bypass capacitors depends on the chosen crystal characteristics. Typically, the crystal should be chosen to have a load capacitance of 7 pF to minimize the oscillator current. Refer to the datasheet of the preferred crystal and take into account the on-chip capacitance.

Alternatively, if an external 32.768 kHz clock is available, it can be used to drive the RTC_CLK_P pin instead of using a crystal. The XO has 6 pF internal capacitance on the RTC_CLK_P pin. To bypass the crystal oscillator, an external signal capable of driving 6 pF can be applied to the RTC_CLK_P terminal, as illustrated in Figure 9-2. RTC_CLK_N must be left unconnected when driving an external source into RTC_CLK_P. Refer to the Table 9-1 for the specification of the external clock to be supplied at RTC_CLK_P.

Figure 9-2. Connections to RTC XO

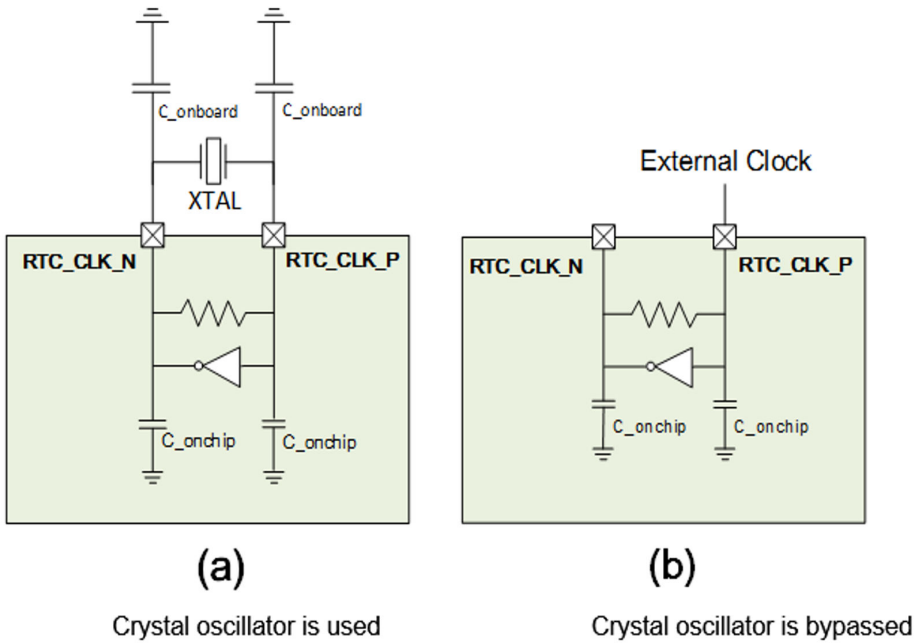


Table 9-1. 32.768 kHz External Clock Specification

Parameter	Min.	Typ.	Max	Unit	Comments
Oscillation frequency		32.768		kHz	Must be able to drive 6 pF load at desired frequency
VinH	0.7		1.2	V	High-level input voltage
VinL	0		0.2		Low-level input voltage
Stability – Temperature	-250		+250	ppm	

Additional internal trimming capacitors (C_onchip) are available. They provide the possibility to tune the frequency output of RTC XO without changing the external load capacitors. Contact technical support for usage of the internal trimming capacitors.

Note:
Refer to the BluSDK BLE API Software Development Guide for details on how to enable the 32.768 kHz clock output and tune the internal trimming capacitors.

Table 9-2. 32.768 kHz XTAL C_onchip Programming

Register: PIERCE_CAP_CTRL[3:0]	C_onchip [pF]
0000	0.0
0001	1.0
0010	2.0
0011	3.0
0100	4.0
0101	5.0
0110	6.0
0111	7.0
1000	8.0
1001	9.0
1010	10.0
1011	11.0
1100	12.0
1101	13.0
1110	14.0
1111	15.0

9.3.2 RTC XO Design and Interface Specification

The RTC consists of two main blocks: The Programmable Gm stage and tuning capacitors. The programmable Gm stage is used to guarantee start-up and to sustain oscillation. Tuning capacitors are used to adjust the XO center frequency and control the XO precision for different crystal models. The output of the XO is driven to the digital domain via a digital buffer stage with a supply voltage of 1.2V.

Table 9-3. RTC XO Interface

Pin Name	Function	Register Default
Digital Control Pins		
PIERCE_RES_CTRL	Control feedback resistance value: 0 = 20 MOhm Feedback resistance 1 = 30 MOhm Feedback resistance	0X4000F404<15>='1'
PIERCE_CAP_CTRL[3:0]	Control the internal tuning capacitors with step of 700 fF: 0000=700 fF 1111=11.2 pF Refer to crystal datasheet to check for optimum tuning cap value	0X4000F404<23:20>="1000"
PIERCE_GM_CTRL[3:0]	Controls the Gm stage gain for different crystal mode: 0011= for crystal with shunt capacitance of 1.2 pF 1000= for crystal with shunt capacitance of >3 pF	0X4000F404<19:16>="1000"
Supply Pins		
VDD_XO	1.2V	-

9.3.3 RTC Characterization with Gm Code Variation at Supply 1.2V and Temp. = 25°C

This section shows the RTC total drawn current and the XO accuracy versus different tuning capacitors and different GM codes, at a supply voltage of 1.2V and temperature = 25°C.

Figure 9-3. RTC Drawn Current vs. Tuning Caps at 25°C

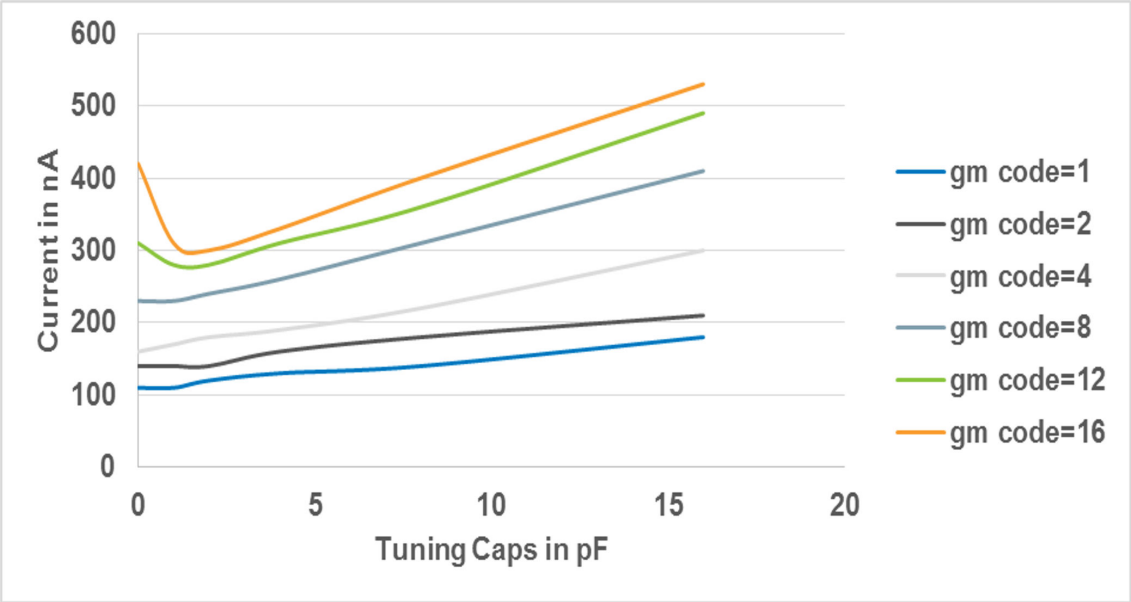
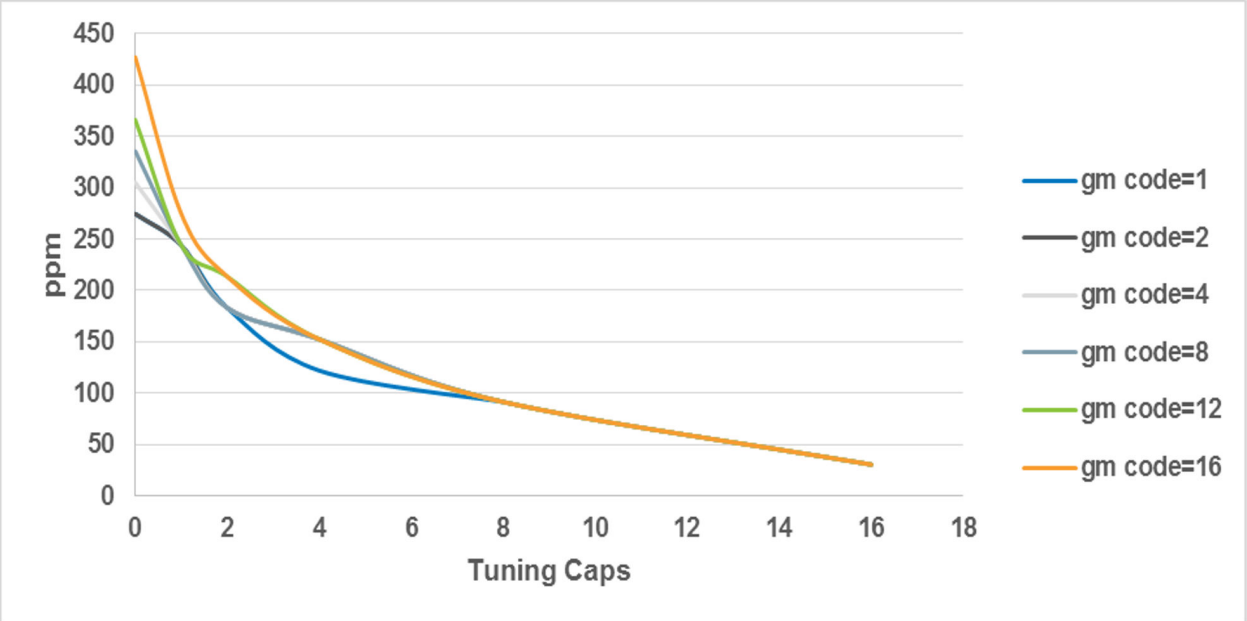


Figure 9-4. RTC Oscillation Frequency Deviation vs. Tuning Caps at 25°C



9.3.4 RTC Characterization with Supply Variation and Temp. = 25°C

Figure 9-5. RTC Drawn Current vs. Supply Variation

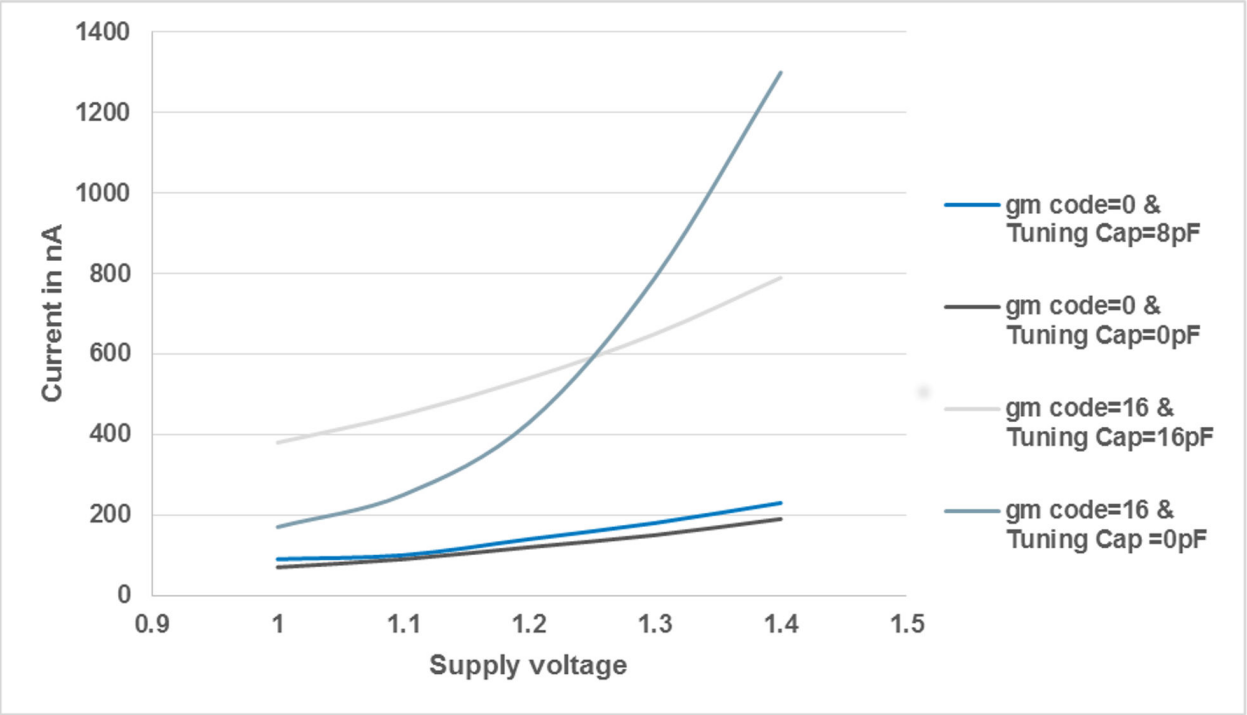
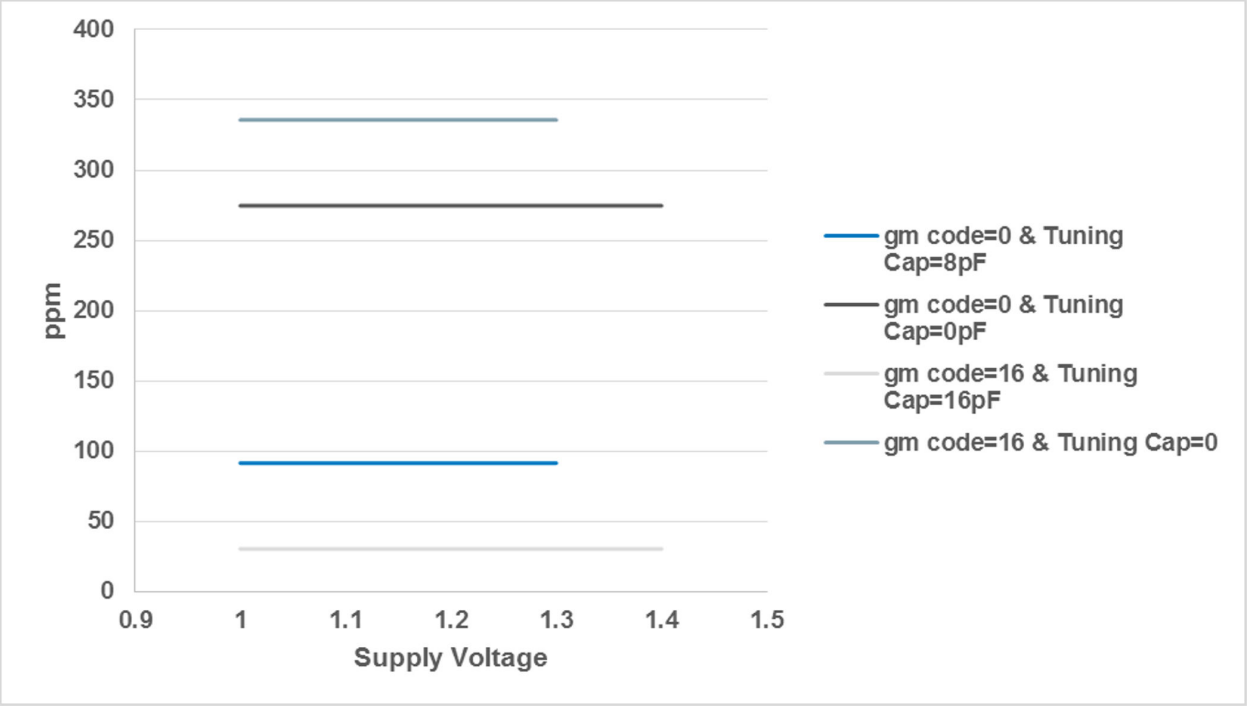


Figure 9-6. RTC Frequency Deviation vs. Supply Voltage



9.4 2 MHz Integrated RC Oscillator

The 2 MHz integrated RC Oscillator circuit without calibration contains a frequency variation of 50% over process, temperature, and voltage variation. As described above, calibration over process, temperature, and voltage is required to maintain the accuracy of this clock.

Figure 9-7. 32 kHz RC Oscillator PPM Variation vs. Calibration Time at Room Temperature

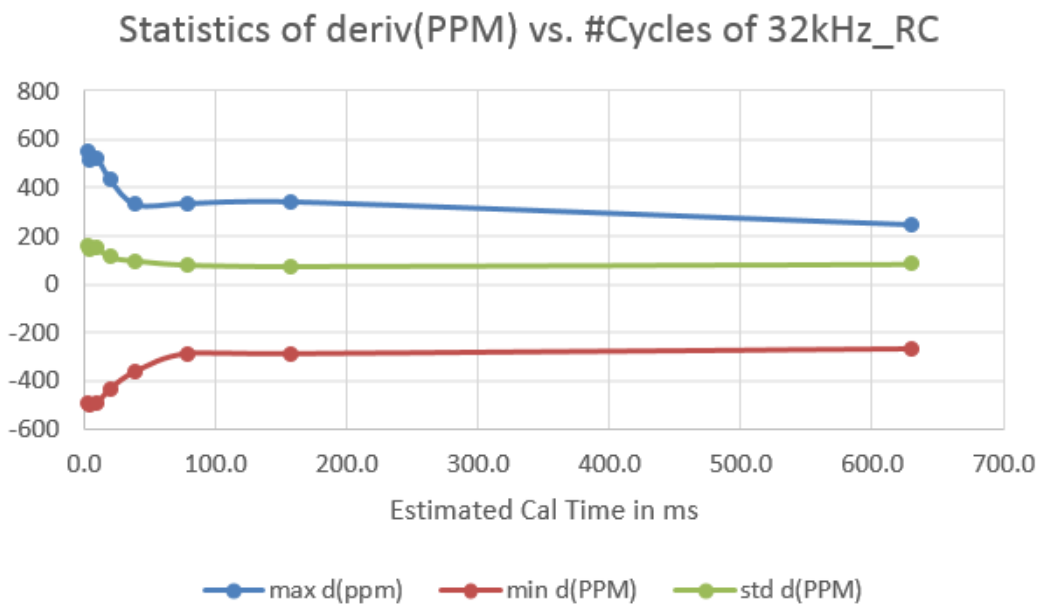
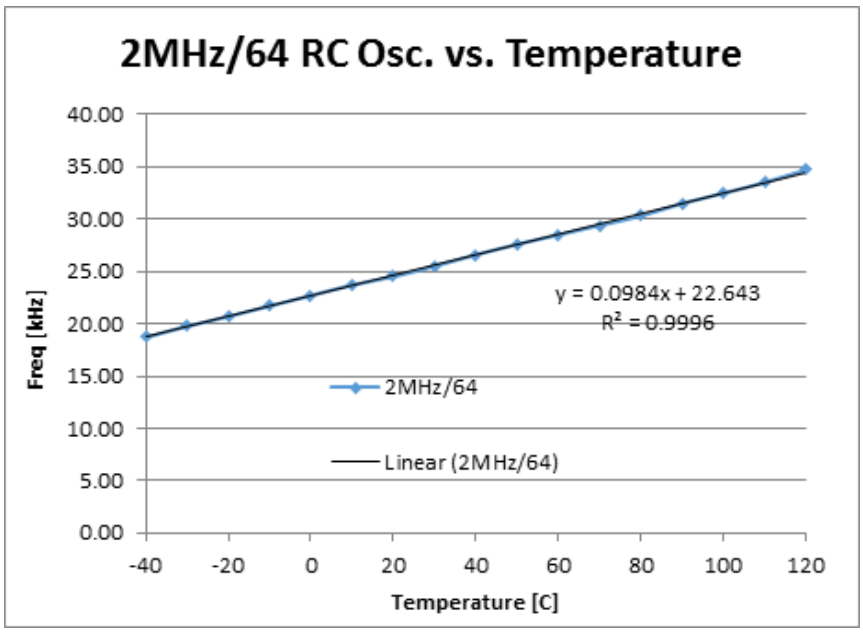


Figure 9-8. 32 kHz RC Oscillator Frequency Variation over Temperature



9.5 Clock Settings for Critical Sections

The three different clock sources 26 MHz XO, 26 MHz internal RC and 2 MHz internal RC can be used as input to ARM processor and other peripheral interfaces. As the clock configuration for some of the critical sections such as ARM, AON power sequencer, AON Sleep Timer, and BLE Sleep Timer are done by ROM firmware, the user is not recommended to change this clock configuration. This might affect the overall functionality. Therefore, the register descriptions related to this clock configurations are not provided.

9.5.1 ARM Processor Clock

26 MHz XO external crystal clock – this is the clock source for ARM processor and many of the peripherals.

9.5.2 AON Power Sequencer Clock

32.768 kHz RTC XO external crystal clock is the low power clock source for AON Power Sequencer module which controls the wake-up and sleep operations of ARM and BLE subsystem.

9.5.3 BLE Sleep Timer Clock

32.768 kHz RTC XO external crystal clock – this is the low power clock source for BLE Sleep Timer. This timer is used to wake-up BLE subsystem.

9.5.4 AON Sleep Timer Clock

32.768 kHz RTC XO external crystal clock – this is the low power clock source for AON Sleep Timer. This timer can be used by user application to wake-up ARM from ULP mode at predefined interval.

9.6 Peripheral Clock Configuration

ARM clock is the source for peripherals except for AON sleep timer. This clock is pre-scaled as per peripheral clock requirement. This clock is gated to peripherals and is enabled or disabled when required. This is to ensure the power is not consumed by the peripherals that are not used.

9.6.1 Enabling Peripheral Clock

Each peripheral clock is gated. Peripheral clock is enabled by setting the specific x_CLK_EN bit in LPMCU_CLOCK_ENABLES_0 or LPMCU_CLOCK_ENABLES_1 register. Few peripherals have gated clock for APB/AHB interface as well as for peripheral core operations. It is required to enable both the clocks for normal peripheral operation. For example, UART1_CORE_CLK_EN and UART1_IF_CLK_EN gate the clock for APB bus of UART1 and UART1 core operations.

Note: Few bits in these registers marked as “INTERNAL” are not recommended to change as those are controlled by ROM firmware.

9.6.2 Disabling Peripheral Clock

Peripheral clock is disabled by clearing the specific x_CLK_EN bit in LPMCU_CLOCK_ENABLES_0 or LPMCU_CLOCK_ENABLES_1 register.

9.6.3 Peripheral Reset

Each peripheral can be reset to default state by clearing the x_RSTN bit of LPMCU_GLOBAL_RESET_0 or LPMCU_GLOBAL_RESET_1 register. As long as x_RSTN bit is ‘0’ the peripheral is in reset state and cannot be configured for normal operations. Ensure that x_RSTN bit is set before configuring the peripheral for normal operation. The peripheral should be disabled before it is reset to avoid undefined

behavior. Some peripherals have gated clock for APB/AHB interface as well as for peripheral core operations. It is required to reset both clock. For example, UART1_CORE_RSTN and UART1_IF_RSTN bits to be cleared for resetting APB bus and UART1 core.

Note: Some bits in these registers marked as “INTERNAL” are not recommended to change as those are controlled by ROM firmware.

9.7 AON Sleep Timer Clock Configuration

The 32.768 kHz RTC XO external crystal clock is given as clock source to AON Sleep Timer by ROM firmware. This clock is gated to enable or disable when required.

9.7.1 Enabling AON Sleep Timer Clock

The AON Sleep Timer clock is enabled by setting AON_SLEEP_TIMER_CLK_EN bit in AON_MISC_CTRL register.

Note: Some bits in these registers marked as “INTERNAL” are not recommended to change as those are controlled by ROM firmware.

9.7.2 Disabling AON Sleep Timer Clock

The AON Sleep Timer clock is disabled by clearing AON_SLEEP_TIMER_CLK_EN bit in AON_MISC_CTRL register.

9.7.3 AON Sleep Timer Reset

The AON Sleep Timer can be reset to default state by clearing the SLEEP_TIMER_RSTN bit of AON_GLOBAL_RESET register. As long as SLEEP_TIMER_RSTN bit is ‘0’ the AON Sleep Timer is in reset state and cannot be configured for normal operations. Ensure that SLEEP_TIMER_RSTN bit is set before configuring the timer for normal operation. AON Sleep Timer should be disabled before it is reset in order to avoid undefined behavior.

Note: Some bits in these register marked as “INTERNAL” are not recommended to change as those are controlled by ROM firmware.

9.7.4 Global (Chip) Reset

By clearing the GLOBAL_RSTN bit of AON_GLOBAL_RESET register resets the entire chip. This is the auto set bit as it resets the entire chip.

9.8 Clock Output

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have an option to output a clock on any of LP_GPIO_x pin through test mux configuration (see I/O Multiplexing).

Note: This feature requires that the ARM and BLE subsystems must not be in the ULP mode.

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA can output the following fixed-frequency clocks:

- 52 MHz derived from XO (RX_52_MHz)
- 26 MHz derived from XO (TX_26_MHz)
- 32.768 kHz derived from the RTC XO (RTC_32 KHz)
- 26 MHz derived from 26 MHz RC Osc (RC_26_MHz)
- 6.5 MHz derived from XO (BT_6.5 MHz)
- 3.25 MHz derived from 26 MHz RC Osc (ARM 3.25 MHz)

Note:

- For clocks with a frequency of 26 MHz and above, ensure that external pad load on the board is minimized to get a clean waveform.
- The clock output on test mux pins are used only for debug purposes and are not recommended to use as input clock for another system/MCU.

Table 9-4. Test Bus Shift Control Value

	Rotate 20	Rotate 19	Rotate 18	Rotate 17	Rotate 16	Rotate 15	Rotate 14	Rotate 13	Rotate 12	Rotate 11	Rotate 10	Rotate 9	Rotate 8	Rotate 7	Rotate 6	Rotate 5	Rotate 4	Rotate 3	Rotate 2	Rotate 1	Rotate 0
LP_GPIO_0												ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz
LP_GPIO_1	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz
LP_GPIO_2	TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz	
LP_GPIO_3		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz
LP_GPIO_4	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz
LP_GPIO_5	RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz			
LP_GPIO_6		RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz		
LP_GPIO_7			RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz	
LP_GPIO_8				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz	BT_6.5 MHz
LP_GPIO_9	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz												ARM 3.25 MHz
LP_GPIO_10	ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz											
LP_GPIO_11		ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz										
LP_GPIO_12			ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz									
LP_GPIO_13				ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz								
LP_GPIO_14					ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz							
LP_GPIO_15						ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz						
LP_GPIO_16							ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz					
LP_GPIO_17								ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz				
LP_GPIO_18									ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz			
LP_GPIO_19										ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz		
LP_GPIO_20											ARM 3.25 MHz	BT_6.5 MHz				RC_26_M Hz	RTC_32K Hz		TX_26_M Hz	RX_52_M Hz	

9.8.1 Generating Clock Output on LP_GPIO_x Pin

1. Select the desired clock and desired LP_GPIO_x to output the clock.
2. Get the test bus Shift Control register rotate value for desired clocks over desired LP_GPIO_x according to Test Bus Shift Control Value.
3. Write the test bus Shift Control register rotate value [ranges from 0 to 20] in bits [4:0] in the TEST_BUS_SHIFT_CTRL_0 register (0x40020250).
   Example: If rotate value = 13, write bits[4:0] to 0x0D.
4. MUX7 on the PINMUX_SEL_n register configures the specific LP_GPIO_x pin in the Test mode. To get clock output on specific LP_GPIO_x pin, configure PINMUX_SEL[2:0] of specific pin on the PINMUX_SEL_n register with value equal to 7.
5. Write the TEST_BUS_CONTROL register (0x400201A0) to 0x1.
6. If TX_26_MHz is selected for clock output, write the register 0x40024A08 to 0x107, to enable the TX path as this clock is only available when TX is active.

Refer the BluSDK Smart BLE API Software Development Guide for example code implementation on how to enable the 32.768 kHz clock output.

9.8.2 Example

The following is the sample example to get RTC_32KHz output on LP_GPIO_10:

```
/* Write the test bus shift register with rotate value 14 */
(*(volatile uint32_t *)(0x40020250)) = 14;
/* Write PINMUX_SEL_1 bits [10:8] with 0x7 */
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x7<<8);
/* Write TEST_BUS_CONTROL register to 0x01 */
(*(volatile uint32_t *)(0x400201A0)) = 1;
```

9.9 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B004	LPMCU_MISC_REGS0	LPMCU_GLOBAL_RESET_0	7:0	SPI1_IF_RSTN	SPI1_CORE_RSTN	SPI0_IF_RSTN	SPI0_CORE_RSTN	SPI_FLASH0_RSTN	SPI_FLASH0_SYS_RSTN	CPU_RSTN	GLOBAL_RSTN
			15:8	UART1_IF_RSTN	UART1_CORE_RSTN	UART0_IF_RSTN	UART0_CORE_RSTN	TIMER0_RSTN	GPIO_RSTN	I2C0_IF_RSTN	I2C0_CORE_RSTN
			23:16	ARM_FREE_CLK_RSTN	DBUG_RSTN	CALIB_XBAR_IF_RSTN	CALIB_RSTN	MBIST_RSTN	IRQ_CTRLR_CORE_RSTN	WDT1_RSTN	WDT0_RSTN
			31:24	PWM3_RSTN	PWM2_RSTN	PWM1_RSTN	PWM0_RSTN	QUAD_DEC2_RSTN	QUAD_DEC1_RSTN	QUAD_DEC0_RSTN	ARM_PRESETN_RSTN
0x4000B008	LPMCU_MISC_REGS0	LPMCU_GLOBAL_RESET_1	7:0	SPI0_SCK_CLK_RSTN	SECURITY_AES_AHB_RSTN	SECURITY_AES_CORE_RSTN	SECURITY_SHA_AHB_RSTN	SECURITY_SHA_CORE_RSTN	I2C1_IF_RSTN	I2C1_CORE_RSTN	DUALTIMER0_RSTN
			15:8					PROV_DMA_CTRL0_RSTN	SPI1_SCK_PHASE_INT_CLK_RSTN	SPI0_SCK_PHASE_INT_CLK_RSTN	SPI1_SCK_CLK_RSTN
0x4000F010	AON_GP_REGS0	AON_GLOBAL_RESET	7:0				PD4_RSTN	BLE_LP_RSTN		SLEEP_TIMER_RSTN	GLOBAL_RSTN
0x4000B00C	LPMCU_MISC_REGS0	LPMCU_CLOCK_ENABLES_0	7:0	GPIO_CLK_EN		DUALTIMER0_CLK_EN	I2C0_CORE_CLK_EN	SPI1_CORE_CLK_EN	SPI0_CORE_CLK_EN	SPI_FLASH0_CLK_EN	
			15:8	UART1_CORE_CLK_EN	UART0_IF_CLK_EN	UART0_CORE_CLK_EN	WDT1_CLK_EN	WDT0_CLK_EN			TIMER0_CLK_EN
			23:16	ARM_PCLK_EN	AON_WRAPPER_CLK_EN	CALIB_XBAR_IF_CLK_EN	ROM_MEM_CLK_EN	IDRAM_2_GL_MEM_CLK_EN	IDRAM_1_GL_MEM_CLK_EN	IRQ_CTRLR_CORE_CLK_EN	UART1_IF_CLK_EN
			31:24		CALIB_CLK_EN	I2C1_CORE_CLK_EN	QUAD_DEC2_CLK_EN	QUAD_DEC1_CLK_EN	QUAD_DEC0_CLK_EN	BLE_MEM_CLK_EN	ARM_PCLKG_EN

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B010	LPMCU_MISC_REGS0	LPMCU_CLOCK_ENABLES_1	7:0	PWM1_CLK_EN	PWM0_CLK_EN	EFUSE5_CLK_EN	EFUSE4_CLK_EN	EFUSE3_CLK_EN	EFUSE2_CLK_EN	EFUSE1_CLK_EN	EFUSE0_CLK_EN
			15:8	SHA_CORE_CLK_EN	TIMER0_PGCLK_EN	GPIO_GCLK_EN	SPI1_SCK_PHASE_INT_CLK_EN	SPI0_SCK_PHASE_INT_CLK_EN	SENS_ADC_CLK_EN	PWM3_CLK_EN	PWM2_CLK_EN
			23:16	IDRAM_2_1_MEM_CLK_EN	IDRAM_2_0_MEM_CLK_EN	IDRAM_1_2_MEM_CLK_EN	IDRAM_1_1_MEM_CLK_EN	IDRAM_1_0_MEM_CLK_EN	AES_AHB_CLK_EN	AES_CORE_CLK_EN	SHA_AHB_CLK_EN
0x4000F00C	AON_GP_REGS0	AON_MISC_CTRL	7:0		LPMCU_CPU_RESET_OVERRIDE_VAL	LPMCU_CPU_RESET_OVERRIDE_EN	LPMCU_USE_BOOT_REGS	LPMCU_BOOT_RESET_MUX_SEL	USE_EXT_32KHZ_CLK_SLEEP_TIMER	USE_RTC_32KHZ_CLK_SLEEP_TIMER	
			15:8	USE_OSC2M_AS_TB_CLK	USE_2M_AON_PWR_SEQ_CLK						
			23:16			FORCE_XO_TO_BYPASS_MODE	FORCE_OFF_XO	INVERT_WAKE_UP_GPIO_0	USE_RTC_AON_PWR_SEQ_CLK	AON_EXT_32KHZ_OUT_EN	AON_SLEEP_TIMER_CLK_EN
0x4000B018	LPMCU_MISC_REGS0	LPMCU_CTRL	7:0			DUALTIMER0_CLK_SEL[1:0]		USE_ARM_LP_CLK	USE_BT26M_CLK	LPMCU_CLK_SEL[1:0]	
			15:8			SPI_FLASH0_CLKSEL[1:0]				USE_XO_FOR_LP_CAL_CLK	BYPASS_WIC
			23:16	SPI_FLASH0_CLOCK_DIV_VALUE[3:0]				IDRAM_1_MEM_IQ_BYP_EN			SPI_FLASH0_DIV_CLKSEL
			31:24	EXT_SPI_MODE_CPOL	EXT_SPI_MODE_CPHA	INVERT_UART1_IF_CLK	INVERT_UART0_IF_CLK	SPI_FLASH0_CLOCK_DIV_VALUE[7:4]			

9.10 Register Description

9.10.1 LPMCU Global Reset 0

Name: LPMCU_GLOBAL_RESET_0
Reset: 0xFFFFFFFD

Absolute Address: 0x4000B004

This register is a part of LPMCU_MISC_REGS0 Registers. This register allows the user to reset the individual peripherals.

Bit	31	30	29	28	27	26	25	24
	PWM3_RSTN	PWM2_RSTN	PWM1_RSTN	PWM0_RSTN	QUAD_DEC2_RSTN	QUAD_DEC1_RSTN	QUAD_DEC0_RSTN	ARM_PRESETN_RSTN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	ARM_FREE_CLK_RSTN	DBUG_RSTN	CALIB_XBAR_IF_RSTN	CALIB_RSTN	MBIST_RSTN	IRQ_CTRLR_CORE_RSTN	WDT1_RSTN	WDT0_RSTN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	UART1_IF_RSTN	UART1_CORE_RSTN	UART0_IF_RSTN	UART0_CORE_RSTN	TIMER0_RSTN	GPIO_RSTN	I2C0_IF_RSTN	I2C0_CORE_RSTN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	SPI1_IF_RSTN	SPI1_CORE_RSTN	SPI0_IF_RSTN	SPI0_CORE_RSTN	SPI_FLASH0_RSTN	SPI_FLASH0_SYS_RSTN	CPU_RSTN	GLOBAL_RSTN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	0	1

Bit 31 – PWM3_RSTN PWM3 Peripheral Reset
Writing '0' to this bit resets PWM3 peripheral
Writing '1' to this bit allows normal PWM3 peripheral operations

Bit 30 – PWM2_RSTN PWM2 Peripheral Reset
Writing '0' to this bit resets PWM2 peripheral
Writing '1' to this bit allows normal PWM2 peripheral operations

Bit 29 – PWM1_RSTN PWM1 Peripheral Reset
Writing '0' to this bit resets PWM1 peripheral
Writing '1' to this bit allows normal PWM1 peripheral operations

Bit 28 – PWM0_RSTN PWM0 Peripheral Reset
Writing '0' to this bit resets PWM0 peripheral
Writing '1' to this bit allows normal PWM0 peripheral operations

Bit 27 – QUAD_DEC2_RSTN Quad Decoder 2 Peripheral Reset
Writing '0' to this bit resets Quad Decoder 2 peripheral
Writing '1' to this bit allows normal Quad Decoder 2 peripheral operations

Bit 26 – QUAD_DEC1_RSTN Quad Decoder 1 Peripheral Reset
Writing '0' to this bit resets Quad Decoder 1 peripheral
Writing '1' to this bit allows normal Quad Decoder 1 peripheral operations

Bit 25 – QUAD_DEC0_RSTN Quad Decoder 0 Peripheral Reset
Writing '0' to this bit resets Quad Decoder 0 peripheral
Writing '1' to this bit allows normal Quad Decoder 0 peripheral operations

Bit 24 – ARM_PRESETN_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 23 – ARM_FREE_CLK_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 22 – DBUG_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 21 – CALIB_XBAR_IF_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 20 – CALIB_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 19 – MBIST_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 18 – IRQ_CTRLR_CORE_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 17 – WDT1_RSTN
Writing '0' to this bit resets Watchdog 1 peripheral
Writing '1' to this bit allows normal Watchdog 1 peripheral operations

Bit 16 – WDT0_RSTN
Writing '0' to this bit resets Watchdog 0 peripheral
Writing '1' to this bit allows normal Watchdog 0 peripheral operations

Bit 15 – UART1_IF_RSTN
Writing '0' to this bit resets UART1 peripheral interface
Writing '1' to this bit allows normal UART1 peripheral interface operations

Bit 14 – UART1_CORE_RSTN
Writing '0' to this bit resets APB operation of UART1 peripheral core
Writing '1' to this bit allows normal UART1 peripheral core APB operations

Bit 13 – UART0_IF_RSTN
Writing '0' to this bit resets UART0 peripheral interface
Writing '1' to this bit allows normal UART0 peripheral interface operations

Bit 12 – UART0_CORE_RSTN
Writing '0' to this bit resets APB operation of UART0 peripheral core
Writing '1' to this bit allows normal UART0 peripheral core APB operations

Bit 11 – TIMER0_RSTN
Writing '0' to this bit resets Timer 0 peripheral
Writing '1' to this bit allows normal Timer 0 peripheral operations

Bit 10 – GPIO_RSTN
Writing '0' to this bit resets GPIO Controllers
Writing '1' to this bit allows normal GPIO Controllers operations

Bit 9 – I2C0_IF_RSTN
Writing '0' to this bit resets I2C0 peripheral interface
Writing '1' to this bit allows normal I2C0 peripheral interface operations

Bit 8 – I2C0_CORE_RSTN
Writing '0' to this bit resets APB operation of I2C0 peripheral core
Writing '1' to this bit allows normal I2C0 peripheral core APB operations

Bit 7 – SPI1_IF_RSTN
Writing '0' to this bit resets SPI1 peripheral interface
Writing '1' to this bit allows normal SPI1 peripheral interface operations

Bit 6 – SPI1_CORE_RSTN
Writing '0' to this bit resets APB operation of SPI1 peripheral core
Writing '1' to this bit allows normal SPI1 peripheral core APB operations

Bit 5 – SPI0_IF_RSTN
Writing '0' to this bit resets SPI0 peripheral interface
Writing '1' to this bit allows normal SPI0 peripheral interface operations

Bit 4 – SPI0_CORE_RSTN
Writing '0' to this bit resets APB operation of SPI0 peripheral core
Writing '1' to this bit allows normal SPI0 peripheral core APB operations

Bit 3 – SPI_FLASH0_RSTN
Writing '0' to this bit resets SPI Flash peripheral interface
Writing '1' to this bit allows normal SPI Flash peripheral interface operations

Bit 2 – SPI_FLASH0_SYS_RSTN
Writing '0' to this bit resets AHB operation of SPI Flash peripheral system
Writing '1' to this bit allows normal SPI Flash peripheral core AHB operations

Bit 1 – CPU_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

Bit 0 – GLOBAL_RSTN
This is an 'INTERNAL' bit. Controlled by ROM firmware and not recommended to change

9.10.2 LPMCU Global Reset 1

Name: LPMCU_GLOBAL_RESET_1
Reset: 0xFFF

Absolute Address: 0x4000B008

This register is a part of LPMCU_MISC_REGS0 Registers. This register allows the user to reset the individual peripherals.

Bit	15	14	13	12	11	10	9	8
					PROV_DMA_CTRL0_RSTN	SPI1_SCK_PHASE_INT_CLK_RSTN	SPI0_SCK_PHASE_INT_CLK_RSTN	SPI1_SCK_CLK_RSTN
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bit	7	6	5	4	3	2	1	0
	SPI0_SCK_CLK_RSTN	SECURITY_AES_AHB_RSTN	SECURITY_AES_CORE_RSTN	SECURITY_SHA_AHB_RSTN	SECURITY_SHA_CORE_RSTN	I2C1_IF_RSTN	I2C1_CORE_RSTN	DUALTIMER0_RSTN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 11 – PROV_DMA_CTRL0_RSTN
Writing '0' to this bit resets DMA Controller
Writing '1' to this bit allows normal DMA Controller operations

Bit 10 – SPI1_SCK_PHASE_INT_CLK_RSTN
Writing '0' to this bit resets logic that is on SPI1 SCK Phase int clock. Phase int clock is same clock as that of SPI Master SCK clock, that can get inverted depending on the phase setting for the SPI.
Writing '1' to this bit allows normal operations

Bit 9 – SPI0_SCK_PHASE_INT_CLK_RSTN
Writing '0' to this bit resets logic that is on SPI0 SCK Phase int clock. Phase int clock is same clock as that of SPI Master SCK clock, that can get inverted depending on the phase setting for the SPI.
Writing '1' to this bit allows normal operations

Bit 8 – SPI1_SCK_CLK_RSTN
Writing '0' to this bit resets logic that is on SPI1 Master clock
Writing '1' to this bit allows normal operations

Bit 7 – SPI0_SCK_CLK_RSTN
Writing '0' to this bit resets logic that is on SPI0 Master clock
Writing '1' to this bit allows normal operations

Bit 6 – SECURITY_AES_AHB_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 5 – SECURITY_AES_CORE_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 4 – SECURITY_SHA_AHB_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 3 – SECURITY_SHA_CORE_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 2 – I2C1_IF_RSTN
Writing '0' to this bit resets I2C1 peripheral interface
Writing '1' to this bit allows normal I2C1 peripheral interface operations

Bit 1 – I2C1_CORE_RSTN
Writing '0' to this bit resets I2C1 APB operations of peripheral core
Writing '1' to this bit allows normal I2C1 peripheral core APB operations

Bit 0 – DUALTIMER0_RSTN
Writing '0' to this bit resets Dual Timer peripheral
Writing '1' to this bit allows normal operations of Dual Timer

9.10.3 AON Global Reset

Name: AON_GLOBAL_RESET
Reset: 0x1B

Absolute Address: 0x4000F010

This register is a part of AON_GP_REGS0 Registers. This register allows the user to reset the individual Always-On power domain peripherals.

Bit	7	6	5	4	3	2	1	0
				PD4_RSTN	BLE_LP_RSTN		SLEEP_TIMER_RSTN	GLOBAL_RSTN
Access				R/W	R/W		R/W	R/W
Reset				1	1		1	1

Bit 4 – PD4_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 3 – BLE_LP_RSTN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 1 – SLEEP_TIMER_RSTN
Writing '0' to this bit resets AON Sleep Timer peripheral
Writing '1' to this bit allows normal AON Sleep Timer operations

Bit 0 – GLOBAL_RSTN
Writing '0' to this bit resets entire chip
This is an auto set bit, resets entire chip

9.10.4 LPMCU Clock Enable 0

Name: LPMCU_CLOCK_ENABLES_0
Reset: 0x 627FF9BE

Absolute Address: 0x4000B00C

This register is a part of LPMCU_MISC_REGS0 Registers. This register allows the user to enable clock for individual peripherals.

Bit	31	30	29	28	27	26	25	24
		CALIB_CLK_EN	I2C1_CORE_CLK_EN	QUAD_DEC2_CLK_EN	QUAD_DEC1_CLK_EN	QUAD_DEC0_CLK_EN	BLE_MEM_CLK_EN	ARM_PCLKG_EN
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		1	1	0	0	0	1	0

Bit	23	22	21	20	19	18	17	16
	ARM_PCLK_EN	AON_WRAPPER_CLK_EN	CALIB_XBAR_IF_CLK_EN	ROM_MEM_CLK_EN	IDRAM_2_GL_MEM_CLK_EN	IDRAM_1_GL_MEM_CLK_EN	IRQ_CTRLR_CORE_CLK_EN	UART1_IF_CLK_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	UART1_CORE_CLK_EN	UART0_IF_CLK_EN	UART0_CORE_CLK_EN	WDT1_CLK_EN	WDT0_CLK_EN			TIMER0_CLK_EN
Access	R/W	R/W	R/W	R/W	R/W			R/W
Reset	1	1	1	1	1			1

Bit	7	6	5	4	3	2	1	0
	GPIO_CLK_EN		DUALTIMER0_CLK_EN	I2C0_CORE_CLK_EN	SPI1_CORE_CLK_EN	SPI0_CORE_CLK_EN	SPI_FLASH0_CLK_EN	
Access	R/W		R/W	R/W	R/W	R/W	R/W	
Reset	1		1	1	1	1	1	

Bit 30 – CALIB_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 29 – I2C1_CORE_CLK_EN
Writing '0' to this bit disables clock to I2C1 core APB interface
Writing '1' to this bit enables clock to I2C1 core APB interface

Bit 28 – QUAD_DEC2_CLK_EN
Writing '0' to this bit disables clock to Quad Decoder2
Writing '1' to this bit enables clock to Quad Decoder2

Bit 27 – QUAD_DEC1_CLK_EN
Writing '0' to this bit disables clock to Quad Decoder1
Writing '1' to this bit enables clock to Quad Decoder1

Bit 26 – QUAD_DEC0_CLK_EN
Writing '0' to this bit disables clock to Quad Decoder0

Writing '1' to this bit enables clock to Quad Decoder0

Bit 25 – BLE_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 24 – ARM_PCLKG_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 23 – ARM_PCLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 22 – AON_WRAPPER_CLK_EN
This is an INTERNAL bit and not recommended to change

Bit 21 – CALIB_XBAR_IF_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 20 – ROM_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 19 – IDRAM_2_GL_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 18 – IDRAM_1_GL_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 17 – IRQ_CTRLR_CORE_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 16 – UART1_IF_CLK_EN
Writing '0' to this bit disables clock to UART1 interface
Writing '1' to this bit enables clock to UART1 interface

Bit 15 – UART1_CORE_CLK_EN
Writing '0' to this bit disables clock to UART1 core APB interface
Writing '1' to this bit enables clock to UART1 core APB interface

Bit 14 – UART0_IF_CLK_EN
Writing '0' to this bit disables clock to UART0 interface
Writing '1' to this bit enables clock to UART0 interface

Bit 13 – UART0_CORE_CLK_EN
Writing '0' to this bit disables clock to UART0 core APB interface
Writing '1' to this bit enables clock to UART0 core APB interface

Bit 12 – WDT1_CLK_EN
Writing '0' to this bit disables clock to Watchdog Timer1
Writing '1' to this bit enables clock to Watchdog Timer1

Bit 11 – WDT0_CLK_EN
Writing '0' to this bit disables clock to Watchdog Timer0

Writing '1' to this bit enables clock to Watchdog Timer0

Bit 8 – TIMER0_CLK_EN
Writing '0' to this bit disables clock to Timer0
Writing '1' to this bit enables clock to Timer0

Bit 7 – GPIO_CLK_EN
Writing '0' to this bit disables clock to GPIO controllers
Writing '1' to this bit enables clock to GPIO controllers

Bit 5 – DUALTIMER0_CLK_EN
Writing '0' to this bit disables clock to Dual Timer
Writing '1' to this bit enables clock to Dual Timer

Bit 4 – I2C0_CORE_CLK_EN
Writing '0' to this bit disables clock to I2C0 core APB interface
Writing '1' to this bit enables clock to I2C0 core APB interface

Bit 3 – SPI1_CORE_CLK_EN
Writing '0' to this bit disables clock to SPI1 core APB interface
Writing '1' to this bit enables clock to SPI1 core APB interface

Bit 2 – SPI0_CORE_CLK_EN
Writing '0' to this bit disables clock to SPI0 core APB interface
Writing '1' to this bit enables clock to SPI0 core APB interface

Bit 1 – SPI_FLASH0_CLK_EN
Writing '0' to this bit disables clock to SPI Flash0
Writing '1' to this bit enables clock to SPI Flash0

9.10.5 LPMCU Clock Enable 1

Name: LPMCU_CLOCK_ENABLES_1
Reset: 0xF8783F

Absolute Address: 0x4000B010

This register is a part of LPMCU_MISC_REGS0 Registers. This register allows the user to enable clock for individual peripherals.

Bit	23	22	21	20	19	18	17	16
	IDRAM_2_1_MEM_CLK_EN	IDRAM_2_0_MEM_CLK_EN	IDRAM_1_2_MEM_CLK_EN	IDRAM_1_1_MEM_CLK_EN	IDRAM_1_0_MEM_CLK_EN	AES_AHB_CLK_EN	AES_CORE_CLK_EN	SHA_AHB_CLK_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	0	0	0

Bit	15	14	13	12	11	10	9	8
	SHA_CORE_CLK_EN	TIMER0_PGCLK_EN	GPIO_GCLK_EN	SPI1_SCK_PHASE_INT_CLK_EN	SPI0_SCK_PHASE_INT_CLK_EN	SENS_ADC_CLK_EN	PWM3_CLK_EN	PWM2_CLK_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	0	0	0

Bit	7	6	5	4	3	2	1	0
	PWM1_CLK_EN	PWM0_CLK_EN	EFUSE5_CLK_EN	EFUSE4_CLK_EN	EFUSE3_CLK_EN	EFUSE2_CLK_EN	EFUSE1_CLK_EN	EFUSE0_CLK_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	1	1	1	1

Bit 23 – IDRAM_2_1_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 22 – IDRAM_2_0_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 21 – IDRAM_1_2_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 20 – IDRAM_1_1_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 19 – IDRAM_1_0_MEM_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 18 – AES_AHB_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 17 – AES_CORE_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 16 – SHA_AHB_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 15 – SHA_CORE_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 14 – TIMER0_PGCLK_EN
Writing '0' to this bit disables clock to Timer0 APB interface
Writing '1' to this bit enables clock to Timer0 APB interface

Bit 13 – GPIO_GCLK_EN
Writing '0' to this bit disables clock to GPIO AHB interface
Writing '0' to this bit enables clock to GPIO AHB interface

Bit 12 – SPI1_SCK_PHASE_INT_CLK_EN
Writing '0' to this bit disables SPI1 SCK Phase int clock. Phase int clock is same clock as that of SPI Master SCK clock, that can get inverted depending on the phase setting for the SPI.
Writing '1' to this bit enables SPI1 SCK Phase int clock

Bit 11 – SPI0_SCK_PHASE_INT_CLK_EN
Writing '0' to this bit disables SPI0 SCK Phase int clock. Phase int clock is same clock as that of SPI Master SCK clock, that can get inverted depending on the phase setting for the SPI.
Writing '1' to this bit enables SPI0 SCK Phase int clock

Bit 10 – SENS_ADC_CLK_EN
Writing '0' to this bit disables ADC peripheral clock
Writing '1' to this bit enables ADC peripheral clock

Bit 9 – PWM3_CLK_EN
Writing '0' to this bit disables PWM3 peripheral clock
Writing '1' to this bit enables PWM3 peripheral clock

Bit 8 – PWM2_CLK_EN
Writing '0' to this bit disables PWM2 peripheral clock
Writing '1' to this bit enables PWM2 peripheral clock

Bit 7 – PWM1_CLK_EN
Writing '0' to this bit disables PWM1 peripheral clock
Writing '1' to this bit enables PWM1 peripheral clock

Bit 6 – PWM0_CLK_EN
Writing '0' to this bit disables PWM0 peripheral clock
Writing '1' to this bit enables PWM0 peripheral clock

Bit 5 – EFUSE5_CLK_EN
Writing '0' to this bit disables EFUSE Bank 6 clock
Writing '1' to this bit enables EFUSE Bank6 clock

Bit 4 – EFUSE4_CLK_EN
Writing '0' to this bit disables EFUSE Bank 5 clock

Writing '1' to this bit enables EFUSE Bank 5 clock

Bit 3 – EFUSE3_CLK_EN
Writing '0' to this bit disables EFUSE Bank 4 clock
Writing '1' to this bit enables EFUSE Bank 4 clock

Bit 2 – EFUSE2_CLK_EN
Writing '0' to this bit disables EFUSE Bank 3 clock
Writing '1' to this bit enables EFUSE Bank 3 clock

Bit 1 – EFUSE1_CLK_EN
Writing '0' to this bit disables EFUSE Bank 2 clock
Writing '1' to this bit enables EFUSE Bank 2 clock

Bit 0 – EFUSE0_CLK_EN
Writing '0' to this bit disables EFUSE Bank 1 clock
Writing '1' to this bit enables EFUSE Bank 1 clock

9.10.6 AON Clock Enable

Name: AON_MISC_CTRL
Reset: 0x010000

Absolute Address: 0x4000F00C

This register is a part of AON_GP_REGS0 Registers. This register allows the user to enable clock for AON power domain peripherals.

Bit	23	22	21	20	19	18	17	16
			FORCE_XO_TO_BYPASS_MODE	FORCE_OFF_XO	INVERT_WAKEUP_GPIO_0	USE_RTC_AON_PWR_SEQ_CLK	AON_EXT_32KHZ_OUT_EN	AON_SLEEP_TIMER_CLK_EN
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	USE_OSC2M_AS_TB_CLK	USE_2M_AON_PWR_SEQ_CLK						
Access	R/W	R/W						
Reset	0	0						

Bit	7	6	5	4	3	2	1	0
		LPMCU_CPU_RESET_OVERRIDE_VAL	LPMCU_CPU_RESET_OVERRIDE_EN	LPMCU_USE_BOOT_REGS	LPMCU_BOOT_RESET_MUX_SEL	USE_EXT_32KHZ_CLK_SLEEP_TIMER	USE_RTC_32KHZ_CLK_SLEEP_TIMER	
Access		R/W	R/W	R/W	R/W	R/W	R/W	
Reset		0	0	0	0	0	0	

Bit 21 – FORCE_XO_TO_BYPASS_MODE
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 20 – FORCE_OFF_XO
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 19 – INVERT_WAKEUP_GPIO_0
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 18 – USE_RTC_AON_PWR_SEQ_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 17 – AON_EXT_32KHZ_OUT_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 16 – AON_SLEEP_TIMER_CLK_EN
Writing '0' to this bit disables AON Sleep Timer peripheral clock
Writing '1' to this bit enables AON Sleep Timer peripheral clock

Bit 15 – USE_OSC2M_AS_TB_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 14 – USE_2M_AON_PWR_SEQ_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 6 – LPMCU_CPU_RESET_OVERRIDE_VAL
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 5 – LPMCU_CPU_RESET_OVERRIDE_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 4 – LPMCU_USE_BOOT_REGS
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 3 – LPMCU_BOOT_RESET_MUX_SEL
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 2 – USE_EXT_32KHZ_CLK_SLEEP_TIMER
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 1 – USE_RTC_32KHZ_CLK_SLEEP_TIMER
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

9.10.7 LPMCU Control Register

Name: LPMCU_CTRL
Reset: 0x00303100

Absolute Address: 0x4000B018

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to select different clock source for different peripheral modules.

Bit	31	30	29	28	27	26	25	24
	EXT_SPI_MODE_CPOL	EXT_SPI_MODE_CPHA	INVERT_UART1_IF_CLK	INVERT_UART0_IF_CLK	SPI_FLASH0_CLOCK_DIV_VALUE[7:4]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SPI_FLASH0_CLOCK_DIV_VALUE[3:0]				IDRAM_1_MEM_IQ_BYP_EN			SPI_FLASH0_DIV_CLKSEL
Access	R/W	R/W	R/W	R/W	R/W			R/W
Reset	0	0	1	1	0			0

Bit	15	14	13	12	11	10	9	8
			SPI_FLASH0_CLKSEL[1:0]				USE_XO_FOR_LP_CAL_CLK	BYPASS_WIC
Access			R/W	R/W			R/W	R/W
Reset			1	1			0	1

Bit	7	6	5	4	3	2	1	0
			DUALTIMER0_CLK_SEL[1:0]		USE_ARM_LP_CLK	USE_BT26M_CLK	LPMCU_CLK_SEL[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 31 – EXT_SPI_MODE_CPOL
Writing '1' to this bit enables the external SPI mode CPOL. This bit is not recommended to change.

Bit 30 – EXT_SPI_MODE_CPHA
Writing '1' to this bit enables the external SPI mode CPHA. This bit is not recommended to change.

Bit 29 – INVERT_UART1_IF_CLK
Writing '1' to this bit inverts UART1 interface clock

Bit 28 – INVERT_UART0_IF_CLK
Writing '1' to this bit inverts UART0 interface clock

Bits 27:20 – SPI_FLASH0_CLOCK_DIV_VALUE[7:0]
This register divides the SPI Flash0 input clock with the divide value when SPI_FLASH0_DIV_CLKSEL bit is set. This uses 26 MHz clock as input clock.
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change.

Bit 16 – SPI_FLASH0_DIV_CLKSEL
Writing '1' to this bit selects the divided clock for SPI Flash0

Bits 13:12 – SPI_FLASH0_CLKSEL[1:0]
These bits select clock input for SPI Flash0

SPI_FLASH0_CLKSEL[1:0]	Description
0x0	3.25 MHz
0x1	6.5 MHz
0x2	13 MHz
0x3	26 MHz

Bit 9 – USE_XO_FOR_LP_CAL_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 8 – BYPASS_WIC
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bits 5:4 – DUALTIMER0_CLK_SEL[1:0]
These bits select clock input for the dual timer. This clock is common for both TIMER1 and TIMER2.

DUALTIMER0_CLK_SEL[1:0]	Description
0x0	26 MHz
0x1	13 MHz
0x2	6.5 MHz
0x3	3.25 MHz

Bit 3 – USE_ARM_LP_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 2 – USE_BT26M_CLK
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bits 1:0 – LPMCU_CLK_SEL[1:0]
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

10. I/O Peripheral Multiplexing and MEGAMUXing

This chapter describes the peripheral multiplexing and MEGAMUXing options of the I/O pins.

10.1 I/O Multiplexing

By default, each pin is controlled by the GPIO controller as a general purpose I/O. Alternatively, it can be assigned to one of the peripheral functions. The I/O pins are categorized into three different groups called LP_GPIO_x (Low Power), AO_GPIO_z (Always ON) and GPIO_MSy (Mixed Signal). To enable a specific peripheral function on a LP_GPIO_x pin, the PINMUX_SEL_n, (n=0,1..4) register corresponding to that pin must be written with the specific MUX value. The specific functionality on AO_GPIO_z pin is selected by configuring MUX value in the AON_PINMUX_SEL register. Only, MUX0 is possible for GPIO_MSy to configure either as digital or analog I/O using the MS_GPIO_MODE register.

Table 10-1. I/O Port Function Multiplexing

Pin Name	XR Pin No.	ZR Pin No.	GPIO Controller	Pull	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
LP_GPIO_0	35	12	GPIO0_00	Up/ Down	GPIO 0	MEGAMUX 0	SWD CLK	-	-	-	-	TEST OUT 0
LP_GPIO_1	36	13	GPIO0_01	Up/ Down	GPIO 1	MEGAMUX 1	SWD I/O	-	-	-	-	TEST OUT 1
LP_GPIO_2	37	14	GPIO0_02	Up/ Down	GPIO 2	MEGAMUX 2	UART0 RXD	-	SPI1 SCK	SPI0 SCK	SPI FLASH0 SCK	TEST OUT 2
LP_GPIO_3	38	15	GPIO0_03	Up/ Down	GPIO 3	MEGAMUX 3	UART0 TXD	-	SPI1 MOSI	SPI0 MOSI	SPI FLASH0 TXD	TEST OUT 3
LP_GPIO_4	39	16	GPIO0_04	Up/ Down	GPIO 4	MEGAMUX 4	UART0 CTS	-	SPI1 SSN	SPI0 SSN	SPI FLASH0 SSN	TEST OUT 4
LP_GPIO_5	2	17	GPIO0_05	Up/ Down	GPIO 5	MEGAMUX 5	UART0 RTS		SPI1 MISO	SPI0 MISO	SPI FLASH0 RXD	TEST OUT 5
LP_GPIO_6	3	18	GPIO0_06	Up/ Down	GPIO 6	MEGAMUX 6	UART1 RXD	-	-	SPI0 SCK	SPI FLASH0 SCK	TEST OUT 6
LP_GPIO_7	4	19	GPIO0_07	Up/ Down	GPIO 7	MEGAMUX 7	UART1 TXD	-	-	SPI0 MOSI	SPI FLASH0 TXD	TEST OUT 7
LP_GPIO_8	5	20	GPIO0_08	Up/ Down	GPIO 8	MEGAMUX 8	I^2C0 SDA	-	-	SPI0 SSN	SPI FLASH0 SSN	TEST OUT 8
LP_GPIO_9	6	21	GPIO0_09	Up/ Down	GPIO 9	MEGAMUX 9	I^2C0 SCL	-	-	SPI0 MISO	SPI FLASH0 RXD	TEST OUT 9
LP_GPIO_10	7	22	GPIO0_10	Up/ Down	GPIO 10	MEGAMUX 10	SPI0 SCK	-	-	-	SPI FLASH0 SCK	TEST OUT 10
LP_GPIO_11	8	23	GPIO0_11	Up/ Down	GPIO 11	MEGAMUX 11	SPI0 MOSI	-	-	-	SPI FLASH0 TXD	TEST OUT 11

.....continued

Pin Name	XR Pin No.	ZR Pin No.	GPIO Controller	Pull	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
LP_GPIO_12	9	24	GPIO0_12	Up/ Down	GPIO 12	MEGAMUX 12	SPI0 SSN	-	-	-	SPI FLASH0 SSN	TEST OUT 12
LP_GPIO_13	10	25	GPIO0_13	Up/ Down	GPIO 13	MEGAMUX 13	SPI0 MISO	-	-	-	SPI FLASH0 RXD	TEST OUT 13
LP_GPIO_14	23	4	GPIO0_14	Up/ Down	GPIO 14	MEGAMUX 14	UART1 CTS	-	I^2C1 SDA	-	-	TEST OUT 14
LP_GPIO_15	24	5	GPIO0_15	Up/ Down	GPIO 15	MEGAMUX 15	UART1 RTS	-	I^2C1 SCL	-	-	TEST OUT 15
LP_GPIO_16	25	6	GPIO1_00	Up/ Down	GPIO 16	MEGAMUX 16	SPI FLASH0 SCK	-	SPI1 SSN	SPI0 SCK	SPI FLASH0 SSN	TEST OUT 16
LP_GPIO_17	28	8	GPIO1_01	Up/ Down	GPIO 17	MEGAMUX 17	SPI FLASH0 TXD	-	SPI1 SCK	SPI0 MOSI	-	TEST OUT 17
LP_GPIO_18	29	9	GPIO1_02	Up/ Down	GPIO 18	MEGAMUX 18	SPI FLASH0 SSN	-	SPI1 MISO	SPI0 SSN	SPI FLASH0 RXD	TEST OUT 18
LP_GPIO_19	30	10	GPIO1_03	Up/ Down	GPIO 19	MEGAMUX 19	SPI FLASH0 RXD	-	SPI1 MOSI	SPI0 MISO	-	TEST OUT 19
LP_GPIO_22	40		GPIO1_06	Up/ Down	GPIO 22	MEGAMUX 22	-	-	-	-	-	-
LP_GPIO_23	1		GPIO1_07	Up/ Down	GPIO 23	MEGAMUX 23	-	-	-	-	-	-
LP_GPIO_24	25		GPIO1_08	Up/ Down	GPIO 24	MEGAMUX 24	-	-	-	-	-	-
AO_GPIO_0	20	1	GPIO1_15	Up	GPIO 31	WAKEUP	RTC CLK IN	32kHZ CLK OUT	-	-	-	-
AO_GPIO_1	21	2	GPIO1_14	Up	GPIO 30	WAKEUP	RTC CLK IN	32kHZ CLK OUT	-	-	-	-
AO_GPIO_2	22	3	GPIO1_13	Up	GPIO 29	WAKEUP	RTC CLK IN	32kHZ CLK OUT	-	-	-	-
GPIO_MS1 (1)	12	17	GPIO2_15	Up/ Down	GPIO 47	-	-	-	-	-	-	-
GPIO_MS2 (1)	13	18	GPIO2_14	Up/ Down	GPIO 46	-	-	-	-	-	-	-
GPIO_MS3 (1)	15	31	GPIO2_13	Up/ Down	GPIO 45	-	-	-	-	-	-	-
GPIO_MS4 (1)	16	32	GPIO2_12	Up/ Down	GPIO 44	-	-	-	-	-	-	-

Note:

1. If analog is selected, then the digital is disabled.
2. MUX2 is the default MUX value for LP_GPIO_0 and LP_GPIO_1 to function as Single Wire Debug (SWD) interface. It is not recommended to use these pins for other peripheral functionality, as these pins are used for programming/debugging.

10.1.1 Example

An example to illustrate the available options for LP_GPIO_3 pin, depending on the PINMUX_SEL [2:0] LP_GPIO_3 value selected on [14:12] bits of PINMUX_SEL_0 register:

- MUX0 – the pin functions general purpose I/O and is controlled by the GPIO controller.
- MUX1 – any option from the Table 10-2 is selected using MEGA_MUX_IO_SEL_n (n=0,1,2..6) register. It can be a quad_dec, pwm, or any of the other functions listed in the Table 10-2.
- MUX2 – the pin functions as UART1 TXD. This is achieved with the MUX1 option via MEGAMUX, but the MUX2 option allows a shortcut for the recommended pinout.
- MUX3 – this option is not used and thus defaults to MUX0.
- MUX4 – the pin functions as SPI1 MOSI.
- MUX5 – the pin functions as SPI0 MOSI.
- MUX6 – the pin functions as SPI FLASH0 SCK.
- MUX7 – the pin functions as bit 3 of the test output bus, giving access to various debug signals.

10.2 MEGAMUXing

In addition to peripheral multiplexing, the MEGAMUXing option allows more flexibility for mapping desired interfaces on I/O pins. The MUX1 option in Table 10-1 allows for any MEGAMUX option from Table 10-2 to be assigned to an I/O pin. When PINMUX_SEL_n (n=0 to 4) is assigned with MUX1 then the MEGA_MUX_IO_SEL_n (n=0 to 6) register is used to configure MEGAMUX for LP_GPIO_x I/O pins. MEGAMUX configuration is not possible for GPIO_MSy and AO_GPIO_z I/O pins.

The use case of the MEGAMUX option is that when a specific peripheral functionality is not available on an intended LP_GPIO_x pin through MUX2 to MUX6 configurations, the MEGAMUX option allows that functionality.

Table 10-2. MEGAMUX Options

MUX_Sel	Function
0x00	UART0 RXD
0x01	UART0 TXD
0x02	UART0 CTS
0x03	UART0 RTS
0x04	UART1 RXD
0x05	UART1 TXD
0x06	UART1 CTS
0x07	UART1 RTS

.....continued

MUX_Sel	Function
0x08	I^2C0 SDA
0x09	I^2C0 SCL
0x0A	I^2C1 SDA
0x0B	I^2C1 SCL
0x0C	PWM 0
0x0D	PWM 1
0x0E	PWM 2
0x0F	PWM 3
0x10	LP CLOCK OUT
0x11	Reserved
0x12	Reserved
0x13	Reserved
0x14	Reserved
0x15	Reserved
0x16	Reserved
0x17	Reserved
0x18	Reserved
0x19	Reserved
0x1A	Reserved
0x1B	Reserved
0x1C	Reserved
0x1D	QUAD DEC X IN A
0x1E	QUAD DEC X IN B
0x1F	QUAD DEC Y IN A
0x20	QUAD DEC Y IN B
0x21	QUAD DEC Z IN A
0x22	QUAD DEC Z IN B

10.2.1 Example

An example of peripheral assignment using these MEGAMUX options is as follows:

- I²C0 PINMUXed on LP_GPIO_10 and LP_GPIO_11 via
  - PINMUX_SEL_1 register PINMUX_SEL[2:0] LP_GPIO_10 = 1 PINMUX_SEL[2:0] LP_GPIO_11 = 1.
  - MEGA_MUX_IO_SEL_2 register MEGAMUX_SEL[5:0] LP_GPIO_10 = 0x08 and MEGAMUX_SEL[5:0] LP_GPIO_11 = 0x09.
- I²C1 PINMUXed on LP_GPIO_0 and LP_GPIO_1 via
  - PINMUX_SEL_0 register PINMUX_SEL[2:0] LP_GPIO_0 = 1 PINMUX_SEL[2:0] LP_GPIO_1 = 1.
  - MEGA_MUX_IO_SEL_1 register MEGAMUX_SEL[5:0] LP_GPIO_0 = 0x0A and MEGAMUX_SEL[5:0] LP_GPIO_1 = 0x0B.

10.3 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B068	LPMCU_MISC_REGS0	PINMUX_SEL_3	7:0		PINMUX_SEL[2:0] LP_SIP_1				PINMUX_SEL[2:0] LP_SIP_0		
			15:8		PINMUX_SEL[2:0] LP_SIP_3				PINMUX_SEL[2:0] LP_SIP_2		
			23:16						PINMUX_SEL[2:0] LP_SIP_4		
			31:24								
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0		MEGAMUX_SEL[5:0] LP_GPIO_0						
			15:8		MEGAMUX_SEL[5:0] LP_GPIO_1						
			23:16		MEGAMUX_SEL[5:0] LP_GPIO_2						
			31:24		MEGAMUX_SEL[5:0] LP_GPIO_3						
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0		MEGAMUX_SEL[5:0] LP_GPIO_4						
			15:8		MEGAMUX_SEL[5:0] LP_GPIO_5						
			23:16		MEGAMUX_SEL[5:0] LP_GPIO_6						
			31:24		MEGAMUX_SEL[5:0] LP_GPIO_7						

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0			MEGAMUX_SEL[5:0] LP_GPIO_8					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_9					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_10					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_11					
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0			MEGAMUX_SEL[5:0] LP_GPIO_12					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_13					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_14					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_15					
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0			MEGAMUX_SEL[5:0] LP_GPIO_16					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_17					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_18					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_19					
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0			MEGAMUX_SEL[5:0] LP_GPIO_20					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_21					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_22					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_23					
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0			MEGAMUX_SEL[5:0] LP_GPIO_24					
0x4000F000	AON_GP_REGS0	AON_PINMUX_SEL	7:0			PINMUX_SEL[1:0] AO_GPIO_1				PINMUX_SEL[1:0] AO_GPIO_0	
			15:8							PINMUX_SEL[1:0] AO_GPIO_2	
0x4000F410	AON_GP_REGS0	MS_GPIO_MODE	7:0				ANALOG_EN GPIO_MS3	ANALOG_EN GPIO_MS2	ANALOG_EN GPIO_MS1	ANALOG_EN GPIO_MS0	

10.4 Register Description

10.4.1 LP_GPIO_x Peripheral Multiplexing 0

Name: PINMUX_SEL_0
Reset: 0x00000022

Absolute Address: 0x4000B044

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_0, LP_GPIO_1, LP_GPIO_2, LP_GPIO_3, LP_GPIO_4, LP_GPIO_5, LP_GPIO_6, LP_GPIO_7 pins as listed in Table 10-1.

Bit	31	30	29	28	27	26	25	24
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	1	0		0	1	0

Bits 30:28, 26:24, 22:20, 18:16, 14:12, 10:8, 6:4, 2:0 – PINMUX_SEL[2:0] LP_GPIO_x (x = 7 to 0) Pin mux configuration
These bits select peripheral function for LP_GPIO_x (x = 7 to 0; Replace x=7 for 30:28 bits, ... x=0 for 2:1 bits)

PINMUX_SEL[2:0]	Description
0x0	MUX0 peripheral function is selected
0x1	MUX1 (MEGAMUX) peripheral function is selected
0x2	MUX2 peripheral function is selected
0x3	MUX3 peripheral function is selected
0x4	MUX4 peripheral function is selected
0x5	MUX5 peripheral function is selected
0x6	MUX6 peripheral function is selected
0x7	MUX7 peripheral function is selected

10.4.2 LP_GPIO_x Peripheral Multiplexing 1

Name: PINMUX_SEL_1
Reset: 0x00000033

Absolute Address: 0x4000B048

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_8, LP_GPIO_9, LP_GPIO_10, LP_GPIO_11, LP_GPIO_12, LP_GPIO_13, LP_GPIO_14, LP_GPIO_15 pins as listed in Table 10-1.

Bit	31	30	29	28	27	26	25	24
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	1	1		0	1	1

Bits 30:28, 26:24, 22:20, 18:16, 14:12, 10:8, 6:4, 2:0 – PINMUX_SEL[2:0] LP_GPIO_x (x = 15 to 8)
Pin mux configuration
These bits select peripheral function for LP_GPIO_x (x = 15 to 8; Replace x=15 for 30:28 bits, ... x=8 for 2:1 bits)

PINMUX_SEL[2:0]	Description
0x0	MUX0 peripheral function is selected
0x1	MUX1 (MEGAMUX) peripheral function is selected
0x2	MUX2 peripheral function is selected
0x3	MUX3 peripheral function is selected
0x4	MUX4 peripheral function is selected
0x5	MUX5 peripheral function is selected
0x6	MUX6 peripheral function is selected
0x7	MUX7 peripheral function is selected

10.4.3 LP_GPIO_x Peripheral Multiplexing 2

Name: PINMUX_SEL_2
Reset: 0x00000000

Absolute Address: 0x4000B04C

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_16, LP_GPIO_17, LP_GPIO_18, LP_GPIO_19, LP_GPIO_20, LP_GPIO_21, LP_GPIO_22, LP_GPIO_23 pins as listed in Table 10-1.

Bit	31	30	29	28	27	26	25	24
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		PINMUX_SEL[2:0] LP_GPIO_x				PINMUX_SEL[2:0] LP_GPIO_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 30:28, 26:24, 22:20, 18:16, 14:12, 10:8, 6:4, 2:0 – PINMUX_SEL[2:0] LP_GPIO_x (x = 23 to 16)
Pin mux configuration
These bits select peripheral function for LP_GPIO_x (x = 23 to 16; Replace x=23 for 30:28 bits, ... x=16 for 2:1 bits)

PINMUX_SEL[2:0]	Description
0x0	MUX0 peripheral function is selected
0x1	MUX1 (MEGAMUX) peripheral function is selected
0x2	MUX2 peripheral function is selected
0x3	MUX3 peripheral function is selected
0x4	MUX4 peripheral function is selected
0x5	MUX5 peripheral function is selected
0x6	MUX6 peripheral function is selected
0x7	MUX7 peripheral function is selected

10.4.4 LP_GPIO_x Peripheral Multiplexing 3

Name: PINMUX_SEL_3
Reset: 0x00000000

Absolute Address: 0x4000B068

This register is a part of the LPMCU_MISC_REGS0 registers. This register allows the user to enable a SPI Flash controller peripheral function on LP_SIP_0, LP_SIP_1, LP_SIP_2, LP_SIP_3, and LP_SIP_4 pins. LP_SIP_x pins are connected internally with stacked Flash within SiP.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						PINMUX_SEL[2:0] LP_SIP_x		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
		PINMUX_SEL[2:0] LP_SIP_x				PINMUX_SEL[2:0] LP_SIP_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		PINMUX_SEL[2:0] LP_SIP_x				PINMUX_SEL[2:0] LP_SIP_x		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 18:16, 14:12, 10:8, 6:4, 2:0 – PINMUX_SEL[2:0] LP_SIP_x (x = 4 to 0, Replace x=4 for 18:16 bits, … x=0 for 2:0 bits) Pin mux configuration
These bits select peripheral function for LP_SIP_x (x = 4 to 0). LP_SIP_x pins are connected internally to SPI Flash. Hence, it is not recommended to use this as another peripheral or GPIO. It is recommended to use only the PINMUX_SEL[2:0] values listed in the following table.

LP_SIP_x pin	SPI Flash Pin Name	PINMUX[2:0]
LP_SIP_0	SPI_FLASH_SCK	1
LP_SIP_1	SPI_FLASH_TXD	3
LP_SIP_2	SPI_FLASH_SSN	2
LP_SIP_3	SPI_FLASH_RXD	4
LP_SIP_4	SPI_FLASH_WP	0

10.4.5 LP_GPIO_x Peripheral Multiplexing 4

Name: PINMUX_SEL_4
Reset: 0x00000000

Absolute Address: 0x4000B080

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_24 pin as listed in Table 10-1.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						PINMUX_SEL[2:0] LP_GPIO_x		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – PINMUX_SEL[2:0] LP_GPIO_x (x = 24) Pin mux configuration
These bits select peripheral function for LP_GPIO_x (x = 24).

PINMUX_SEL[2:0]	Description
0x0	MUX0 peripheral function is selected
0x1	MUX1 (MEGAMUX) peripheral function is selected
0x2	MUX2 peripheral function is selected
0x3	MUX3 peripheral function is selected
0x4	MUX4 peripheral function is selected
0x5	MUX5 peripheral function is selected
0x6	MUX6 peripheral function is selected
0x7	MUX7 peripheral function is selected

10.4.6 LP_GPIO_x Mega Multiplexing 0

Name: MEGA_MUX_IO_SEL_0
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1A0

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_0, LP_GPIO_1, LP_GPIO_2, LP_GPIO_3 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 3 to 0; Replace x=3 for 29:24 bits, ... x=0 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.7 LP_GPIO_x Mega Multiplexing 1

Name: MEGA_MUX_IO_SEL_1
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1A4

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_4, LP_GPIO_5, LP_GPIO_6, LP_GPIO_7 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 7 to 4; Replace x=7 for 29:24 bits, ... x=4 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.8 LP_GPIO_x Mega Multiplexing 2

Name: MEGA_MUX_IO_SEL_2
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1A8

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_8, LP_GPIO_9, LP_GPIO_10, LP_GPIO_11 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 11 to 8; Replace x=11 for 29:24 bits, ... x=8 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.9 LP_GPIO_x Mega Multiplexing 3

Name: MEGA_MUX_IO_SEL_3
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1AC

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_12, LP_GPIO_13, LP_GPIO_14, LP_GPIO_15 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 15 to 12; Replace x=15 for 29:24 bits, ... x=12 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.10 LP_GPIO_x Mega Multiplexing 4

Name: MEGA_MUX_IO_SEL_4
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1B0

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_16, LP_GPIO_17, LP_GPIO_18, LP_GPIO_19 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 19 to 16; Replace x=19 for 29:24 bits, ... x=16 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.11 LP_GPIO_x Mega Multiplexing 5

Name: MEGA_MUX_IO_SEL_5
Reset: 0x3F3F3F3F

Absolute Address: 0x4000B1B4

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_20, LP_GPIO_21, LP_GPIO_22, LP_GPIO_23 pins using MEGAMUX configuration as listed in Table 10-2.

Bit	31	30	29	28	27	26	25	24
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 29:24, 21:16, 13:8, 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 23 to 20; Replace x=23 for 29:24 bits, ... x=20 for 5:0 bits) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.12 LP_GPIO_x Mega Multiplexing 6

Name: MEGA_MUX_IO_SEL_6
Reset: 3F

Absolute Address: 0x4000B1B8

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable a specific peripheral function on LP_GPIO_24 pin using MEGAMUX configuration as listed in Table 10-2.

Bit	7	6	5	4	3	2	1	0
			MEGAMUX_SEL[5:0] LP_GPIO_x					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bits 5:0 – MEGAMUX_SEL[5:0] LP_GPIO_x (x = 24) MEGAMUX configuration
These bits select peripheral function for LP_GPIO_x when, corresponding PINMUX_SEL[2:0] value in PINMUX_SEL_n register is 1 (MUX1).
See Table 10-2 for different possible peripheral options for MEGAMUX_SEL[5:0].

10.4.13 AO_GPIO_z Peripheral Multiplexing

Name: AON_PINMUX_SEL
Reset: 0x0001

Absolute Address: 0x4000F000

This register is a part of AON_GP_REGS0 registers. This register allows the user to enable a specific peripheral function on AO_GPIO_z pins as listed in I/O Port Function Multiplexing table.

Bit	15	14	13	12	11	10	9	8
							PINMUX_SEL[1:0] AO_GPIO_z	
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
			PINMUX_SEL[1:0] AO_GPIO_z				PINMUX_SEL[1:0] AO_GPIO_z	
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	1

Bits 9:8, 5:4, 1:0 – PINMUX_SEL[1:0] AO_GPIO_z (z = 2 to 0; Replace z=2 for 9:8 bits , ... z=0 for 1:0 bits) Pin mux configuration
These bits select peripheral function for AO_GPIO_z pin.

PINMUX_SEL[1:0]	Description
0x0	MUX0 peripheral function is selected
0x1	MUX1 peripheral function is selected
0x2	MUX2 peripheral function is selected
0x3	MUX3 peripheral function is selected

10.4.14 GPIO_MSy Mixed Signal Mode Select

Name: MS_GPIO_MODE
Reset: 0x0F

Absolute Address: 0x4000F410

This register is a part of AON_GP_REGS0 registers. This register allows the user to configure GPIO_MSy as either digital I/O pin or analog input pin.

Bit	7	6	5	4	3	2	1	0
					ANALOG_EN GPIO_MSy	ANALOG_EN GPIO_MSy	ANALOG_EN GPIO_MSy	ANALOG_EN GPIO_MSy
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bits 3,2,1,0 – ANALOG_EN GPIO_MSy (y = 1 to 4 ; y=1 for bit 3, ... y=4 for bit 0) Mixed Signal Mode Select
Writing '0' to a bit configures GPIO_MSy as digital I/O.
Writing '1' to a bit configures GPIO_MSy as an analog input pin.

11. Muxable Interrupt

The Nested Vectored Interrupt Controller (NVIC) supports 32 interrupt lines. Each of the 32 interrupt lines is connected to one peripheral instance, as shown in the ATSAMB11 Interrupt Vector Table table. It is possible to change this default interrupt mapping to a different peripheral through muxable interrupt configuration. IRQ number 0 to IRQ number 20 are muxable interrupts.

The following table provides the list of interrupt options available for each of the muxable interrupts. IRQ_MUX_IO_SEL_n (n = 0 to 20) is the register used to configure the muxable interrupt options. The default value of IRQ_MUX_IO_SEL_n is zero; that is, the IRQ source for the IRQ number (0 to 20) is default as mentioned in the ATSAMB11 Interrupt Vector Table.

The use case of the muxable interrupt option is that when a specific peripheral interrupt (eg: DMA Status) is not mapped in the default ATSAMB11 Interrupt Vector Table, then the peripheral interrupt can be mapped through the muxable interrupt configuration.

Table 11-1. ATSAMB11 Muxable Interrupt Options

Option	Interrupt Source
0x1	UART0 RX
0x2	UART0 TX
0x3	UART1 RX
0x4	UART1 TX
0x5	SPI0 RX
0x6	SPI0 TX
0x7	SPI1 RX
0x8	SPI1 TX
0x9	I^2C0 RX
0xA	I^2C0 TX
0xB	I^2C1 RX
0xC	I^2C1 TX
0xD	WDT0 – RESERVED
0xE	WDT1
0xF	ARM DUALTIMER
0x10	DMA STATUS
0x11	SECURITY
0x12	RESERVED
0x13	QUAD DECODER
0x14	RESERVED
0x15	RESERVED

.....continued	
Option	**Interrupt Source**
0x16	RESERVED
0x17	RESERVED
0x18	BROWNOUT DETECTED

11.1 Example

An example of providing interrupt source as QUAD DECODER for IRQ number 11 using the muxable interrupt options is as follows:

1. Configure IRQ_MUX_IO_SEL_2 register MUX_11[4:0] = 0x13.
2. Follow the steps provided in Functional Description for enabling the interrupt.

11.2 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0						MUX_0[4:0]		
			15:8						MUX_1[4:0]		
			23:16						MUX_2[4:0]		
			31:24						MUX_3[4:0]		
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0						MUX_4[4:0]		
			15:8						MUX_5[4:0]		
			23:16						MUX_6[4:0]		
			31:24						MUX_7[4:0]		
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0						MUX_8[4:0]		
			15:8						MUX_9[4:0]		
			23:16						MUX_10[4:0]		
			31:24						MUX_11[4:0]		
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0						MUX_12[4:0]		
			15:8						MUX_13[4:0]		
			23:16						MUX_14[4:0]		
			31:24						MUX_15[4:0]		
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0						MUX_16[4:0]		
			15:8						MUX_17[4:0]		
			23:16						MUX_18[4:0]		
			31:24						MUX_19[4:0]		
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0						MUX_20[4:0]		

11.3 Register Description

11.3.1 IRQ Multiplexing 0

Name: IRQ_MUX_IO_SEL_0
Reset: 0x00000000

Absolute Address: 0x4000B0C0

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 0, IRQ number 1, IRQ number 2, IRQ number 3 as listed in the ATSAMB11 Interrupt Vector table.

Bit	31	30	29	28	27	26	25	24
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 28:24, 20:16, 12:8, 4:0 – MUX_n[4:0] IRQ number (n) = 3 to 0 (n=3 for 28:24 bits and n=0 for 4:0 bits), IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

11.3.2 IRQ Multiplexing 1

Name: IRQ_MUX_IO_SEL_1
Reset: 0x00000000

Absolute Address: 0x4000B0C4

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 4, IRQ number 5, IRQ number 6, IRQ number 7 as listed in ATSAMB11 Interrupt Vector table.

Bit	31	30	29	28	27	26	25	24
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 28:24, 20:16, 12:8, 4:0 – MUX_n[4:0] IRQ number (n) = 7 to 4 (n=7 for 28:24 bits and n=4 for 4:0 bits, IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

11.3.3 IRQ Multiplexing 2

Name: IRQ_MUX_IO_SEL_2
Reset: 0x00000000

Absolute Address: 0x4000B0C8

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 8, IRQ number 9, IRQ number 10, IRQ number 11 as listed in the ATSAMB11 Interrupt Vector table.

Bit	31	30	29	28	27	26	25	24
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 28:24, 20:16, 12:8, 4:0 – MUX_n[4:0] IRQ number (n) = 11 to 8 (n=11 for 28:24 bits and n=8 for 4:0 bits, IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

11.3.4 IRQ Multiplexing 3

Name: IRQ_MUX_IO_SEL_3
Reset: 0x00000000

Absolute Address: 0x4000B0CC

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 12, IRQ number 13, IRQ number 14, IRQ number 15 as listed in the ATSAMB11 Interrupt Vector table.

Bit	31	30	29	28	27	26	25	24
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 28:24, 20:16, 12:8, 4:0 – MUX_n[4:0] IRQ number (n) = 15 to 12 (n=15 for 28:24 bits and n=12 for 4:0 bits, IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

11.3.5 IRQ Multiplexing 4

Name: IRQ_MUX_IO_SEL_4
Reset: 0x00000000

Absolute Address: 0x4000B0D0

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 16, IRQ number 17, IRQ number 18, IRQ number 19 as listed in the ATSAMB11 Interrupt Vector table.

Bit	31	30	29	28	27	26	25	24
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 28:24, 20:16, 12:8, 4:0 – MUX_n[4:0] IRQ number (n) = 19 to 16 (n=19 for 28:24 bits and n=16 for 4:0 bits, IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

11.3.6 IRQ Multiplexing 5

Name: IRQ_MUX_IO_SEL_5
Reset: 0x00000000

Absolute Address: 0x4000B0D4

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to select the interrupt source for IRQ number 20 as listed in the ATSAMB11 Interrupt Vector table.

Bit	7	6	5	4	3	2	1	0
				MUX_n[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 4:0 – MUX_n[4:0] IRQ number (n) = 20, IRQ mux configuration
These bits select interrupt source for specific IRQ number.
Refer to the ATSAMB11 Muxable Interrupt Options table for different possible interrupt source options for MUX_n[4:0].

12. GPIO Pin Controller

The GPIO pin controller controls the I/O pins of the device. The I/O pins are organized in a series of groups, collectively referred to as a Port group. Each Port group can have up to 16 pins that can be configured and controlled individually or as a group. There are three Port groups on a SAMB11-XR/ZR device, controlled by three GPIO pin controllers. Each pin is either used for general-purpose I/O under direct application control or assigned to an embedded device peripheral. When used for general-purpose I/O, each pin is configured as input or output, with configurable pull-up/pull-down settings.

All I/O pins have true read-modify-write functionality when used for general purpose I/O. The direction or the output value of one or more pins may be changed explicitly without unintentionally changing the state of any other pins in the same port group by a single, atomic 16-bit write.

The GPIO controller is connected to the high-speed bus matrix through the AHB bus.

12.1 Features

The AHB GPIO controller provides a 16-bit I/O interface with the following properties:

- Selectable input and output configuration for each individual pin.
- Software-controlled multiplexing of peripheral functions on I/O pins.
- Configurable pull settings:
  - Internal pull-up or pull-down.
- Programmable interrupt generation capability:
  - Interrupt generation masking.
  - Edge-triggered on rising, falling.
  - Level-sensitive on high or low values.
- Thread safe operation by providing separate set and clear addresses for control registers.
- Inputs are sampled using a double flip-flop to avoid metastability issues.

12.2 Signal Description

The following table describes the signal description of the GPIO pins.

Table 12-1. Signal Description for GPIO Pins

Pin Name	Type	Description
LP_GPIO_x	Digital I/O	General-purpose I/O pin x
AO_GPIO_z	Digital I/O	General-purpose I/O pin z. It can also be used to wake-up the core from the Ultra-Low Power mode
GPIO_MSy	Mixed signal I/O	General-purpose I/O pin y. Analog input can be connected

12.3 I/O Lines

The I/O lines are mapped to pins of the physical device. The I/O lines are categorized based on different functionalities into three groups called LP_GPIO_x, AO_GPIO_z, GPIO_MSy, as shown in Table 12-1. However, all these GPIO's are controlled by one of three GPIO controllers.

The I/O lines are divided into three groups with each group of 16 pins controlled by an individual GPIO controller. Referring to the GPIO Controller column of Table 10-1, the I/O lines from GPIO0_00 to GPIO0_15 are controlled using GPIO0 controller; GPIO1_00 to GPIO1_15 are controlled using GPIO1 controller, and GPIO2_00 to GPIO2_15 are controlled using GPIO2 controller.

Each pin can be controlled by pinmux settings, which allow the pad to route internally to a dedicated peripheral function. When set to specific PINMUX value in PINMUX_SEL_n/AON_PINMUX_SEL_n, the selected peripheral controls the output state of the pad, and reads the current physical pad state.

For more details see, I/O Multiplexing.

12.4 Clock Configuration

The GPIO controller must be provided with clock source before configuring for normal operation. Two bits GPIO_CLK_EN, and GPIO_GCLK_EN enable the clock for GPIO controllers and its AHB interface. Both the bits must be set for GPIO controller normal operation. For more details on configuration, see Peripheral Clock Configuration.

12.5 Functional Description of LP_GPIO_x I/O Pins

The overview and peripheral multiplexing of LP_GPIO_x I/O pins are shown in the following figures.

Figure 12-1. Overview of LP_GPIO_x Port

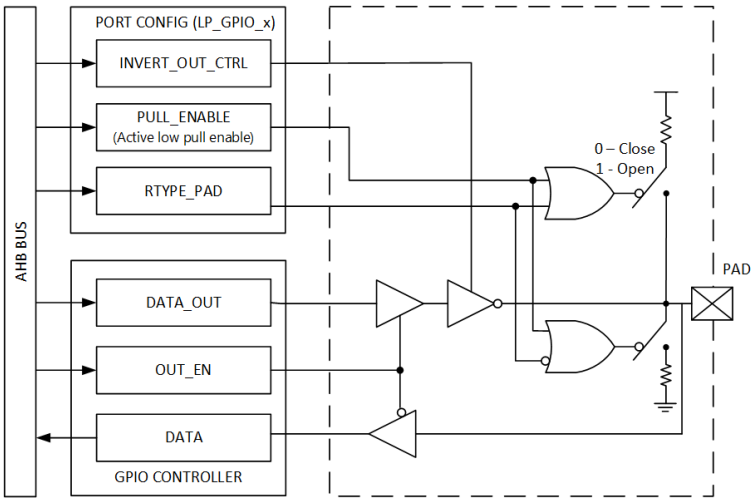
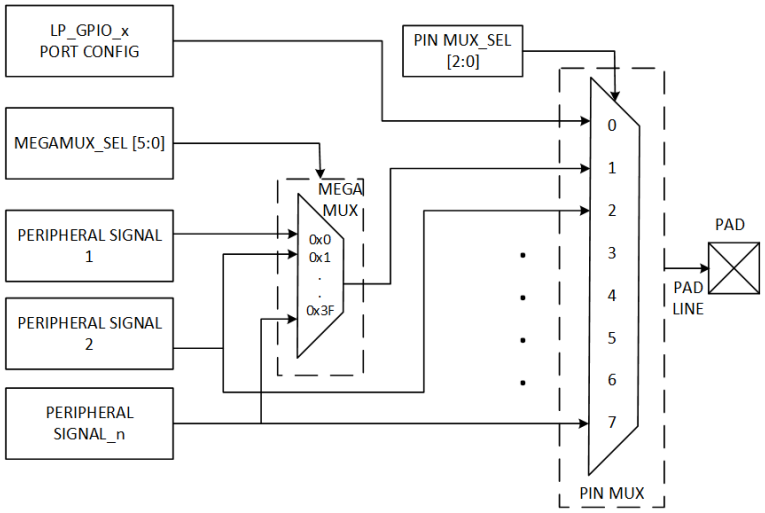


Figure 12-2. Overview of Peripheral Multiplexing for LP_GPIO_x Port



12.5.1 Initialization

After reset, all LP_GPIO_x pads are enabled with input and pull up. For more details, see the I/O Pin Behavior in the Different Device States table.

However, specific pins, such as those used for connection to a debugger are configured using PINMUX_SEL_n, as required by its special function.

12.5.2 Operation

1. Each LP_GPIO_x can be configured by the registers in PORT Config as shown in Figure 12-1. It is required to map a specific pin (LP_GPIO_x) to the GPIO controller which controls it. To control pin LP_GPIO_x, get the corresponding GPIO pin number ‘n’ and the GPIO controller number ‘p’ which controls it from I/O Multiplexing under GPIO Controller column (GPIOp_n).
   For example, LP_GPIO_18 is mapped to GPIO1_02. Here ‘p’ equals to 1, ‘n’ equals to 02 and ‘x’ equals to 18. In the configuration steps below, ‘n’ and ‘p’ are used to configure GPIO controller specific registers, and ‘x’ is used to configure LPMCU_MISC_REGS0 specific registers. 'n', 'p', 'x' value of specific LP_GPIO_x pin to be noted to configure that pin as general purpose I/O.
2. Configure PINMUX_SEL[2:0] bits of PINMUX_SEL_n register corresponding to x pin with MUX value equals to zero. Three bits are used to configure MUX for LP_GPIO_x pin.
   For example, to configure LP_GPIO_18: PINMUX_SEL_2 register PINMUX_SEL[2:0] LP_GPIO_18 = 0.
3. To use LP_GPIO_x pin as an output, write n bit of OUTENSET register of ‘p’ GPIO controller to '1'. The n bit in the DATAOUT register must be written to the desired output value. The output on pin ‘n’ can be inverted by configuring INVERT_OUT_CTRL register. To invert ‘n’ pin write ‘1’ of LP_GPIO_x bit on INVERT_OUT_CTRL register.
   For example, to configure LP_GPIO_18 as output: assign bit 2 of OUTENSET register of GPIO1(p) Controller to '1'.
4. To use pin LP_GPIO_x as an input, bit ‘n’ in the OUTENCLR register of ‘p’ GPIO controller must be written to '1'. The input value can be read from bit ‘n’ in register DATA.
   For example, to configure LP_GPIO_18 as input: assign bit 2 of OUTENCLR register of GPIO1(p) Controller to '1'.

The (PINMUX_SEL_n) registers select the peripheral function for the corresponding pin. When PINMUX_SEL[2:0] value is other than zero, then this overrides the connection between the GPIO

controller and the I/O pin, and connects the selected peripheral signal to the particular I/O pin instead of the GPIO Controller. For more information on different MUX configurations, see I/O Multiplexing.

12.5.3 I/O Pin Pull Configuration

The PULL_ENABLE and RTYPE_PAD registers are used to configure pull up or pull down on the LP_GPIO_x I/O pins as shown in the following pull configuration table. The internal pull up or pull down is mainly used for input; if the input pin is not connected, then Hi-z occurs. Enabling pull up or pull down leads to defined state on input line. The pull up or pull down circuit is common for input and output.

Table 12-2. Pull Configuration

OUTENSET	PULL_ENABLE	RTYPE_PAD	Configuration
0	0	0	Input with pull up
0	0	1	Input with pull down
0	1	0	Input; pull disabled If input line is floating then the value on pad is Hi-z. This means the sampled value in DATA register can be 0 or 1, and not a fixed value.
0	1	1	
1	0	0	Output with pull up
1	0	1	Output with pull down
1	1	0	Output; pull disabled
1	1	1	Output; pull disabled

12.5.4 Example

The following is the sample example to get logic high output on LP_GPIO_10 pin:

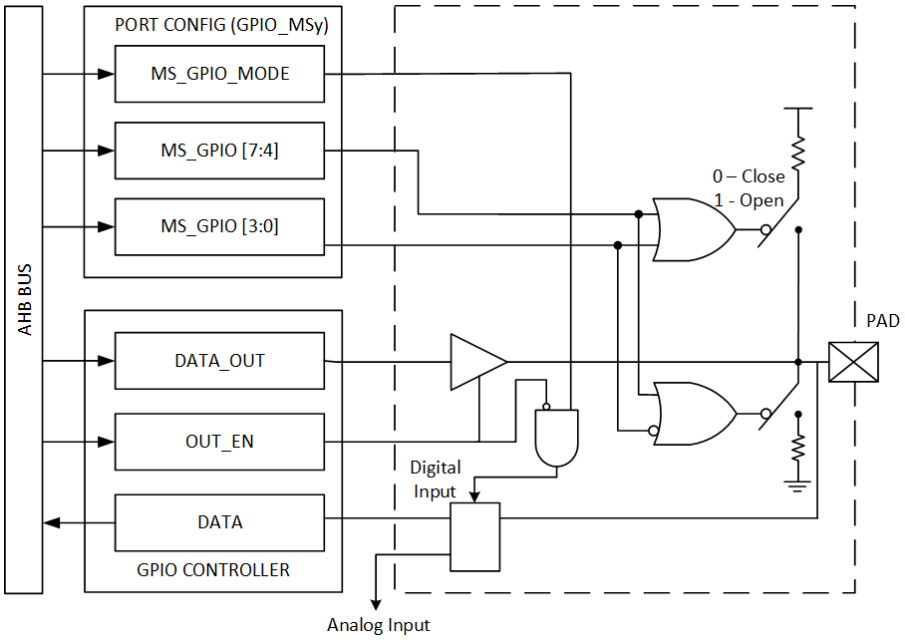
Note: The LP_GPIO_10 pin is mapped to GPIO0_10 from I/O Multiplexing and belongs to the GPIO0 controller.

```
/* Write PINMUX_SEL_1 bits [10:8] with 0x0 (MUX0) */
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x0<<8);
/* Write '1' to OUTENSET to enable output */
GPIO0->OUTENSET.reg |= (1 << 10);
/* Write '1' to DATAOUT to output logic high*/
GPIO0->DATAOUT.reg |= (1 << 10);
```

12.6 Functional Description of GPIO_MSy I/O Pins

The overview of GPIO_MSy I/O pins is shown in the following figure.

Figure 12-3. Overview of GPIO_MSy Port



12.6.1 Initialization

After reset, all GPIO_MSy pads are configured as analog input pin with pull disabled.

12.6.2 Operation

The GPIO_MSy pins are mixed signal pins which are configured as analog or digital. Each GPIO_MSy can be configured by the registers in PORT Config as shown in Figure 12-3.

The GPIO_MSy pins are configured either as analog input pins or general purpose digital I/O pins using register MS_GPIO_MODE. By default, the GPIO_MSy pins are analog input pins with the MS_GPIO_MODE bit set for individual pins.

1. To use GPIO_MSy as general purpose I/O pin, clear the ANALOG_EN GPIO_MSy bit of MS_GPIO_MODE register.
2. Similar to LP_GPIO_x, it is required to get the corresponding GPIO_MSy pin number ‘n’ from I/O multiplexing under GPIO controller column. The GPIO_MSy I/O pins are controlled by GPIO2 controller. To use ‘n’ pin as an output, write ‘n’ bit of OUTENSET register of GPIO2 controller. The ‘n’ bit in the DATAOUT register must be written to the desired output value.
   For example, to configure GPIO_MS1 as output: assign bit 15 of OUTENSET register of GPIO2 Controller to '1'.
3. To use pin GPIO_MSy as an input, bit ‘n’ in the OUTENCLR register of GPIO2 Controller must be written to '1'. The input value can be read from bit ‘n’ in register DATA.
   For example, to configure GPIO_MS1 as input: assign bit 15 of OUTENCLR register of GPIO2 Controller to '1'.

12.6.3 I/O Pin Pull Configuration

The MS_GPIO register is used to configure pull up or pull down on the GPIO_MSy I/O pins as shown in the following pull configuration table. The pull up/down circuit is common for input and output.

Table 12-3. Pull Configuration

OUTENSET	MS_GPIO[7:4]	MS_GPIO[3:0]	Configuration
0	0	0	Input with pull up
0	0	1	Input with pull down
0	1	0	Input; pull disabled If input line is floating then the value on PAD is Hi-z. This means the value be sampled in DATA register can be 0 or 1, not a fixed value.
0	1	1	
1	0	0	Output with pull up
1	0	1	Output with pull down
1	1	0	Output; pull disabled
1	1	1	Output; pull disabled

12.6.4 Example

The following is the sample example to read input from GPIO_MS1 pin:

Note: The GPIO_MS1 pin is mapped to GPIO2_15 from I/O Multiplexing and belongs to the GPIO2 controller.

```
/* Write MS_GPIO_MODE, ANALOG_EN GPIO_MS1 bit to 0x0  */
AON_GP_REGS0->MS_GPIO_MODE.bit.ANALOG_EN GPIO_MS1 = 0;
/* Write '1' to OUTENCLR to enable input */
GPIO2->OUTENCLR.reg |= (1 << 15);
/* Read input on Bit 15 of DATA register*/
regval = GPIO2->DATA.reg;
regval &= (1 << 15);
```

12.7 Functional Description of AO_GPIO_z I/O Pins

The overview and peripheral multiplexing of AO_GPIO_z I/O pins are shown in the following figures.

Figure 12-4. Overview of AO_GPIO_z Port

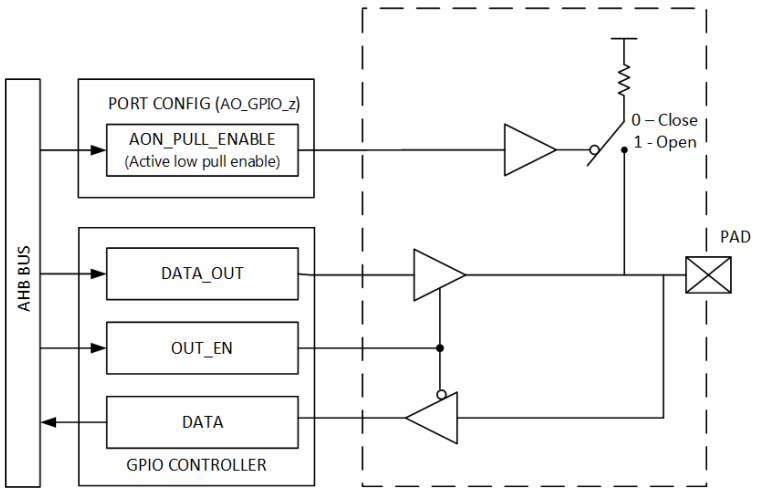
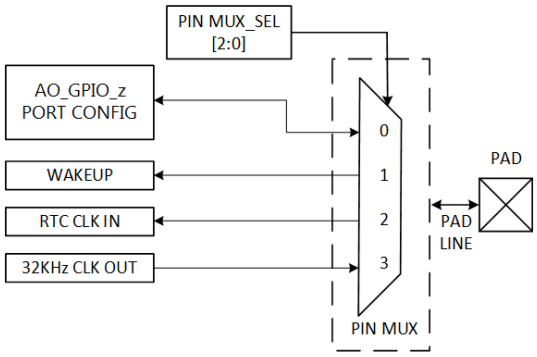


Figure 12-5. Overview of Peripheral Multiplexing for AO_GPIO_z Port



12.7.1 Initialization

After reset, all AO_GPIO_z pads are enabled with input and pull up. By default, the AO_GPIO_0 pin is configured to wake-up the core from the Ultra-Low Power (ULP) mode.

12.7.2 Operation

The AO_GPIO_z pins are special function I/O pins that are configured as general purpose I/O and also configured to wake up (Wake-up Source) the core from the ULP mode.

1. To use AO_GPIO_z pin as an general purpose I/O, configure PINMUX_SEL[1:0] of AON_PINMUX_SEL register corresponding to AO_GPIO_y pin with MUX value equal to zero. Two bits are used to configure MUX for AO_GPIO_z GPIO pin.
   For example, to configure AO_GPIO_1: AON_PINMUX_SEL register PINMUX_SEL[1:0] AO_GPIO_1 = 0.
2. Similar to LP_GPIO_x, it is required to get the corresponding AO_GPIO_z pin number ‘n’ from I/O Multiplexing under GPIO controller column. The AO_GPIO_z I/O lines are controlled by GPIO1 controller. To use AO_GPIO_z ‘n’ pin as an output, write n bit of OUTENSET register of GPIO1 controller. The n bit in the DATAOUT register must be written to the desired output value.
   For example, to configure AO_GPIO_1 as output: assign bit 14 of OUTENSET register of GPIO1 Controller to '1'.
3. To use pin AO_GPIO_z ‘n’ as an input, bit ‘n’ in the OUTENCLR register of GPIO1 controller must be written to '1'. The input value can be read from bit ‘n’ in register DATA.
   For example, to configure AO_GPIO_1 as input: assign bit 14 of OUTENCLR register of GPIO1 Controller to '1'.

The AON_PINMUX_SEL register selects the peripheral function for the corresponding AO_GPIO_z pin. When PINMUX_SEL[1:0] value is other than zero, then this overrides the connection between the GPIO controller and the I/O pin, and connects the selected peripheral signal to the particular I/O pin instead of the GPIO controller. For more information on different MUX configuration, see I/O Multiplexing.

12.7.3 I/O Pin Pull Configuration

The AON_PULL_ENABLE register is used to configure pull up on the AO_GPIO_z I/O pins as shown in the following pull configuration table. The pull up circuit is common for input and output.

Table 12-4. Pull Configuration

OUTENSET	AON_PULL_ENABLE	Configuration
0	0	Input with pull up

.....continued		
OUTENSET	**AON_PULL_ENABLE**	**Configuration**
0	1	Input; pull disabled If input line is floating then the value on pad is Hi-z. This means the sampled value in DATA register can be 0 or 1, and not a fixed value.
1	0	Output with pull up
1	1	Output; pull disabled

12.7.4 Example

The following is the sample example to get logic low output on the AO_GPIO_2 pin:
Note: The AO_GPIO_2 pin is mapped to GPIO1_13 from I/O Multiplexing and belongs to the GPIO1 controller.

```
/* Write AON_PINMUX_SEL, bit AO_GPIO_2 with 0x0 (MUX0) */
AON_GP_REGS0->AON_PINMUX_SEL.bit.AO_GPIO_2 = 0;
/* Write '1' to OUTENSET to enable output */
GPIO1->OUTENSET.reg |= (1 << 13);
/* Write '0' to DATAOUT to output logic high*/
GPIO1->DATAOUT.reg &= ~(1 << 13);
```

12.7.5 Wake-up Source

The AO_GPIO_z pin can also wake-up the ARM Subsystem and BLE Subsystem from the ULP mode. Along with wake-up configuration, enabling an external interrupt generates an interrupt request. On reset, the AO_GPIO_0 as wake-up source is enabled by default.

1. To use AO_GPIO_z as wake-up source:
   - Wake-up source for ARM subsystem – configure AON_PINMUX_SEL with MUX1 and set the value as one for specific AO_GPIO_z pin.
   - Wake-up source for BLE subsystem – set the BLE_ENABLE bit on GPIO_WAKEUP_CTRL register. This configuration is common for all AO_GPIO_z pins.
     Note:
     - It is recommended to enable wake-up of ARM subsystem and BLE subsystem together. Enabling only the wake-up of BLE subsystem should not be performed.
     - Only AO_GPIO_0 can be configured as wake-up source with the present firmware. The AO_GPIO_1 and AO_GPIO_2 cannot be used as wake-up source at present.
     - As the firmware on ROM handles the wake-up and sleep operation, it is restricted to configure the AO_GPIO_z as rising interrupt only. Rising edge on AO_GPIO_z wakes-up the ARM and falling edge triggers the sleep. On rising edge, interrupt ROM firmware configures the AO_GPIO_z as falling edge. Until AO_GPIO_z is held high, the device is awake and awaits for falling edge on this pin. At falling edge, it triggers the interrupt and ROM firmware, configures the pin as rising edge, and enables sleep operation.
2. Configure AO_GPIO_z pin as input with external interrupt enable. For more details, see External Interrupt section.

12.8 External Interrupt

The GPIO Controller allows all GPIO pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z) to be configured as interrupt lines. Each interrupt line can be individually masked and can generate an interrupt to CPU on rising, falling, or on high or low levels.

12.8.1 Initialization

After reset, the interrupt is disabled on all GPIO pins.

12.8.2 Operation

To configure a GPIO pin as external interrupt, get the corresponding GPIO pin number ‘n’ and GPIO controller number ‘p’ from I/O Multiplexing table under the GPIO Controller column and perform the following initialization steps:

- Configure PINMUX_SEL_n or MS_GPIO_MODE or AON_PINMUX_SEL based in GPIO group with MUX0. When AO_GPIO_z is configured as wake-up source, then configure AON_PINMUX_SEL with MUX1.
- Initialize the GPIO pin direction as input. For LP_GPIO_x pin operation, see 12.5.2 Operation section. For GPIO_MSy pin operation, see 12.6.2 Operation section. For AO_GPIO_z pin operation, see 12.7.2 Operation section.
- Refer to default pull configuration of the intended GPIO pin group and enable or disable pull based on the requirement.
- The GPIO controller provides programmable interrupt generation features. As shown in the following table, three registers control the operation, and each register has separate set and clear addresses. To configure each bit of the I/O pin (‘n’ bit on ‘p’ GPIO register) generate interrupt based on these three registers.

Table 12-5. Interrupt Configuration

Interrupt Enable Set (INTENSET)	Interrupt Polarity Set (INTPOLSET)	Interrupt Type Set (INTTYPESET)	Interrupt Feature
0	X	X	Disabled
1	0	0	Low-Level
1	0	1	Falling Edge
1	1	0	High-Level
1	1	1	Rising Edge

Note: External interrupts are double synchronized to FCLK (Free-running clock) before being converted to edge/level types and then is registered on FCLK. Hence, there are only three FCLK cycles latency on these interrupts for all of the four types (high level/low level/ rise edge/ fall edge) of interrupt configurations before being sensed by the processor.

- The interrupt request line is connected to the Interrupt Controller (NVIC). To use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. For more details, see Nested Vector Interrupt Controller.
- After an interrupt request is triggered, the corresponding bit in the INTSTATUSCLEAR register is set. The interrupt status can be cleared by writing 1 to the corresponding bit of the INTSTATUSCLEAR register.

12.8.3 Example

The following is the sample example to configure the GPIO_MS1 pin as interrupt on rising edge:
Note: The GPIO_MS1 pin is mapped to GPIO2_15 pin from I/O Multiplexing and belongs to the GPIO2 controller.

```
/* Register ISR handler and enable interrupt as explained in Section 6.3  Nested Vector
Interrupt Controller */

/* Write MS_GPIO_MODE, ANALOG_EN GPIO_MS1 bit to 0x0  */
AON_GP_REGS0->MS_GPIO_MODE.bit.ANALOG_EN GPIO_MS1 = 0;

/* Write '1' to OUTENCLR to enable input */
GPIO2->OUTENCLR.reg |= (1 << 15);
/* Write '1' to INTYPESET and INTPOLSET registers to enable rising edge interrupt*/
GPIO2->INTTYPESET.reg |= (1 << 15);
GPIO2->INTPOLSET.reg |= (1 << 15);
/* Write '1' to INTENSET register to enable interrupt on GPIO */
GPIO2->INTENSET.reg |= (1 << 15);
```

12.9 Power Management

If the system goes to Ultra-Low Power mode, the GPIO controller is shut down and the latches in the pad retain their current configuration in the Sleep mode, such as the output value and pull settings. However, the GPIO Controller configuration registers lose their content, and these cannot be restored when the GPIO Controller is powered-up again. Therefore, user must reconfigure the GPIO Controller at power-up to ensure it is in a well-defined state before use. For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains on how sleep and wake-up are controlled.

12.10 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40010000 (GPIO0), 0x40011000 (GPIO1), 0x40013000 (GPIO2)	GPIO Controller	DATA	7:0	DATA[7:0]							
			15:8	DATA[15:8]							
0x40010004 (GPIO0), 0x40011004 (GPIO1), 0x40013004 (GPIO2)	GPIO Controller	DATAOUT	7:0	DATAOUT[7:0]							
			15:8	DATAOUT[15:8]							
0x40010010 (GPIO0), 0x40011010 (GPIO1), 0x40013010 (GPIO2)	GPIO Controller	OUTENSET	7:0	OUTENSET[7:0]							
			15:8	OUTENSET[15:8]							
0x40010014 (GPIO0), 0x40011014 (GPIO1), 0x40013014 (GPIO2)	GPIO Controller	OUTENCLR	7:0	OUTENCLR[7:0]							
			15:8	OUTENCLR[15:8]							
0x40010020 (GPIO0), 0x40011020 (GPIO1), 0x40013020 (GPIO2)	GPIO Controller	INTENSET	7:0	INTENSET[7:0]							
			15:8	INTENSET[15:8]							

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x40010024 (GPIO0), 0x40011024 (GPIO1), 0x40013024 (GPIO2)	GPIO Controller	INTENCLR	7:0	INTENCLR[7:0]							
			15:8	INTENCLR[15:8]							
0x40010028 (GPIO0), 0x40011028 (GPIO1), 0x40013028 (GPIO2)	GPIO Controller	INTTYPESET	7:0	INTTYPESET[7:0]							
			15:8	INTTYPESET[15:8]							
0x4001002C (GPIO0), 0x4001102C (GPIO1), 0x4001302C (GPIO2)	GPIO Controller	INTTYPECLR	7:0	INTTYPECLR[7:0]							
			15:8	INTTYPECLR[15:8]							
0x40010030 (GPIO0), 0x40011030 (GPIO1), 0x40013030 (GPIO2)	GPIO Controller	INTPOLSET	7:0	INTPOLSET[7:0]							
			15:8	INTPOLSET[15:8]							
0x40010034 (GPIO0), 0x40011034 (GPIO1), 0x40013034 (GPIO2)	GPIO Controller	INTPOLCLR	7:0	INTPOLCLR[7:0]							
			15:8	INTPOLCLR[15:8]							
0x40010038 (GPIO0), 0x40011038 (GPIO1), 0x40013038 (GPIO2)	GPIO Controller	INTSTATUSCLEAR	7:0	INTSTATUSCLEAR[7:0]							
			15:8	INTSTATUSCLEAR[15:8]							
0x4000B040	LPMCU_MISC_REGS0	INVERT_OUTPUT_CTRL	7:0	INVERT_OUT LP_GPIO_x (x = 7:0)							
			15:8	INVERT_OUT LP_GPIO_x (x = 15:8)							
			23:16	INVERT_OUT LP_GPIO_x (x = 23:16)							
			31:24	INVERT_OUT LP_SIP_x (x = 5:0)							INVERT_OUT LP_GPIO_24
0x4000B050	LPMCU_MISC_REGS0	PULL_ENABLE	7:0	PULL_EN LP_GPIO_x (x = 7:0)							
			15:8	PULL_EN LP_GPIO_x (x = 15:8)							
			23:16	PULL_EN LP_GPIO_x (x = 23:16)							
			31:24	PULL_EN LP_SIP_x (x = 5:0)							PULL_EN LP_GPIO_24
0x4000B054	LPMCU_MISC_REGS0	RTYPE_PAD_0	7:0	PULL_TYPE LP_GPIO_x (x = 7:0)							
			15:8	PULL_TYPE LP_GPIO_x (x = 15:8)							
			23:16	PULL_TYPE LP_GPIO_x (x = 23:16)							
			31:24								PULL_TYPE LP_GPIO_24
0x4000B414	LPMCU_MISC_REGS0	MS_GPIO	7:0	PULL_EN GPIO_MSy (y = 1:4)				PULL_TYPE GPIO_MSy (y = 1:4)			
0x4000F014	AON_GP_REGS0	AON_PULL_ENABLE	7:0						PULLUP_EN AO_GPIO_z (z= 0:2)		
0x4000E000	AON_PWR_SEQ0	GPIO_WAKEUP_CTRL	7:0							BLE_ENABLE	
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0			MEGAMUX_SEL[5:0] LP_GPIO_0					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_1					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_2					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_3					
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0			MEGAMUX_SEL[5:0] LP_GPIO_4					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_5					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_6					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_7					
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0			MEGAMUX_SEL[5:0] LP_GPIO_8					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_9					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_10					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_11					
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0			MEGAMUX_SEL[5:0] LP_GPIO_12					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_13					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_14					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_15					
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0			MEGAMUX_SEL[5:0] LP_GPIO_16					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_17					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_18					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_19					
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0			MEGAMUX_SEL[5:0] LP_GPIO_20					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_21					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_22					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_23					
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0			MEGAMUX_SEL[5:0] LP_GPIO_24					
0x4000F000	AON_GP_REGS0	AON_PINMUX_SEL	7:0			PINMUX_SEL[1:0] AO_GPIO_1				PINMUX_SEL[1:0] AO_GPIO_0	
			15:8							PINMUX_SEL[1:0] AO_GPIO_2	
0x4000F410	AON_GP_REGS0	MS_GPIO_MODE	7:0				ANALOG_EN GPIO_MS1	ANALOG_EN GPIO_MS2	ANALOG_EN GPIO_MS3	ANALOG_EN GPIO_MS4	

12.11 Register Description

12.11.1 Data Input Value

Name: DATA
Reset: 0x---- (Based on level on I/O pin)

Absolute Address: 0x40010000 (GPIO0), 0x40011000 (GPIO1), 0x40013000 (GPIO2)

This register is a part of GPIO Controller registers. This register reads the input drive value on individual I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to register bit.

Bit	15	14	13	12	11	10	9	8
	DATA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bits 15:0 – DATA[15:0] Port Data Input Value

Read Value	Description
0	The corresponding I/O pin input sampler detects a logical low level on the input pin.
1	The corresponding I/O pin input sampler detects a logical high level on the input pin

Writing this register sends the DATA register value to DATAOUT register.

12.11.2 Data Output Value

Name: DATAOUT
Reset: 0x0000

Absolute Address: 0x40010004 (GPIO0), 0x40011004 (GPIO1), 0x40013004 (GPIO2)

This register is a part of GPIO Controller registers. This register sets the data output drive value for individual I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	DATAOUT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATAOUT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – DATAOUT[15:0] Port Data Output Value
When the register is configured as output via the Data Direction register (OUTENSET), these bits set the logical output drive.

Read Value	Description
0	The corresponding I/O pin is driven logical low level.
1	The corresponding I/O pin is driven logical high level.

12.11.3 Data Direction Set

Name: OUTENSET
Reset: 0x0000

Absolute Address: 0x40010010 (GPIO0), 0x40011010 (GPIO1), 0x40013010 (GPIO2)

This register is a part of GPIO Controller registers. This register allows the user to set one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z) as an output. This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to register bit.

Bit	15	14	13	12	11	10	9	8
	OUTENSET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTENSET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – OUTENSET[15:0] Port Data Direction Set
Writing '0' to a bit has no effect.
Writing '1' to a bit sets the corresponding bit in the OUTENSET register, which configures the I/O pin as an output.

Read Value	Description
0	Indicates the corresponding I/O pin in the GPIO Controller group as input.
1	Indicates the corresponding I/O pin in the GPIO Controller group as output.

12.11.4 Data Direction Clear

Name: OUTENCLR
Reset: 0x0000

Absolute Address: 0x40010014 (GPIO0), 0x40011014 (GPIO1), 0x40013014 (GPIO2)

This register is a part of GPIO Controller registers. This register allows the user to set one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z) as input. This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	OUTENCLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OUTENCLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – OUTENCLR[15:0] Port Data Direction Clear
Writing '0' to a bit has no effect.
Writing '1' to a bit clears the corresponding bit in the OUTENSET register, which configures the I/O pin as an input.

Read Value	Description
0	Indicates the corresponding I/O pin in the GPIO Controller group as input.
1	Indicates the corresponding I/O pin in the GPIO Controller group as output.

12.11.5 Interrupt Enable Set

Name: INTENSET
Reset: 0x0000

Absolute Address: 0x40010020 (GPIO0), 0x40011020 (GPIO1), 0x40013020 (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to enable interrupt on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTENSET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTENSET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTENSET[15:0] Interrupt Enable Set
Writing '0' to a bit has no effect.
Writing '1' to a bit sets the corresponding bit in the INTENSET register, which enables interrupt on I/O pin.

Read Value	Description
0	Indicates the interrupt is disabled on corresponding I/O pin in GPIO controller group.
1	Indicates the interrupt is enabled on corresponding I/O pin in GPIO controller group.

12.11.6 Interrupt Enable Clear

Name: INTENCLR
Reset: 0x0000

Absolute Address: 0x40010024 (GPIO0), 0x40011024 (GPIO1), 0x40013024 (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to disable interrupt on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTENCLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTENCLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTENCLR[15:0] Interrupt Enable Clear
Writing '0' to a bit has no effect.
Writing '1' to a bit clears the corresponding bit in the INTENSET register, which disables interrupt on I/O pin.

Read Value	Description
0	Indicates the interrupt is disabled on corresponding I/O pin in GPIO controller group.
1	Indicates the interrupt is enabled on corresponding I/O pin in GPIO controller group.

12.11.7 Interrupt Type Set

Name: INTTYPESET
Reset: 0x0000

Absolute Address: 0x40010028 (GPIO0), 0x40011028 (GPIO1), 0x40013028 (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to set interrupt type on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTTYPESET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTTYPESET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTTYPESET[15:0] Interrupt Type Set
Writing '0' to a bit has no effect.
Writing '1' to a bit sets the corresponding bit in the INTTYPESET register, which configures interrupt as falling edge or rising edge. For more details, see Interrupt Configuration table.

Read Value	Description
0	Indicates the interrupt as LOW level or HIGH level decided by INTPOL register.
1	Indicates the interrupt as Falling edge or Rising edge decided by INTPOL register.

12.11.8 Interrupt Type Clear

Name: INTTYPECLR
Reset: 0x0000

Absolute Address: 0x4001002C (GPIO0), 0x4001102C (GPIO1), 0x4001302C (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to clear interrupt type on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTTYPECLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTTYPECLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTTYPECLR[15:0] Interrupt Type Clear
Writing '0' to a bit has no effect.
Writing '1' to a bit clears the corresponding bit in the INTTYPESET register.

Read Value	Description
0	Indicates the interrupt as LOW level or HIGH level decided by INTPOL register.
1	Indicates the interrupt as Falling edge or Rising edge decided by INTPOL register.

12.11.9 Interrupt Polarity Set

Name: INTPOLSET
Reset: 0x0000

Absolute Address: 0x40010030 (GPIO0), 0x40011030 (GPIO1), 0x40013030 (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to set interrupt polarity on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTPOLSET[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTPOLSET[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTPOLSET[15:0] Interrupt Polarity Set
Writing '0' to a bit has no effect.
Writing '1' to a bit sets the corresponding bit in the INTPOLSET register, which configures interrupt as High level or Rising edge. For more details, see Interrupt Configuration table.

Read Value	Description
0	Indicates the interrupt as LOW level or Falling edge decided by INTTYPE register.
1	Indicates the interrupt as HIGH level or Rising edge decided by INTTYPE register.

12.11.10 Interrupt Polarity Clear

Name: INTPOLCLR
Reset: 0x0000

Absolute Address: 0x40010034 (GPIO0), 0x40011034 (GPIO1), 0x40013034 (GPIO2)

This register is a part of GPIO Controller Registers. This register allows the user to clear interrupt polarity on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of the I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to the register bit.

Bit	15	14	13	12	11	10	9	8
	INTPOLCLR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTPOLCLR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTPOLCLR[15:0] Interrupt Polarity Clear
Writing '0' to a bit has no effect.
Writing '1' to a bit clears the corresponding bit in the INTPOLSET register.

Read Value	Description
0	Indicates the interrupt as LOW level or Falling edge decided by INTTYPE register.
1	Indicates the interrupt as HIGH level or Rising edge decided by INTTYPE register.

12.11.11 Interrupt Status Clear

Name: INTSTATUSCLEAR
Reset: 0x0000

Absolute Address: 0x40010038 (GPIO0), 0x40011038 (GPIO1), 0x40013038 (GPIO2)

This register is a part of GPIO Controller Registers. This register indicates status of interrupt trigger and allows the user to clear interrupt status on one or more I/O pins (LP_GPIO_x, GPIO_MSy, AO_GPIO_z). This register is duplicated for each GPIO Controller, with increasing base address.

For more details on mapping between LP_GPIO_x, GPIO_MSy, AO_GPIO_z pin number with GPIO controller pin number, see GPIO Controller column of I/O Port Function Multiplexing table. The corresponding GPIO controller pin number 'n' is mapped to register bit.

Bit	15	14	13	12	11	10	9	8
	INTSTATUSCLEAR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTSTATUSCLEAR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – INTSTATUSCLEAR[15:0] Interrupt Status Clear
Writing '0' to a bit has no effect.
Writing '1' to a bit clears the interrupt request.

Read Value	Description
0	Indicates the interrupt request is not triggered.
1	Indicates the interrupt request is triggered.

12.11.12 LP_GPIO_x Invert Output Level

Name: INVERT_OUTPUT_CTRL
Reset: 0x00000000

Absolute Address: 0x4000B040

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to invert the logical output level which is set using DATAOUT register for LP_GPIO_x pins.

Bit	31	30	29	28	27	26	25	24
	INVERT_OUT LP_SIP_x	INVERT_OUT LP_SIP_x	INVERT_OUT LP_SIP_x	INVERT_OUT LP_SIP_x	INVERT_OUT LP_SIP_x			INVERT_OUT LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W			R/W
Reset	0	0	0	0	0			0

Bit	23	22	21	20	19	18	17	16
	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x	INVERT_OUT LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31,30,29,28,27 – INVERT_OUT LP_SIP_x (x = 4 to 0; Replace x=4 for bit 31, ... x=0 for bit 27)
Output level invert
The LP_SIP_x pins are connected to internal SPI Flash0 which holds the application code. It is not recommended to use these pins as general purpose I/O.

Bits 24,23,22,21,20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – INVERT_OUT LP_GPIO_x
(x = 24 to 0; Replace x=24 for bit 24, ... x=0 for bit 0) Output level invert
Writing '1'inverts the output logical level set using DATAOUT register on LP_GPIO_x pin. For example, if low level is set using DATAOUT register, writing '1' to INVERT_OUTPUT_CTRL changes to high level.
Writing '0' sets the output logical level same as DATAOUT register on LP_GPIO_x pin.

12.11.13 LP_GPIO_x Pull Enable

Name: PULL_ENABLE
Reset: 0x00000000

Absolute Address: 0x4000B050

This register is a part of LPMCU_MISC_REGS0 Registers. This register allows the user to enable or disable the internal pull up or pull down for LP_GPIO_x pins.

Bit	31	30	29	28	27	26	25	24
	PULL_EN LP_SIP_x	PULL_EN LP_SIP_x	PULL_EN LP_SIP_x	PULL_EN LP_SIP_x	PULL_EN LP_SIP_x			PULL_EN LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W			R/W
Reset	0	0	0	0	0			0

Bit	23	22	21	20	19	18	17	16
	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x	PULL_EN LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31,30,29,28,27 – PULL_EN LP_SIP_x (x = 4 to 0; Replace x=4 for bit 31, ... x=0 for bit 27) Pull Enable Register
The LP_SIP_x pins are connected to internal SPI Flash which holds the application code. It is not recommended to use these pins as general purpose I/O.

Bits 24,23,22,21,20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – PULL_EN LP_GPIO_x (x = 24 to 0; Replace x=24 for bit 24, ... x=0 for bit 0) Pull Enable Register
Writing '1'disables the internal Pull up/Pull down on LP_GPIO_x pin.
Writing '0' enables the internal Pull up/Pull down on LP_GPIO_x pin. It is an active LOW pull enable configuration.

12.11.14 LP_GPIO_x Pull Type Set

Name: RTYPE_PAD_0
Reset: 0x00000000

Absolute Address: 0x4000B054

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to set the pull type to pull up or pull down for LP_GPIO_x pins.

Bit	31	30	29	28	27	26	25	24
								PULL_TYPE LP_GPIO_x
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x	PULL_TYPE LP_GPIO_x
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 24,23,22,21,20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0 – PULL_TYPE LP_GPIO_x (x = 24 to 0; Replace x=24 for bit 24, ... x=0 for bit 0) Pull Type Set Register
Writing '1' enables the internal pull down on LP_GPIO_x pin.
Writing '0' enables the internal pull up on LP_GPIO_x pin.

12.11.15 GPIO_MSy Pull Configuration

Name: MS_GPIO
Reset: 0xF0

Absolute Address: 0x4000B414

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to set and enable the internal pull up or pull down for GPIO_MSy pins.

Bit	7	6	5	4	3	2	1	0
	PULL_EN GPIO_MSy	PULL_EN GPIO_MSy	PULL_EN GPIO_MSy	PULL_EN GPIO_MSy	PULL_TYPE GPIO_MSy	PULL_TYPE GPIO_MSy	PULL_TYPE GPIO_MSy	PULL_TYPE GPIO_MSy
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	0	0	0	0

Bits 7,6,5,4 – PULL_EN GPIO_MSy (y = 1 to 4; Replace y=1 for bit7, ... y=4 for bit4)
Writing '1'disables the internal pull up/pull down on GPIO_MSy pin.
Writing '0' enables the internal pull up/pull down on GPIO_MSy pin. It is an active LOW pull enable configuration.

Bits 3,2,1,0 – PULL_TYPE GPIO_MSy (y = 1 to 4; Replace y=1 for bit3, ... y=4 for bit0)
Writing '1' enables the internal pull down on GPIO_MSy pin.
Writing '0' disables the internal pull up on GPIO_MSy pin.

12.11.16 AO_GPIO_z Pull Enable

Name: AON_PULL_ENABLE
Reset: 0x00

Absolute Address: 0x4000F014

This register is a part of the AON_GP_REGS0 Registers. This register allows the user to enable the internal pull up for AO_GPIO_z pins.

Bit	7	6	5	4	3	2	1	0
						PULLUP_EN AO_GPIO_z	PULLUP_EN AO_GPIO_z	PULLUP_EN AO_GPIO_z
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2,1,0 – PULLUP_EN AO_GPIO_z (z = 2 to 0; Replace z=2 for bit 2, ... z=0 for bit0)
Writing '1'disables the internal pull up on AO_GPIO_z pin.
Writing '0' enables the internal pull up on AO_GPIO_z pin. It is an active LOW pull enable configuration

12.11.17 WakeUp Control on AO_GPIO_z

Name: GPIO_WAKEUP_CTRL
Reset: 0x02

Absolute Address: 0x4000E000

This register is a part of the AON_PWR_SEQ0 registers. This register allows the user to enable the AO_GPIO_z to wake-up BLE subsystem from the ULP mode.

Bit	7	6	5	4	3	2	1	0
							BLE_ENABLE	
Access							R/W	
Reset							1	

Bit 1 – BLE_ENABLE Wake-up BLE subsystem on AO_GPIO_z (z = 0,1,2)
Writing '0' to this bit disables the wake-up of BLE subsystem on AO_GPIO_z.
Writing '1' to this bit enables the wake-up of BLE subsystem on AO_GPIO_z. This configuration is common for all AO_GPIO_z pins.
For more details on wake-up procedure, see Wake-up Source.

13. Always-On (AON) Sleep Timer

This timer is a 32-bit countdown timer that operates on the 32 kHz sleep clock. It can be used as a general-purpose timer for the ARM or as a wake-up source for the chip.

13.1 Features

The following are the AON Sleep Timer features:

- 32 bit decrement counter operation
- Supported modes:
  - Single Count mode
  - Reload mode
- Wake-up source for ARM and BLE subsystem

13.2 Clock Configuration

The AON Sleep Timer must be provided with clock source before configuring for normal operation. For more details on configuration, see AON Sleep Timer Clock Configuration.

13.3 Functional Description

13.3.1 Initialization

After device reset, the AON Sleep Timer is not active.

13.3.2 Operation

The Always-On (AON) Sleep Timer generates a timer tick at a programmed interval. The counter decrements at the frequency of the P_CLK (32.768 kHz is configured) clock signal. For more details see, Clock Settings for Critical Sections. When the counter reaches zero, a tick is generated and interrupt is triggered.

There are two different modes; single count mode and reload mode. The following are the steps to initialize and enable the AON Sleep Timer.

- Counter is decremented for each P_CLK. Calculate the counter value for desired interval (ms) using the following formula:
  SINGLE_COUNT_DURATION = Interval (ms) * P_CLK (kHz)
- Load the counter value in SINGLE_COUNT_DURATION register.
- Configure the interrupt vector line to connect the interrupt request to NVIC. To use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. For more information, see Nested Vector Interrupt Controller.
- To enable timer with the Single Count mode, set SINGLE_COUNT_ENABLE bit in CONTROL register.
- To enable timer with the Reload mode, set RELOAD_ENABLE bit in CONTROL register. This loads the CURRENT_COUNT_VALUE register with the value set in SINGLE_COUNT_DURATION register. It is required to wait for reload delay (Reload mode) or single count delay (Single Count mode) to ensure that the timer is loaded with the set counter value.

- The CURRENT_COUNT_VALUE register starts decrementing and when it reaches zero, AON Sleep Timer interrupt is triggered. Interrupt request can be cleared by writing '1' to IRQ_CLEAR bit of CONTROL register.
- In Single Count mode, the AON Sleep Timer counter is not reloaded again and timer stops. In Reload mode AON Sleep Timer is again reloaded with SINGLE_COUNT_DURATION and timer starts decrementing again.

13.3.3 Example

The following is the sample example to enable the AON Sleep Timer in Reload mode with interval of one second:

```
/* Register ISR handler as explained in Section 6.3  Nested Vector Interrupt Controller */
/* Calculate SINGLE_COUNT_DURATION = 1000 (ms) * 32.768(kHz) = 32768 */
AON_SLEEP_TIMER0->SINGLE_COUNT_DURATION.reg = 32768;
/* Enable Reload mode */
AON_SLEEP_TIMER0->CONTROL.reg = (1<<0);
/* Wait for reload delay */
       while ((AON_SLEEP_TIMER0->CONTROL.reg & (2<<8) !=(2<<8))
/* Enable NVIC interrupt as explained in Section 6.3  Nested Vector Interrupt Controller */
```

13.4 Restart the Running AON Sleep Timer

The AON Sleep Timer is started in Single Count mode or Reload mode by configuring CONTROL register as explained in Operation. Once the timer starts decrementing the CURRENT_COUNT_VALUE, to stop the counter, SINGLE_COUNT_ENABLE or RELOAD_ENABLE bit must be cleared. Even after clearing the bits, CURRENT_COUNT_VALUE register is not cleared immediately. It continues to decrement till it reaches zero, then the AON Sleep Timer stops.

Therefore, while restarting the timer again with new value loaded on SINGLE_COUNT_DURATION gets reloaded on CURRENT_COUNT_VALUE only when the current counter reaches zero.

AON Sleep Timer must be reset to clear the CURRENT_COUNT_VALUE to zero immediately. The AON Sleep Timer can be reset by clearing the SLEEP_TIMER_RSTN bit of AON_GLOBAL_RESET register. This procedure resets all the AON Sleep Timer registers to default values. For more details, see AON Sleep Timer Reset.

The AON Sleep Timer must be disabled before it is reset in order to avoid undefined behavior.

13.5 Wake-up Source

The AON Sleep Timer is also a wake-up source for the ARM subsystem and BLE subsystem to wake-up from the ULP mode after the set interval. Enabling the AON Sleep Timer as wake-up source can be done using the AON_ST_WAKEUP_CTRL register. By default, on reset the AON Sleep Timer as wake-up source is disabled.

Note: It is recommended to enable wake-up of ARM subsystem and BLE subsystem together or wake-up of only the ARM subsystem. Enabling only the wake-up of the BLE subsystem should not be performed.

13.6 Power Management

As AON Timer belongs to the Always On power domain, the configuration registers are intact when device goes to the ULP mode. For more details on how sleep and wake-up are controlled by ROM firmware, see ATSAMB11 BluSDK Smart Interrupts and ULP Architecture.

13.7 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000D000	AON_SLEEP_TIMER0	CONTROL	7:0				IRQ_CLEAR			SINGLE_COUNT_ENABLE	RELOAD_ENABLE
			15:8		SLP_TIMER_SINGLE_COUNT_ENABLE_DLY [2:0]					SLP_TIMER_CLK_RELOAD_DLY[1:0]	
			23:16								
			31:24	SLEEP_TIMER_NOT_ACTIVE	SLEEP_TIMER_ACTIVE						
0x4000D004	AON_SLEEP_TIMER0	SINGLE_COUNT_DURATION	7:0	COUNT_DURATION[7:0]							
			15:8	COUNT_DURATION[15:8]							
			23:16	COUNT_DURATION[23:16]							
			31:24	COUNT_DURATION[31:24]							
0x4000D00C	AON_SLEEP_TIMER0	CURRENT_COUNT_VALUE	7:0	COUNT[7:0]							
			15:8	COUNT[15:8]							
			23:16	COUNT[23:16]							
			31:24	COUNT[31:24]							
0x4000E00C	AON_PWR_SEQ0	AON_ST_WAKEUP_CTRL	7:0							BLE_ENABLE	ARM_ENABLE

13.8 Register Description

13.8.1 AON Sleep Timer Control

Name: CONTROL
Reset: 0x80000000

Absolute Address: 0x4000D000

This register is a part of AON_SLEEP_TIMER0 Registers. This register allows the user to configure and enable AON Sleep Timer in Single Count mode or Reload mode.

Bit	31	30	29	28	27	26	25	24
	SLEEP_TIMER_NOT_ACTIVE	SLEEP_TIMER_ACTIVE						
Access	R	R						
Reset	1	0						

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2:0]					SLP_TIMER_CLK_RELOAD_DLY[1:0]	
Access		R	R	R			R	R
Reset		0	0	0			0	0

Bit	7	6	5	4	3	2	1	0
				IRQ_CLEAR			SINGLE_COUNT_ENABLE	RELOAD_ENABLE
Access				R/W			R/W	R/W
Reset				0			0	0

Bit 31 – SLEEP_TIMER_NOT_ACTIVE
Read value '1', indicates that the CURRENT_COUNT_VALUE is 0 and AON Sleep Timer is not active.

Bit 30 – SLEEP_TIMER_ACTIVE
Read value '1', indicates that the CURRENT_COUNT_VALUE value is not 0 and AON Sleep Timer is running.

Bits 14:12 – SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2:0]
This provides the current status of the Single Count mode. Before setting or clearing SINGLE_COUNT_ENABLE bit the SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2] bit must be checked.

Read Value of SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2]	Description
0	If SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2] is 0, which means the SINGLE_COUNT_ENABLE bit is cleared and can write SINGLE_COUNT_ENABLE bit to a 1 when needed
1	If SLP_TIMER_SINGLE_COUNT_ENABLE_DLY[2] is 1, which means the SINGLE_COUNT_ENABLE bit is set and can write SINGLE_COUNT_ENABLE bit to a 0 when needed

Bits 9:8 – SLP_TIMER_CLK_RELOAD_DLY[1:0]
This provides the current status of the reload mode. Before setting or clearing RELOAD_ENABLE bit the SLP_TIMER_CLK_RELOAD_DLY[1] bit must be checked.

Read Value of SLP_TIMER_CLK_RELOAD_DLY[1]	Description
0	If SLP_TIMER_CLK_RELOAD_DLY[1] bit is 0, which means the RELOAD_ENABLE bit is cleared and it is safe to write RELOAD_ENABLE bit to a 1 when needed
1	If SLP_TIMER_CLK_RELOAD_DLY[1] bit is 1, which means the RELOAD_ENABLE bit is set and it is safe to write RELOAD_ENABLE bit to a 0 when needed

Bit 4 – IRQ_CLEAR
Writing '0' to a bit has no effect.
Writing '1' to a bit will clear the AON Sleep Timer interrupt request.

Bit 1 – SINGLE_COUNT_ENABLE
Writing '0' to a bit disables AON Sleep Timer in Single Count mode.
Writing '1' to a bit enables AON Sleep Timer in Single Count mode and loads the AON Sleep Timer with the SINGLE_COUNT_DURATION value.

Bit 0 – RELOAD_ENABLE
Writing '0' to a bit disables AON Sleep Timer in Reload mode.
Writing '1' to a bit enable AON Sleep Timer in Reload mode and loads the AON Sleep Timer with the SINGLE_COUNT_DURATION value whenever the timer expires.

13.8.2 AON Sleep Timer Duration

Name: SINGLE_COUNT_DURATION
Reset: 0x00000000

Absolute Address: 0x4000D004

This register is a part of AON_SLEEP_TIMER0 Registers. This register allows the user to load AON Sleep Timer duration counter value for Single Count mode or Reload mode.

Bit	31	30	29	28	27	26	25	24
	COUNT_DURATION[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COUNT_DURATION[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COUNT_DURATION[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT_DURATION[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – COUNT_DURATION[31:0]
These bits hold the counter value equivalent to timer interval that is loaded into the CURRENT_COUNT_VALUE register. For more details on formula, see Operation.

13.8.3 AON Sleep Timer Current Counter Value

Name: CURRENT_COUNT_VALUE
Reset: 0x00000000

Absolute Address: 0x4000D00C

This register is a part of the AON_SLEEP_TIMER0 Registers. This register contains the AON Sleep Timer current counter value of Single Count mode or Reload mode. This starts decrementing and when it reaches zero AON Sleep Timer interrupt is triggered.

Bit	31	30	29	28	27	26	25	24
	COUNT[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COUNT[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – COUNT[31:0]
These bits contain the current counter value.

13.8.4 WakeUp Control on AON Sleep Timer

Name: AON_ST_WAKEUP_CTRL
Reset: 0x00

Absolute Address: 0x4000E00C

This register is a part of the AON_PWR_SEQ0 Registers. This register allows the user to enable the AON Sleep Timer to wake-up ARM subsystem and BLE subsystem from the ULP mode.

Bit	7	6	5	4	3	2	1	0
							BLE_ENABLE	ARM_ENABLE
Access							R/W	R/W
Reset							0	0

Bit 1 – BLE_ENABLE
Writing '0' disables BLE subsystem wake-up by AON Sleep Timer.
Writing '1' enables BLE subsystem wake-up by AON Sleep Timer.

Bit 0 – ARM_ENABLE
Writing '0' disables ARM subsystem wake-up by AON Sleep Timer.
Writing '1' enables ARM subsystem wake-up by AON Sleep Timer.

14. Pulse Width Modulation

The Pulse Width Modulation (PWM) feature provides a way to generate a periodic pulse waveform. The ATSAMB11 contains four individually configurable PWM blocks to provide external control voltages.

14.1 Features

- Four individually configurable PWM blocks
- Automatic generation of PWM output by hardware without software intervention
- Two PWM modes of operation
- Configurable period and duty cycle of PWM output
- Four different selectable base frequencies for PWM (26 MHz, 13 MHz, 6.5 MHz, 3.25 MHz)
- Configurable PWM output on LP_GPIO_x pins via MEGAMUX options

14.2 Clock Configuration

Before configuring the PWM registers, reset the PWM core by setting PWMx_ RSTN bit in LPMCU_GLOBAL_RESET_0 register. The PWM peripheral must be provided with clock source before configuring for normal operation. For more details on configuration, see Peripheral Clock Configuration.

14.3 Functional Description

14.3.1 Initialization

After device reset, PWM registers are zero.

PWM waveform can be output on any of LP_GPIO_x pins configured through MEGAMUX options.

- To get output of PWM waveform on specific LP_GPIO_x pin, configure PINMUX_SEL[2:0] of specific pin on PINMUX_SEL_n register with value equal to 1. This configuration enables MEGAMUXing on this pin.
- See the MEGAMUX Options table and configure the MEGA_MUX_IO_SEL_n register with PWM option.
- Select the PWM module clock source by configuring the CLOCK_SEL[1:0] bits of PWMn_CTRL register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz and 3 MHz.
- There are two modes of operation that are selectable through the PWM_MODE_SEL bit of the PWMn_CTRL register.
  - PWM Mode 1–Clear PWM_MODE_SEL bit
  - PWM Mode 2–Set PWM_MODE_SEL bit

14.3.2 PWM Mode 1 Operation

The PWM Mode 1 operation is based on 15-bit timer with compare match logic. The PWM period (T) is controlled by PWM_PERIOD [3:0] and the duty cycle is controlled by AGCDATA_IN[9:0] bits of PWMn_CTRL register. The timer TOP value is set based on PWM_PERIOD[3:0] and SAMPLE_METHOD as shown in the following table. When the PWM is enabled by writing 1’ to PWM_EN bit of PWMn_CTRL register, the timer counter is incremented for every clock cycle. When up-counting, the PWM output on LP_GPIO_x is set when the masked value of TOP value with timer counter becomes ZERO. The PWM

output is cleared when the timer counter reaches CC value as per the table. This PWM output polarity on LP_GPIO_x is reversed when OUTPUT_POLARITY bit of PWMn_CTRL register is set.

14.3.2.1 Method of Operation in PWM Mode 1

There are three methods of operation in PWM Mode 1:

1. USE_AGCUPDATE=1 and AGCUPDATE=1
2. USE_AGCUPDATE=0 , AGCUPDATE=0 and SAMPLE_METHOD=1
3. USE_AGCUPDATE=0 , AGCUPDATE=0 and SAMPLE_METHOD=0

Table 14-1. PWM Mode 1 Functional Values

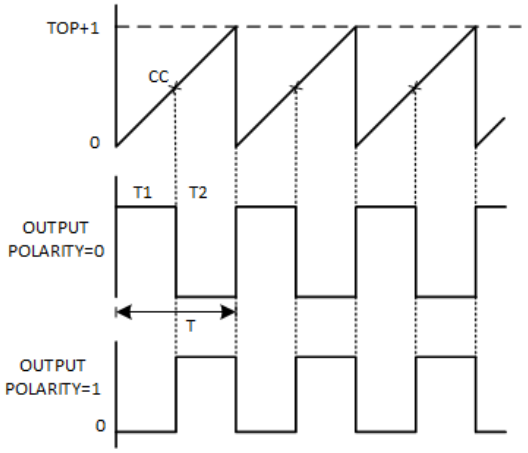
PWM_PERIOD[3:0]	CC	SHIFTED_OUT			TOP			MAX		
		SAMPLE_METHOD=0	SAMPLE_METHOD=1	USE_AGCUPDATE = 1, AGC_UPDATE=1	SAMPLE_METHOD=0	SAMPLE_METHOD=1	USE_AGCUPDATE = 1, AGC_UPDATE=1	SAMPLE_METHOD=0	SAMPLE_METHOD=1	USE_AGCUPDATE = 1, AGC_UPDATE=1
0	AGCDATA_IN_POST_INVERT[9:0]>>4	AGCDATA_IN_POST_INVERT[3:0]	0	0	0x3F	0x3F	0x3F	0x3FF	0x3F	0x3F
1	AGCDATA_IN_POST_INVERT[9:0]>>3	AGCDATA_IN_POST_INVERT[2:0]	0	0	0x7F	0x7F	0x7F	0x3FF	0x7F	0x7F
2	AGCDATA_IN_POST_INVERT[9:0]>>2	AGCDATA_IN_POST_INVERT[1:0]	0	0	0xFF	0xFF	0xFF	0x3FF	0xFF	0xFF
3	AGCDATA_IN_POST_INVERT[9:0]>>1	AGCDATA_IN_POST_INVERT[0:0]	0	0	0x1FF	0x1FF	0x1FF	0x3FF	0x1FF	0x1FF
4	AGCDATA_IN_POST_INVERT[9:0]>>0	0	0	0	0x3FF	0x3FF	0x3FF	0x3FF	0x3FF	0x3FF
5	AGCDATA_IN_POST_INVERT[9:0]<<1	0	0	0	0x7FF	0x7FF	0x7FF	0x3FF	0x7FF	0x7FF
6	AGCDATA_IN_POST_INVERT[9:0]<<2	0	0	0	0xFFF	0xFFF	0xFFF	0x3FF	0xFFF	0xFFF
7	AGCDATA_IN_POST_INVERT[9:0]<<3	0	0	0	0x1FFF	0x1FFF	0x1FFF	0x3FF	0x1FFF	0x1FFF
8	AGCDATA_IN_POST_INVERT[9:0]<<4	0	0	0	0x3FFF	0x3FFF	0x3FFF	0x3FF	0x3FFF	0x3FFF
Others	AGCDATA_IN_POST_INVERT[9:0]>>0	0	0	0	0x3FF	0x3FF	0x3FF	0x3FF	0x3FF	0x3FF

Note: The AGCDATA_FMT bit of PWMn_CTRL register decides the sign of AGCDATA_IN[9:0] data. If AGCDATA_FMT is '1' then AGCDATA_IN is unsigned data and if AGCDATA_FMT is '0' then AGCDATA_IN is signed data.
AGCDATA_IN_POST_INVERT[9:0] = {AGCDATA_IN[9] ^ (~AGCDATA_FMT), AGCDATA_IN[8:0]}

14.3.2.1.1 USE_AGCUPDATE=1 and AGCUPDATE=1

If USE_AGCUPDATE bit is '1', then the AGCUPDATE bit must be '1' for PWM pulse output. The TOP and MAX values are as per PWM Mode 1 Functional Values under USE_AGCUPDATE =1, AGCUPDATE=1. The TOP and MAX values are same in this method. The CC value is the shifted value of AGCDATA_IN[9:0]. The SHIFTED_OUT bits of AGCDATA_IN[9:0] are lost in this method. Hence, there PWM pulse output is not generated for AGCDATA_IN[9:0] values 0x1 to 0xF.

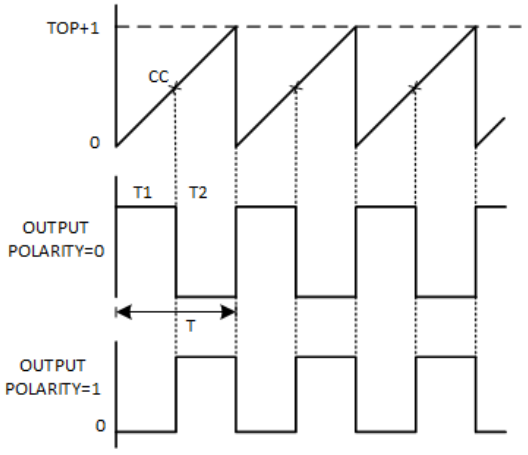
Figure 14-1. PWM Mode1 Operation USE_AGCUPDATE=0 , AGCUPDATE=0 and SAMPLE_METHOD=1



14.3.2.1.2 USE_AGCUPDATE=0 , AGCUPDATE=0 and SAMPLE_METHOD=1

If USE_AGCUPDATE is '0', then PWM pulse output is changed based on SAMPLE_METHOD bit. Refer the TOP and MAX in table under SAMPLE_METHOD=1. The operation of SAMPLE_METHOD=1 is same as the first method.

Figure 14-2. PWM Mode1 Operation USE_AGCUPDATE =1 and AGCUPATE=1



14.3.2.1.3 USE_AGCUPDATE=0 , AGCUPDATE=0 and SAMPLE_METHOD=0

If USE_AGCUPDATE is '0' and SAMPLE_METHOD=0, then, the TOP and MAX values are referred under SAMPLE_METHOD=0 in table. The TOP and MAX values are not same in this method. In this method, the SHIFTED_OUT bits of AGCDATA_IN adds extra cycle in the PWM pulse stream. The SHIFTED_OUT value is getting added with EXTRA_CYCLE[3:0] every time the counter reaches TOP

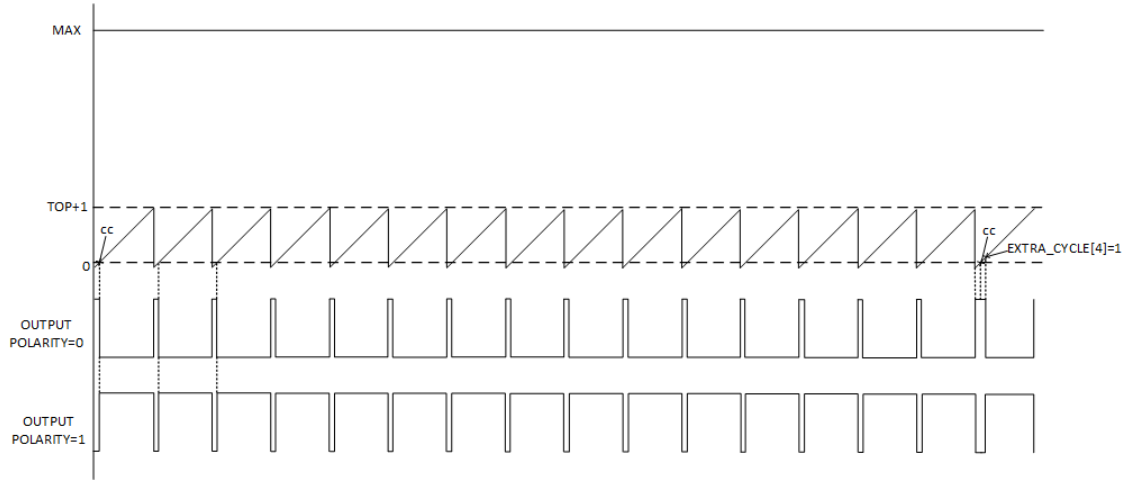
value as per the following equation. An extra pulse is generated when the bit ‘4’ of EXTRA_CYCLE[4:0] is set.

EXTRA_CYCLE[4:0] = SHITED_OUT + EXTRA_CYCLE[3:0]

Example1: The PWM pulse output for AGCDATA_IN[9:0] = 0x11 and PWM_PERIOD[3:0]=0x00 is shown in the following figure. When upcounting, the PWM output on LP_GPIO_x is set when the masked value of TOP (TOP = 0x3F) value with timer counter becomes ZERO. The PWM output is cleared when the timer counter reaches CC (CC = 0x01). An extra cycle is added in the pulse stream when EXTRA_CYCLE[4:0] bit '4' is set. This is set when timer counter reaches 0x3CO for this AGCDATA_IN[9:0] = 0x11 as shown in the following figure.

Figure 14-3. PWM Mode1 Operation SAMPLE_METHOD=0

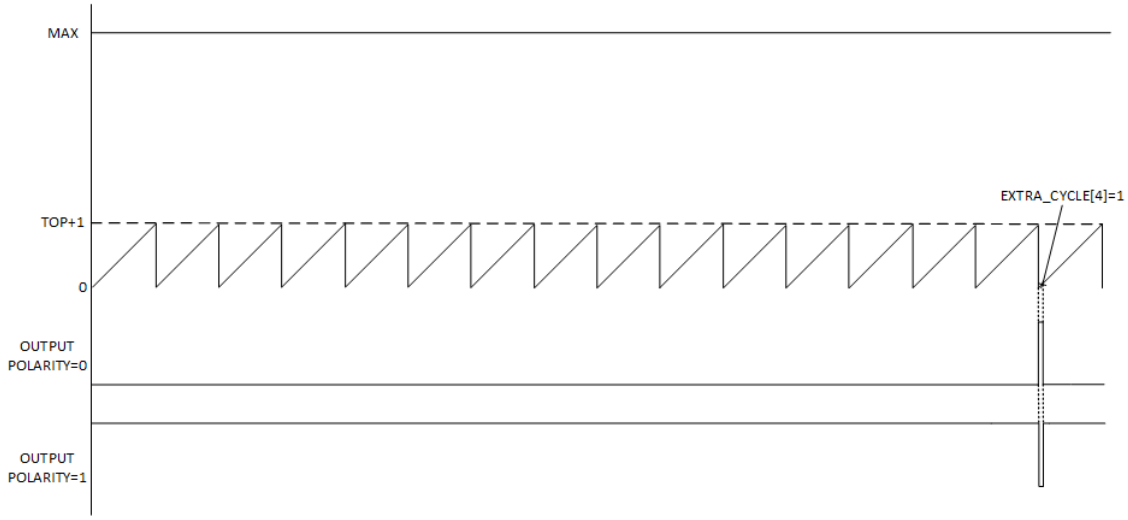
Example: AGCDATA_IN[9:0] = 0x11 and PWM_PERIOD[3:0] = 0



Example2: The PWM pulse output for AGCDATA_IN[9:0] = 0x01 and PWM_PERIOD[3:0]=0x00 is shown in the following figure. As CC=0x00, there is no pulse stream, but an extra cycle is added in the pulse stream when EXTRA_CYCLE[4:0] bit '4' is set. The extra pulse is added when timer counter reaches 0x3CO for this AGCDATA_IN[9:0] = 0x01 as shown in the following figure.

Figure 14-4. Figure 14-3. PWM Mode1 Operation SAMPLE_METHOD=0

Example: AGCDATA_IN[9:0] = 0x01 and PWM_PERIOD[3:0] = 0



14.3.2.2 PWM Pulse Frequency

The clock input to PWM module can be selected through CLOCK_SEL[1:0] bits of PWMn_CTRL register. Four different clock inputs are possible. The PWM pulse frequency (f_{PWM}) of the PWM block is derived from selected input clock frequency (f_{PWM_base}) using the following formula.

When, PWM_PERIOD[3:0] = 0,1,2, … , 8

$$f_{PWM}(\text{Mode 1}) = \frac{f_{PWM\_base}}{64*2^{\text{PWM\_PERIOD[3:0]}}}$$

When, PWM_PERIOD[3:0] > 8

$$f_{PWM}(\text{Mode 1}) = \frac{f_{PWM\_base}}{64*2\text{^}4}$$

f_{PWM_base} can be selected to have different values according to Table 14-2. The minimum and maximum frequencies supported for each clock selection are also listed in the table.

Table 14-2. f_{PWM} Range for Different f_{PWM} Base Frequencies

f_{PWM_base}	f_{PWM} max.	f_{PWM} min.
26 MHz	406.25 kHz	1.586 kHz
13 MHz	203.125 kHz	793.25 Hz
6.5 MHz	101.562 kHz	396.72 Hz
3.25 MHz	50.781 kHz	198.36 Hz

14.3.2.3 PWM Duty Cycle

The duty cycle is configured through AGCDATA_IN[9:0] bits. The duty cycle is calculated using the following formula:

$$\text{Duty cycle (Mode1)} = \frac{\text{CCx100}}{\text{TOP+1}}$$

14.3.2.4 Enabling and Disabling PWM Mode 1 Operation

After configuring PWMn_CTRL register with desired duty cycle and period the PWM output on specific LP_GPIO_x is enabled by writing a ‘1’ to PWM_EN bit of PWMn_CTRL register.

14.3.2.5 Updating New Duty Cycle and Period for Mode 1

USE_AGCUPDATE and AGCUPDATE decide the update new AGCDATA_IN[9:0] value and output the PWM pulse accordingly. If USE_AGCUPDATE is ‘1’ then only when AGCUPDATE bit is ‘1’ the AGCDATA_IN[9:0] is loaded into internal registers. The PWM output is available on LP_GPIO_x pin after PWM_EN is set. If AGCUPDATE bit is ‘0’ then PWM pulse is not output.

If USE_AGCUPDATE is ‘0’ then the internal PWM register is updated with new values when the timer counter reaches MAX value in SAMPLE_METHOD =1 and SAMPLE_METHOD =0.

14.3.2.6 Example

The following is the sample example to enable PWM0 output on LP_GPIO_10 pin with pulse frequency = 406.25 kHz, Duty cycle = 75% using PWM Mode 1 USE_AGCUPDATE=1 and AGCUPDATE=1 method:

```
/* Write PINMUX_SEL_1 bits [10:8] with 0x1 (MUX1) to enable MEGAMUX option */
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x1<<8);
/* MEGAMUX option value for PWM0 from Table 10-2. MEGAMUX Options is 0x0C */
/* Assign 0x0C MEGAMUX value for LP_GPIO_10 specific bits [20:16] */
LPMCU_MISC_REGS0->MEGA_MUX_IO_SEL_2.reg |= (0x0C<<16);
/* Select clock source for PWM0 as 26MHz */
```

```
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x00<< 21);
/* Write '0' to PWM_MODE_SEL for Mode 1 operation */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x0<< 4);
/* Calculate PWM_PERIOD value: For pulse frequency fPWM_base = 26MHz, fPMW = 406.25kHz then
PWM_PERIOD[3:0] = 0 */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x00<< 5);
/* Write USE_AGCUPDATE to '1' */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 19);
/* Write AGC_UPDATE to '1' */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 20);
/* Calculate AGCDATA_IN[9:0] value from duty cycle
                Duty cycle = CC*100/(TOP+1)
                        CC = AGCDATA_IN_POST_INVERT[9:0]>>4 from Table 14.1 for PWM_PERIOD
=0
                     TOP = 0x3F from Table 14.1 for PWM_PERIOD = 0
                Duty Cycle = 75%
                        CC = (75*(0x3F+1))/100 = 0x30
AGCDATA_IN_POST_INVERT[9:0] = 0x30<<4 = 0x300
AGCDATA_IN_POST_INVERT[9:0] = {AGCDATA_IN[9] ^ (~AGCDATA_FMT), AGCDATA_IN[8:0]}
For AGCDATA_FMT = 1,
AGCDATA_IN = 0x300 */
/* Write AGCDATA_IN */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x300<< 9);
/* Write AGCDATA_FMT to '1' */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 2);
/* Enable PWM output */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 0);
```

14.3.3 PWM Mode 2 Operation

The PWM Mode 2 operation is based on accumulator logic. Both the PWM period (T) and duty cycle are controlled by AGCDATA_IN[9:0] bits of PWMn_CTRL register. The PWM Mode 2 operation is selected by setting PWM_MODE_SEL bit of PWMn_CTRL register. When AGCUPDATE bit is '1' the AGCDATA_IN_POST_INVERT[9:0] is loaded into internal register. Enable PWM pulse output on specific LP_GPIO_x by writing a '1' to PWM_EN bit of PWMn_CTRL register. When AGCUPDATE bit is '0', on every clock cycle (clock input selected using CLOCK_SEL[2:0] bits of PWMn_CTRL register) accumulation happens as per below calculation:

AGC_ACC[10:0] = AGCDATA_IN_POST_INVERT[9:0] + AGC_ACC[9:0]

The PWM output on LP_GPIO_x is set whenever AGC_ACC[10] is '1' and cleared when AGC_ACC[10] is '0'. This PWM output polarity on LP_GPIO_x is reversed when OUTPUT_POLARITY bit of PWMn_CTRL register is set.

14.3.3.1 PWM Pulse Frequency

The clock input to PWM module can be selected through CLOCK_SEL[1:0] bits of PWMn_CTRL register. Four different clock inputs are possible. The PWM pulse frequency (f_{PWM}) of the PWM block is derived from selected input clock frequency (f_{PWM_base}) using following formula.

When, AGCDATA_IN_POST_INVERT[9:0] <= 512:

$$f_{PWM}(\text{Mode 2}) = f_{PWM\_base} * \text{AGCDATA\_IN\_POST\_INVERT}[9:0]/1024$$

When, AGCDATA_IN_POST_INVERT[9:0] > 512:

$$f_{PWM}(\text{Mode 2}) = f_{PWM\_base} * (1024 - \text{AGCDATA\_IN\_POST\_INVERT}[9:0])/1024$$

Note: The AGCDATA_FMT bit of PWMn_CTRL register decides the sign of AGCDATA_IN[9:0] data. If AGCDATA_FMT is '1' then AGCDATA_IN is unsigned data and if AGCDATA_FMT is '0' then AGCDATA_IN is signed data.
AGCDATA_IN_POST_INVERT[9:0] = {AGCDATA_IN[9] ^ (~AGCDATA_FMT), AGCDATA_IN[8:0]}

14.3.3.2 PWM Duty Cycle

The duty cycle is configured through AGCDATA_IN[9:0] bits. The duty cycle is calculated using the following formula:

$$\text{Duty cycle (Mode2)} = \frac{\text{AGCDATA\_IN\_POST\_INVERT[9:0]} \times 100}{1024}$$

Note: The AGCDATA_FMT bit of the PWMn_CTRL register decides the sign of AGCDATA_IN[9:0] data. If AGCDATA_FMT is '1' then AGCDATA_IN is unsigned data and if AGCDATA_FMT is '0' then AGCDATA_IN is signed data.
AGCDATA_IN_POST_INVERT[9:0] = {AGCDATA_IN[9] ^ (~AGCDATA_FMT), AGCDATA_IN[8:0]}

14.3.3.3 Enabling and Disabling PWM Mode 2 Operation

After configuring PWMn_CTRL register with desired duty cycle and period the PWM output on specific LP_GPIO_x is enabled by writing a '1' to PWM_EN bit of PWMn_CTRL register.

14.3.3.4 Updating New Duty Cycle and Period

The AGCUPDATE bit decides the update of new AGCDATA_IN[9:0] and output of the PWM pulse accordingly. If AGCUPDATE bit is '1', the AGCDATA_IN[9:0] will be loaded into internal register. If AGCUPDATE bit is '0', the AGC accumulation as per new AGCDATA_IN[9:0] value starts and PWM pulse is available on LP_GPIO_x pin when PWM_EN is bit is set. The USE_AGCUPDATE bit is not used for PWM Mode 2 operation.

14.3.3.5 Example

The following is the sample example to enable PWM0 output on LP_GPIO_10 pin with Duty cycle = 75% using PWM Mode 2:

```
/* Write PINMUX_SEL_1 bits [10:8] with 0x1 (MUX1) to enable MEGAMUX option */
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x1<<8);
/* MEGAMUX option value for PWM0 from Table 10-2. MEGAMUX Options is 0x0C */
/* Assign 0x0C MEGAMUX value for LP_GPIO_10 specific bits [20:16] */
LPMCU_MISC_REGS0->MEGA_MUX_IO_SEL_2.reg |= (0x0C<<16);
/* Select clock source for PWM0 as 26MHz */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x00<< 21);
/* Write '1' to PWM_MODE_SEL for Mode 2 operation */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= ((0x1<< 4);
/* Calculate AGCDATA_IN[9:0] value from duty cycle = 75%
 Duty cycle = (AGCDATA_IN_POST_INVERT[9:0] /1024)*100
 AGCDATA_IN_POST_INVERT[9:0] = (75*1024)/100 = 0x300
AGCDATA_IN_POST_INVERT[9:0] = {AGCDATA_IN[9] ^ (~AGCDATA_FMT), AGCDATA_IN[8:0]}
For AGCDATA_FMT = 1,
AGCDATA_IN = 0x300 */
/* Write AGCDATA_IN */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x300<< 9);
/* Write AGCDATA_FMT to '1' */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 2);
/* The expected PWM pulse frequency for fPWM_base = 26MHz, and  AGCDATA_IN = 0x300
When, AGCDATA_IN_POST_INVERT[9:0] > 512:
fPWM = fPWM_base* (1024 - AGCDATA_IN_POST_INVERT[9:0])/1024
= 6.5MHz
/* PWM_PERIOD[3:0] bits is not used in this PWM mode 2 hence write 0x00 */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x00<< 5);
/* Write '1' to AGCUPDATE bit */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |=  (0x1<<20);

/* Enable PWM output */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |= (0x1<< 0);
/* Write '0' to AGCUPDATE bit */
LPMCU_MISC_REGS0->PWM0_CTRL.reg |=  (0x0<<20);
```

14.4 Power Management

If the system goes to the Ultra-Low Power mode, the PWM peripheral is shut down and the PWM pulse output is stopped. The PWM configuration registers lose its content, and can not be restored when powered-up again. Therefore, the user must reconfigure the PWM peripheral at power-up to ensure it is in a well-defined state before use. For details on reconfiguration, refer to ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User Guide. This document also explains how sleep and wake-up are controlled.

14.5 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B160	LPMCU_MISC_REGS0 Registers	PWM0_CTRL	7:0	PWM_PERIOD[2:0]	PWM_MODE_SEL			SAMPLE_METHOD	AGCDATA_FMT	OUTPUT_POLARITY	PWM_EN
			15:8	AGCDATA_IN[6:0]							PWM_PERIOD[3]
			23:16		CLOCK_SEL[1:0]		AGC_UPDATE	USE_AGCUPDATE	AGCDATA_IN[9:7]		
0x4000B164	LPMCU_MISC_REGS0 Registers	PWM1_CTRL	7:0	PWM_PERIOD[2:0]	PWM_MODE_SEL			SAMPLE_METHOD	AGCDATA_FMT	OUTPUT_POLARITY	PWM_EN
			15:8	AGCDATA_IN[6:0]	PWM_PERIOD[3]						
			23:16		CLOCK_SEL[1:0]		AGC_UPDATE	USE_AGCUPDATE	AGCDATA_IN[9:7]		
0x4000B168	LPMCU_MISC_REGS0 Registers	PWM2_CTRL	7:0	PWM_PERIOD[2:0]	PWM_MODE_SEL			SAMPLE_METHOD	AGCDATA_FMT	OUTPUT_POLARITY	PWM_EN
			15:8	AGCDATA_IN[6:0]	PWM_PERIOD[3]						
			23:16		CLOCK_SEL[1:0]		AGC_UPDATE	USE_AGCUPDATE	AGCDATA_IN[9:7]		
0x4000B16C	LPMCU_MISC_REGS0 Registers	PWM3_CTRL	7:0	PWM_PERIOD[2:0]	PWM_MODE_SEL			SAMPLE_METHOD	AGCDATA_FMT	OUTPUT_POLARITY	PWM_EN
			15:8	AGCDATA_IN[6:0]	PWM_PERIOD[3]						
			23:16		CLOCK_SEL[1:0]		AGC_UPDATE	USE_AGCUPDATE	AGCDATA_IN[9:7]		
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0					MEGAMUX_SEL[5:0] LP_GPIO_0			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_1			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_2			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_3			
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0					MEGAMUX_SEL[5:0] LP_GPIO_4			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_5			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_6			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_7			
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0					MEGAMUX_SEL[5:0] LP_GPIO_8			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_9			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_10			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_11			
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0					MEGAMUX_SEL[5:0] LP_GPIO_12			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_13			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_14			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_15			
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0					MEGAMUX_SEL[5:0] LP_GPIO_16			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_17			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_18			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_19			
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0					MEGAMUX_SEL[5:0] LP_GPIO_20			
			15:8					MEGAMUX_SEL[5:0] LP_GPIO_21			
			23:16					MEGAMUX_SEL[5:0] LP_GPIO_22			
			31:24					MEGAMUX_SEL[5:0] LP_GPIO_23			
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0					MEGAMUX_SEL[5:0] LP_GPIO_24			

14.6 Register Description

14.6.1 PWM Control Register

Name: PWMn_CTRL
Reset: 0x000000

Absolute Address: 0x4000B160(PWM0),0x4000B164(PWM1),0x4000B168(PWM2), 0x4000B16C(PWM3)

This register is a part of the LPMCU_MISC_REGS0 Registers. This register allows the user to configure PWM functionality and enables the PWM output. There are four individual PWM blocks in SAMB11 and four PWMn_CTRL registers (n=0,1,2,3).

Bit	23	22	21	20	19	18	17	16
		CLOCK_SEL[1:0]		AGC_UPDATE	USE_AGCUPDATE	AGCDATA_IN[9:7]		
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	AGCDATA_IN[6:0]							PWM_PERIOD[3:3]
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PWM_PERIOD[2:0]			PWM_MODE_SEL	SAMPLE_METHOD	AGCDATA_FMT	OUTPUT_POLARITY	PWM_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 22:21 – CLOCK_SEL[1:0]
These bits select clock input for PWM.

CLOCK_SEL[1:0]	Description
0x0	26 MHz
0x1	13 MHz
0x2	6.5 MHz
0x3	3.25 MHz

Bit 20 – AGC_UPDATE
In combination with USE_AGCUPDATE, PWM_MODE_SEL, and SAMPLE_METHOD controls the loading of AGCDATA_IN into internal registers.

Bit 19 – USE_AGCUPDATE
In combination with AGC_UPDATE, and PWM_MODE_SEL controls the loading of AGCDATA_IN into internal registers.

PWM_MODE_SEL	USE_AGCUPDATE	AGC_UPDATE	SAMPLE_METHOD	Operation
0	1	1	X	AGCDATA_IN loaded into internal register
0	1	0	X	AGCDATA_IN is not loaded into internal register
0	0	X	1	The SHIFTED_OUT bits of AGCDATA_IN[9:0] are lost in this method
0	0	X	0	The SHIFTED_OUT bits of AGCDATA_IN adds extra cycle in the PWM pulse stream
1	X	0->1->0	X	AGCDATA_IN loaded into internal register when AGC_UPDATE is '1' and PWM output is seen after AGC_UPDATE is made as '0'

Bits 18:9 – AGCDATA_IN[9:0]
Configures the duty cycle.

Bits 8:5 – PWM_PERIOD[3:0]
Configures the period of PWM pulse frequency for PWM Mode 1 operation. For more details on formula, see PWM Pulse Frequency.

Bit 4 – PWM_MODE_SEL
Selects the two different PWM modes of operations.

PWM_MODE_SEL	Mode
0	Mode 1
1	Mode 2

Bit 3 – SAMPLE_METHOD

SAMPLE_METHOD	Description
0	SHIFTED_OUT bits of AGCDATA_IN adds extra cycle in the PWM pulse stream
1	The SHIFTED_OUT bits of AGCDATA_IN[9:0] are lost in this method

Bit 2 – AGCDATA_FMT
Configures the sign of AGCDATA_IN[9:0] value.

Bit 1 – OUTPUT_POLARITY
Writing '1' to this bit reverses PWM pulse output polarity.

Bit 0 – PWM_EN
Writing '1' to this bit enables PWM functionality.
Writing '0' to this bit disables PWM functionality.

15. I²C Interface

The ATSAMB11-XR2100A and the ATSAMB11-ZR210CA provide an I²C interface that can be configured as slave or master. The I²C interface is a two-wire serial interface consisting of a serial data line (SDA) and a serial clock line (SCL). The ATSAMB11-XR2100A and the ATSAMB11-ZR210CA I²C support I²C bus Version 2.1 – 2000.

15.1 Features

The following are the features of I²C module:

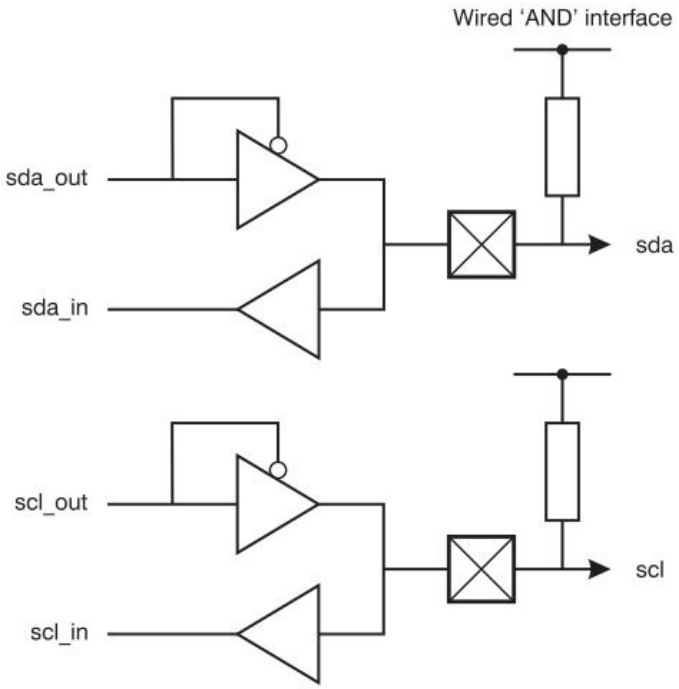
- Two I²C peripherals (I²C0 and I²C1) available
- Modes of operation:
  - Master mode
  - Slave mode
- Automatic address recognition in hardware
- Automatic acknowledgment generation
- Standard mode (100 kbps)
- Fast mode (400 kbps)
- High-Speed mode (3.4 Mbps)

15.2 Principal of Operation

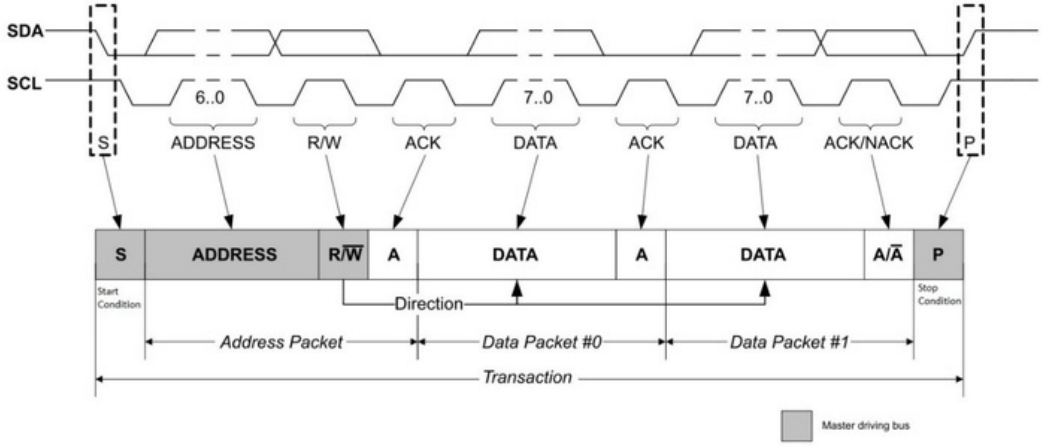
The I²C interface uses two physical lines for communication:

- Serial Data Line (SDA) for packet transfer
- Serial Clock Line (SCL) for the bus clock

The I²C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only when the SCL line is low, except for STOP and START conditions. A transaction starts with the I²C master sending the start condition, followed by a 7-bit address and a direction bit (read or write to/from the slave). The addressed I²C slave will then acknowledge (ACK) the address, and data packet transactions can begin. Every 9-bit data packet consists of 8 data bits followed by a one-bit reply indicating whether or not the data was acknowledged. If a data packet is not acknowledged (NACK), whether by the I²C slave or master, the I²C master acts by terminating the transaction by sending the STOP condition. The I²C peripheral in ATSAMB11-XR2100A and the ATSAMB11-ZR210CA does not support repeated start, clock stretching by slave, and 10-bit slave addressing condition. The output drivers are open-drain to perform wire-AND functions on the bus.



The maximum number of devices on the bus are limited by only the maximum capacitance specification of 400 pF. The following figure illustrates the I²C transaction formats.



The I²C pins SCL and SDA can be selected based on the configuration selected in the Peripheral Multiplexing and MEGAMUXing register. For more details, see Peripheral Multiplexing and MEGAMUXing.

15.3 Clock Configuration

Before configuring the I²C registers, reset the I²C core by setting I2CX_CORE_RSTN bit in LPMCU_GLOBAL_RESET_0 register. Then the clock to the I²C peripheral needs to be enabled. This is done by setting the I2C0_CORE_CLK_EN, I2C1_CORE_CLK_EN bit in LPMCU_CLOCK_ENABLES_0 register for I²C0 and I²C1 respectively. For more details on configuration, see Peripheral Clock Configuration.

15.4 Functional Description

15.4.1 Initialization

- Configure the pin mux register and/or MEGAMUX register (see I/O Peripheral Multiplexing and MegaMuxing) to select the IO pins that need to be used as SCL and SDA lines.
- Select the I^2C module clock source by configuring the CLOCK_SOURCE_SELECT register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz and 3 MHz. In Slave mode, the clock source selected should be a factor of at least five times more than the SCL clock from master device. For example, to operate the SCL at 400 kHz a minimum 2 MHz clock is required.
- Set the clock divider value by configuring I2C_CLK_DIVIDER register. In Master mode, the value configured in this register determine the SCL frequency. The clock source selected in CLOCK_SOURCE_SELECT register is divided by (n+1), where n is the value set in I2C_CLK_DIVIDER register.
  SCL Frequency = CLOCK_SOURCE_SELECT/(I2C_CLK_DIVIDER+1)
- To configure the I^2C as master, set the MASTER_ENABLE bit in I2C_MASTER_MODE register.
- To configure the I^2C as slave, write the 7-bit slave address in register I2C_SLAVE_ADDRESS. Clear MASTER_ENABLE bit in I2C_MASTER_MODE register.

15.4.2 Enabling, Disabling, and Flushing

- The I^2C peripheral is enabled by setting the ENABLE bit in I2C_MODULE_ENABLE register. Writing '1' to the ENABLE bit enables the I^2C module clock and allows to perform in I^2C transactions. To generate interrupts on I^2C transaction, the appropriate bits in the interrupt mask registers must be set, as follows:
  - Register the I^2C transmit and receive ISR function
  - Enable the interrupts in NVIC interrupt controller registers. For more details, see Interrupt section.
- The I^2C peripheral is disabled by clearing the ENABLE bit in I2C_MODULE_ENABLE register. If this bit is cleared then the module does not take part in any I^2C transactions and the internal state of the module is reset.
- The I2C_FLUSH register can be used to flush the contents of both the transmit and receive FIFOs. Flushing the transmit FIFO terminates any ongoing transactions when the current byte is transmitted. This allows the software to flush the FIFOs and abort any ongoing transactions.
- The I2C_ACTIVE bit in I2C_STATUS register indicates whether the I^2C peripheral is Idle or Active state. If the I2C_ACTIVE bit is set, the I^2C module is in Active state and the configuration registers should not be changed during this period. If the registers are modified while a transaction is ongoing, the state of the I^2C module cannot be ensured.

15.4.3 I^2C Master Operation

- The I^2C transaction is started by setting the ONBUS_ENABLE bit in the I2C_ONBUS register. When operating as a master the module initiates transactions when data are placed in the transmit FIFO, and continues to transmit the content of the FIFO until it is empty. If ONBUS_ENABLE bit is reset to 0 then the module completes the transmission of the current byte and generates a stop condition on the bus.
- The start condition is generated by setting the ADDRESS_FLAG bit in TRANSMIT_DATA register and ONBUS_ENABLE bit in I2C_ONBUS register is enabled. The byte written into the lower 8 [7:1] bits of TRANSMIT_DATA register must then be the address of the device associated with this transaction and the least significant bit 0 of the data indicates the direction of the transaction.

- The NAK bit in RECEIVE_STATUS register is set when a NAK is received. The I²C module automatically retries the transmission, the transaction can be aborted by writing to the I2C_FLUSH register.

15.4.4 I²C Slave Operation

In Slave mode, the NAK bit in RECEIVE_STATUS register is set at the end of a transaction. The module automatically recognizes the end of a transaction and stops; this bit can be used by the software to recognize the end of a transaction.

15.4.5 Transmit and Receive Operation

- The I²C Module has two FIFOs, one for transmit and one for receive to automate transmission/ reception. The data written on TRANSMIT_DATA register pushes one byte into the transmit FIFO.
- The TRANSMIT_STATUS register reflects the state of the I²C transmitter. If the corresponding bits are set in the TX_INTERRUPT_MASK register then interrupts can be generated upon the bit in this register being set. To enable the I²C interrupt on transmit, register the I²C transmit ISR function and enable the interrupts in NVIC interrupt controller registers. For more details, see Interrupt.
- Reading from the RECEIVE_DATA register pops one byte from the receive FIFO.
- The RECEIVE_STATUS register reflects the state of the I²C receiver. If the corresponding bits are set in the RX_INTERRUPT_MASK mask register, then interrupts can be generated upon the bit in this register being set. To enable the I²C interrupt on reception, register the I²C receive ISR function and enable the interrupts in NVIC interrupt controller registers. For more details, see Interrupt.

15.4.6 Example

The following is the sample example to enable I2C0 as master on LP_GPIO_8 (SDA) and LP_GPIO_9 (SCL) pins with 100 kHz SCL frequency, 0x20 as slave address and write 1 byte data:

```
/* From Table 10-1. I/O Port Function Multiplexing,  LP_GPIO_8 (SDA) and LP_GPIO_9 (SCL) are
configured as I2C through MUX2 configuration */
/* Write PINMUX_SEL_1 bits [2:0], [4:6] with 0x2 (MUX2) */
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x2<<0);
LPMCU_MISC_REGS0->PINMUX_SEL_1.reg |= (0x2<<4);

/* Select clock source for I2C0 as 26MHz */
I2C0 ->CLOCK_SOURCE_SELECT.reg = 0;

/* Calculate CLK_DIVIDER for SCL clock frequency
SCL Frequency = CLOCK_SOURCE_SELECT/(I2C_CLK_DIVIDER+1)
I2C_CLK_DIVIDER = (26000000/100000)-1 = 259 */
I2C0->I2C_CLK_DIVIDER.reg = 259;

/* Enable I2C0 in master mode */
I2C0->I2C_MASTER_MODE.reg = 1;

/* Enable I2C0 Module */
I2C0->I2C_MODULE_ENABLE.reg = 1;

/* Enable I2C0 bus (start condition) */
I2C0>I2C_ONBUS.reg = 1;

/*Write  I2C slave address ADDRESS_FLAG = 1, Direction is write */
I2C0->TRANSMIT_DATA.reg = (1<<8) | (0x20<< 1) | 0x0;

/* Wait for data to be transmitted. TX_FIFO_NOT_FULL bit to be set*/
while (!(I2C0->TRANSMIT_STATUS.reg& (1<<0))) ;

/*Write  1 byte of data */
I2C0->TRANSMIT_DATA.reg = 0x67;

/* Now check whether the core has sent all the data in FIFO out and free the bus.
TX_FIFO_EMPTY is set */
while (!(I2C0->TRANSMIT_STATUS.reg& (1<<4));
```

```
/* Send stop condition */
I2C0>I2C_ONBUS.reg =0;
```

15.5 Power Management

If the system goes to the Ultra-Low Power mode, the I^2C peripheral shuts down. The I^2C configuration registers lose their content, and are not restored when powered- up again. User must reconfigure the I^2C peripheral at power-up to ensure it is in a well-defined state before use. For details on reconfiguration, refer to ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User Guide. This document also explains on how sleep and wake-up are controlled.

15.6 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40003000	I2C0 Register	TRANSMIT_DATA	7:0	TX_DATA[7:0]							
0x40003400	I2C1 Register		15:8								ADDRESS_FLAG
0x40003004	I2C0 Register	RECEIVE_DATA	7:0	RX_BYTE[7:0]							
0x40003404	I2C1 Register										
0x40003008	I2C0 Register	TRANSMIT_STATUS	7:0				TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL
0x40003408	I2C1 Register										
0x4000300C	I2C0 Register	RECEIVE_STATUS	7:0			NAK	FIFO_OVERRUN	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY
0x4000340C	I2C1 Register										
0x40003010	I2C0 Register	CLOCK_SOURCE_SELECT	7:0							CLOCK[1:0]	
0x40003410	I2C1 Register										
0x40003014	I2C0 Register	I2C_MODULE_ENABLE	7:0								ENABLE
0x40003414	I2C1 Register										
0x40003018	I2C0 Register	I2C_CLK_DIVIDER	7:0	I2C_DIVIDE_RATIO[7:0]							
0x40003418	I2C1 Register		15:8	I2C_DIVIDE_RATIO[15:8]							
0x4000301C	I2C0 Register	I2C_MASTER_MODE	7:0								MASTER_ENABLE
0x4000341C	I2C1 Register										
0x40003020	I2C0 Register	I2C_ONBUS	7:0								ONBUS_ENABLE
0x40003420	I2C1 Register										
0x40003024	I2C0 Register	I2C_SLAVE_ADDRESS	7:0		ADDRESS[6:0]						
0x40003424	I2C1 Register										
0x40003028	I2C0 Register	I2C_STATUS	7:0								I2C_ACTIVE
0x40003428	I2C1 Register										
0x4000302C	I2C0 Register	TX_INTERRUPT_MASK	7:0				TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK
0x4000342C	I2C1 Register										
0x40003030	I2C0 Register	RX_INTERRUPT_MASK	7:0			NAK_MASK	FIFO_OVERRUN_MASK	RX_FIFO_0P75_FULL_MASK	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK
0x40003430	I2C1 Register										
0x40003034	I2C0 Register	I2C_FLUSH	7:0								I2C_FLUSH
0x40003434	I2C1 Register										

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0			MEGAMUX_SEL[5:0] LP_GPIO_0					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_1					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_2					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_3					
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0			MEGAMUX_SEL[5:0] LP_GPIO_4					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_5					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_6					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_7					
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0			MEGAMUX_SEL[5:0] LP_GPIO_8					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_9					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_10					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_11					
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0			MEGAMUX_SEL[5:0] LP_GPIO_12					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_13					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_14					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_15					
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0			MEGAMUX_SEL[5:0] LP_GPIO_16					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_17					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_18					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_19					
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0			MEGAMUX_SEL[5:0] LP_GPIO_20					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_21					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_22					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_23					
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0			MEGAMUX_SEL[5:0] LP_GPIO_24					
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0				MUX_0[4:0]				
			15:8				MUX_1[4:0]				
			23:16				MUX_2[4:0]				
			31:24				MUX_3[4:0]				

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0					MUX_4[4:0]			
			15:8					MUX_5[4:0]			
			23:16					MUX_6[4:0]			
			31:24					MUX_7[4:0]			
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0					MUX_8[4:0]			
			15:8					MUX_9[4:0]			
			23:16					MUX_10[4:0]			
			31:24					MUX_11[4:0]			
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0					MUX_12[4:0]			
			15:8					MUX_13[4:0]			
			23:16					MUX_14[4:0]			
			31:24					MUX_15[4:0]			
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0					MUX_16[4:0]			
			15:8					MUX_17[4:0]			
			23:16					MUX_18[4:0]			
			31:24					MUX_19[4:0]			
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0					MUX_20[4:0]			

15.7 Register Description

15.7.1 I2C Transmit Data

Name: TRANSMIT_DATA
Reset: 0x0000

Absolute Address: 0x40003000 (I2C0), 0x40003400 (I2C1)

This register is a part of I2C Registers. Writing this register pushes one byte of data into the transmit FIFO of I2C module.

Bit	15	14	13	12	11	10	9	8
								ADDRESS_FLAG
Access								W
Reset								0

Bit	7	6	5	4	3	2	1	0
	TX DATA[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 8 – ADDRESS_FLAG
Writing '1' to bit indicate the start of I2C transaction. When this bit is set, the byte written into the lower 8 bits of this register should then be the address of the device associated with this transaction. When this bit is set the least significant bit of the data is used to indicate the direction of the transaction.

ADDRESS_FLAG	TX_DATA
0	Bits 7:0 is data
1	Bits 7:1 is Address of the slave Bit 0: Direction of Transaction

Bits 7:0 – TX DATA[7:0]
These eight bits are the data or address to transmit (see ADDRESS_FLAG table).
Writing '0' to Bit 0, indicates the write command and the direction is writing to slave from master.
Writing '1' to Bit 0, indicates the read command and the direction is reading from slave by master.

15.7.2 I2C Receive Data

Name: RECEIVE_DATA
Reset: 0x00

Absolute Address: 0x40003004 (I2C0), 0x40003404 (I2C1)

This register is a part of I2C Registers. Reading this register pops one byte of received data from receive FIFO.

Bit	7	6	5	4	3	2	1	0
	RX_BYTE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RX_BYTE[7:0]
Reading from this 8-bit read only register pops one byte from the receive FIFO.

15.7.3 I2C Transmit Status

Name: TRANSMIT_STATUS
Reset: 0x1F

Absolute Address: 0x40003008 (I2C0), 0x40003408 (I2C1)

This register is a part of I2C Registers. This register provides the status of I2C transmit operation.

Bit	7	6	5	4	3	2	1	0
				TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL
Access				R	R	R	R	R
Reset				1	1	1	1	1

Bit 4 – TX_FIFO_EMPTY
This bit is set if the FIFO is completely empty.

Bit 3 – TX_FIFO_0P75_EMPTY
This bit is set if the FIFO is three-quarters empty.

Bit 2 – TX_FIFO_0P5_EMPTY
This bit is set if the FIFO is half empty.

Bit 1 – TX_FIFO_0P25_EMPTY
This bit is set if the FIFO is one quarter empty.

Bit 0 – TX_FIFO_NOT_FULL
This bit is set if there is at least space for one more byte in the FIFO.
Reading '0' indicates TX FIFO is full.

15.7.4 I2C Receive Status

Name: RECEIVE_STATUS
Reset: 0x00

Absolute Address: 0x4000300C (I2C0), 0x4000340C (I2C1)

This register is a part of I2C Registers. This register provides the status of I2C receive operation.

Bit	7	6	5	4	3	2	1	0
			NAK	FIFO_OVERRUN	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – NAK
This bit is set when NAK is received. The I2C module retries transmission unless transaction aborted by the flush register. This bit is reset after the Status register is read.

Bit 4 – FIFO_OVERRUN
This bit is set when a character is received but there is no place left in the FIFO to store it.
This bit is reset after the Status register is read.

Bit 3 – RX_FIFO_0P75_FULL
This bit is set if the FIFO is three-quarters full.

Bit 2 – RX_FIFO_0P5_FULL
This bit is set if the FIFO is half full.

Bit 1 – RX_FIFO_0P25_FULL
This bit is set if the FIFO is one quarter full.

Bit 0 – RX_FIFO_NOT_EMPTY
This bit is set if there is at least space for one more byte in the FIFO.
Reading '0' indicates RX FIFO is Empty.

15.7.5 I2C Clock Source Select

Name: CLOCK_SOURCE_SELECT
Reset: 0x00

Absolute Address: 0x40003010 (I2C0), 0x40003410 (I2C1)

This register is a part of I2C Registers. This register allows the user to select the input clock source for I2C peripheral.

Bit	7	6	5	4	3	2	1	0
							CLOCK[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 1:0 – CLOCK[1:0]
Selects the input clock for I2C module.

CLOCK[1:0]	Description
0	26 MHz clock
1	13 MHz clock
2	6.5 MHz clock
3	3.25 MHz clock

15.7.6 I2C Module Enable

Name: I2C_MODULE_ENABLE
Reset: 0x00

Absolute Address: 0x40003014 (I2C0), 0x40003414 (I2C1)

This register is a part of I2C Registers. This register allows the user to enable/disable the I2C peripheral.

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R/W
Reset								0

Bit 0 – ENABLE
Writing '0' to this bit disables I2C module
Writing '1' to this bit enables I2C module.

15.7.7 I2C SCK Clock Divider

Name: I2C_CLK_DIVIDER
Reset: 0x0000

Absolute Address: 0x40003018 (I2C0), 0x40003418 (I2C1)

This register is a part of I2C Registers. This register sets the divide ratio used to generate the SCL clock from the module's input clock.

Bit	15	14	13	12	11	10	9	8
	I2C_DIVIDE_RATIO[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	I2C_DIVIDE_RATIO[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – I2C_DIVIDE_RATIO[15:0]
Sets the divide ratio used to generate the SCK clock signal from the clock selected by the CLOCK_SOURCE_SELECT register. The minimum division is by 2; a value of 0 is not valid.
SCL Frequency = CLOCK[1:0] / (I2C_DIVIDE_RATIO[15:0]+1)

15.7.8 I2C Master Mode Enable

Name: I2C_MASTER_MODE
Reset: 0x00

Absolute Address: 0x4000301C (I2C0), 0x4000341C (I2C1)

This register allows the user to select I2C between Master and Slave modes.

Bit	7	6	5	4	3	2	1	0
								MASTER_ENABLE
Access								R/W
Reset								0

Bit 0 – MASTER_ENABLE
Writing '0' to this bit enables I2C in Slave mode.
Writing '1' to this bit enables I2C in Master mode.

15.7.9 I2C Start Master Transaction

Name: I2C_ONBUS
Reset: 0x00

Absolute Address: 0x40003020 (I2C0), 0x40003420 (I2C1)

This register is a part of I2C Registers. This register initiates the I2C transactions when in Master mode.

Bit	7	6	5	4	3	2	1	0
								ONBUS_ENABLE
Access								R/W
Reset								0

Bit 0 – ONBUS_ENABLE
Active High Enable to initiate transactions when in Master mode
Writing '0' to this bit Master completes current byte and generates stop condition on bus
Writing '1' to this bit Master transmits contents of FIFO until empty

15.7.10 I2C Slave Address

Name: I2C_SLAVE_ADDRESS
Reset: 0x00

Absolute Address: 0x40003024 (I2C0), 0x40003424 (I2C1)

This register is a part of I2C Registers. This seven bit read/write register sets the I2C slave address.

Bit	7	6	5	4	3	2	1	0
		ADDRESS[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 6:0 – ADDRESS[6:0]
These bits hold the I2C slave address.

15.7.11 I2C Status

Name: I2C_STATUS
Reset: 0x00

Absolute Address: 0x40003028 (I2C0), 0x40003428 (I2C1)

This register is a part of I2C Registers.

Bit	7	6	5	4	3	2	1	0
								I2C_ACTIVE
Access								R
Reset								0

Bit 0 – I2C_ACTIVE
The I2C configuration registers must not be changed when this bit is set. If the registers are modified during transaction is ongoing, the state of the I2C module is not guaranteed.

Read Value	Description
0	I2C is idle
1	I2C is active

15.7.12 I2C Transmit Interrupt Mask

Name: TX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x4000302C (I2C0), 0x4000342C (I2C1)

This register is a part of I2C Registers. This register is used to enable or disable the generation of I2C transmission interrupts. During the I2C transmission interrupt, if a bit in TX_INTERRUPT_MASK register is set and its corresponding bit in TRANSMIT_STATUS register is set then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
				TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – TX_FIFO_EMPTY_MASK
Writing '0' disables the TX_FIFO_EMPTY interrupt.
Writing '1' enables the TX_FIFO_EMPTY interrupt.

Bit 3 – TX_FIFO_0P75_EMPTY_MASK
Writing '0' disables the TX_FIFO_0P75_EMPTY interrupt.
Writing '1' enables the TX_FIFO_0P75_EMPTY interrupt.

Bit 2 – TX_FIFO_0P5_EMPTY_MASK
Writing '0' disables the TX_FIFO_0P5_EMPTY interrupt.
Writing '1' enables the TX_FIFO_0P5_EMPTY interrupt.

Bit 1 – TX_FIFO_0P25_EMPTY_MASK
Writing '0' disables the TX_FIFO_0P25_EMPTY interrupt.
Writing '1' enables the TX_FIFO_0P25_EMPTY interrupt.

Bit 0 – TX_FIFO_NOT_FULL_MASK
Writing '0' disables the TX_FIFO_NOT_FULL interrupt.
Writing '1' enables the TX_FIFO_NOT_FULL interrupt.

15.7.13 I2C Receive Interrupt Mask

Name: RX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x40003030 (I2C0), 0x40003430 (I2C1)

This register is a part of I2C Registers. This register is used to enable or disable the generation of I2C receive interrupts. During the I2C receive interrupt, if a bit in RX_INTERRUPT_MASK register is set and its corresponding bit in RECEIVE_STATUS register is set then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
			NAK_MASK	FIFO_OVERRUN_MASK	RX_FIFO_0P75_FULL_MASK	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – NAK_MASK
Writing '0' disables the NAK interrupt.
Writing '1' enables the NAK interrupt.

Bit 4 – FIFO_OVERRUN_MASK
Writing '0' disables the FIFO_OVERRUN interrupt.
Writing '1' enables the FIFO_OVERRUN interrupt.

Bit 3 – RX_FIFO_0P75_FULL_MASK
Writing '0' disables the RX_FIFO_0P75_FULL interrupt.
Writing '1' enables the RX_FIFO_0P75_FULL interrupt.

Bit 2 – RX_FIFO_0P5_FULL_MASK
Writing '0' disables the RX_FIFO_0P5_FULL interrupt.
Writing '1' enables the RX_FIFO_0P5_FULL interrupt.

Bit 1 – RX_FIFO_0P25_FULL_MASK
Writing '0' disables the RX_FIFO_0P25_FULL interrupt.
Writing '1' enables the RX_FIFO_0P25_FULL interrupt.

Bit 0 – RX_FIFO_NOT_EMPTY_MASK
Writing '0' disables the RX_FIFO_NOT_EMPTY interrupt.
Writing '1' enables the RX_FIFO_NOT_EMPTY interrupt.

15.7.14 I2C FLUSH

Name: I2C_FLUSH
Reset: 0x00

Absolute Address: 0x40003034 (I2C0), 0x40003434 (I2C1)

This register is a part of I2C Registers. This register allows the software to flush the FIFOs and abort any ongoing transactions.

Bit	7	6	5	4	3	2	1	0
								I2C_FLUSH
Access								W
Reset								0

Bit 0 – I2C_FLUSH
Writing to this address flushes the content of both the Tx and Rx FIFOs. The written value does not have effect. Flushing the Tx FIFO aborts the ongoing transactions when the current byte is being transmitted.

16. Dual Timer

The dual timer is an APB dual-input timer module consisting of two programmable 32-bit or 16-bit down-counters (TIMER1, TIMER2) that can generate interrupts when they reach zero. The operation of each timer module is identical.

16.1 Features

The following are the dual timer features:

- Selectable configuration
  - 16- or 32-bit down counter operation
- Three different modes of operation
  - Free-Running mode
  - One Shot Count mode
  - Periodic mode
- Interrupts on
  - Counter underflow
- Internal three selectable prescaler
- Four different clock inputs

16.2 Block Diagram

The following are the block diagrams of the dual timer and pre-scale clock enable generation.

Figure 16-1. Dual Timer Block

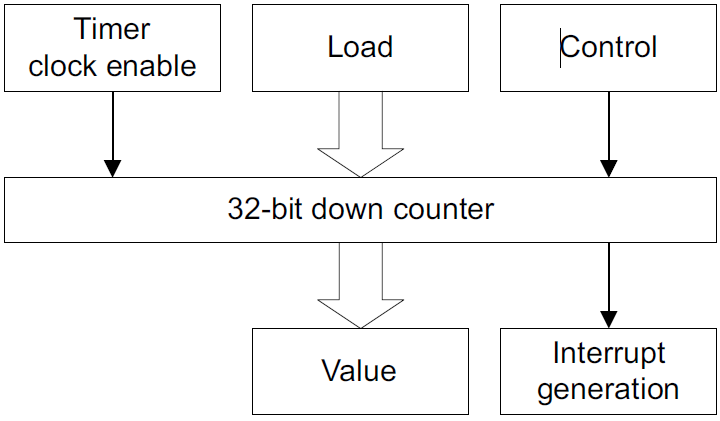
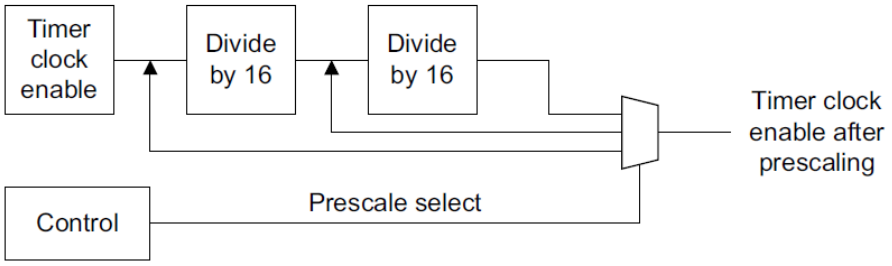


Figure 16-2. Pre-Scale Clock Enable Generation



16.3 Clock Configuration

Before configuring the dual timer registers, the clock for dual timer peripheral must be enabled. This is performed by setting the DUALTIMER0_CLK_EN bit in LPMCU_CLOCK_ENABLES_0 register. This is the common clock source for both the timer modules. For more details on configuration, see Peripheral Clock Configuration.

16.4 Direct Memory Access

The Direct Memory Access (DMA) request lines are connected to the DMA Controller (DMAC). To use DMA requests with this peripheral, the DMAC must first be configured. For more details on configuration, see DMA Controller.

16.5 Functional Description

16.5.1 Initialization

After device Reset, the timer interrupt of both the timer module is enabled by default and all other registers are set as zero. The following initialization procedure is identical for both TIMER1 and TIMER2 modules.

1. Select the dual timer module clock source by configuring the DUALTIMER0_CLK_SEL [1:0] bits of LPMCU_CTRL register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz and 3.25 MHz. The clock source is common for both TIMER1 and TIMER2.
2. Set the pre-scaler value by configuring TIMERPRE[1:0] bits of TIMERnCONTROL (n=1, 2) register. The clock source selected in the above step is divided by selected pre-scale value. This acts as the input clock to the TIMERn module. The input clock (P_CLK) for dual timer is calculated using the following formula:

$$\text{P\_CLK} = \frac{\text{Clock set by DUALTIMER0\_CLK\_SEL [1:0] bits}}{\text{Prescaler set by TIMERPRE[1:0] bits}}$$

16.5.2 Timer Modes

For each timer, the following modes of operation are available:

16.5.2.1 Free-Running Mode

In the Free-Running mode, the counter wraps after reaching zero and generates an interrupt every time when it reaches zero. It also continues to count down from the maximum value. This is the default mode.

16.5.2.2 Periodic Timer Mode

In the Periodic Timer mode, the counter generates an interrupt at a constant interval, reloading the original value after wrapping past zero.

16.5.2.3 One-Shot Timer Mode

In the One-Shot Timer mode, the counter generates an interrupt only once. When the counter reaches zero, it stops until the timer reprograms. The timer can be restarted again by any one of following method:

- Clear the ONE_SHOT_COUNT bit in TIMERnCONTROL (n=1, 2) register, with which the count proceeds according to the selection of the Free-Running or Periodic mode.
- Write a new value to the TIMERnLOAD (n=1, 2) register. This method starts the timer again in One-Shot Timer mode.

16.5.3 Operation

Each timer is identical in its operation and has an identical set of registers.

- Select the 16-bit or 32-bit timer operation by configuring TIMER_SIZE bit of the TIMERnCONTROL (n=1, 2) register.
- Select the Timer mode of operations by configuring ONE_SHOT_COUNT, TIMER_MODE bits of TIMERnCONTROL (n=1, 2) register. For more details on specific mode, see Timer Modes.
- Enable timer interrupt by writing '1' to INTERRUPT_ENABLE bit of the TIMERnCONTROL (n=1, 2) register. To generate an interrupt on the Dual Timer, the appropriate bits in the interrupt mask registers must be set as following:
  - Register the dual timer ISR function in interrupt vector table.
  - Enable the dual timer interrupt in NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.

  Note: Only one ISR index is allocated in interrupt vector table for both TIMER1 and TIMER2 modules.
- Counter is decremented for each P_CLK. Calculate the counter value for desired interval (ms) using the following formula:
  TIMERnLOAD = Interval (ms) * P_CLK (kHz)

  The interval is loaded by writing to the LOAD[31:0] bits of the TIMERnLOAD (n=1, 2) register.
- The loaded counter value is reloaded on current counter of the TIMERnVALUE (n=1, 2) register.
- Enable the timer by writing '1' to TIMER_ENABLE bit of the TIMERnCONTROL (n=1, 2) register and CNTR_n_ENABLE bit of the DUALTIMER0_CTRL register.
- The loaded counter value counts down to 0.
- When 0 is reached, an interrupt is generated. This is indicated through masked interrupt status bit in the TIMERnMIS (n=1, 2) register. Interrupt can be cleared by writing to the TIMERnINTCLR (n=1, 2) register. If interrupt is not enabled though INTERRUPT_ENABLE bit, when the counter reaches 0, the raw interrupt status register TIMERnRIS (n=1, 2) indicates the counter expiry.
- In the One-Shot mode, the timer halts when it reaches 0. In Periodic mode, the counter continues to decrement to 0, and then reloads the TIMERnVALUE (n=1, 2) from the TIMERnLOAD (n=1, 2) register and continues to decrement. In this mode, the counter effectively generates a periodic interrupt. If the timer is operating in Free-Running mode, it continues to decrement from its maximum value (0xFFFFFFFF for 32-Bit mode, 0xFFFF for 16-Bit mode).

16.5.4 Restart the Running Timer

When a counter is already running, write a new value to the TIMERnLOAD (n=1, 2) register to immediately restart the current counter TIMERnVALUE (n=1, 2) at the new value.

In the Periodic mode, write the new value to the TIMERnBGLOAD (n=1, 2) register. This has no effect on the current count. The counter continues to decrement to zero, and then restarts from the new load value.

16.5.5 Example

The following is the sample example to enable Dual Timer TIMER1 in the One-Shot mode with interval of one second and 32-bit mode:

```
/* Register ISR handler as explained in Section 6.3  Nested Vector Interrupt Controller */
/* Set clock source as 26MHz */
LPMCU_MISC_REGS0->LPMCU_CTRL.bit.DUALTIMER0_CLK_SEL = 0;
/* Set input clock prescaler as 1 */
DUALTIMER0->TIMER1CONTROL.bit.TIMERPRE = 0;
/* Calculate TIMERnLOAD = Interval (ms) * P_CLK (kHz)
P_CLK = Clock set by DUALTIMER0_CLK_SEL [1:0] bits/
Prescaler set by TIMERPRE[1:0] bits
P_CLK = 26000000/1 = 26000kHz
TIMERnLOAD = 1000 (ms) * 26000 (kHz)= 26000000*/
DUALTIMER0->TIMER1LOAD.reg = 26000000;
/* Select the 32-bit timer operation */
DUALTIMER0->TIMER1CONTROL.bit. TIMERSIZE= 1;
/* Select one shot mode */
DUALTIMER0->TIMER1CONTROL.bit. ONE_SHOT_COUNT= 1;
/* Enable Dual timer interrupt */
DUALTIMER0->TIMER1CONTROL.bit. INTERRUPT_ENABLE= 1;
/* Enable Dual Timer */
LPMCU_MISC_REGS0->DUALTIMER0_CTRL.bit.CNTR_1_ENABLE = 1;
DUALTIMER0->TIMER1CONTROL.bit.TIMER_ENABLE = 1;
/* Enable NVIC interrupt as explained in Section 6.3  Nested Vector Interrupt Controller */
```

16.6 Power Management

If the system goes to the Ultra-Low Power mode, the dual timer peripheral shuts down. The Dual Timer Configuration registers lose their content, and cannot be restored when powered-up again. Therefore, the user must reconfigure the dual timer peripheral at power-up to ensure it is in a well-defined state before use.

For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains how sleep and wake-up are controlled.

16.7 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B188	LPMCU_MISC_REGS0	DUALTIMER0_CTRL	7:0							CNTR_2_ENABLE	CNTR_1_ENABLE
0x40001000 (TIMER1), 0x40001020 (TIMER2)	DUALTIMER0	TIMERnLOAD (n=1, 2)	7:0	LOAD[7:0]							
			15:8	LOAD[15:8]							
			23:16	LOAD[23:16]							
			31:24	LOAD[31:24]							
0x40001004 (TIMER1), 0x40001024 (TIMER2)	DUALTIMER0	TIMERnVALUE (n=1, 2)	7:0	VALUE[7:0]							
			15:8	VALUE[15:8]							
			23:16	VALUE[23:16]							
			31:24	VALUE[31:24]							
0x40001008 (TIMER1), 0x40001028 (TIMER2)	DUALTIMER0	TIMERnCONTROL (n=1, 2)	7:0	TIMER_ENABLE	TIMER_MODE	INTERRUPT_ENABLE		TIMERPRE[1:0]		TIMERSIZE	ONE_SHOT_COUNT

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000100C (TIMER1), 0x4000102C (TIMER2)	DUALTIMER0	TIMERnINTCLR (n=1, 2)	7:0								INTCLR
0x40001010 (TIMER1), 0x40001030 (TIMER2)	DUALTIMER0	TIMERnRIS (n=1, 2)	7:0								RIS
0x40001014 (TIMER1), 0x40001034 (TIMER2)	DUALTIMER0	TIMERnMIS (n=1, 2)	7:0								MIS
0x40001018 (TIMER1), 0x40001038 (TIMER2)	DUALTIMER0	TIMERnBGLOAD (n=1, 2)	7:0	BGLOAD[7:0]							
			15:8	BGLOAD[15:8]							
			23:16	BGLOAD[23:16]							
			31:24	BGLOAD[31:24]							
0x40001F00	DUALTIMER0	TIMERITCR	7:0								ITCR
0x40001F04	DUALTIMER0	TIMERITOP	7:0							INT_TEST_TIMINT2_VALUE	INT_TEST_TIMINT1_VALUE
0x40001FD0	DUALTIMER0	TIMERPERIPHID4	7:0	BLOCK_COUNT[3:0]				JEP106_C_CODE[3:0]			
0x40001FD4	DUALTIMER0	TIMERPERIPHID5	7:0								
0x40001FD8	DUALTIMER0	TIMERPERIPHID6	7:0								
0x40001FDC	DUALTIMER0	TIMERPERIPHID7	7:0								
0x40001FE0	DUALTIMER0	TIMERPERIPHID0	7:0	PART_NUMBER[7:0]							
0x40001FE4	DUALTIMER0	TIMERPERIPHID1	7:0	JEP106_ID_3_0[3:0]				PART_NUMBER[3:0]			
0x40001FE8	DUALTIMER0	TIMERPERIPHID2	7:0	REVISION[3:0]				JEDEC_USED	JEP106_ID_6_4[2:0]		
0x40001FEC	DUALTIMER0	TIMERPERIPHID3	7:0	ECO_REV_NUMBER[3:0]				CUSTOMER_MOD_NUMBER[3:0]			
0x40001FF0	DUALTIMER0	TIMERPCELLID0	7:0	TIMERPCELLID0[7:0]							
0x40001FF4	DUALTIMER0	TIMERPCELLID1	7:0	TIMERPCELLID1[7:0]							
0x40001FF8	DUALTIMER0	TIMERPCELLID2	7:0	TIMERPCELLID2[7:0]							
0x40001FFC	DUALTIMER0	TIMERPCELLID3	7:0	TIMERPCELLID3[7:0]							

16.8 Register Description

16.8.1 Dual Timer Control

Name: DUALTIMER0_CTRL
Reset: 0x00

Absolute Address: 0x4000B188

This register is a part of the LPMCU_MISC_REGS0 registers. This register allows the user to enable dual timer TIMER1 and TIMER2.

Bit	7	6	5	4	3	2	1	0
							CNTR_2_ENABLE	CNTR_1_ENABLE
Access							R/W	R/W
Reset							0	0

Bit 1 – CNTR_2_ENABLE
Writing '0' to this bit disables TIMER2 of dual timer
Writing '1' to this bit enables TIMER2 of dual timer when TIMER_ENABLE bit of the TIMER2CONTROL register is also set

Bit 0 – CNTR_1_ENABLE
Writing '0' to this bit disables TIMER1 of dual timer
Writing '1' to this bit enables TIMER1 of dual timer when TIMER_ENABLE bit of the TIMER1CONTROL register is also set

16.8.2 TIMERn LOAD

Name: TIMERnLOAD (n=1, 2)
Reset: 0x00000000

Absolute Address: 0x40001000(TIMER1), 0x40001020(TIMER2)

This register is a part of DUALTIMER0 registers. This register contains the value from which the counter must decrement. This is the value used to reload the counter when the Periodic mode is enabled, when the current count reaches 0.There are two individual timer blocks in dual timer and therefore there are two TIMERnLOAD (n=1,2) registers.

Bit	31	30	29	28	27	26	25	24
	LOAD[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	LOAD[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LOAD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LOAD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – LOAD[31:0]
These bits hold the counter value equivalent to timer interval that is loaded into the TIMERnVALUE (n=1, 2) register. For more details on formula, see Operation.
This value is also used to reload the counter when the Periodic mode is enabled and reaches 0.

16.8.3 TIMERnVALUE

Name: TIMERnVALUE (n=1, 2)
Reset: 0xFFFFFFFF

Absolute Address: 0x40001004(TIMER1), 0x40001024(TIMER2)

This register is a part of DUALTIMER0 registers. This register contains the current counter value. There are two individual timer blocks in dual timer and therefore there are two TIMERnVALUE (n=1,2) registers.

Bit	31	30	29	28	27	26	25	24
	VALUE[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	VALUE[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	VALUE[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	VALUE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bits 31:0 – VALUE[31:0]
These bits hold the current counter value.

16.8.4 TIMERn Control

Name: TIMERnCONTROL (n=1, 2)
Reset: 0x20

Absolute Address: 0x40001008(TIMER1), 0x40001028(TIMER2)

This register is a part of DUALTIMER0 registers. This register allows the user to configure the dual timer. There are two individual timer blocks in dual timer and therefore there are two TIMERnCONTROL (n=1,2) registers.

Bit	7	6	5	4	3	2	1	0
	TIMER_ENABLE	TIMER_MODE	INTERRUPT_ENABLE		TIMERPRE[1:0]		TIMERSIZE	ONE_SHOT_COUNT
Access	R/W	R/W	R/W		R/W	R/W	R/W	R/W
Reset	0	0	1		0	0	0	0

Bit 7 – TIMER_ENABLE
Writing '0' to this bit disables the TIMERn
Writing '1' to this bit enables the TIMERn module when, CNTR_n_ENABLE bit of the DUALTIMER0_CTRL register is also set

Bit 6 – TIMER_MODE
Writing '0' to this bit enables TIMERn in the Free-Running mode, when ONE_SHOT_COUNT is disabled
Writing '1' to this bit enables TIMERn in the Periodic mode, when ONE_SHOT_COUNT is disabled

Bit 5 – INTERRUPT_ENABLE
Writing '0' to this bit disables TIMERn interrupt
Writing '1' to this bit enables TIMERn interrupt

Bits 3:2 – TIMERPRE[1:0]
These bits select the prescaler for TIMERn input clock selected through DUALTIMER0_CLK_SEL[1:0] bits.

TIMERPRE[1:0]	Description
0x00	Input clock is divided by 1
0x01	Input clock is divided by 16
0x02	Input clock is divided by 256
0x03	Undefined, do not use

Bit 1 – TIMERSIZE
This bit selects 16-bit or 32-bit operations
Writing '0' to this bit configures TIMERn as 16-bit counter
Writing '1' to this bit configures TIMERn as 32-bit counter

Bit 0 – ONE_SHOT_COUNT
Writing '0' to this bit configures TIMERn as the Free-Running or Periodic mode selected by TIMER_MODE bit
Writing '1' to this bit configures TIMERn as the One-Shot Count mode

16.8.5 TIMERn Interrupt Clear

Name: TIMERnINTCLR (n=1, 2)
Reset: 0x00

Absolute Address: 0x4000100C(TIMER1), 0x4000102C(TIMER2)

This register is a part of DUALTIMER0 registers. This register allows the user to clear the TIMERn interrupt request. There are two individual timer blocks in dual timer and therefore there are two TIMERnINTCLR (n=1,2) registers.

Bit	7	6	5	4	3	2	1	0
								INTCLR
Access								W
Reset								0

Bit 0 – INTCLR
Writing '0' or '1' to this bit clears the TIMERn interrupt request

16.8.6 TIMERn Raw Interrupt Status

Name: TIMERnRIS (n=1, 2)
Reset: 0x00

Absolute Address: 0x40001010(TIMER1), 0x40001030(TIMER2)

This register is a part of DUALTIMER0 registers. This register provides the raw interrupt status of TIMERn. There are two individual timer blocks in dual timer and therefore there are two TIMERnRIS (n=1,2) registers.

Bit	7	6	5	4	3	2	1	0
								RIS
Access								R
Reset								0

Bit 0 – RIS
This bit indicates the raw interrupt status of TIMERn. When the counter value decrements and reaches zero this bit is set to indicate the trigger status of the timer. This bit is set even when the INTERRUPT_ENABLE bit of TIMERnCONTROL (n=1, 2) register is disabled.

16.8.7 TIMERn Masked Interrupt Status

Name: TIMERnMIS (n=1, 2)
Reset: 0x00

Absolute Address: 0x40001014(TIMER1), 0x40001034(TIMER2)

This register is a part of DUALTIMER0 registers. This register provides the masked interrupt status of TIMERn. There are two individual timer blocks in dual timer and therefore there are two TIMERnMIS (n=1,2) registers.

Bit	7	6	5	4	3	2	1	0
								MIS
Access								R
Reset								0

Bit 0 – MIS
This bit indicates the masked interrupt status of TIMERn. When the counter value decrements and reaches zero this bit is set to indicate the trigger status of the timer. This bit is set only when the INTERRUPT_ENABLE bit of the TIMERnCONTROL (n=1, 2) register is enabled.

16.8.8 TIMERn Background LOAD

Name: TIMERnBGLOAD (n=1, 2)
Reset: 0x00000000

Absolute Address: 0x40001018(TIMER1), 0x40001038(TIMER2)

This register is a part of DUALTIMER0 registers. This register contains the value from which the counter must decrement. This is the value used to reload the counter when the Periodic mode is enabled. This register provides an alternative method for accessing the TIMERnLOAD (n=1, 2) register. There are two individual timer blocks in dual timer and therefore there are two TIMERnBGLOAD (n=1,2) registers.

Bit	31	30	29	28	27	26	25	24
	BGLOAD[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BGLOAD[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BGLOAD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BGLOAD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – BGLOAD[31:0]
These bits hold the counter value equivalent to timer interval that is loaded into the TIMERnVALUE (n=1, 2) register. This value is also used to reload the counter when the Periodic mode is enabled.
The difference between TIMERnLOAD (n=1, 2) and TIMERnBGLOAD register is that writing to TIMERnBGLOAD register does not cause the current counter to restart immediately from the new value. The value is loaded on TIMERnVALUE (n=1, 2) register when it reaches zero.

16.8.9 Dual Timer Integrated Test Control

Name: TIMERITCR
Reset: 0x00

Absolute Address: 0x40001F00

This register is a part of DUALTIMER0 registers. This register allows the Test mode to use when integrating the dual timer peripheral with the ARM processor through APB bus. This is not usable for user application. When the Integration Test mode is enabled through this register, the TIMERITOP register directly controls the dual timer masked interrupt outputs.

Bit	7	6	5	4	3	2	1	0
								ITCR
Access								R/W
Reset								0

Bit 0 – ITCR - Integrated Test Control Register
Writing '0' to this bit disables the Integrated Test mode
Writing '1' to this bit enables the Integrated Test mode

16.8.10 Dual Timer Integrated Test Output Set

Name: TIMERITOP
Reset: 0x00

Absolute Address: 0x40001F04

This register is a part of DUALTIMER0 registers. This register allows the Test mode to use when integrating the dual timer peripheral with the ARM processor through APB bus. This is not usable for user application. When the Integration Test mode is enabled through TIMERITCR register, the values in TIMERITOP register directly drives the dual timer interrupt outputs.

Bit	7	6	5	4	3	2	1	0
							INT_TEST_TIM INT2_VALUE	INT_TEST_TIM INT1_VALUE
Access							W	W
Reset							0	0

Bit 1 – INT_TEST_TIMINT2_VALUE
Writing '0' to this bit disables integrated test output on TIMER2 interrupt
When the Test mode is enabled using the TIMERITCR register, writing '1' to this bit triggers the dual timer interrupt on TIMER2 interrupt

Bit 0 – INT_TEST_TIMINT1_VALUE
Writing '0' to this bit disables integrated test output on TIMER1 interrupt
When the Test mode is enabled using the TIMERITCR register, writing '1' to this bit triggers the dual timer interrupt on TIMER1 interrupt

16.8.11 Dual Timer Peripheral ID 4

Name: TIMERPERIPHID4
Reset: 0x04

Absolute Address: 0x40001FD0

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	BLOCK_COUNT[3:0]				JEP106_C_CODE[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	0

Bits 7:4 – BLOCK_COUNT[3:0]
These bits return zero when read

Bits 3:0 – JEP106_C_CODE[3:0]
These bits return 0x04 when read

16.8.12 Dual Timer Peripheral ID 5

Name: TIMERPERIPHID5
Reset: 0x00

Absolute Address: 0x40001FD4

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification Read-Only register. This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

16.8.13 Dual Timer Peripheral ID 6

Name: TIMERPERIPHID6
Reset: 0x00

Absolute Address: 0x40001FD8

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification Read-Only register. This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

16.8.14 Dual Timer Peripheral ID 7

Name: TIMERPERIPHID7
Reset: 0x00

Absolute Address: 0x40001FDC

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification Read-Only register. This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

16.8.15 Dual Timer Peripheral ID 0

Name: TIMERPERIPHID0
Reset: 0x23

Absolute Address: 0x40001FE0

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	PART_NUMBER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	0	0	1	1

Bits 7:0 – PART_NUMBER[7:0] Lower part of the Part number
These bits hold lower bits of part number. These bits return 0x23 when read.

16.8.16 Dual Timer Peripheral ID 1

Name: TIMERPERIPHID1
Reset: 0xB8

Absolute Address: 0x40001FE4

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	JEP106_ID_3_0[3:0]				PART_NUMBER[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	1	0	0	0

Bits 7:4 – JEP106_ID_3_0[3:0] Lower part of the JEP-106 Identity Code
These bits hold the lower part of JEP-106 identity code. These bits return 0x0B when read (JEP-106 identity code is 0x3B).

Bits 3:0 – PART_NUMBER[3:0] Higher part of the Part number
These bits hold higher bits of the part number. These bits return 0x08 when read.

16.8.17 Dual Timer Peripheral ID 2

Name: TIMERPERIPHID2
Reset: 0x1B

Absolute Address: 0x40001FE8

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	REVISION[3:0]				JEDEC_USED	JEP106_ID_6_4[2:0]		
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	1	0	1	1

Bits 7:4 – REVISION[3:0]
This bit hold the revision number of the dual timer peripheral. These bits return 0x01 when read.

Bit 3 – JEDEC_USED
This bit returns one when read, indicating that JEP-106 code is used.

Bits 2:0 – JEP106_ID_6_4[2:0] Higher part of the Part number
These bits hold the high part of JEP-106 identity code. These bits return 0x03 when read (JEP-106 identity code is 0x3B).

16.8.18 Dual Timer Peripheral ID 3

Name: TIMERPERIPHID3
Reset: 0x00

Absolute Address: 0x40001FEC

This register is a part of DUALTIMER0 registers. This is the dual timer peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	ECO_REV_NUMBER[3:0]				CUSTOMER_MOD_NUMBER[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:4 – ECO_REV_NUMBER[3:0]
These bits return 0x0 when read

Bits 3:0 – CUSTOMER_MOD_NUMBER[3:0]
These bits return 0x0 when read

16.8.19 Dual Timer Component ID 0

Name: TIMERPCELLID0
Reset: 0x0D

Absolute Address: 0x40001FF0

This register is a part of DUALTIMER0 registers. This is the dual timer component identification read-only register.

Bit	7	6	5	4	3	2	1	0
	TIMERPCELLID0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	1	0	1

Bits 7:0 – TIMERPCELLID0[7:0]
These bits return 0x0D when read

16.8.20 Dual Timer Component ID 1

Name: TIMERPCELLID1
Reset: 0xF0

Absolute Address: 0x40001FF4

This register is a part of DUALTIMER0 registers. This is the dual timer component identification read-only register.

Bit	7	6	5	4	3	2	1	0
	TIMERPCELLID1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	0	0	0	0

Bits 7:0 – TIMERPCELLID1[7:0]
These bits return 0xF0 when read

16.8.21 Dual Timer Component ID 2

Name: TIMERPCELLID2
Reset: 0x05

Absolute Address: 0x40001FF8

This register is a part of DUALTIMER0 registers. This is the dual timer component identification read-only register.

Bit	7	6	5	4	3	2	1	0
	TIMERPCELLID2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	1

Bits 7:0 – TIMERPCELLID2[7:0]
These bits return 0x05 when read

16.8.22 Dual Timer Component ID 3

Name: TIMERPCELLID3
Reset: 0xB1

Absolute Address: 0x40001FFC

This register is a part of DUALTIMER0 registers. This is the dual timer component identification read-only register.

Bit	7	6	5	4	3	2	1	0
	TIMERPCELLID3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	0	0	0	1

Bits 7:0 – TIMERPCELLID3[7:0]
These bits return 0xB1 when read

17. ARM Timer

The 32-bit timer block allows the CPU to generate a time tick at a programmed interval. This feature can be used for a wide variety of functions such as counting, interrupt generation, and time tracking.

Note: ARM Timer is one of the reserved resources being used by the BLE stack. Application must refrain from using this peripheral.

18. Serial Peripheral Interface

The SPI module is an implementation of a Serial Peripheral Interface (SPI) for communicating with peripherals with this interface, either as a master or a slave. The SPI interface is a full-duplex slave-synchronous serial interface. Fixed LP_GPIO_x pins can be configured as SPI using PINMUX_SEL_n, (n=0,1..4) registers.

18.1 Features

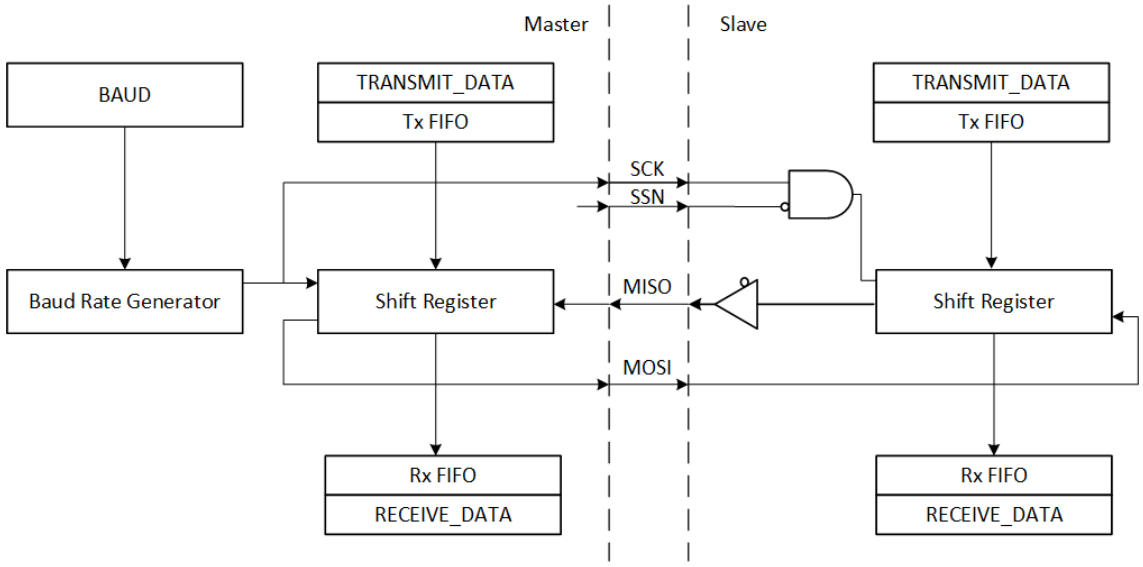
The following are the Serial Peripheral Interface features:

- Two SPI (SPI0, SPI1) peripherals
- Full-duplex, four-wire interface (MISO, MOSI, SCK, SSN)
- Eight bytes FIFO transmitter, and eight bytes FIFO receiver
- Supports all four SPI modes of operation
- Single data direction operation allows alternate function on MISO or MOSI pin
- Selectable LSB- or MSB-first data transfer
- Supports DMA
- Master operation:
  - Configurable serial clock speed from four different clock sources
  - Software controlled Slave Select through GPIO
  - Bus arbitration to avoid bus contention in multi-master environment

18.2 Block Diagram

The following is the block diagram of SPI.

Figure 18-1. Full-Duplex SPI Master Slave Interconnection



18.3 Signal Description

The SPI interface pins are mapped, as illustrated in the following table.

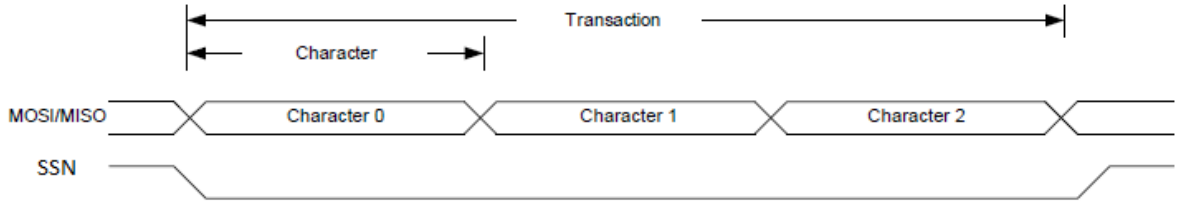
Table 18-1. SPI Interface Pin Mapping

Pin Name	SPI Function
SSN	Active-Low Slave Select
SCK	Serial Clock
MOSI	Master Out Slave In (Data)
MISO	Master In Slave Out (Data)

18.4 Principle of Operation

The SPI is a high-speed synchronous data transfer interface. It allows high-speed communication between the device and peripheral devices. The SPI can operate as master or slave. As master, the SPI initiates and controls all data transactions. When transmitting data, the data register can be loaded with the next character to be transmitted during the current transmission. When receiving data, the data is transferred to the receive FIFO, and the receiver is ready for a new character. The SPI transaction format is shown in the following figure. Each transaction can contain one or more characters. The character size is eight bits.

Figure 18-2. SPI Transaction Format



The SPI master must pull the Slave Select Not line (SSN) of the desired slave low to initiate a transaction. The master and slave prepare data to send via their respective Shift registers, and the master generates the serial clock on the SCK line.

Data are shifted from master to slave on the Master Output Slave Input line (MOSI) and data are shifted from slave to master on the Master Input Slave Output line (MISO).

Each time a character is shifted out from the master, and a character is shifted out from the slave simultaneously. To signal the end of a transaction, the master pulls the SSN line high.

18.5 Clock Configuration

Before configuring the SPI registers, enable the clock of the SPI peripheral. Both SPI0, and SPI1 clocks are controlled individually. This is done by setting the SPI0_CORE_CLK_EN, SPI0_SCK_PHASE_INT_CLK_EN, SPI1_CORE_CLK_EN, and SPI1_SCK_PHASE_INT_CLK_EN bits in the LPMCU_CLOCK_ENABLES_0, and LPMCU_CLOCK_ENABLES_1 registers. The SPIn_CORE_CLK_EN bit enables the SPIn APB bus interface clock, whereas SPIn_SCK_PHASE_INT_CLK_EN clock enables the SPI master SCK clock. For more details on configuration, see Peripheral Clock Configuration.

18.6 Direct Memory Access

The Direct Memory Access (DMA) request lines are connected to the DMA Controller (DMAC). To use DMA requests with this peripheral, the DMAC must first be configured. For more details on configuration, see Direct Memory Access Controller.

18.7 Functional Description

18.7.1 Initialization

The initialization procedures are identical for both SPI0 and SPI1 modules.

- Configure the PINMUX_SEL_n registers (see I/O Peripheral Multiplexing and MEGAMUXing) to select the IO pins that need to be used as MOSI, MISO and SCK lines. By configuring pinmux to select SPI functionality on LP_GPIO_x, the GPIO controller functionality on that pin is bypassed.
- Configure the SSN pinmux:
  - When SPI is in the Master mode, configure the PINMUX_SEL_n of SSN with MUX0 to enable the SSN pin as general purpose I/O pin. The GPIO controller of specific SSN pin must be configured for output. For more details, see Functional Description of LP_GPIO_x I/O Pins.
    Note: Auto slave select feature is not supported in ATSAMB11 SPI. Therefore, the SSN pin must be controlled manually from the software.

    Any LP_GPIO_x pin (not only the pin marked with SPIn SSN) can be used as SSN pin for slave select.
  - When SPI is in the Slave mode, configure the PINMUX_SEL_n of SSN with specific MUX value as per I/O Port Function Multiplexing. Only LP_GPIO_x pins marked with SPIn SSN can be used as slave select pin.
  - When SPI is in the Master mode in multi-master environment, configure the PINMUX_SEL_n for SSN with specific MUX value as per I/O Port Function Multiplexing. Only LP_GPIO_x pins marked with SPIn SSN can be used as slave select pin.
- Select the SPIn module clock source by configuring the CLOCK[1:0] bits of the CLOCK_SOURCE_SELECT register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz and 3.25 MHz. The SPI module synchronizes the external data and clock using this internal clock source, selected from one of the four possible clock sources. To ensure that the data are reliably received and transmitted, it is necessary to wait during timing constraints.
  - In the Slave mode, the clock source must be at least eight times the external baud rate clock received by the module.
  - In the Master mode, the internal clock must be at least four times the external baud rate clock generated by the module.
- Set the SPI Transfer mode by configuring SCK_POLARITY and SCK_PHASE bits in the SPI_CONFIGURATION register. For more details, see SPI Transfer Modes.
- Write data order in LSB_FIRST_ENABLE bit of the SPI_CONFIGURATION register.
- To configure the SPI as Master:
  - Set the clock divider value by configuring the SPI_CLK_DIVIDER register. The value configured in this register determines the SCK frequency. The clock source selected in the CLOCK_SOURCE_SELECT register is divided by (n+1), where 'n' is the value set in the SPI_CLK_DIVIDER register.
    SCK Frequency = CLOCK_SOURCE_SELECT/(SPI_CLK_DIVIDER+1)

  - Set MASTER_ENABLE bit in the SPI_MASTER_MODE register.
- To configure the SPI as slave, clear MASTER_ENABLE bit in the SPI_MASTER_MODE register.

18.7.2 Enabling and Disabling the SPI Peripheral

- Enable the SPI peripheral by setting the ENABLE bit in the SPI_MODULE_ENABLE register. Setting the ENABLE bit enables the SPI module clock and can perform SPI transactions.
  The appropriate bits in the Interrupt Mask registers must be set to generate interrupts on SPI transaction as follows:
  - Register the SPI transmit and receive ISR function.
  - Enable the interrupts in the NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.
- Disable the SPI peripheral by clearing the ENABLE bit in the SPI_MODULE_ENABLE register. If this bit is cleared then the module does not take part in any SPI transactions.
  The SPI_ACTIVE bit in the SPI_BUS_STATUS register indicates whether the SPI peripheral is in Idle or Active state.
  Note: If the SPI_ACTIVE bit is set, the SPI module is in Active state and the configuration registers must not be changed during this period. If the registers are modified while a transaction is ongoing, the state of the SPI module cannot be ensured.

18.7.3 SPI Transfer Modes

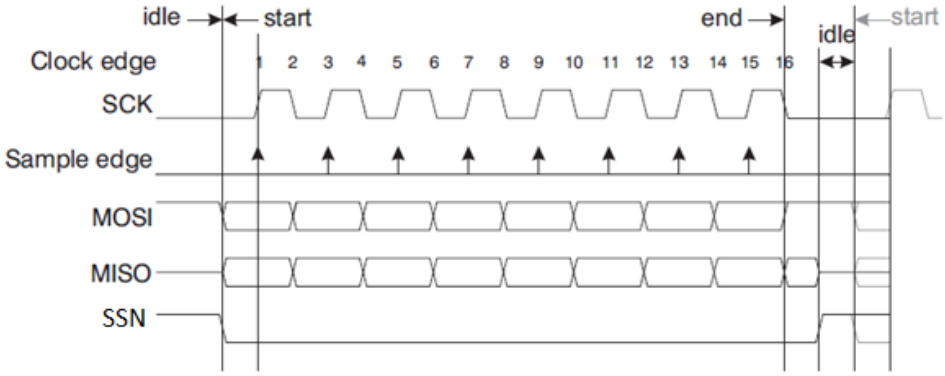
The SPI interface supports four standard modes as determined by the Clock Polarity (SCK_POLARITY) and Clock Phase (SCK_PHASE) settings. These modes are illustrated in the following table.

Table 18-2. SPI Modes

Mode	SCK_POLARITY	SCK_PHASE
0	0	0
1	0	1
2	1	0
3	1	1

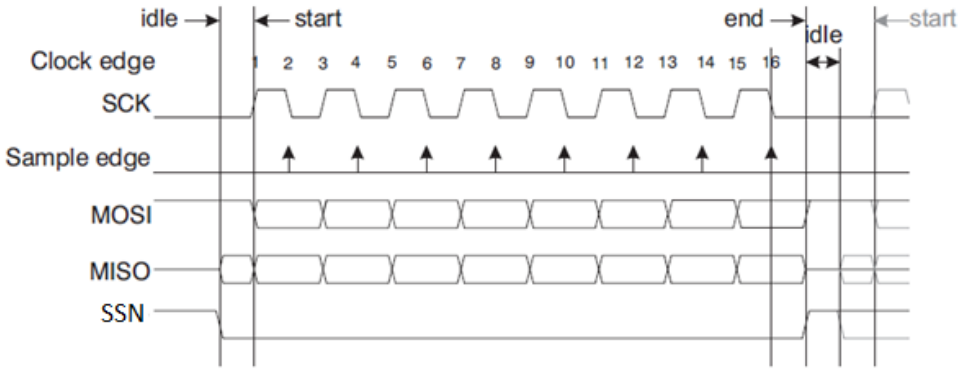
The following figures show the possible combinations of clock polarity and clock phase.

Figure 18-3. SCK_POLARITY = 0; SCK_PHASE = 0



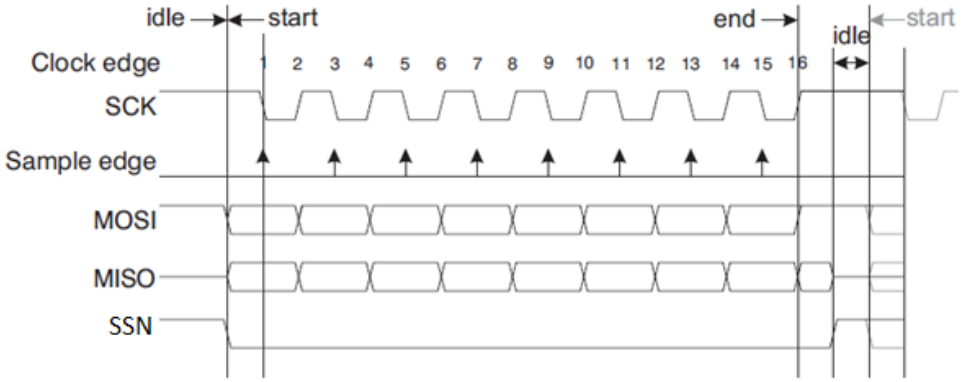
Data is shifted out on the even clock edges and sampled on the odd clock edges. The clock signal is normally low.

Figure 18-4. SCK_POLARITY = 0; SCK_PHASE = 1



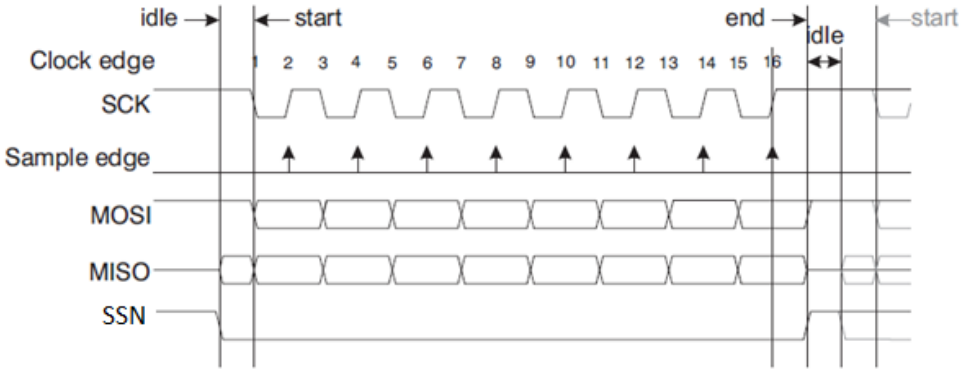
Data is shifted out on the odd clock edges and sampled on the even clock edges. The clock signal is normally low.

Figure 18-5. SCK_POLARITY = 1; SCK_PHASE = 0



Data is shifted out on the even clock edges and sampled on the odd clock edges. The clock signal is normally high.

Figure 18-6. SCK_POLARITY = 1; SCK_PHASE = 1



Data is shifted out on the odd clock edges and sampled on the even clock edges. The clock signal is normally high.

The SSN signal is asserted to start the transmission, and de-asserted at the end of a transmission. Data is transferred synchronously with a clock signal, and the master provides the clock signal for all transfers.

18.7.4 Transferring Data

- In the Master mode (MASTER_ENABLE bit is '1'), the SSN line must be configured as output. SSN can be assigned to any general purpose I/O pin. When the SPI is ready for data transaction, pull the SSN line low in the user application.

  When there is a space in transmit FIFO, indicated through TX_FIFO_NOT_FULL bit of the TRANSMIT_STATUS register, writing a character to the TRANSMIT_DATA register writes the data to transmit FIFO. The character is then transferred to the Shift register.

  When one character is shifted out from the master, another character is shifted in from the slave simultaneously. The content of the Shift register is transferred to the receive FIFO. The transfer takes place in the same clock cycle when the last data bit is shifted in. The arrival data on receive is indicated by '1' in RX_FIFO_NOT_EMPTY bit of the RECEIVE_STATUS register. The received data can be retrieved by reading the RECEIVE_DATA register.

  When the last character is transmitted and there is no valid data in the TRANSMIT_DATA register and transmit FIFO, the TX_FIFO_EMPTY bit indicates the transfer is complete. When the transaction is finished, the master pulls the SSN line high to notify the slave. The user must pull the SSN line high in the application as auto slave select option is not supported in ATSAMB11 SPI.
- In the Slave mode (MASTER_ENABLE bit is '0'), the SPI interface remains inactive with the MISO line tri-stated as long as the SSN pin is pulled high. Software may update the content of the TRANSMIT_DATA register at any time as long as the TX_FIFO_NOT_FULL bit in the TRANSMIT_STATUS register is set.
  When SSN is pulled low and SCK is running, the slave samples and shifts out the data according to the Transaction mode set. The content of the TRANSMIT_DATA register is loaded into the Shift register.

  Similar to the master, the slave receives one character for each character transmitted. A character is transferred into the receive FIFO within the same clock cycle and its last data bit is received. The received character can be retrieved from the RECEIVE_DATA register when the RX_FIFO_NOT_EMPTY bit is set.

  When the master pulls the SSN line high, the transaction is complete.

18.7.5 Receiver Error Bit

The SPI receiver has the FIFO Overflow bit (FIFO_OVERRUN), which is an error bit read from the Status register (RECEIVE_STATUS). When an error occurs, the bit is set until the RECEIVE_STATUS register is read.

If FIFO_OVERRUN bit is set upon FIFO overflow, the software must empty the receive FIFO by reading the RECEIVE_DATA register until the RX_FIFO_NOT_EMPTY flag is cleared.

18.7.6 Interrupt

The SPI module has two FIFOs, one for transmit and one for receive to automate transmission/reception.

The data written on the TRANSMIT_DATA register pushes one byte into the transmit FIFO. The TRANSMIT_STATUS register reflects the state of the SPI transmitter. If the corresponding bits are set in the TX_INTERRUPT_MASK register, then interrupts can be generated based on the bit that are set in this register. To enable the SPI interrupt on transmit, register the SPI transmit ISR function and enable the interrupts in the NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.

Reading from the RECEIVE_DATA register pops one byte from the receive FIFO. The RECEIVE_STATUS register reflects the state of the SPI receiver. If the corresponding bits are set in the

RX_INTERRUPT_MASK register, then interrupts can be generated based on the bits that are set in this register. To enable the SPI interrupt on reception, register the SPI receive ISR function and enable the interrupts in NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.

18.7.7 Example

The following is the sample example to enable SPI1 as master on LP_GPIO_17 (SCK), LP_GPIO_19 (MOSI), LP_GPIO_18 (MISO), and LP_GPIO_16(SS) pins with 100 kHz SCK frequency, and write 1 byte data:

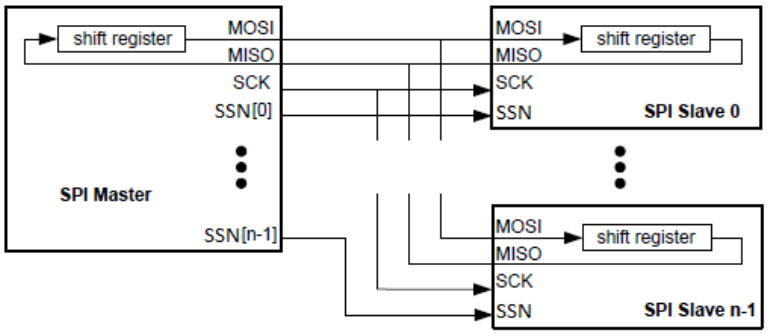
```
/* From Table 10-1. I/O Port Function Multiplexing,  LP_GPIO_17 (SCK), LP_GPIO_19 (MOSI),
LP_GPIO_18 (MISO), are configured as SPI1 through MUX4 configuration */
/* Write PINMUX_SEL_2 bits  [4:6],[8:10],[12:14], with 0x4 (MUX4) */
LPMCU_MISC_REGS0->PINMUX_SEL_2.reg |= (0x4<<4);
LPMCU_MISC_REGS0->PINMUX_SEL_2.reg |= (0x4<<8);
LPMCU_MISC_REGS0->PINMUX_SEL_2.reg |= (0x4<<12);
/* Configure Slave select pin LP_GPIO_16 as general purpose output */
/* Write PINMUX_SEL_2 bits [0:2] with 0x0 (MUX0) */
LPMCU_MISC_REGS0->PINMUX_SEL_2.reg |= (0x0<<0);
/* Write '1' to OUTENSET to enable output. LP_GPIO_16 is mapped to GPIO1_00 from I/O
multiplexing table*/
GPIO1->OUTENSET.reg |= (1 << 0);
/* Write '1' to DATAOUT to output logic high to disable SS*/
GPIO1->DATAOUT.reg |= (1 << 0);for(uint16_t i = 0; i < 0xFF; i++) {
/* Wait for the last data shift out */
}

/* Write '1' to DATAOUT to output logic high to disable SS*/
GPIO1->DATAOUT.reg |= (1 << 0);

/* Select clock source for SPI1 as 26MHz */
SPI1->CLOCK_SOURCE_SELECT.reg = 0;

/* Calculate CLK_DIVIDER for SCK clock frequency of 100kHz;
 SCK Frequency = CLOCK_SOURCE_SELECT/(SPI_CLK_DIVIDER+1)
SPI_CLK_DIVIDER = (26000000/100000)-1 = 259 */
SPI1->SPI_CLK_DIVIDER.reg = 259;
 /* Set clock polarity and clock phase for MODE0*/
SPI1->SPI_CONFIGURATION.bit.SCK_PHASE = 0x0;
SPI1>SPI_CONFIGURATION.bit.SCK_POLARITY = 0x0;
/* Set data order as MSB bit first */
SPI1->SPI_CONFIGURATION.bit.LSB_FIRST_ENABLE = 0x0;
/* Enable SPI1 in master mode */
SPI1->SPI_MASTER_MODE.reg = 1;
/* Enable SPI1 module */
SPI1->SPI_MODULE_ENABLE.reg = 1;
/* Select the slave by making Logic Low on SS pin */
GPIO1->DATAOUT.reg &=~(1 << 0);
/* Check for TX buffer is not full TX_FIFO_NOT_FULL */
while(!(SPI1->TRANSMIT_STATUS.bit.TX_FIFO_NOT_FULL));
SPI1->TRANSMIT_DATA.reg = 0x77; //data
/* Check for all the data is transmitted Tx buffer is  empty TX_FIFO_EMPTY*/
while(!(SPI1->TRANSMIT_STATUS.bit.TX_FIFO_EMPTY));
```

18.8 Additional Features

18.8.1 Master with Several Slaves

Master with multiple slaves is supported in the ATSAMB11 SPI. If the bus has several SPI slaves, an SPI master can use general purpose I/O pins to control the SSN line on each of the slaves on the bus, as shown in the following figure. In this configuration, a single selected SPI slave drives the tri-state MISO line.

Figure 18-7. Multiple Slaves in Parallel



18.8.2 Master in Multi-Master Environment

When the SPI is configured as Master, the user application can determine the direction of the SSN pin. If SSN is configured as an output, the pin is a general output pin which does not affect the SPI system. Typically, the pin drives the SSN pin of the SPI Slave. If SSN is configured as SSN pin through pinmux configuration, it must be held high to ensure Master SPI operation.

In addition to the Master mode, it is also possible that a system has multiple masters. In this case, there must be some bus arbitration to avoid bus contention. If a master initiates a transaction it drives the SSN pin of another master to low by preventing it from taking control of the interface.

If both masters decide to initiate a transaction at the same time, a failure is detected and the FAULT bit in the RECEIVE_STATUS register is set. This Fault detection feature can be enabled by setting FAULT_ENABLE bit in the SPI_FAULT_ENABLE register.

When a Fault is detected on SPI master, the MASTER_ENABLE bit should be reset to zero, which aborts the transaction and places the SPI module into the Slave mode and Idle state.

18.8.3 Single Data Direction Operation

Single data direction operation provides an alternate function on MISO or MOSI pin. When the BIDIRECTIONAL_ENABLE bit is set in the SPI_CONFIGURATION register, then the SPI module goes to Bidirectional mode and only one input/output is used. The OUTPUT_ENABLE bit in the SPI_CONFIGURATION register selects whether the communication pin is an input or an output as shown in the following table.

Table 18-3. Pin Assignments when BIDIRECTIONAL_ENABLE = 1

MASTER_ENABLE	OUTPUT_ENABLE	MISO	MOSI
0	0	Slave Input	Not used
0	1	Slave Output	Not used
1	0	Not used	Master Input
1	1	Not used	Master Output

18.8.4 DMA Operation

The SPI generates the following DMA requests:

- Data received (RX) – The request is set when data is available in the receive FIFO. The request is cleared when the RECEIVE_DATA register is read.

- Data transmit (TX) – The request is set when the transmit FIFO is empty. The request is cleared when the TRANSMIT_DATA register is written.

18.9 Power Management

If the system goes to the Ultra-Low Power mode, the SPI peripheral shuts down. The SPI Configuration registers lose their content, and cannot be restored when powered-up again. Therefore, the user must reconfigure the SPI peripheral at power-up to ensure it is in a well-defined state before use.

For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains how sleep and wake-up are controlled.

18.10 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.									
0x40006000 (SPI0), 0x40007000 (SPI1)	SPI	TRANSMIT_DATA	7:0	TX_BYTE[7:0]								
0x40006004 (SPI0), 0x40007004 (SPI1)	SPI	RECEIVE_DATA	7:0	RX_BYTE[7:0]								
0x40006008 (SPI0), 0x40007008 (SPI1)	SPI	TRANSMIT_STATUS	7:0				TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL	
0x4000600C (SPI0), 0x4000700C (SPI1)	SPI	RECEIVE_STATUS	7:0			FIFO_OVERRUN	FAULT	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY	
0x40006010 (SPI0), 0x40007010 (SPI1)	SPI	CLOCK_SOURCE_SELECT	7:0							CLOCK[1:0]		
0x40006014 (SPI0), 0x40007014 (SPI1)	SPI	SPI_CLK_DIVIDER	7:0	SPI_DIVIDE_RATIO[7:0]								
			15:8	SPI_DIVIDE_RATIO[15:8]								
0x40006018 (SPI0), 0x40007018 (SPI1)	SPI	SPI_MODULE_ENABLE	7:0								ENABLE	
0x4000601C (SPI0), 0x4000701C (SPI1)	SPI	SPI_MASTER_MODE	7:0								MASTER_ENABLE	
0x40006020 (SPI0), 0x40007020 (SPI1)	SPI	SPI_FAULT_ENABLE	7:0								FAULT_ENABLE	
0x40006024 (SPI0), 0x40007024 (SPI1)	SPI	SPI_CONFIGURATION	7:0	RX_DONE_SYNC_ENABLE	SSN_SYNC_ENABLE	SSN_SHIFT_ENABLE	OUTPUT_ENABLE	BIDIRECTIONAL_ENABLE	LSB_FIRST_ENABLE	SCK_PHASE	SCK_POLARITY	
0x40006028 (SPI0), 0x40007028 (SPI1)	SPI	SPI_BUS_STATUS	7:0								SPI_ACTIVE	
0x4000602C (SPI0), 0x4000702C (SPI1)	SPI	TX_INTERRUPT_MASK	7:0				TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK	
0x40006030 (SPI0), 0x40007030 (SPI1)	SPI	RX_INTERRUPT_MASK	7:0			FIFO_OVERRUN_MASK	FAULT_DETECT_MASK	RX_FIFO_0P75_FULL_MASK	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK	

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0				MUX_0[4:0]				
			15:8				MUX_1[4:0]				
			23:16				MUX_2[4:0]				
			31:24				MUX_3[4:0]				
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0				MUX_4[4:0]				
			15:8				MUX_5[4:0]				
			23:16				MUX_6[4:0]				
			31:24				MUX_7[4:0]				
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0				MUX_8[4:0]				
			15:8				MUX_9[4:0]				
			23:16				MUX_10[4:0]				
			31:24				MUX_11[4:0]				
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0				MUX_12[4:0]				
			15:8				MUX_13[4:0]				
			23:16				MUX_14[4:0]				
			31:24				MUX_15[4:0]				
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0				MUX_16[4:0]				
			15:8				MUX_17[4:0]				
			23:16				MUX_18[4:0]				
			31:24				MUX_19[4:0]				
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0				MUX_20[4:0]				

18.11 Register Description

18.11.1 SPI Transmit Data

Name: TRANSMIT_DATA
Reset: 0x00

Absolute Address: 0x40006000(SPI0), 0x40007000(SPI1)

This register is a part of SPI registers. This register pushes one byte of data into the transmit FIFO of SPI module.

Bit	7	6	5	4	3	2	1	0
	TX_BYTE[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TX_BYTE[7:0]
Writing these eight bit pushes one byte into the transmit FIFO

18.11.2 SPI Receive Data

Name: RECEIVE_DATA
Reset: 0x00

Absolute Address: 0x40006004(SPI0), 0x40007004(SPI1)

This register is a part of SPI registers. This register pops one byte of data from the receive FIFO of the SPI module.

Bit	7	6	5	4	3	2	1	0
	RX_BYTE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RX_BYTE[7:0]
Reading these eight bit pops one byte from the receive FIFO

18.11.3 SPI Transmit Status

Name: TRANSMIT_STATUS
Reset: 0x00

Absolute Address: 0x40006008(SPI0), 0x40007008(SPI1)

This register is a part of SPI registers and provides the status of SPI transmit operation.

Bit	7	6	5	4	3	2	1	0
				TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – TX_FIFO_EMPTY
This bit is set if the Tx FIFO is completely empty

Bit 3 – TX_FIFO_0P75_EMPTY
This bit is set if the Tx FIFO is three-quarters empty

Bit 2 – TX_FIFO_0P5_EMPTY
This bit is set if the Tx FIFO is half empty

Bit 1 – TX_FIFO_0P25_EMPTY
This bit is set if the Tx FIFO is one quarter empty

Bit 0 – TX_FIFO_NOT_FULL
This bit is set if there is space for at least one more byte in the Tx FIFO

18.11.4 SPI Receive Status

Name: RECEIVE_STATUS
Reset: 0x00

Absolute Address: 0x4000600C (SPI0), 0x4000700C(SPI1)

This register is a part of SPI registers. This register provides the status of SPI receive operation.

Bit	7	6	5	4	3	2	1	0
			FIFO_OVERRUN	FAULT	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – FIFO_OVERRUN
This bit is set when a character is received but there is no space in the Rx FIFO to store
This bit resets after the status register is read

Bit 4 – FAULT
This bit is set when two masters access the SPI bus at the same time and the SPI interface sets in to the Slave mode
This bit resets after the status register is read

Bit 3 – RX_FIFO_0P75_FULL
This bit is set if the Rx FIFO is three-quarters full

Bit 2 – RX_FIFO_0P5_FULL
This bit is set if the Rx FIFO is half full

Bit 1 – RX_FIFO_0P25_FULL
This bit is set if the Rx FIFO is one quarter full

Bit 0 – RX_FIFO_NOT_EMPTY
This bit is set if there is at least space for one more byte in the Rx FIFO

18.11.5 SPI Clock Source Select

Name: CLOCK_SOURCE_SELECT
Reset: 0x00

Absolute Address: 0x40006010 (SPI0), 0x40007010(SPI1)

This register is a part of SPI registers. This register allows the user to select the input clock source for SPI module.

Bit	7	6	5	4	3	2	1	0
							CLOCK[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 1:0 – CLOCK[1:0]
Selects the input clock for I2C module

CLOCK[1:0]	Description
0	26 MHz clock
1	13 MHz clock
2	6.5 MHz clock
3	3.25 MHz clock

18.11.6 SPI SCK Clock Divider

Name: SPI_CLK_DIVIDER
Reset: 0x0000

Absolute Address: 0x40006014 (SPI0), 0x40007014(SPI1)

This register is a part of SPI registers. This register sets the divide ratio used to generate the SCK clock from input clock of the SPI module.

Bit	15	14	13	12	11	10	9	8
	SPI_DIVIDE_RATIO[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SPI_DIVIDE_RATIO[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – SPI_DIVIDE_RATIO[15:0]
Sets the divide ratio to generate the SCK clock signal from the clock selected by the CLOCK_SOURCE_SELECT register.
SCK Frequency = CLOCK[1:0] / (SPI_DIVIDE_RATIO[15:0]+1)
Note: The minimum division is by 2; a value of 0 is not recommended for SPI_DIVIDE_RATIO[15:0] as it may result in unpredictable behavior.

18.11.7 SPI Module Enable

Name: SPI_MODULE_ENABLE
Reset: 0x00

Absolute Address: 0x40006018 (SPI0), 0x40007018(SPI1)

This register is a part of SPI registers and allows the user to enable/disable the SPI peripheral.

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R/W
Reset								0

Bit 0 – ENABLE
Writing '0' to this bit disables the SPI module
Writing '1' to this bit enables the SPI module

18.11.8 SPI Master Mode Enable

Name: SPI_MASTER_MODE
Reset: 0x00

Absolute Address: 0x4000601C (SPI0), 0x4000701C(SPI1)

This register is a part of SPI registers. This register allows the user to select SPI between Master and Slave modes.

Bit	7	6	5	4	3	2	1	0
								MASTER_ENABLE
Access								R/W
Reset								0

Bit 0 – MASTER_ENABLE
Writing '0' to this bit enables SPI in the Slave mode
Writing '1' to this bit enables SPI in the Master mode

18.11.9 SPI Fault Detection Enable

Name: SPI_FAULT_ENABLE
Reset: 0x00

Absolute Address: 0x40006020 (SPI0), 0x40007020(SPI1)

This register is a part of SPI registers. This register allows the user to enable the bus connection when SPI bus is in multi-master environment. If the FAULT_ENABLE bit is enabled, SPI bus contention is detected and the FAULT bit in the RECEIVE_STATUS register is set, forcing the SPI module to switch to the Idle state. When a Fault is detected, the current SPI transaction is abandoned and the interface switches to the Slave mode in the Wait state.

Bit	7	6	5	4	3	2	1	0
								FAULT_ENABLE
Access								R/W
Reset								0

Bit 0 – FAULT_ENABLE
Active High to enable SPI Fault Detection
Writing '0' to this bit ignores the Fault on SPI in the Master mode
Writing '1' to this bit enables Fault detection on SPI in the Master mode

18.11.10 SPI Configuration

Name: SPI_CONFIGURATION
Reset: 0xE0

Absolute Address: 0x40006024 (SPI0), 0x40007024(SPI1)

This register is a part of SPI registers. This register allows the user to configure the SPI Configuration modes.

Bit	7	6	5	4	3	2	1	0
	RX_DONE_SYNC_ENABLE	SSN_SYNC_ENABLE	SSN_SHIFT_ENABLE	OUTPUT_ENABLE	BIDIRECTIONAL_ENABLE	LSB_FIRST_ENABLE	SCK_PHASE	SCK_POLARITY
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	0	0	0	0	0

Bit 7 – RX_DONE_SYNC_ENABLE
Writing '0' to this bit bypasses internal synchronization of RX_DONE signal
Writing '1' to this bit allows RX_DONE signal to synchronize internally before being used

Bit 6 – SSN_SYNC_ENABLE
Writing '0' to this bit bypasses internal synchronization of SSN signal
Writing '1' to this bit allows SSN signal to synchronize internally before being used

Bit 5 – SSN_SHIFT_ENABLE
Writing '0' to this bit enables the Shift register only when SSN pin is asserted
Writing '1' to this bit enables the Shift register always

Bit 4 – OUTPUT_ENABLE
When SPI is in the Bidirectional mode this bit selects whether the communication pin acts as an input or an output as shown in the following table.

MASTER_ENABLE	OUTPUT_ENABLE	MISO	MOSI
0	0	Slave Input	Not used
0	1	Slave Output	Not used
1	0	Not used	Master Input
1	1	Not used	Master Output

Bit 3 – BIDIRECTIONAL_ENABLE
This bit enables the Bidirectional mode of operation
Writing '0' to this bit configures the SPI module in the Unidirectional mode, when the SPI module uses one input and one output (MISO and MOSI).
Writing '1' to this bit configures the SPI module in the Bidirectional mode, when the SPI module uses only one input/output.

Bit 2 – LSB_FIRST_ENABLE
This bit selects the data bit order
Writing '0' to this bit transmits the most significant bit of the data first
Writing '1' to this bit transmits the least significant bit of the data first

Bit 1 – SCK_PHASE
This bit selects clock edge for data sampling and launching
Writing '0' to this bit samples the data bits on the first edge and the next bit is output on the next edge
Writing '1' to this bit outputs the data bits on the first edge and samples the data on the next edge

Bit 0 – SCK_POLARITY
This bit selects the level of SCK in the Idle state
Writing '0' to this bit sets SCK low in the Idle state
Writing '1' to this bit sets SCK high in the Idle state

18.11.11 SPI Bus Status

Name: SPI_BUS_STATUS
Reset: 0x00

Absolute Address: 0x40006028 (SPI0), 0x40007028(SPI1)

This register is a part of SPI registers and indicates the SPI module status.

Bit	7	6	5	4	3	2	1	0
								SPI_ACTIVE
Access								R
Reset								0

Bit 0 – SPI_ACTIVE
The SPI configuration registers must not be changed while this bit is set. If the registers are modified whilst a transaction is ongoing, the state of the SPI module is not ensured.

Read Value	Description
0	SPI is idle
1	SPI is active

18.11.12 SPI Transmit Mask Interrupt

Name: TX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x4000602C (SPI0), 0x4000702C(SPI1)

This register is a part of SPI registers. This register is used to enable or disable the generation of SPI transmission interrupts. During the SPI transmission interrupt, if a bit in the TX_INTERRUPT_MASK register is set and its corresponding bit in the TRANSMIT_STATUS register is set, then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
				TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – TX_FIFO_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_EMPTY interrupt

Bit 3 – TX_FIFO_0P75_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P75_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P75_EMPTY interrupt

Bit 2 – TX_FIFO_0P5_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P5_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P5_EMPTY interrupt

Bit 1 – TX_FIFO_0P25_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P25_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P25_EMPTY interrupt

Bit 0 – TX_FIFO_NOT_FULL_MASK
Writing '0' to this bit disables the TX_FIFO_NOT_FULL interrupt
Writing '1' to this bit enables the TX_FIFO_NOT_FULL interrupt

18.11.13 SPI Receive Mask Interrupt

Name: RX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x40006030 (SPI0), 0x40007030(SPI1)

This register is a part of SPI registers. This register is used to enable or disable the generation of SPI receive interrupts. During the SPI receive interrupt, if a bit in the RX_INTERRUPT_MASK register is set and its corresponding bit in the RECEIVE_STATUS register is set, then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
			FIFO_OVERRUN_MASK	FAULT_DETECT_MASK	RX_FIFO_0P75_FULL_MASK	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – FIFO_OVERRUN_MASK
Writing '0' to this bit disables the FIFO_OVERRUN interrupt
Writing '1' to this bit enables the FIFO_OVERRUN interrupt

Bit 4 – FAULT_DETECT_MASK
Writing '0' to this bit disables the FAULT_DETECT interrupt
Writing '1' to this bit enables the FAULT_DETECT interrupt

Bit 3 – RX_FIFO_0P75_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P75_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P75_FULL interrupt

Bit 2 – RX_FIFO_0P5_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P5_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P5_FULL interrupt

Bit 1 – RX_FIFO_0P25_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P25_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P25_FULL interrupt

Bit 0 – RX_FIFO_NOT_EMPTY_MASK
Writing '0' to this bit disables the RX_FIFO_NOT_EMPTY interrupt
Writing '1' to this bit enables the RX_FIFO_NOT_EMPTY interrupt

19. UART Interface

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA provide Universal Asynchronous Receiver/ Transmitter (UART) interfaces for serial communication. It contains two UART interfaces; 2-Pin mode for data only, and a 4-pin interface for flow control and data transfer. The UART interfaces are compatible with the RS-232 standard, where theATSAMB11-XR2100A and ATSAMB11-ZR210CA operate as Data Terminal Equipment (DTE).

19.1 Features

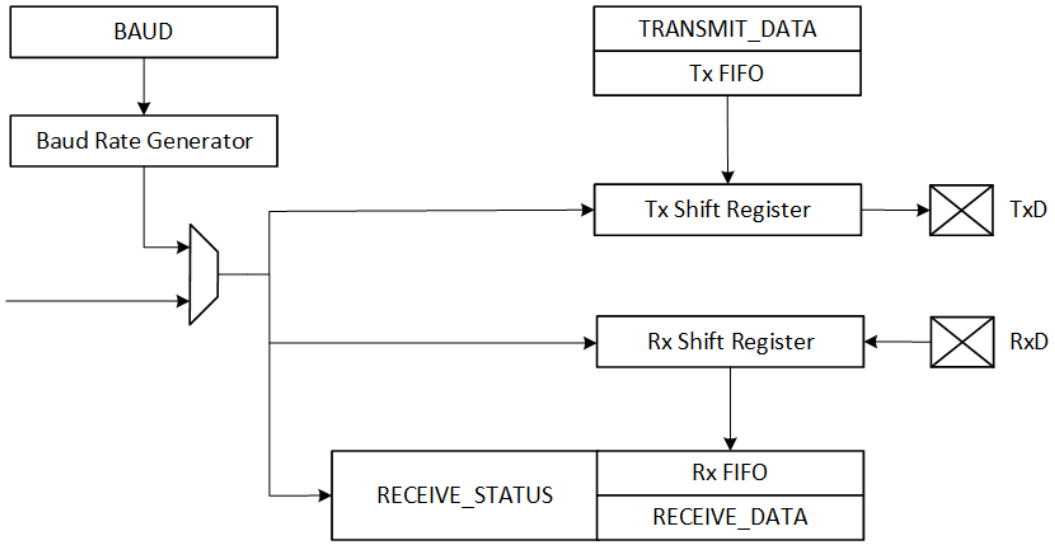
The following are the UART interface features:

- Two UART (UART0, UART1) peripherals
- Full-duplex operation
- Asynchronous operation
- Four different internal clock inputs (between 26 MHz, 13 MHz, 6.5 MHz, and 3.25 MHz) for Baud-rate generation.
- Supports serial frames with 7, 8 data bits and 1 or 2 stop bits
- Odd, even, mark, or space parity generation and parity check
- FIFO overflow and frame error detection
- Receive timeout indication
- RTS and CTS flow control
- Four bytes FIFO transmitter, Four bytes FIFO receiver
- DMA support

19.2 Block Diagram

The following is the block diagram of UART.

Figure 19-1. UART Block Diagram



19.3 Principle of Operation

The UART uses the following lines for data transfer:

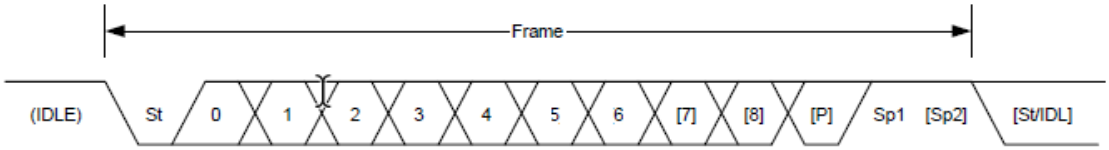
- RXD for receiving
- TXD for transmitting

The UART data transfer is frame-based. A serial frame consists of:

- One Start bit
- Seven or eight Data bits (LSB bit is transmitted first)
- No Parity bit, Mark Parity bit, Space Parity bit, and Even or Odd Parity bit
- One or two Stop bits

A frame starts with the Start bit followed by one character of Data bits. If parity is enabled, the Parity bit is inserted after the Data bits and before the first Stop bit. After the Stop bit(s) in frame, either the next frame can follow immediately, or the communication line can return to the Idle (high) state. The following figure illustrates the possible frame formats.

Figure 19-2. UART Data Transfer Frame



Note: Brackets denote optional bits.

St–Start bit. The signal is always low for Start bit.

n, [n]–Data bits. The Data bits ranges from 0 to [7..8]

[P]–Parity bit. The Parity bit can be either odd, even, mark or space.

Sp, [Sp]–Stop bit. The signal is always high for Stop bit.

IDLE–The signal is always high in this state and no frame is transferred on the communication line.

19.4 Clock Configuration

Before configuring the UART registers, enable the clock of the UART peripheral. Both UART0, and UART1 clocks are controlled individually. This is done by setting the UART0_CORE_CLK_EN, UART0_IF_CLK_EN, UART1_CORE_CLK_EN, and UART1_IF_CLK_EN bits in the LPMCU_CLOCK_ENABLES_0 register. The UARTn_CORE_CLK_EN enables the UARTn APB bus interface clock, whereas UARTn_IF_CLK_EN bit enables the UART interface clock. For more details on configuration, see Peripheral Clock Configuration.

19.5 Direct Memory Access

The Direct Memory Access (DMA) request lines are connected to the DMA Controller (DMAC). To use DMA requests with this peripheral, the DMAC must be configured first. For more details on configuration, see Direct Memory Access Controller.

19.6 Functional Description

19.6.1 Initialization

The initialization procedures are identical for both UART0 and UART1 modules.

- Configure the PINMUX_SEL_n and/or MEGAMUX registers (see I/O Peripheral Multiplexing and MEGAMUXing) to select the IO pins that need to be used as RXD, TXD, RTS, CTS (RTS, CTS are optional and required only when flow control is enabled) lines. By configuring pinmux to select UART functionality on LP_GPIO_x, the GPIO controller functionality is bypassed on that pin.
- Select the UARTn module clock source by configuring the CLOCK_SELECT[1:0] bits of UART_CLOCK_SOURCE register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz and 3.25 MHz. The UART module synchronizes the external data and clock using the internal clock source, selected from one of the four clock sources.
- Set the baud rate by configuring the UART_BAUD_RATE register. The register consists of two fields, INTEGER_DIVISION[15:3] specifies the integral division of the clock, and FRACTIONAL_DIVISION[2:0] specifies a fractional division of the clock. The integral division specifies the division of the clock in terms of a division by n.
  Baud rate is calculated using following formulas:

  Baud Rate = CLOCK_SOURCE_SELECT / Clock_Division

  Clock_Division = INTEGER_DIVISION.(FRACTIONAL_DIVISION/8)

  Where, '.' indicates decimal point.
- Set the number of bits per character in the UART_CONFIGURATION register by configuring NUMBER_OF_BITS.
- Enable or disable the parity check and generation by configuring PARITY_ENABLE bit in the UART_CONFIGURATION register. The Parity modes such as, space, mark, odd, even can be configured through PARITY_MODE[1:0] bits.
- Configure number of stop bits in STOP_BITS bit in the UART_CONFIGURATION register.
- The hardware flow control can be enabled by setting CTS_ENABLE bit in the UART_CONFIGURATION register. This is an optional feature, and if hardware flow control is enabled then the pinmuxing must be configured for RTS and CTS pins.

19.6.2 Data Transmission

Data transmission is initiated by writing the data that are to be transferred into the TRANSMIT_DATA register, which places the data in Tx FIFO. Then, the data in Tx FIFO is moved to the Shift register when the Shift register is empty and ready to send a new frame. After the Shift register is loaded with data, the data frame is transmitted.

When the entire data frame including Stop bit(s) is transmitted and no new data is written to the TRANSMIT_DATA register, the transmit completion can be indicated by reading TX_FIFO_EMPTY flag in the TRANSMIT_STATUS register. The optional interrupt is generated when the corresponding bit in the TX_INTERRUPT_MASK register is set.

19.6.3 Data Reception

The receiver accepts data when a valid Start bit is detected. Each bit following the Start bit is sampled according to the baud rate, and shifted into the Receive Shift register until the first Stop bit of a frame is received. The second Stop bit is ignored by the receiver.

When the first Stop bit is received and a complete serial frame is available in the Receive Shift register, the content of the Shift register is moved into the Rx FIFO. This is indicated by setting RX_FIFO_NOT_EMPTY flag in the RECEIVE_STATUS register. The optional interrupt is generated when the corresponding bit in the RX_INTERRUPT_MASK register is set.

The received data can be read from the RECEIVE_DATA register.

19.6.3.1 Error Bits

The UART receiver contains three error bits in the RECEIVE_STATUS register as follows:

- Frame error (FRAMING_ERROR)
- FIFO overflow (FIFO_OVERRUN)
- Parity error (PARITY_ERROR)

When an error occurs, the corresponding error bit is set in the RECEIVE_STATUS register.

The FRAMING_ERROR bit is set when seven or eight bits data is received (according to the configuration) and no Stop bit is received.

The FIFO_OVERRUN bit is set when a character is received but there is no space in the Rx FIFO for storage. When the FIFO_OVERRUN is set, the software must empty the receive FIFO by reading the RECEIVE_DATA register until the RX_FIFO_NOT_EMPTY bit is cleared.

The PARITY_ERROR bit is set when a character is received with an unexpected Parity bit.

19.6.3.2 Asynchronous Operational Range

The operational range of an asynchronous reception depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size (in number of bits). In addition, the operational range of the receiver depends on the difference between the received bit rate and the internally generated baud rate. If the baud rate of an external transmitter is too high or too low compared to the internally-generated baud rate, then the receiver cannot synchronize the frames to the Start bit.

There are two sources for mismatch in baud rate, such as:

- The internal clock source with some minor instability.
- The Baud Rate Generator cannot perform an division of the internal clock frequency to get the desired baud rate. In this case, the UART_BAUD_RATE register value must be set to give the lowest error.

19.6.3.3 Example

The following is the sample example to transmit a byte on UART0 using LP_GPIO_2 (RXD), and LP_GPIO_3 (TXD) pins without flow control with 115200 baud rate, odd parity, and 1 stop bit:

```
/* From Table 10-1. I/O Port Function Multiplexing,  LP_GPIO_2 (RXD), LP_GPIO_3 (TXD), are
configured as UART0 through MUX2 configuration */
/* Write PINMUX_SEL_0 bits  ,[8:10],[12:14], with 0x2 (MUX2) */
LPMCU_MISC_REGS0->PINMUX_SEL_0.reg |= (0x2<<8);
LPMCU_MISC_REGS0->PINMUX_SEL_0.reg |= (0x2<<12);
/* Select clock source for UART0 as 26MHz */
UART0->UART_CLOCK_SOURCE.reg = 0x0;

/* Calculate CLK_DIVIDER for baud rate = 115200
 Baud Rate = CLOCK_SOURCE_SELECT / Clock_Division
Clock_Division = 26000000/115200 = 225.694
Clock_Division = INTEGER_DIVISION.(FRACTIONAL_DIVISION/8)
INTEGER_DIVISION = 225
FRACTIONAL_DIVISION = 0.694*8 = 5*/
UART0->UART_BAUD_RATE.reg = (225<<3) | (5<<0)
 /* Configure 8-bit data mode*/
UART0->UART_CONFIGURATION.bit.NUMBER_OF_BITS = 0;
/* Set odd parity */
UART0->UART_CONFIGURATION.bit.PARITY_MODE= 0x1;
```

```
UART0->UART_CONFIGURATION.bit.PARITY_ENABLE= 0x1;
/* Set 1 stop bit */
UART0->UART_CONFIGURATION.bit.STOP_BITS = 0;
/* Disable hardware flow control */
UART0->UART_CONFIGURATION.bit.CTS_ENABLE = 0;
/* Check for Tx buffer has space  */
while(!(UART0->TRANSMIT_STATUS.bit.TX_FIFO_NOT_FULL));
/* Transmit 1 byte data */
UART0->TRANSMIT_DATA.reg = 0x77;
```

19.6.4 Interrupt

The UART module has two FIFOs, one for transmit and one for receive to automate transmission/ reception.

The data written on the TRANSMIT_DATA register pushes one byte into the transmit FIFO. The TRANSMIT_STATUS register reflects the state of the UART transmitter. If the corresponding bits are set in the TX_INTERRUPT_MASK register then interrupts can be generated based on the bit that are set in this register. To enable the UART interrupt on transmit, register the UART transmit ISR function and enable the interrupts in NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.

Reading from the RECEIVE_DATA register pops one byte from the receive FIFO. The RECEIVE_STATUS register reflects the state of the UART receiver. If the corresponding bits are set in the RX_INTERRUPT_MASK register, then interrupts can be generated based on the bit that are set in this register. To enable the UART interrupt on reception, register the UART receive ISR function and enable the interrupts in NVIC interrupt controller registers. For more details, see Nested Vector Interrupt Controller.

19.6.5 Additional Features

19.6.5.1 Parity

Even parity, odd parity, mark parity, or space parity can be selected for error checking by writing to PARITY_MODE[1:0] bits and writing '1' to PARITY_ENABLE bit in the UART_CONFIGURATION register.

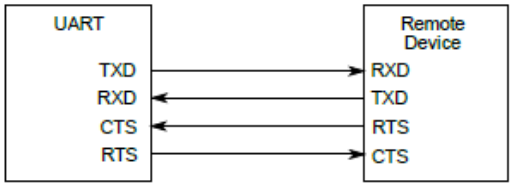
- If even parity is selected (PARITY_MODE[1:0]=0), the parity bit of an outgoing frame is '1' when the data contain odd number of bits that are '1', making the total number of '1' as even.
- If odd parity is selected (PARITY_MODE[1:0]=1), the parity bit of an outgoing frame is '1' when the data contain even number of bits that are '1', making the total number of '1' as odd.
- If space parity is selected (PARITY_MODE[1:0]=2), the parity bit of an outgoing frame is always ‘0’.
- If mark parity is selected (PARITY_MODE[1:0]=3), the parity bit of an outgoing frame is always ‘1’.

When parity checking is enabled, the parity checker calculates the parity of the data bits in incoming frames and compares the result with the parity bit of the corresponding frame. If a parity error is detected, the PARITY_ERROR bit in the RECEIVE_STATUS is set.

19.6.5.2 Hardware Handshaking

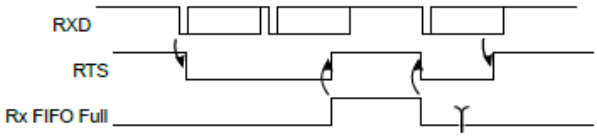
The UART features an out-of-band hardware handshaking flow control mechanism, implemented by connecting the RTS and CTS pins with the remote device, as shown in the following figure.

Figure 19-3. Connection with a Remote Device for Hardware Handshaking



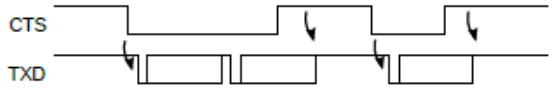
When the Rx FIFO is full, the receiver drives the RTS pin high. This notifies the remote device to stop the transfer after the ongoing transmission. When the receive FIFO is full, RTS is set and the frame which is received is stored in the Shift register until the receive FIFO is not full.

Figure 19-4. Receiver Behavior when Operating with Hardware Handshaking



The current CTS_ACTIVE status is indicated in the TRANSMIT_STATUS register. Character transmission starts only when CTS pin is low. When CTS pin is high, the transmitter completes the ongoing transmission and stops transmitting.

Figure 19-5. Transmitter Behavior when Operating with Hardware Handshaking



19.6.5.3 Receive Time-out

The RECEIVE_TIMEOUT register sets the time-out period. A time-out counter is reloaded with this value when, the RECEIVE_STATUS register is read or a character is received. The counter is decremented at the frequency of the baud rate clock.

When the counter reaches zero, the TIMEOUT bit in the RECEIVE_STATUS register is set. If the corresponding bit in the RX_INTERRUPT_MASK register is set then an interrupt is generated. This functionality is particularly useful in high throughput, low latency systems.

19.6.5.4 DMA Operation

The UART generates the following DMA requests:

- Data received (RX) – The request is set when data is available in the receive FIFO. The request is cleared when the RECEIVE_DATA register is read.
- Data transmit (TX) – The request is set when the transmit FIFO is empty. The request is cleared when the TRANSMIT_DATA register is written.

19.7 Power Management

If the system goes to the Ultra-Low Power mode, the UART peripheral shuts down. The UART configuration registers lose their content, and cannot be restored when powered-up again. Therefore, the user must reconfigure the UART peripheral at power-up to ensure it is in a well-defined state before use.

For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains how sleep and wake-up are controlled.

19.8 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40004000 (UART0), 0x40005000 (UART1)	UART	TRANSMIT_DATA	7:0	TX_BYTE[7:0]							
0x40004004 (UART0), 0x40005004 (UART1)	UART	TRANSMIT_STATUS	7:0			CTS_ACTIVE	TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL
0x40004008 (UART0), 0x40005008 (UART1)	UART	TX_INTERRUPT_MASK	7:0			CTS_ACTIVE_MASK	TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK
0x40004010 (UART0), 0x40005010 (UART1)	UART	RECEIVE_DATA	7:0	RX_BYTE[7:0]							
0x40004014 (UART0), 0x40005014 (UART1)	UART	RECEIVE_STATUS	7:0	FRAMING_ERROR	FIFO_OVERRUN	PARITY_ERROR	TIMEOUT	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY
0x40004018 (UART0), 0x40005018 (UART1)	UART	RX_INTERRUPT_MASK	7:0	FRAMING_ERROR_MASK	FIFO_OVERRUN_MASK	PARITY_ERROR_MASK	TIMEOUT_MASK	RX_FIFO_0P75_FULL_MAS	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK
0x4000401C (UART0), 0x4000501C (UART1)	UART	RECEIVE_TIMEOUT	7:0	TIMEOUT_VALUE[7:0]							
0x40004020 (UART0), 0x40005020 (UART1)	UART	UART_CONFIGURATION	7:0			CTS_ENABLE	STOP_BITS	PARITY_MODE[1:0]		PARITY_ENABLE	NUMBER_OF_BITS
0x40004024 (UART0), 0x40005024 (UART1)	UART	UART_BAUD_RATE	7:0	INTEGER_DIVISION[4:0]					FRACTIONAL_DIVISION[2:0]		
			15:8	INTEGER_DIVISION[12:5]							
0x40004028 (UART0), 0x40005028 (UART1)	UART	UART_CLOCK_SOURCE	7:0							CLOCK_SELECT[1:0]	
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0			MEGAMUX_SEL[5:0] LP_GPIO_0					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_1					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_2					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_3					

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0			MEGAMUX_SEL[5:0] LP_GPIO_4					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_5					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_6					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_7					
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0			MEGAMUX_SEL[5:0] LP_GPIO_8					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_9					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_10					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_11					
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0			MEGAMUX_SEL[5:0] LP_GPIO_12					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_13					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_14					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_15					
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0			MEGAMUX_SEL[5:0] LP_GPIO_16					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_17					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_18					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_19					
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0			MEGAMUX_SEL[5:0] LP_GPIO_20					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_21					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_22					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_23					
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0			MEGAMUX_SEL[5:0] LP_GPIO_24					
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0				MUX_0[4:0]				
			15:8				MUX_1[4:0]				
			23:16				MUX_2[4:0]				
			31:24				MUX_3[4:0]				
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0				MUX_4[4:0]				
			15:8				MUX_5[4:0]				
			23:16				MUX_6[4:0]				
			31:24				MUX_7[4:0]				
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0				MUX_8[4:0]				
			15:8				MUX_9[4:0]				
			23:16				MUX_10[4:0]				
			31:24				MUX_11[4:0]				
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0				MUX_12[4:0]				
			15:8				MUX_13[4:0]				
			23:16				MUX_14[4:0]				
			31:24				MUX_15[4:0]				
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0				MUX_16[4:0]				
			15:8				MUX_17[4:0]				
			23:16				MUX_18[4:0]				
			31:24				MUX_19[4:0]				
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0				MUX_20[4:0]				

19.9 Register Description

19.9.1 UART Transmit Data

Name: TRANSMIT_DATA
Reset: 0x00

Absolute Address: 0x40004000(UART0), 0x40005000(UART1)

This register is a part of UART registers. This register allows the user to push one byte of data into the transmit FIFO of the UART module.

Bit	7	6	5	4	3	2	1	0
	TX_BYTE[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TX_BYTE[7:0]
Writing to this eight bits Write-Only register pushes one byte into the transmit FIFO
Note: If the seven bit data is transmitted, the most significant bit must be set as zero.

19.9.2 UART Transmit Status

Name: TRANSMIT_STATUS
Reset: 0x00

Absolute Address: 0x40004004(UART0), 0x40005004(UART1)

This register is a part of UART registers. This register provides the status of UART transmit operation.

Bit	7	6	5	4	3	2	1	0
			CTS_ACTIVE	TX_FIFO_EMPTY	TX_FIFO_0P75_EMPTY	TX_FIFO_0P5_EMPTY	TX_FIFO_0P25_EMPTY	TX_FIFO_NOT_FULL
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – CTS_ACTIVE
This bit is set when CTS signal is low

Bit 4 – TX_FIFO_EMPTY
This bit is set if the Tx FIFO is completely empty

Bit 3 – TX_FIFO_0P75_EMPTY
This bit is set if the Tx FIFO is three-quarters empty

Bit 2 – TX_FIFO_0P5_EMPTY
This bit is set if the Tx FIFO is half empty

Bit 1 – TX_FIFO_0P25_EMPTY
This bit is set if the Tx FIFO is one quarter empty

Bit 0 – TX_FIFO_NOT_FULL
This bit is set if there is at least space for one more byte in the Tx FIFO

19.9.3 UART Transmit Mask Interrupt

Name: TX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x40004008(UART0), 0x40005008(UART1)

This register is a part of UART registers. This register is used to enable or disable the generation of UART transmission interrupts. During the UART transmission interrupt, if a bit in this TX_INTERRUPT_MASK register is set and its corresponding bit in the TRANSMIT_STATUS register is set then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
			CTS_ACTIVE_MASK	TX_FIFO_EMPTY_MASK	TX_FIFO_0P75_EMPTY_MASK	TX_FIFO_0P5_EMPTY_MASK	TX_FIFO_0P25_EMPTY_MASK	TX_FIFO_NOT_FULL_MASK
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – CTS_ACTIVE_MASK
Writing '0' to this bit disables the CTS_ACTIVE_MASK interrupt
Writing '1' to this bit enables the CTS_ACTIVE_MASK interrupt

Bit 4 – TX_FIFO_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_EMPTY interrupt

Bit 3 – TX_FIFO_0P75_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P75_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P75_EMPTY interrupt

Bit 2 – TX_FIFO_0P5_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P5_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P5_EMPTY interrupt

Bit 1 – TX_FIFO_0P25_EMPTY_MASK
Writing '0' to this bit disables the TX_FIFO_0P25_EMPTY interrupt
Writing '1' to this bit enables the TX_FIFO_0P25_EMPTY interrupt

Bit 0 – TX_FIFO_NOT_FULL_MASK
Writing '0' to this bit disables the TX_FIFO_NOT_FULL interrupt
Writing '1' to this bit enables the TX_FIFO_NOT_FULL interrupt

19.9.4 UART Receive Data

Name: RECEIVE_DATA
Reset: 0x00

Absolute Address: 0x40004010(UART0), 0x40005010(UART1)

This register is a part of UART registers. This register allows the user to pop one byte of data from the receive FIFO of the UART module.

Bit	7	6	5	4	3	2	1	0
	RX_BYTE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – RX_BYTE[7:0]
Reading from this eight bits read-only register pops one byte from the receive FIFO
Note: If the seven bit data is received, the most significant bit is set as zero.

19.9.5 UART Receive Status

Name: RECEIVE_STATUS
Reset: 0x00

Absolute Address: 0x40004014(UART0), 0x40005014(UART1)

This register is a part of UART registers. This register provides the status of UART receive operation.

Bit	7	6	5	4	3	2	1	0
	FRAMING_ERROR	FIFO_OVERRUN	PARITY_ERROR	TIMEOUT	RX_FIFO_0P75_FULL	RX_FIFO_0P5_FULL	RX_FIFO_0P25_FULL	RX_FIFO_NOT_EMPTY
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – FRAMING_ERROR
This bit is set when seven or eight bits are received (according to the configuration) but Stop bit is not received
This bit resets after the Status register is read

Bit 6 – FIFO_OVERRUN
This bit is set when a character is received but there is no space in the Rx FIFO to store
This bit resets after the Status register is read

Bit 5 – PARITY_ERROR
This bit is set when a character is received with a parity bit other than the expected character
This bit resets after the Status register is read

Bit 4 – TIMEOUT
This bit is set when the timeout set in the RECEIVE_TIMEOUT register is elapsed since the last interaction with the receiver
This bit resets after the Status register is read

Bit 3 – RX_FIFO_0P75_FULL
This bit is set if the Rx FIFO is three-quarters full

Bit 2 – RX_FIFO_0P5_FULL
This bit is set if the Rx FIFO is half full

Bit 1 – RX_FIFO_0P25_FULL
This bit is set if the Rx FIFO is one quarter full

Bit 0 – RX_FIFO_NOT_EMPTY
This bit is set if there is at least space for one more byte in the Rx FIFO

19.9.6 UART Receive Mask Interrupt

Name: RX_INTERRUPT_MASK
Reset: 0x00

Absolute Address: 0x40004018(UART0), 0x40005018(UART1)

This register is a part of UART registers. This register is used to enable or disable the generation of UART receive interrupts. During the UART receive interrupt, if a bit in this RX_INTERRUPT_MASK register is set and its corresponding bit in the RECEIVE_STATUS register is set, then an interrupt is generated.

Bit	7	6	5	4	3	2	1	0
	FRAMING_ERROR_MASK	FIFO_OVERRUN_MASK	PARITY_ERROR_MASK	TIMEOUT_MASK	RX_FIFO_0P75_FULL_MASK	RX_FIFO_0P5_FULL_MASK	RX_FIFO_0P25_FULL_MASK	RX_FIFO_NOT_EMPTY_MASK
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – FRAMING_ERROR_MASK
Writing '0' to this bit disables the FRAMING_ERROR interrupt
Writing '1' to this bit enables the FRAMING_ERROR interrupt

Bit 6 – FIFO_OVERRUN_MASK
Writing '0' to this bit disables the FIFO_OVERRUN interrupt
Writing '1' to this bit enables the FIFO_OVERRUN interrupt

Bit 5 – PARITY_ERROR_MASK
Writing '0' to this bit disables the PARITY_ERROR interrupt
Writing '1' to this bit enables the PARITY_ERROR interrupt

Bit 4 – TIMEOUT_MASK
Writing '0' to this bit disables the TIMEOUT interrupt
Writing '1' to this bit enables the TIMEOUT interrupt

Bit 3 – RX_FIFO_0P75_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P75_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P75_FULL interrupt

Bit 2 – RX_FIFO_0P5_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P5_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P5_FULL interrupt

Bit 1 – RX_FIFO_0P25_FULL_MASK
Writing '0' to this bit disables the RX_FIFO_0P25_FULL interrupt
Writing '1' to this bit enables the RX_FIFO_0P25_FULL interrupt

Bit 0 – RX_FIFO_NOT_EMPTY_MASK
Writing '0' to this bit disables the RX_FIFO_NOT_EMPTY interrupt
Writing '1' to this bit enables the RX_FIFO_NOT_EMPTY interrupt

19.9.7 UART Receive Time-out

Name: RECEIVE_TIMEOUT
Reset: 0xFF

Absolute Address: 0x4000401C(UART0), 0x4000501C(UART1)

This register is a part of UART registers. This register sets the receive time-out period. This functionality is useful for high throughput, and low latency systems.

Bit	7	6	5	4	3	2	1	0
	TIMEOUT_VALUE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 7:0 – TIMEOUT_VALUE[7:0]
This eight bit read/write register sets the time-out period. A time-out counter is reloaded with this value when the status register is read or a character is received.

19.9.8 UART Configuration

Name: UART_CONFIGURATION
Reset: 0x00

Absolute Address: 0x40004020(UART0), 0x40005020(UART1)

This register is a part of UART registers. This register allows the user to configure the UART Configuration modes.

Bit	7	6	5	4	3	2	1	0
			CTS_ENABLE	STOP_BITS	PARITY_MODE[1:0]		PARITY_ENABLE	NUMBER_OF_BITS
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 5 – CTS_ENABLE
Writing '0' to this bit disables flow control via CTS. The CTS signal is ignored and data is transmitted whenever the bytes are available in the transmit FIFO
Writing '1' to this bit enables flow control via CTS. The transmitter transmits only when the CTS signal is asserted

Bit 4 – STOP_BITS
This bit indicates the number of stop bits as shown in the following table.

STOP_BITS	Description
0	One stop bit
1	Two stop bits

Bits 3:2 – PARITY_MODE[1:0]
These bits select the Parity mode when parity is enabled as shown in the following table.

PARITY_MODE[1:0]	Description
0	Even parity
1	Odd parity
2	Space parity
3	Mark parity

Bit 1 – PARITY_ENABLE
Writing '0' to this bit disables parity checking and generation
Writing '1' to this bit enables parity checking and generation

Bit 0 – NUMBER_OF_BITS
This bit indicates the number of bits per character, without including parity as shown in the following table.

NUMBER_OF_BITS	Description
0	8 bits data

.....continued

NUMBER_OF_BITS	Description
1	7 bits data

19.9.9 UART Baud Rate Clock Divider

Name: UART_BAUD_RATE
Reset: 0x0000

Absolute Address: 0x40004024(UART0), 0x40005024(UART1)

This register is a part of UART registers. This register sets the divide ratio used to generate the baud clock from input clock of the UART module.

Bit	15	14	13	12	11	10	9	8
	INTEGER_DIVISION[12:5]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	INTEGER_DIVISION[4:0]					FRACTIONAL_DIVISION[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:3 – INTEGER_DIVISION[12:0]
These bits set the integral division part of the clock divider. This bit must be at least set to '2'.

Bits 2:0 – FRACTIONAL_DIVISION[2:0]
These bits set the fractional division part of the clock divider. If the value is non-zero, the integer part must be at least set to '3'.
Clock_Division = INTEGER_DIVISION.(FRACTIONAL_DIVISION/8)
Here, . indicates decimal point.
Baud Rate = CLOCK_SOURCE_SELECT/ Clock_Division

19.9.10 UART Clock Source Select

Name: UART_CLOCK_SOURCE
Reset: 0x00

Absolute Address: 0x40004028(UART0), 0x40005028(UART1)

This register is a part of UART registers. This register allows the user to select the input clock source for the UART peripheral.

Bit	7	6	5	4	3	2	1	0
							CLOCK_SELECT[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 1:0 – CLOCK_SELECT[1:0]
These bits select the input clock for the UART module.

CLOCK_SELECT[1:0]	Description
0	26 MHz clock
1	13 MHz clock
2	6.5 MHz clock
3	3.25 MHz clock

20. SPI Flash Controller

The AHB SPI Flash Controller is used to access the internal stacked SPI Flash to access various instruction/data code required for storing application code, code patches, and OTA images. It supports several SPI modes including 0, 1, 2, and 3.

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA provide an SPI Master interface for accessing the internal stacked SPI Flash memory. The TXD pin is same as the Master Output, Slave Input (MOSI), and the RXD pin is the same as the Master Input, Slave Output (MISO). The SPI Master interface supports all four standard modes of clock polarity and clock phase, as shown in SPI Modes. Internal stacked SPI Flash memory is accessed by a processor programming commands to the SPI Master interface, which in turn initiates SPI master access to the Flash.

20.1 Features

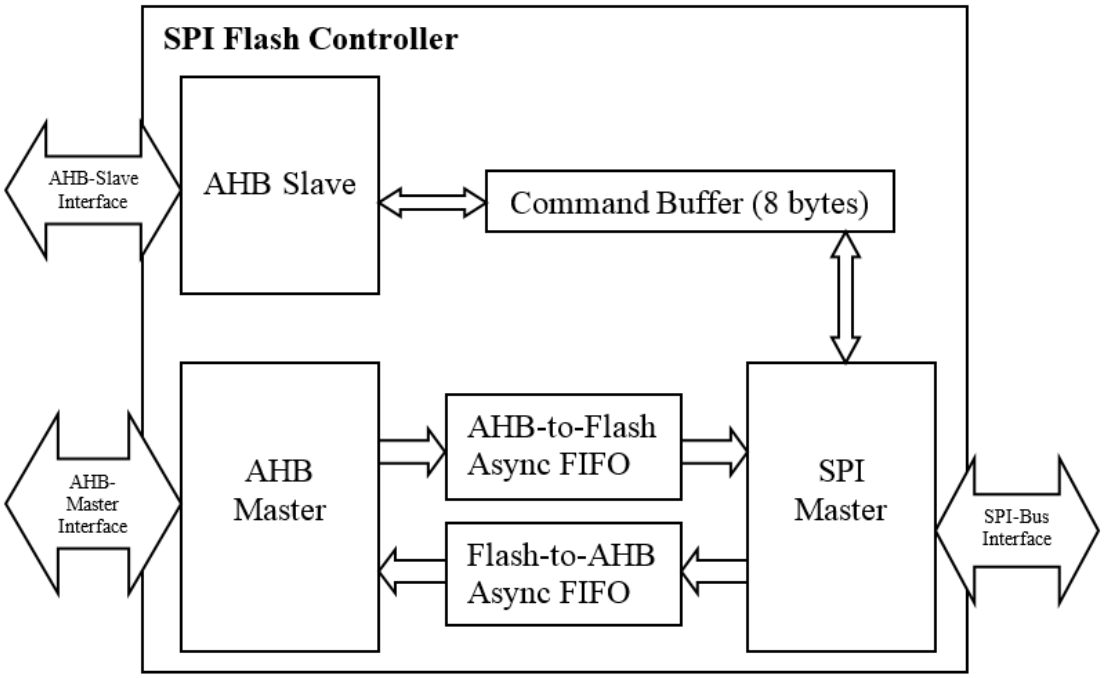
The following are the SPI Flash controller features:

- One SPI Flash controller in SPI master configuration
- Four-wire interface (TXD, RXD, SCK, and SSN)
- Supports all four SPI modes of operation
- Supports data transfer between SPI Flash memory and AHB in both directions
- Flexibility of accessing SPI Flash memory through a software programmable command buffer
- Selectable LSB- or MSB-first data transfer
- Selectable endianess for write/read data to/from Flash
- Dedicated DMA internal to SPI Flash controller
- Configurable serial clock speed from four different clock sources
- Hardware controlled SSN
- Stacked SPI Flash inside SiP:
  - Dedicated LP_SIP_x pins (TXD, RXD, SCK, SSN, WP(Write-protect)) connected internally with stacked Flash (SPI Flash Part Number: W25X20CL)
  - Software controllable VDDIO power supply for stacked Flash to reduce power consumption when stacked Flash is not in use

20.2 Block Diagram

The SPI Flash controller is composed of three main blocks, such as AHB Slave, AHB Master and SPI Master as shown in the following figure.

Figure 20-1. SPI Flash Controller Block Diagram



The AHB master interface is used to transfer the data to/from the Flash memory, from/to a software programmable location. The AHB slave interface is used to configure the SPI Flash controller. It also has asynchronous FIFOs that transfers the data between the AHB and the Flash.

20.3 Clock Configuration

Before configuring the SPI Flash controller registers, the clock for SPI Flash controller must be enabled. This is done by setting the SPI_FLASH0_CLK_EN bit in the LPMCU_CLOCK_ENABLES_0 register. For more details on configuration, see Peripheral Clock Configuration.

20.4 Functional Description

20.4.1 Initialization

- Enable the power supply to the internal stacked Flash by writing ‘1’ to ENABLE bit of the SPIFLASH_VDDIO_CTRL register.
- Configure the PINMUX_SEL_n register (see I/O Peripheral Multiplexing and MEGAMUXing) to select the IO pins that need to be used as SPI FLASH0 TXD (MOSI), SPI FLASH0 RXD (MISO), SPI FLASH0 SCK, and SPI FLASH0 SSN lines. By configuring pinmux to select SPI Flash controller functionality, LP_GPIO_x bypasses the GPIO controller function on selected LP_GPIO_x pin. This pinmux configuration is applicable only if SPI Flash controller is used to communicate with external SPI Flash
- Internal stacked Flash pinmux configuration: Dedicated LP_SIP_x pins TXD, RXD, SCK, SSN, WP (Write-protect) are connected internally with stacked Flash (Part Number: W25X20CL). The following table shows the required pinmux configuration.

LP_SIP_x pin	SPI Flash Pin Name	PINMUX[2:0] (PINMUX_SEL_3) Value
LP_SIP_0	SPI_FLASH_SCK	1
LP_SIP_1	SPI_FLASH_TXD	3
LP_SIP_2	SPI_FLASH_SSN	2
LP_SIP_3	SPI_FLASH_RXD	4
LP_SIP_4	SPI_FLASH_WP	0

- Select the SPI Flash controller clock source by configuring the SPI_FLASH0_CLKSEL[1:0] bits of the LPMCU_CTRL register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz, and 3.25 MHz.
- Set the SPI Transfer mode by configuring MODE[1:0] bits in the MODE_CTRL register. For more details on supported modes by specific SPI Flash device, see SPI Flash datasheet.
- The SPI Flash read/write data bit ordering (MSB or LSB first) is programmable through WDATA_REVERSE, RDATA_REVERSE bits of the CONFIG register.
- The MOSI is programmable to drive ‘0’, ‘1’, toggle between '1' and '0' or programmable 8-bit data while receiving the data from Flash through MISO. This is selected by programming the TX_CONTROL register.
- The SPI Flash read/write data byte ordering (little-endian or big-endian) is programmable through ENDIANNESS bit of the CONFIG register. For more details, see Endianness.
- Wake up the SPI Flash from Low-Power mode to Normal mode by sending specific commands. For more details on commands, see the specific SPI Flash datasheet.

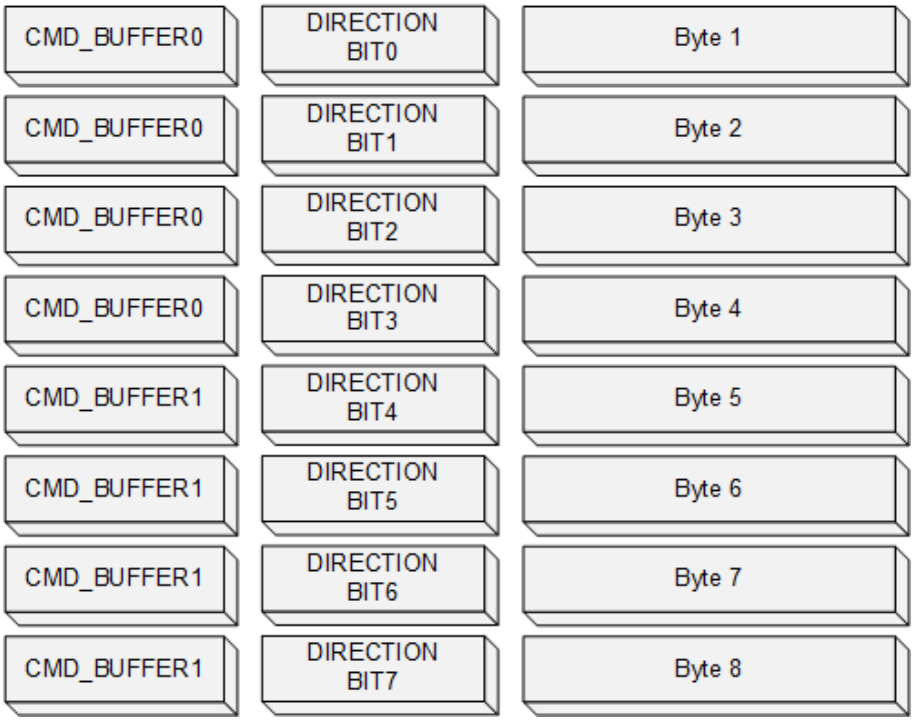
20.4.2 SPI Flash Controller Configurations

To configure the SPI Flash controller for accessing the Flash memory, write the SPI commands and configuration parameters to the command buffer (CMD_BUFFER0, CMD_BUFFER1, and CMD_DIRECTION), which initiates the SPI Flash write/read operation.

The status registers data read from the SPI Flash is also available in the command buffer, which can be read by the application.

The following figure shows the 8-byte command buffer and its associated direction bit.

Figure 20-2. Command Buffer Direction



20.4.2.1 Writing Commands and Parameters to SPI Flash

Any number of bytes between one to eight are written in the CMD_BUFFER0 (holds 1 to 4 bytes) register, CMD_BUFFER1 (holds five to eight bytes) register, and its corresponding R/W bit in the DIRECTION register is set to 1.

For example, to transfer four bytes of commands/parameters to SPI Flash perform the following:

- Byte 1, 2, 3, 4 is written to the CMD_BUFFER0 register and its R/W bit of each byte is set to 1 in the DIRECTION register. Bit 0 of the DIRECTION register corresponds to R/W bit for byte 1 of the CMD_BUFFER0, bit 1 corresponds to byte 1 of the CMD_BUFFER0 register, and so on and bit 7 corresponds to byte 4 of the CMD_BUFFER1 register as shown in Command Buffer Direction.
- Write CMD_COUNT[2:0] of the TRANSACTION_CTRL register with value as 4 (for transferring 4 bytes of command).
- Set FLASH_TRANS_START bit of the TRANSACTION_CTRL register to ‘1’. This operation initiates a SPI access to transfer the programmed four bytes.
- The end of transaction is indicated by setting FLASH_TRANS_DONE bit of the IRQ_STATUS register to ‘1’.

20.4.2.2 Reading Status from SPI Flash

The R/W bit of the DIRECTION register must be set to 0 for the bytes that must be read.

For example, to read 32-bit value (4 bytes) from the SPI Flash register, perform the following:

- Write the SPI Flash register address (address of SPI Flash register) to byte 1 of the CMD_BUFFER0 register.
- Set byte 1 R/W bit to 1 in the DIRECTION register to ‘1’.
- Set R/W bit of byte 2, 3, 4, 5 (4 bytes to read) to ‘0’.
- Write CMD_COUNT[2:0] of the TRANSACTION_CTRL register with value as 5 (1 byte command +32-bit read register).

- Set FLASH_TRANS_START bit of the TRANSACTION_CTRL register to ‘1’.
- The end of transaction is indicated by setting FLASH_TRANS_DONE bit of the IRQ_STATUS register to ‘1’. The application can read the register value from byte 2 through 5 from the CMD_BUFFER0 and CMD_BUFFER1 registers.

20.4.2.3 Reading Data from SPI Flash

To read the data from Flash, perform the following two steps.

1. Write the commands in the command buffer (see Writing Commands and Parameters to SPI Flash) and program the data count register with the actual number of bytes that must be read.
2. Set the Start bit. Thus, the AHB master (DMA) writes the data which is read from the SPI Flash to the destination buffer address.

For example, to read eight bytes of data from the SPI Flash with three bytes address, perform the following:

- Write read command value into byte 1 of the CMD_BUFFER0 register and set its R/W bit (Bit 0) to 1 in the DIRECTION register.
- Write three bytes address (SPI Flash data address from where data to be read) into bytes 2, 3, and 4 of the CMD_BUFFER0 register and set its R/W bits (bits 1, 2, and 3) to '1' in the DIRECTION register.
- Write dummy byte (SPI read requires writing dummy byte) into byte 1 of the CMD_BUFFER1 register and set its R/W bit (bit 4) to '1' in the DIRECTION register.
- Write CMD_COUNT[2:0] of the TRANSACTION_CTRL register with the value as 5 (1 byte command +3 byte address+1 byte dummy data).
- Write RDATA_COUNT[23:0] of the READ_CTRL register with the value as 8 (to read 8 bytes of data).
- Write the DMA_START_ADDRESS register with the destination buffer address where the read data must be written.
- Set FLASH_TRANS_START bit of the TRANSACTION_CTRL register to ‘1’. This operation initiates SPI access to transfer the programmed five bytes and reads eight bytes of data from the SPI Flash. The DMA transfers the read data into destination buffer.
- The end of transaction is indicated by setting FLASH_TRANS_DONE bit of the IRQ_STATUS register to ‘1’.

20.4.2.4 Writing Data to SPI Flash

Writing data to SPI Flash is similar to writing commands (Writing Commands and Parameters to SPI Flash), but WDATA_COUNT[23:0] bit in the TRANSACTION_CTRL register is also programmed by writing number of bytes to the SPI Flash and setting the start bit. Then, the AHB master starts reading the data from the source address and the SPI master writes this data to the SPI Flash.

For example, to program a SPI Flash page with 256 bytes of data:

- Write the write command value into byte 1 of the CMD_BUFFER0 register and set its R/W bit (bit 0) as 1 in the DIRECTION register.
- Write three byte address into bytes 2, 3, and 4, of the CMD_BUFFER0 register and set its R/W bits (bits 1, 2, and 3) to 1 in the DIRECTION register.
- Write CMD_COUNT[2:0] of the TRANSACTION_CTRL register with the value as 4 (1 byte command +3 byte address).
- Write WDATA_COUNT[23:0] of the TRANSACTION_CTRL register with the value as 256 (to write 256 bytes of data).

- Write the DMA_START_ADDRESS register with the source buffer address, where the write data is stored.
- Set FLASH_TRANS_START bit of the TRANSACTION_CTRL register to '1'. This operation initiates SPI access to transfer the programmed four bytes and initiates to write data into SPI flash. The DMA transfers the data from source buffer to SPI flash.
- The end of transaction is indicated by setting FLASH_TRANS_DONE bit of the IRQ_STATUS register to '1'.

20.4.3 Internal Stacked Flash Power Down

After the transaction with SPI Flash is complete, allow the SPI Flash to go to low power by sending low-power command. For more details on commands, see SPI Flash datasheet.

Switch-off the power supply to the internal stacked Flash by clearing ENABLE bit of the SPIFLASH_VDDIO_CTRL register.

20.5 Endianness

When the little-endian configuration is selected (ENDIANNESS bit of the CONFIG register is '1'), the following logic applies to read/write from the SPI Flash.

Table 20-1. Little-endian Configuration Logic

Transfer Size	Address Offset	DATA [31:24]	DATA [23:16]	DATA [15:8]	DATA [7:0]
Word	0	3	2	1	0
Half word	0	-	-	1	0
Half word	2	3	2	-	-
Byte	0	-	-	-	0
Byte	1	-	-	1	-
Byte	2	-	2	-	-
Byte	3	3	-	-	-

When the big-endian configuration is selected (ENDIANNESS bit of the CONFIG register is '0'), the following logic applies to read/write from the SPI Flash.

Table 20-2. Big-endian Configuration Logic

Transfer Size	Address Offset	DATA [31:24]	DATA [23:16]	DATA [15:8]	DATA [7:0]
Word	0	0	1	2	3
Half word	0	0	1	-	-
Half word	2	-	-	2	3
Byte	0	0	-	-	-
Byte	1	-	1	-	-

.....continued

Transfer Size	Address Offset	DATA [31:24]	DATA [23:16]	DATA [15:8]	DATA [7:0]
Byte	2	-	-	2	-
Byte	3	-	-	-	3

Where, B* indicates the order of the bytes read/written from/to the SPI Flash memory.

20.6 Power Management

If the system goes to the Ultra-Low Power mode, the SPI Flash controller shuts down. The SPI Flash Controller Configuration registers lose its content, and cannot be restored when powered-up again. Therefore, user must reconfigure the SPI Flash controller at power-up to ensure it is in a well-defined state before use.

For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains on how sleep and wake-up are controlled.

20.7 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40012000	SPI_FLASH0	MODE_CTRL	7:0	MODE_SELECT				BYPASS_CS_POST	BYPASS_CS_PRE	MODE[1:0]	
0x40012004	SPI_FLASH0	TRANSACTION_CTRL	7:0	FLASH_TRANS_START					CMD_COUNT[2:0]		
			15:8	WDATA_COUNT[7:0]							
			23:16	WDATA_COUNT[15:8]							
			31:24	WDATA_COUNT[23:16]							
0x40012008	SPI_FLASH0	READ_CTRL	7:0	RDATA_COUNT[7:0]							
			15:8	RDATA_COUNT[15:8]							
			23:16	RDATA_COUNT[23:16]							
0x4001200C	SPI_FLASH0	CMD_BUFFER0	7:0	CMD_BUFFER0[7:0]							
			15:8	CMD_BUFFER0[15:8]							
			23:16	CMD_BUFFER0[23:16]							
			31:24	CMD_BUFFER0[31:24]							
0x40012010	SPI_FLASH0	CMD_BUFFER1	7:0	CMD_BUFFER1[7:0]							
			15:8	CMD_BUFFER1[15:8]							
			23:16	CMD_BUFFER1[23:16]							
			31:24	CMD_BUFFER1[31:24]							
0x40012014	SPI_FLASH0	DIRECTION	7:0	DIRECTION[7:0]							
0x40012018	SPI_FLASH0	IRQ_STATUS	7:0							AHB_ERROR_RESPONSE	FLASH_TRANS_DONE
0x4001201C	SPI_FLASH0	DMA_START_ADDRESS	7:0	DMA_START_ADDRESS[7:0]							
			15:8	DMA_START_ADDRESS[15:8]							
			23:16	DMA_START_ADDRESS[23:16]							
			31:24	DMA_START_ADDRESS[31:24]							

.....continued

Absolute Address	Register Group	Name	Bit Pos.									
0x40012020	SPI_FLASH0	CONFIG	7:0	SPI_INTERFACE_CONFIG_DEBUG[3:0]				REVERSE_INCOMING_DATA	RDATA_REVERSE	REVERSE_CMD_BUFFER	ENDIANNESS
			15:8								WDATA_REVERSE
0x40012024	SPI_FLASH0	TX_CONTROL	7:0					DUMMY_ON_TX	TOGGLE_ON_TX	DRIVE_1_ON_TX	DRIVE_0_ON_TX
			15:8	DUMMY_BYTE[7:0]							
0x40012028	SPI_FLASH0	STATUS	7:0		FIFO_CTRL_STATE[2:0]					AHB_MASTER_STATE[1:0]	
			15:8				SPI_MASTER_CSN		SPI_MASTER_STATE[2:0]		
0x4000B0A0	LPMCU_MISC_REGS0	SPIFLASH_VDDIO_CTRL	7:0								ENABLE
0x4000B0A4	LPMCU_MISC_REGS0	SPIFLASH_BYPASS	7:0								ENABLE
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B068	LPMCU_MISC_REGS0	PINMUX_SEL_3	7:0		PINMUX_SEL[2:0] LP_SIP_1				PINMUX_SEL[2:0] LP_SIP_0		
			15:8		PINMUX_SEL[2:0] LP_SIP_3				PINMUX_SEL[2:0] LP_SIP_2		
			23:16						PINMUX_SEL[2:0] LP_SIP_4		
			31:24								
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								

20.8 Register Description

20.8.1 SPI Flash Controller Mode Selection

Name: MODE_CTRL
Reset: 0x00

Absolute Address: 0x40012000

This register is a part of SPI_FLASH0 registers. This register allows the user to select the SPI modes (0, 1, 2, and 3).

Bit	7	6	5	4	3	2	1	0
	MODE_SELECT				BYPASS_CS_POST	BYPASS_CS_PRE	MODE[1:0]	
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				0	0	0	0

Bit 7 – MODE_SELECT
Writing '1' to this bit selects the SPI mode selection from external pins
Note: This mode is not supported and therefore it is not recommended the user write '1' to this bit.

Bit 3 – BYPASS_CS_POST
Writing '1' to this bit bypasses the one clock cycle delay between the SSN signal which is going high and the clock which is held low after a transaction finishes
Writing '0' to this bit allows the one clock cycle delay between the SSN signal which is going high and the clock which is held low after a transaction finishes

Bit 2 – BYPASS_CS_PRE
Writing '1' to this bit bypasses the one clock cycle delay between the SSN signal which is going low and the clock toggling before a transaction starts
Writing '0' to this bit allows the one clock cycle delay between the SSN signal which is going low and the clock toggling before a transaction starts

Bits 1:0 – MODE[1:0]
These bits select the SPI modes

MODE[1:0]	Description
0	CPOL = 0, CPHA=0
1	CPOL = 0, CPHA=1
2	CPOL = 1, CPHA=0
3	CPOL = 1, CPHA=1

20.8.2 SPI Flash Controller Transaction Control

Name: TRANSACTION_CTRL
Reset: 0x00000004

Absolute Address: 0x40012004

This register is a part of SPI_FLASH0 registers. This register allows the user to initiate the SPI Flash controller transactions.

Bit	31	30	29	28	27	26	25	24
	WDATA_COUNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WDATA_COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WDATA_COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FLASH_TRANS_START					CMD_COUNT[2:0]		
Access	R/W					R/W	R/W	R/W
Reset	0					1	0	0

Bits 31:8 – WDATA_COUNT[23:0]
These bits hold the number of bytes to be programmed to SPI Flash
These bits reset to ‘0’ when the transaction is complete

Bit 7 – FLASH_TRANS_START
Writing '1' to this bit starts the SPI Flash controller transactions
These bits reset to ‘0’ when the transaction is complete

Bits 2:0 – CMD_COUNT[2:0]
These bits hold the number of command bytes to be transferred/received
These bits reset to ‘0’ when the transaction is complete

20.8.3 SPI Flash Controller Read Control

Name: READ_CTRL
Reset: 0x00000000

Absolute Address: 0x40012008

This register is a part of SPI_FLASH0 registers. This register allows the user to configure the number of data to be read from SPI Flash.

Bit	23	22	21	20	19	18	17	16
	RDATA_COUNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RDATA_COUNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RDATA_COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – RDATA_COUNT[23:0]
These bits hold the number of bytes to be read from SPI Flash
These bits reset to ‘0’ when the transaction is complete

20.8.4 SPI Flash Controller Command Buffer 0

Name: CMD_BUFFER0
Reset: 0x00000000

Absolute Address: 0x4001200C

This register is a part of SPI_FLASH0 registers. This register holds the byte 1, 2, 3, and 4 of the commands that must be sent to SPI Flash.

Bit	31	30	29	28	27	26	25	24
	CMD_BUFFER0[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CMD_BUFFER0[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CMD_BUFFER0[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CMD_BUFFER0[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CMD_BUFFER0[31:0]
These bits hold the command data

CMD_BUFFER0[31:0]	Description
CMD_BUFFER0[7:0]	Command byte 1
CMD_BUFFER0[15:8]	Command byte 2
CMD_BUFFER0[23:16]	Command byte 3
CMD_BUFFER0[31:24]	Command byte 4

20.8.5 SPI Flash Controller Command Buffer 1

Name: CMD_BUFFER1
Reset: 0x00000000

Absolute Address: 0x40012010

This register is a part of SPI_FLASH0 registers. This register holds byte 5, 6, 7 and 8 of the commands that must be sent to SPI Flash.

Bit	31	30	29	28	27	26	25	24
	CMD_BUFFER1[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CMD_BUFFER1[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CMD_BUFFER1[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CMD_BUFFER1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CMD_BUFFER1[31:0]
These bits hold the command data

CMD_BUFFER1[31:0]	Description
CMD_BUFFER0[7:0]	Command byte 5
CMD_BUFFER0[15:8]	Command byte 6
CMD_BUFFER0[23:16]	Command byte 7
CMD_BUFFER0[31:24]	Command byte 8

20.8.6 Command Data Direction Control

Name: DIRECTION
Reset: 0x0F

Absolute Address: 0x40012014

This register is a part of SPI_FLASH0 registers. This register controls the R/W direction for command data stored in the CMD_BUFFER registers.

Bit	7	6	5	4	3	2	1	0
	DIRECTION[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bits 7:0 – DIRECTION[7:0]
These bits hold the R/W direction bit for command bytes 8 to 1 (Bit 0 for Byte 1 .. Bit 7 for Byte 8)

20.8.7 SPI Flash Transaction Status

Name: IRQ_STATUS
Reset: 0x00

Absolute Address: 0x40012018

This register is a part of SPI_FLASH0 registers. This register provides the status of current transaction.

Bit	7	6	5	4	3	2	1	0
							AHB_ERROR_RESPONSE	FLASH_TRANS_DONE
Access							R/W	R/W
Reset							0	0

Bit 1 – AHB_ERROR_RESPONSE
Reading '1' on this bit indicates that the AHB master receives an error response from any slave
This bit resets to 0 after being read or by writing 0 to this bit

Bit 0 – FLASH_TRANS_DONE
Reading '1' on this bit indicates current Flash transaction is complete
This bit resets to 0 after being read or by writing 0 to this bit

20.8.8 DMA Start Address

Name: DMA_START_ADDRESS
Reset: 0x00000000

Absolute Address: 0x4001201C

This register is a part of SPI_FLASH0 registers. The internal DMA in SPI Flash controller transfers the data between the application buffer and SPI Flash. This register holds the start address of the application buffer.

Bit	31	30	29	28	27	26	25	24
	DMA_START_ADDRESS[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DMA_START_ADDRESS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DMA_START_ADDRESS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DMA_START_ADDRESS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – DMA_START_ADDRESS[31:0]
These bits hold the start address of write data buffer/read data buffer

20.8.9 SPI Flash Controller Configuration

Name: CONFIG
Reset: 0x0103

Absolute Address: 0x40012020

This register is a part of SPI_FLASH0 registers. This register allows the user to configure the SPI Flash controller for different configuration options.

Bit	15	14	13	12	11	10	9	8
								WDATA_REVERSE
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SPI_INTERFACE_CONFIG_DEBUG[3:0]				REVERSE_INCOMING_DATA	RDATA_REVERSE	REVERSE_CMD_BUFFER	ENDIANNESS
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	1	1

Bit 8 – WDATA_REVERSE
Writing '0' to this bit enables MSB bit first for every data byte to be written to SPI Flash
Writing '1' to this bit enables LSB bit first for every data byte to be written to SPI Flash

Bits 7:4 – SPI_INTERFACE_CONFIG_DEBUG[3:0]
These are 'INTERNAL' bits and not recommended to change

Bit 3 – REVERSE_INCOMING_DATA
Writing '1' to this bit reverses the bits of status registers that are read from SPI Flash before stored in the command buffers (CMD_BUFFER0, andCMD_BUFFER1)

Bit 2 – RDATA_REVERSE
Writing '1' to this bit reverses the bits of every data byte that are read from SPI Flash before stored in the application buffer

Bit 1 – REVERSE_CMD_BUFFER
Writing '0' to this bit enables MSB bit first for every command byte in CMD_BUFFER while sending the data to SPI Flash
Writing '1' to this bit enables LSB bit first for every command byte in CMD_BUFFER while sending the data to SPI Flash

Bit 0 – ENDIANNESS
Writing '0' to this bit enables big-endianess for write/read data to/from flash
Writing '1' to this bit enables little-endianess for write/read data to/from flash

20.8.10 Transmit Control Configuration

Name: TX_CONTROL
Reset: 0x0000

Absolute Address: 0x40012024

This register is a part of SPI_FLASH0 registers. Using this register the MOSI is programmable to drive ‘0’, ‘1’, toggle between '0' and '1' or programmable data while receiving data from SPI flash through MISO.

Bit	15	14	13	12	11	10	9	8
	DUMMY_BYTE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					DUMMY_ON_TX	TOGGLE_ON_TX	DRIVE_1_ON_TX	DRIVE_0_ON_TX
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 15:8 – DUMMY_BYTE[7:0]
These bits hold the dummy byte which must be sent on MOSI line while receiving the data on MISO when DUMMY_ON_TX option is enabled

Bit 3 – DUMMY_ON_TX
Writing '1' to this bit enables the configurable dummy data on MOSI line while receiving data on MISO line

Bit 2 – TOGGLE_ON_TX
Writing '1' to this bit enables alternate '1' and '0' bits on MOSI line while receiving data on MISO line

Bit 1 – DRIVE_1_ON_TX
Writing '1' to this drives logic '1' on MOSI line while receiving data on MISO line

Bit 0 – DRIVE_0_ON_TX
Writing '1' to this drives logic '0' on MOSI line while receiving data on MISO line

20.8.11 SPI Flash Controller Internal State Machine Current State

Name: STATUS
Reset: 0x0000

Absolute Address: 0x40012028

This register is a part of SPI_FLASH0 registers. This register provides the current FSM state of the SPI Flash Controller.

Bit	15	14	13	12	11	10	9	8
				SPI_MASTER_CSN		SPI_MASTER_STATE[2:0]		
Access				R		R	R	R
Reset				0		0	0	0

Bit	7	6	5	4	3	2	1	0
		FIFO_CTRL_STATE[2:0]					AHB_MASTER_STATE[1:0]	
Access		R	R	R			R	R
Reset		0	0	0			0	0

Bit 12 – SPI_MASTER_CSN
This bit indicates the synchronization of SPI Flash controller system clock with SSN signal

Bits 10:8 – SPI_MASTER_STATE[2:0]
These bits provide the SPI master FSM current state

Bits 6:4 – FIFO_CTRL_STATE[2:0]
These bits provide the FIFO control FSM current state

Bits 1:0 – AHB_MASTER_STATE[1:0]
These bits provide the AHB master FSM current state

20.8.12 Internal Stacked Flash Power Control

Name: SPIFLASH_VDDIO_CTRL
Reset: 0x01

Absolute Address: 0x4000B0A0

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to switch off and switch on the power supply to the internal stacked SPI Flash.

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R/W
Reset								1

Bit 0 – ENABLE
Writing '0' to this bit disables VDDIO power supply to internal stacked SPI Flash
Writing '1' to this bit enables VDDIO power supply to internal stacked SPI Flash

20.8.13 SPI Flash Controller Bypass

Name: SPIFLASH_BYPASS
Reset: 0x00

Absolute Address: 0x4000B0A4

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to bypass the SPI Flash controller if not required.

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R/W
Reset								0

Bit 0 – ENABLE
Writing '1' to this bit bypasses SPI Flash controller operations

21. Three-axis Quadrature Decoder

The ATSAMB11-XR2100A and ATSAMB11-ZR210CA have a three-axis quadrature decoder (X, Y, and Z) that can be used to decode the quadrature encoder signals to determine the direction, position, and speed of movement on three axes. The quadrature encoder is expected to provide pulse trains as input to the quadrature decoder.

The quadrature encoders are used to determine the position and speed of rotary devices such as, servo-motors, volume control wheels, and PC mice. The decoded quadrature signals are used as a sensory input for the system to determine the absolute or relative position of the rotary device. This relative position can be used in a control loop for applications such as servo-motor.

21.1 Features

The following are the Quad Decoder features:

- Three Quad Decoders individually control the three axis (X, Y, and Z) inputs, two quadrature signals (two signals 90 degrees phase shifted) per axis
- Enables the direction and position indication
- Enables the speed measurement when configured with a timer
- Interrupt generation when the set threshold value is reached
- Four different input clock sources (26 MHz, 13 MHz, 6.5 MHz, and 3.25 MHz) to enable range of input pulse frequencies

21.2 Principle of Operation

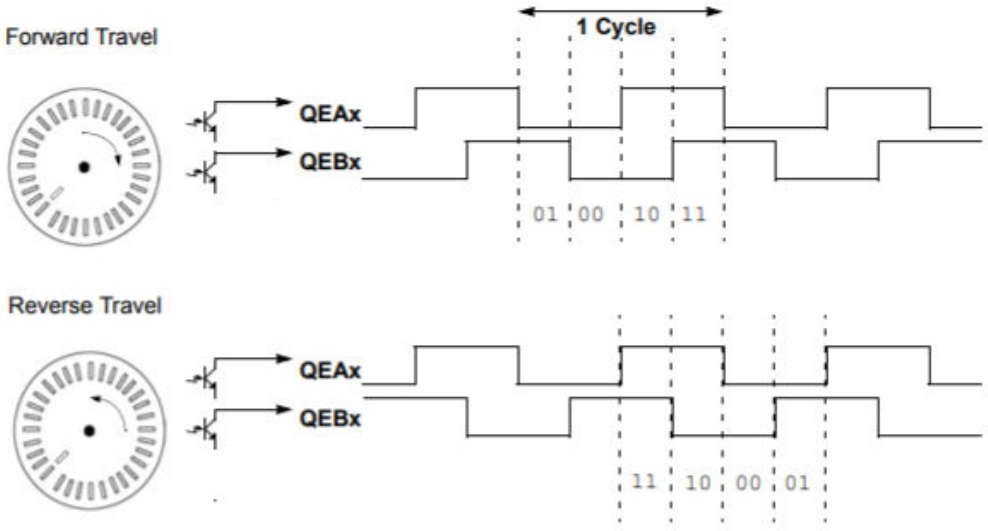
The quadrature decoder module provides an interface to the incremental encoders for obtaining mechanical position data. The quadrature decoders detects position and speed of rotating motion systems by decoding the signals from the quadrature encoders.

A typical quadrature encoder includes, a slotted wheel attached to the shaft of the motor and an emitter/detector module that senses the slots in the wheel. Typically, two output channels, Phase A (QEAx), and Phase B (QEBx) provide information on the movement of the motor shaft, including the distance and direction.

The Phase A and Phase B channels have a unique relationship. If Phase A leads Phase B, the direction of the motor is deemed positive, or forward. If Phase A lags Phase B, the direction of the motor is deemed negative or reverse. For the relative timing diagram of these two signals, see the following figure.

The quadrature signals produced by the encoder has four unique states (01, 00, 10, and 11) that reflects the relationship between QEAx and QEBx. The following figure shows these states for one count cycle. The order of the states reverses when the direction of travel changes.

Figure 21-1. Quadrature Decoder Interface Signals



The quadrature decoder module contains 16-bit counter. When these two signals (QEAx and QEBx) are provided as input to one of the quadrature decoder module, it increments or decrements the 16-bit up/down counter for each change of state. The counter increments when QEAx leads QEBx and decrements when QEBx leads QEAx. Therefore, for one cycle of input signal the counter increments or decrements four counts.

21.3 Clock Configuration

Before configuring the Quad Decoder registers, enable the clock of the Quad Decoder peripheral. The clock for three Quad Decoders (QUAD_DEC0, QUAD_DEC1, and QUAD_DEC2) are controlled individually. This is done by setting QUAD_DECn_CLK_EN (n=0,1,2) bits in the LPMCU_CLOCK_ENABLES_0 register. For more details on configuration, see Peripheral Clock Configuration.

21.4 Functional Description

The Quad Decoder is the three axis decoder with which the QUAD_DEC0 controls X-axis, the QUAD_DEC1 controls Y-axis, and the QUAD_DEC2 controls Z-axis. The three Quad Decoders are individually controllable, therefore it is not required to provide all three axis (three pair of signals) inputs. A pair of 90-degree phase shifted quad signals of one axis must be provided to one QUAD_DECn.

21.4.1 Initialization

The initialization procedures are identical for all three Quad Decoder modules.

- Configure the PINMUX_SEL_n and MEGA_MUX_IO_SEL_n registers (see I/O Peripheral Multiplexing and MEGAMUXing) to select the LP_GPIO_x pins that must be used as quad decoder input lines. The quad decoder peripheral on LP_GPIO_x pin is configured through MEGAMUX configuration only. The QUAD DEC X IN A and QUAD DEC X IN B megamux values are the pair of inputs to be selected for QUAD_DEC0. Similarly, QUAD DEC Y IN A and QUAD DEC Y IN B is used for QUAD_DEC1, and QUAD DEC Z IN A and QUAD DEC Z IN B is used for QUAD_DEC2.

Note: If only one axis measurement is required, then configure the pinmux for corresponding signals and one decoder.

- Select the QUAD_DECn module clock source by configuring the CLOCK_SEL[1:0] bits of the QUAD_DECn_CTRL (n=0,1,2) register. The clock sources that can be selected are 26 MHz, 13 MHz, 6.5 MHz, and 3.25 MHz.
- Set the upper and lower threshold values in UPPER[15:0] and LOWER[15:0] bits of the QUAD_DECn_THRESHOLD (n=0,1,2) register. The upper threshold value must be higher than or equal to lower threshold value.
- If interrupt generation is required, when upper or lower threshold is reached, then configure NVIC to enable the interrupt. The quad decoder peripheral is not mapped to the default interrupt sources in the vector table. Therefore, it is required to use muxable interrupt option to map quad decoder as interrupt source. To generate an interrupt, the appropriate bits in the interrupt mask registers must be set, as following.
  - Select the IRQ number from ATSAMB11 Interrupt Vector Table which is not used in application.
  - Configure the quad decoder peripheral to this IRQ number (see Muxable Interrupt).
  - Register the quad decoder ISR function in the interrupt vector table.
  - Enable the quad decoder interrupt in the NVIC interrupt controller registers. For more details, see Interrupt section.

Note: Only one ISR index is allocated in interrupt vector table for all the three QUAD_DECn modules.

21.4.2 Operation

- Enable the quad decoder by writing ‘1’ to ENABLE bit of the QUAD_DECn_CTRL (n=0,1,2) register.
- The 16-bit counter starts incrementing or decrementing based on the input signals direction as follows.
  - If the direction is forward (QEAx leads QEBx), COUNT[15:0] of the QUAD_DECn_STATUS (n=0,1,2) register increments from zero and when it reaches the LOWER[15:0] threshold, QUAD_DECn_IRQ bit of the QUAD_DEC_IRQS register is set.
  - If the direction is reverse (QEBx leads QEAx), COUNT[15:0] of the QUAD_DECn_STATUS (n=0,1,2) register decrements from zero and when it reaches the UPPER[15:0] threshold, QUAD_DECn_IRQ bit of the QUAD_DEC_IRQS register is set.
  - The counter does not stop to increment/decrement until the quad decoder is disabled by writing ‘0’ to ENABLE bit of the QUAD_DECn_CTRL (n=0,1,2) register. Disabling the quad decoder stops the COUNT[15:0] increment/decrement. COUNT[15:0] can be reset to zero by resetting QUAD_DECn. Refer to 9.6.3 Peripheral Reset for details.
- If interrupt is enabled, then ISR handler is called. The QUAD_DECn_IRQ bit can be cleared by writing ‘1’ to CLR_IRQ bit of the QUAD_DECn_CTRL (n=0,1,2) register.
  - The interrupt is triggered again and again even after clearing the Status bit until the COUNT[15:0] is within UPPER[15:0] and LOWER[15:0] values. The interrupt stops when the COUNT[15:0] is not within the upper and lower threshold limits.
  - To trigger the interrupt again for the next cycle when the COUNT[15:0] reaches lower or upper threshold, CLR_IRQ bit must be written with '0'. The interrupt again gets triggered when the COUNT[15:0] reaches upper/lower threshold.

21.4.3 Example

The following is the sample example to use LP_GPIO_16 and LP_GPIO_17 pins as A, and B quad inputs for Y axis quad decoder (quad1) with upper threshold = 0x4000 and lower threshold = 0x3500:

```
/* As there is no default ISR mapping for quad decoder interrupt in Table 6-2. ATSAMB11
Interrupt Vector Table, use muxable interrupt option as explained in 11.1 Example and
register ISR handler */
/* Write PINMUX_SEL_2 bits [2:0], [6:4] with 0x1 (MUX1) to enable MEGAMUX option on
LP_GPIO_16, LP_GPIO_17*/
LPMCU_MISC_REGS0->PINMUX_SEL_2.reg |= (0x1<<0) | (0x1<<4);
/* MEGAMUX option value for QUAD DEC Y IN A from Table 10-2. MEGAMUX Options is 0x1F */
/* Assign 0x1F MEGAMUX value for LP_GPIO_16 specific bits [4:0] */
LPMCU_MISC_REGS0->MEGA_MUX_IO_SEL_4.reg |= (0x1F<<0);
/* MEGAMUX option value for QUAD DEC Y IN B from Table 10-2. MEGAMUX Options is 0x20 */
/* Assign 0x20 MEGAMUX value for LP_GPIO_17 specific bits [12:8] */
LPMCU_MISC_REGS0->MEGA_MUX_IO_SEL_4.reg |= (0x20<<8);
/* Select clock source for quad decoder 1 as 26MHz */
LPMCU_MISC_REGS0->QUAD_DEC1_CTRL.bit.CLOCK_SEL = 0;

/* Set upper and lower threshold */
LPMCU_MISC_REGS0->QUAD_DEC1_THRESHOLD.bit.UPPER =0x4000;
LPMCU_MISC_REGS0->QUAD_DEC1_THRESHOLD.bit.LOWER = 0x3500;
/* Enable quad decoder1 (Y axis)*/
LPMCU_MISC_REGS0->QUAD_DEC1_CTRL.bit.ENABLE = 1;
/* Enable NVIC interrupt as explained in Section 6.3  Nested Vector Interrupt Controller*/
```

21.4.4 Count Direction

The direction of quadrature counting is determined by reading two consecutive COUNT[15:0] values. If the COUNT[15:0] value is greater than the previous value, then direction is forward. If the COUNT[15:0] value is lesser than the previous value, then the direction is reverse.

21.4.5 Using Counter Value for Position Measurement

The application can use the current COUNT[15:0] data in several ways. In some systems, the position count is accumulated consistently and taken as an absolute value representing the total position of the system.

For example, assume that a quadrature encoder is affixed to a motor controlling the print head in a printer. In operation, the system is initialized by moving the print head to the maximum left position and the COUNT[15:0] register becomes zero. As the print head moves to the right, the quadrature encoder begins to accumulate counts in the COUNT[15:0] register. As the print head moves to the left, the accumulated count decreases and reaches zero.

The UPPER[15:0] or LOWER[15:0] threshold setting can be used to reset the position counter. When the COUNT[15:0] reaches a predetermined threshold count set in UPPER[15:0] or LOWER[15:0] while incrementing or decrementing. An interrupt is generated when the COUNT[15:0] reaches a predetermined threshold count set in UPPER[15:0] or LOWER[15:0] and during interrupt the user application can reset the COUNT [15:0] by resetting QUAD_DECn. Refer to 9.6.3 Peripheral Reset for details.

In some systems, the position count can be cyclic. The position count references the position of the wheel within number of rotations determined by the index pulse. For example, a tool platform moved by a screw rod uses a quadrature encoder attached to the screw rod. In operation, the screw may require five and a half rotations to achieve the desired position. The user software detects five index pulses to count the full rotations and uses the position count to measure the remaining half rotation. In this method, the index pulse resets the position counter (COUNT[15:0]) to initialize the counter at each rotation and generates an interrupt for each rotation. The index pulse input can be provided to any of the external interrupt GPIO pins. The user application must reset the COUNT[15:0] by disabling Quad decoder in GPIO ISR and enable the quad decoder again to restart the counter again.

21.5 Power Management

If the system goes to the Ultra-Low Power mode, the Quad Decoder peripheral shuts down. The Quad Decoder Configuration registers lose their content and cannot be restored when powered-up again. Therefore, user must reconfigure the Quad Decoder peripheral at power-up to ensure it is in a well-defined state before use.

For more details on reconfiguration, refer to the ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User guide. This document also explains on how sleep and wake-up are controlled.

21.6 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B280	LPMCU_MISC_REGS0	QUAD_DEC_IRQS	7:0						QUAD_DEC2_IRQ	QUAD_DEC1_IRQ	QUAD_DEC0_IRQ
0x4000B284 (QUAD_DEC0), 0x4000B288 (QUAD_DEC1), 0x4000B28C (QUAD_DEC2)	LPMCU_MISC_REGS0	QUAD_DECn_STATUS (n=0,1,2)	7:0	COUNT[7:0]							
			15:8	COUNT[15:8]							
0x4000B290 (QUAD_DEC0), 0x4000B2A0 (QUAD_DEC1), 0x4000B2B0 (QUAD_DEC2)	LPMCU_MISC_REGS0	QUAD_DECn_THRESHOLD (n=0,1,2)	7:0	UPPER[7:0]							
			15:8	UPPER[15:8]							
			23:16	LOWER[7:0]							
			31:24	LOWER[15:8]							
0x4000B294 (QUAD_DEC0), 0x4000B2A4 (QUAD_DEC1), 0x4000B2B4 (QUAD_DEC2)	LPMCU_MISC_REGS0	QUAD_DECn_CTRL (n=0,1,2)	7:0					CLOCK_SEL[1:0]		CLR_IRQ	ENABLE
0x4000B044	LPMCU_MISC_REGS0	PINMUX_SEL_0	7:0		PINMUX_SEL[2:0] LP_GPIO_1				PINMUX_SEL[2:0] LP_GPIO_0		
			15:8		PINMUX_SEL[2:0] LP_GPIO_3				PINMUX_SEL[2:0] LP_GPIO_2		
			23:16		PINMUX_SEL[2:0] LP_GPIO_5				PINMUX_SEL[2:0] LP_GPIO_4		
			31:24		PINMUX_SEL[2:0] LP_GPIO_7				PINMUX_SEL[2:0] LP_GPIO_6		
0x4000B048	LPMCU_MISC_REGS0	PINMUX_SEL_1	7:0		PINMUX_SEL[2:0] LP_GPIO_9				PINMUX_SEL[2:0] LP_GPIO_8		
			15:8		PINMUX_SEL[2:0] LP_GPIO_11				PINMUX_SEL[2:0] LP_GPIO_10		
			23:16		PINMUX_SEL[2:0] LP_GPIO_13				PINMUX_SEL[2:0] LP_GPIO_12		
			31:24		PINMUX_SEL[2:0] LP_GPIO_15				PINMUX_SEL[2:0] LP_GPIO_14		
0x4000B04C	LPMCU_MISC_REGS0	PINMUX_SEL_2	7:0		PINMUX_SEL[2:0] LP_GPIO_17				PINMUX_SEL[2:0] LP_GPIO_16		
			15:8		PINMUX_SEL[2:0] LP_GPIO_19				PINMUX_SEL[2:0] LP_GPIO_18		
			23:16		PINMUX_SEL[2:0] LP_GPIO_21				PINMUX_SEL[2:0] LP_GPIO_20		
			31:24		PINMUX_SEL[2:0] LP_GPIO_23				PINMUX_SEL[2:0] LP_GPIO_22		
0x4000B080	LPMCU_MISC_REGS0	PINMUX_SEL_4	7:0						PINMUX_SEL[2:0] LP_GPIO_24		
			15:8								
			23:16								
			31:24								
0x4000B1A0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_0	7:0			MEGAMUX_SEL[5:0] LP_GPIO_0					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_1					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_2					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_3					

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x4000B1A4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_1	7:0			MEGAMUX_SEL[5:0] LP_GPIO_4					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_5					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_6					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_7					
0x4000B1A8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_2	7:0			MEGAMUX_SEL[5:0] LP_GPIO_8					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_9					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_10					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_11					
0x4000B1AC	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_3	7:0			MEGAMUX_SEL[5:0] LP_GPIO_12					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_13					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_14					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_15					
0x4000B1B0	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_4	7:0			MEGAMUX_SEL[5:0] LP_GPIO_16					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_17					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_18					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_19					
0x4000B1B4	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_5	7:0			MEGAMUX_SEL[5:0] LP_GPIO_20					
			15:8			MEGAMUX_SEL[5:0] LP_GPIO_21					
			23:16			MEGAMUX_SEL[5:0] LP_GPIO_22					
			31:24			MEGAMUX_SEL[5:0] LP_GPIO_23					
0x4000B1B8	LPMCU_MISC_REGS0	MEGA_MUX_IO_SEL_6	7:0			MEGAMUX_SEL[5:0] LP_GPIO_24					
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0				MUX_0[4:0]				
			15:8				MUX_1[4:0]				
			23:16				MUX_2[4:0]				
			31:24				MUX_3[4:0]				
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0				MUX_4[4:0]				
			15:8				MUX_5[4:0]				
			23:16				MUX_6[4:0]				
			31:24				MUX_7[4:0]				
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0				MUX_8[4:0]				
			15:8				MUX_9[4:0]				
			23:16				MUX_10[4:0]				
			31:24				MUX_11[4:0]				
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0				MUX_12[4:0]				
			15:8				MUX_13[4:0]				
			23:16				MUX_14[4:0]				
			31:24				MUX_15[4:0]				
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0				MUX_16[4:0]				
			15:8				MUX_17[4:0]				
			23:16				MUX_18[4:0]				
			31:24				MUX_19[4:0]				
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0				MUX_20[4:0]				

21.7 Register Description

21.7.1 Quad Decoder IRQ Status

Name: QUAD_DEC_IRQS
Reset: 0x00

Absolute Address: 0x4000B280

This register is a part of LPMCU_MISC_REGS0 registers. This register indicates the IRQ status of QUAD_DECn (n=0,1,2).

Bit	7	6	5	4	3	2	1	0
						QUAD_DEC2_IRQ	QUAD_DEC1_IRQ	QUAD_DEC0_IRQ
Access						R	R	R
Reset						0	0	0

Bit 2 – QUAD_DEC2_IRQ
Reading '1' on this bit indicates the COUNT[15:0] of Quad Decoder 2 that reaches the threshold limit set in UPPER[15:0] or LOWER[15:0]

Bit 1 – QUAD_DEC1_IRQ
Reading '1' on this bit indicates the COUNT[15:0] of Quad Decoder 1 that reaches the threshold limit set in UPPER[15:0] or LOWER[15:0]

Bit 0 – QUAD_DEC0_IRQ
Reading '1' on this bit indicates the COUNT[15:0] of Quad Decoder 0 that reaches the threshold limit set in UPPER[15:0] or LOWER[15:0]

21.7.2 Quad Decoder Current Count

Name: QUAD_DECn_STATUS (n=0,1,2)
Reset: 0x0000

Absolute Address: 0x4000B284 (QUAD_DEC0), 0x4000B288 (QUAD_DEC1), 0x4000B28C (QUAD_DEC2)

This register is a part of LPMCU_MISC_REGS0 registers. This register contains the current counter value. There are three individual quad decoder blocks and therefore there are three QUAD_DECn_STATUS (n=0,1,2) registers.

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – COUNT[15:0]
These bits provide the current count value of QUAD_DECn

21.7.3 Quad Decoder Threshold Limit Set

Name: QUAD_DECn_THRESHOLD (n=0,1,2)
Reset: 0x00FFFF00

Absolute Address: 0x4000B290 (QUAD_DEC0), 0x4000B2A0 (QUAD_DEC1), 0x4000B2B0 (QUAD_DEC2)

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to set the upper and lower threshold limit to trigger the interrupt when the current counter matches with the set limit of the QUAD_DECn_THRESHOLD register. There are three individual quad decoder blocks and therefore there are three QUAD_DECn_THRESHOLD (n=0,1,2) registers.

Bit	31	30	29	28	27	26	25	24
	LOWER[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	LOWER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	UPPER[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	UPPER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – LOWER[15:0]
These bits hold the lower threshold of counter for QUAD_DECn register

Bits 15:0 – UPPER[15:0]
These bits hold the upper threshold of counter for QUAD_DECn register

21.7.4 Quad Decoder Control

Name: QUAD_DECn_CTRL (n=0,1,2)
Reset: 0x00

Absolute Address: 0x4000B294 (QUAD_DEC0), 0x4000B2A4 (QUAD_DEC1), 0x4000B2B4 (QUAD_DEC2)

This register is a part of LPMCU_MISC_REGS0 registers. This register allows the user to enable the quad decoder operation and also allows selection of the input clock source. There are three individual quad decoder blocks, therefore there are three QUAD_DECn_CTRL (n=0,1,2) registers.

Bit	7	6	5	4	3	2	1	0
					CLOCK_SEL[1:0]		CLR_IRQ	ENABLE
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:2 – CLOCK_SEL[1:0]
These bits select the input clock for QUAD_DECn module.

CLOCK_SEL[1:0]	Description
0	26 MHz clock
1	13 MHz clock
2	6.5 MHz clock
3	3.25 MHz clock

Bit 1 – CLR_IRQ
Writing '1' to this bit clears the QUAD_DECn_IRQ bit

Bit 0 – ENABLE
Writing '0' to this bit disables QUAD_DECn module
Writing '1' to this bit enables QUAD_DECn module

22. Analog-to-Digital Converter

The Analog-to-Digital Converter (ADC) converts analog signals to digital values. The ADC has up to 11-bit resolution, and sampling rate of up to 1 MSPS. Ultra-low power Successive Approximation Register (SAR) ADC with single-ended input and resolution up to 11-bit is intended to measure low frequency input signal either received from external sensor or on-chip signal.

An integrated temperature sensor is available for use with the ADC. The V_{batt} voltage can also be measured by the ADC.

The Bluetooth Low Energy ROM firmware uses the ADC to measure temperature and battery voltage to do periodic clock calibration. When user application requires ADC, it is required to disable this periodic measurement before using ADC. When the application completes with the measurement, enable the periodic measurement. For more details on API, refer to the BluSDKSmart API reference guide.

22.1 Features

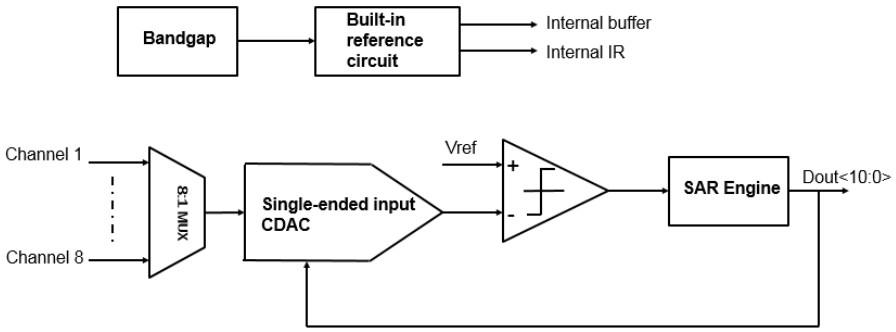
The following are the ADC features:

- 11-bit resolution
- Up to 1,000,000 samples per second (1 MSPS)
- Single-ended inputs:
  - Up to eight analog inputs, including internal and external
- Internal inputs:
  - Internal temperature sensor
  - Scaled core supply
  - Low power domain voltage
- Continuous and sequencing (time domain multiplexing) measurement options
- Built-in internal reference and external reference options
- Selectable sampling rate

22.2 Block Diagram

The following figure shows the hierarchical interface of the ADC with eight single-ended input channels. The reference voltage can be either generated internally (internal buffered or internal IR) or set externally from an accurate reference circuit. The key building block of the ADC which are single-ended capacitive DAC, comparator and synchronous SAR engine.

Figure 22-1. SAR ADC Block Diagram



22.3 Clock Configuration

The ADC peripheral clock must be enabled before configuring the ADC registers. This is done by setting SENS_ADC_CLK_EN bit in the LPMCU_CLOCK_ENABLES_1 register. For more details on configuration, see Peripheral Clock Configuration.

22.4 Functional Description

22.4.1 Initialization

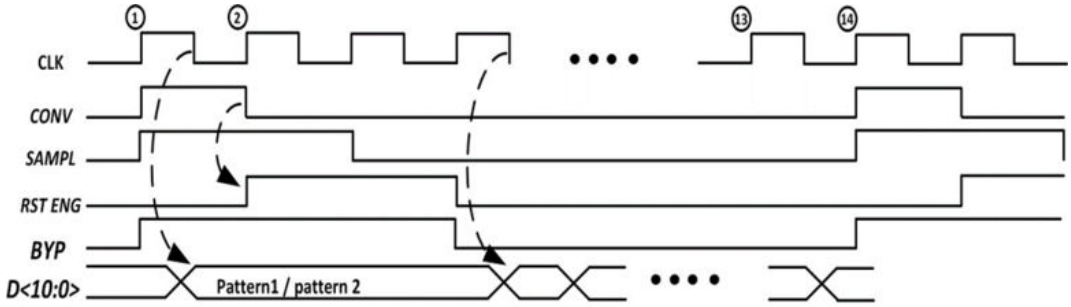
The GPIO_MSy pins are used as analog input. To initialize the ADC, perform the following:

1. Enable the Analog mode on GPIO_MSy pin by setting ANALOG_EN GPIO_MSy bit of the MS_GPIO_MODE register. This configuration is required if GPIO_MSy pin is used for ADC input signal or ADC external reference signal.
2. Select the comparator biasing current based on the sampling rate using SADC_LP_CTRL [1:0] bits of the RF_PMU_REGS_1 register.

22.4.1.1 Conversion Timing and Sampling Rate

The ADC timing is shown in the following figure. The ADC input signal is sampled twice. In the first sampling cycle, the input range is defined either to be above or below the reference voltage, and in the second sampling cycle, the ADC starts its normal operation.

Figure 22-2. SAR ADC Timing



The ADC takes two sampling cycles, N-1 conversion cycle (N=ADC resolution), and one cycle to sample the data out. Therefore, for the 11-bit resolution, it takes 13 clock cycles to complete one sample conversion.

N+2 is the sampling clock frequency (N is the ADC resolution) required for ADC conversion to complete.

CONV signal : Gives indication about the end of conversion.

SAMPL : The input signal is sampled when this signal is high.

RST ENG : When high SAR engine is in the Reset mode.

The 26 MHz is the input clock source for ADC. There is also a prescaler in the ADC to enable conversion at lower clock rates by configuring the SENS_ADC_CLK_CTRL register. The register consists of two fields, INT_PART[11:0] specifying the integral division of the clock, and FRAC_PART[7:0] specifying the fractional division of the clock. Sampling rate is calculated using the following formulas.

Clock_Division = INT_PART.(FRAC_PART/256)

(Here '.' is decimal point)

Sampling Rate = 26 MHz/(Clock_Division * (N+2))

(where N=11 for 11-bit resolution)

22.4.1.2 Reference Configuration

The ADC has eight sources for reference voltage (V_{REF}). The reference voltage selection bits, SADC_REF_SEL[2:0] in the RF_PMU_REGS_1 register determine the reference source. By default, the internal buffered voltage reference is selected. Based on the application requirements, the external or internal reference can be selected.

When the SADC_REF_SEL[2:0] bits are selected for internal voltage reference or buffered voltage reference then SADC_BIAS_RES_CTRL[3:0] of the RF_PMU_REGS_1 register select different reference voltage levels.

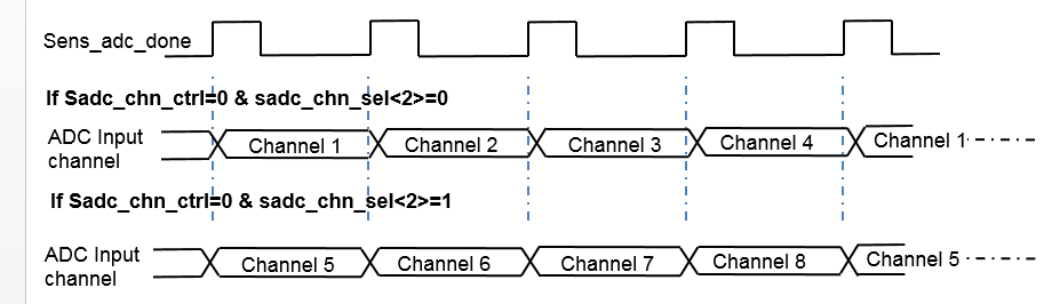
The effective reference voltage is double the value of set reference. For example, if the set reference voltage is ADC_INTERNAL_BUF_0_5 (SADC_BIAS_RES_CTRL[3:0] = 0x0) with internal buffered reference voltage (SADC_REF_SEL[2:0]=0x0) then the effective reference voltage is 1V (0.5*2).

22.4.1.3 Input Channel Selection

There are two modes for input channel selection using single control bit (SADC_CHN_CTRL) of the RF_PMU_REGS_1 register.

1. Specific input channel using channel selection bits SADC_CHN_SEL[2:0] is selected when SADC_CHN_CTRL bit is written as ‘1’. End of conversion is indicated through ADC_DONE bit of the SENS_ADC_RAW_STATUS register. The conversions happen repeatedly and automatically at defined sampling rate on the selected input channel.
2. Time domain multiplexing happens between four input channels when SADC_CHN_CTRL bit is written as ‘0’, based on the rising edge of sens_adc_done which indicates the end of conversion. There are two scenarios in time domain multiplexing; that is, when bit 2 of SADC_CHN_SEL[2] is set as '0' then time multiplexing occurs between input channel 1 to input channel 4, else time multiplexing occurs between input channel 5 to input channel 8, as shown in the following figure.

Figure 22-3. Time Domain Multiplexing



22.4.2 Enabling and Disabling ADC

The ADC is enabled by writing '0' to PMU_SENS_ADC_RST bit, writing '1' to PMU_SENS_ADC_EN and PMU_BGR_EN bits of the AON_PMU_CONTROL register.

When ADC enables, the conversion is started and repeated based on the sampling rate set. The conversion can be stopped by writing '0' to PMU_SENS_ADC_EN, and PMU_BGR_EN bits of the AON_PMU_CONTROL register.

22.4.3 Operation

The ADC samples values from the configured internal or external sources. The rate of conversion depends on the clock prescaler. For more details, see Conversion Timing and Sampling Rate.

To convert analog values to digital values, perform the following:

1. Initialize the ADC. For more details, see Initialization.
2. Enable the ADC to start the data conversion. The ADC operates in the free-running mode which continuously converts the analog input to digital output. In the free-running mode, initiate the first conversion and the subsequent conversions automatically start at the end of a previous conversion.

The end of conversion is indicated by setting ADC_DONE bit of the SENS_ADC_RAW_STATUS register. The transition of the ADC_DONE bit from '0' to '1' indicates that the ADC conversion is complete. The ADC_DONE bit is auto clear bit. The result of the conversion is stored in the Result register SENS_ADC_CHn_DATA (n=0,1,2,3) overwriting the result from the previous conversion.

22.4.4 Example

The following is the sample example to convert analog input on GPIO_MS1 pin using internal buffered reference voltage of 1.0V, and 100 ksps sampling rate:

```
/* Write MS_GPIO_MODE, ANALOG_EN GPIO_MS1 bit to 0x1 to enable analog mode  */
MS_GPIO_MODE.bit.ANALOG_EN GPIO_MS1 = 1;
 /* Calculate Clock division for sampling rate 100ksps
Sampling Rate = 26 MHz/(Clock_Division * (N+2))
Clock_Division = 26000 KHz /(100ksps * 13) = 20
Clock_Division = INT_PART.(FRAC_PART/256)
INT_PART = 20, FRAC_PART = 0 */
LPMCU_MISC_REGS0->SENS_ADC_CLK_CTRL.bit.INT_PART = 20;
 LPMCU_MISC_REGS0->SENS_ADC_CLK_CTRL.bit.FRAC_PART = 0;
/* Set the bias current control for 100ksps to 0 */
AON_GP_REGS0->RF_PMU_REGS_1.bit.SADC_LP_CTRL = 0;
/* Set the ADC reference as internal buffered*/
AON_GP_REGS0->RF_PMU_REGS_1.bit.SADC_REF_SEL = 0;
/* Set the reference voltage 1.0V */
AON_GP_REGS0->RF_PMU_REGS_1.bit.SADC_BIAS_RES_CTRL = 5;
/* Channel selection */
AON_GP_REGS0->RF_PMU_REGS_1.bit.SADC_CHN_CTRL =1;
AON_GP_REGS0->RF_PMU_REGS_1.bit.SADC_CHN_SEL =0; //GPIO_MS1
/* Enable ADC */
AON_GP_REGS0->AON_PMU_CTRL.bit.PMU_SENS_ADC_RST=0;
AON_GP_REGS0->AON_PMU_CTRL.bit.PMU_BGR_EN = 1;
```

```
AON_GP_REGS0->AON_PMU_CTRL.bit.PMU_SENS_ADC_EN = 1;
/* Wait for ADC conversion to complete */
/* The transition of the ADC_DONE signal from LO to HI indicates that the ADC conversion is
done. */
while((LPMCU_MISC_REGS0->SENS_ADC_RAW_STATUS.bit.ADC_DONE));
while(!(LPMCU_MISC_REGS0->SENS_ADC_RAW_STATUS.bit.ADC_DONE));
/* Read the result register. GPIO_MS1 result is available on CH0 result register */
result = LPMCU_MISC_REGS0->SENS_ADC_CH0_DATA.reg;
```

22.4.5 Offset and Gain Correction

Inherent gain and offset errors affect the absolute accuracy of the ADC. The offset error is defined as the deviation of the actual ADC transfer function from an ideal straight line at zero input voltage. The offset correction value to be calculated by giving zero voltage on input line. The converted value at zero input voltage to be subtracted from the converted value for actual input.

The gain error is defined as the deviation of the last output step's midpoint from the ideal straight line, after compensating for offset error. Offset and gain error compensation results are calculated according to:

Result = (Conversion value +/− OFFSETCORR) · GAINCORR

The SAMB11 does not have registers for gain and offset correction, therefore user application must perform compensation as per above formula to get the end result.

22.4.6 Device Temperature Measurement

The ATSAMB11 has an integrated temperature sensor. The analog signal of the temperature sensor is converted into a digital value by the ADC. The digital value can be converted into a temperature, in °C using the following the steps.

1. Select the input channel as V_{temp} by writing 0x04 to SADC_CHN_SEL[2:0] and writing '1' to SADC_CHN_CTRL bits.
2. Use internal buffered reference signal as V_{ref} by writing 0x00 to SADC_REF_SEL[2:0] bits.
3. Use 0.9 V as reference voltage by writing 0x04 to SADC_BIAS_RES_CTRL[3:0] bits.
4. Set the sampling rate (see Conversion Timing and Sampling Rate).
5. Select the comparator biasing current based on the sampling rate using SADC_LP_CTRL bits of the RF_PMU_REGS_1 register.
6. Enable the ADC and accumulate the ADC output value and get the average value (temp_accu_ave).
   Formula to calculate temperature in Celsius:

   Temperature in Celsius = (float)(temp_accu_ave-1690.0)/(-4.2)

   Note: The values 1690 and (-4.2) are the offset and gain correction values derived at 0 deg Celsius with 0.9V internal buffered voltage. These values to be derived if different reference voltage is used by the user.

22.5 Power Management

The ADC configuration registers which are part of AON power domain are intact when the device switches to the ULP mode. However, during end of conversion ADC cannot wake up the device. Therefore, it is required to disable ADC before switching to the ULP mode to save power.

22.6 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x4000F004	AON_GP_REGS0	AON_PMU_CONTROL	7:0	EFUSE_LDO_EN		PMU_RETN_VAL_SEL[1:0]				PMU_RTC_CLK_EN	PMU_REGS_4TO1_SEL
			15:8	PMU_MUX_SEL[3:0]				PMU_MUX_A[2:0]			PMU_MUX_EN
			23:16				PMU_26MHZ_CLK_FORCE_OFF	PMU_2MHZ_CLK_EN	PMU_BGR_EN	PMU_SENS_ADC_RST	PMU_SENS_ADC_EN
			31:24								
0x4000F404	AON_GP_REGS0	RF_PMU_REGS_1	7:0	BOD_EN	SADC_REF_SEL[2:0]			SADC_BIAS_RES_CTRL[3:0]			
			15:8	PIERCE_RES_CTRL	EFUSE_LDO_IBIAS_CTRL[1:0]		EFUSE_LDO_VOUT_CTRL[2:0]			EFUSE_LDO_BYP	LPD_CLK_INJECT_EN
			23:16	PIERCE_CAP_CTRL[3:0]				PIERCE_GM_CTRL[3:0]			
			31:24	SADC_LP_CTRL[1:0]		CODE_IN[1:0]		SADC_CHN_SEL[2:0]			SADC_CHN_CTRL
0x4000B1C0	LPMCU_MISC_REGS0	SENS_ADC_CLK_CTRL	7:0	FRAC_PART[7:0]							
			15:8	INT_PART[7:0]							
			23:16				INVERT	INT_PART[11:8]			
0x4000B1C4	LPMCU_MISC_REGS0	SENS_ADC_RAW_STATUS	7:0	ADC_OUT[7:0]							
			15:8			ADC_CH[1:0]			ADC_OUT[10:8]		
			23:16								ADC_DONE
0x4000B1C8	LPMCU_MISC_REGS0	SENS_ADC_CH0_DATA	7:0	SENS_ADC_CH0_DATA[7:0]							
			15:8						SENS_ADC_CH0_DATA[10:8]		
0x4000B1CC	LPMCU_MISC_REGS0	SENS_ADC_CH1_DATA	7:0	SENS_ADC_CH1_DATA[7:0]							
			15:8						SENS_ADC_CH1_DATA[10:8]		
0x4000B1D0	LPMCU_MISC_REGS0	SENS_ADC_CH2_DATA	7:0	SENS_ADC_CH2_DATA[7:0]							
			15:8						SENS_ADC_CH2_DATA[10:8]		
0x4000B1D4	LPMCU_MISC_REGS0	SENS_ADC_CH3_DATA	7:0	SENS_ADC_CH3_DATA[7:0]							
			15:8						SENS_ADC_CH3_DATA[10:8]		
0x4000F410	AON_GP_REGS0	MS_GPIO_MODE	7:0				ANALOG_EN GPIO_MS3	ANALOG_EN GPIO_MS2	ANALOG_EN GPIO_MS1	ANALOG_EN GPIO_MS0	

22.7 Register Description

22.7.1 AON PMU Control

Name: AON_PMU_CONTROL
Reset: 0x000A0022

Absolute Address: 0x4000F004

This register is a part of AON_GP_REGS0 registers. This register is the AON power domain control register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
				PMU_26MHZ_CLK_FORCE_OFF	PMU_2MHZ_CLK_EN	PMU_BGR_EN	PMU_SENS_ADC_RST	PMU_SENS_ADC_EN
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	1	0	1	0

Bit	15	14	13	12	11	10	9	8
	PMU_MUX_SEL[3:0]				PMU_MUX_A[2:0]			PMU_MUX_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFUSE_LDO_EN		PMU_RETN_VAL_SEL[1:0]				PMU_RTC_CLK_EN	PMU_REGS_4TO1_SEL
Access	R/W		R/W	R/W			R/W	R/W
Reset	0		1	0			1	0

Bit 20 – PMU_26MHZ_CLK_FORCE_OFF
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 19 – PMU_2MHZ_CLK_EN
This is an INTERNAL bit. Controlled by ROM firmware and not recommended to change

Bit 18 – PMU_BGR_EN
Writing '1' to this bit enables band gap reference voltage
Writing '0' to this bit disables band gap reference voltage

Bit 17 – PMU_SENS_ADC_RST
Writing '0' to this bit enables ADC Reset engine. To start ADC conversion, set this bit to '0'
Writing '1' to this bit disables ADC Reset engine

Bit 16 – PMU_SENS_ADC_EN
Writing '0' to this bit disables ADC conversion
Writing '1' to this bit enables ADC conversion

Bits 15:12 – PMU_MUX_SEL[3:0]
This is an INTERNAL bit and not recommended to change

Bits 11:9 – PMU_MUX_A[2:0]
This is an INTERNAL bit and not recommended to change

Bit 8 – PMU_MUX_EN
This is an INTERNAL bit and not recommended to change

Bit 7 – EFUSE_LDO_EN
Writing '1' to this bit enables power to eFuse banks to write the data into eFuse memory

Bits 5:4 – PMU_RETN_VAL_SEL[1:0]
This is an INTERNAL bit and not recommended to change

Bit 1 – PMU_RTC_CLK_EN
This is an INTERNAL bit and not recommended to change

Bit 0 – PMU_REGS_4TO1_SEL
This is an INTERNAL bit and not recommended to change

22.7.2 ADC Control

Name: RF_PMU_REGS_1
Reset: 0x31888C82

Absolute Address: 0x4000F404

This register is a part of AON_GP_REGS0 registers. This register allows the user to configure ADC.

Bit	31	30	29	28	27	26	25	24
	SADC_LP_CTRL[1:0]		CODE_IN[1:0]		SADC_CHN_SEL[2:0]			SADC_CHN_CTRL
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	0	0	0	1

Bit	23	22	21	20	19	18	17	16
	PIERCE_CAP_CTRL[3:0]				PIERCE_GM_CTRL[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	1	0	0	0

Bit	15	14	13	12	11	10	9	8
	PIERCE_RES_CTRL	EFUSE_LDO_IBIAS_CTRL[1:0]		EFUSE_LDO_VOUT_CTRL[2:0]			EFUSE_LDO_BYP	LPD_CLK_INJECT_EN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	1	1	0	0

Bit	7	6	5	4	3	2	1	0
	BOD_EN	SADC_REF_SEL[2:0]			SADC_BIAS_RES_CTRL[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	1	0

Bits 31:30 – SADC_LP_CTRL[1:0]
These bits control comparator biasing current for different sampling rates

SADC_LP_CTRL[1:0]	Description
0	Comparator bias current is 4 uA for throughput <=10 KS/s
1	Comparator bias current is 8 uA
2	Comparator bias current is 20 uA
3	Comparator bias current is 24 uA for throughput 1 MS/s

Bits 29:28 – CODE_IN[1:0]
The default value is 0x3, and it is not recommended to change this value

Bits 27:25 – SADC_CHN_SEL[2:0]
These bits select the input channel for measurement

SADC_CHN_SEL[2:0]	Channel Number	Description
0	1	Selects GPIO_MS1 as input
1	2	Selects GPIO_MS2 as input
2	3	Selects GPIO_MS3 as input
3	4	Selects GPIO_MS4 as input
4	5	Selects internal temperature sensor as input
5	6	Selects V_{batt}/4 as input
6	7	Selects 1.2V low power domain voltage as input
7	8	Selects one of eight V_{ref} as input. **Note:** When this channel is selected, the input signal and reference signal for ADC are same. Since the effective reference is double the actual reference voltage, the input signal is half of reference voltage.

Bit 24 – SADC_CHN_CTRL
Writing '1' to this bit selects specific input channel selected through SADC_CHN_SEL[2:0] bits
Writing '0' to this bit allows time domain multiplexing between four input channels which depend on bit '2' (SADC_CHN_SEL)

SADC_CHN_SEL[2]	Description
0	Channel 1 to 4
1	Channel 5 to 8

Bits 23:20 – PIERCE_CAP_CTRL[3:0]
These bits control the value of internal tuning capacitors for 32 KHz crystal input
Adds 1 pF per LSB. The value of 0x00 adds 0 pF and value of 0xF adds 15 pF. For more details, see 32.768 kHz XTAL C_onchip Programming.

Bits 19:16 – PIERCE_GM_CTRL[3:0]
These bits control the value of Gm stage gain for different Crystal mode for 32 KHz crystal. For more details, see RTC Characterization with Gm Code Variation at Supply 1.2V and Temp. = 25°C.

Bit 15 – PIERCE_RES_CTRL
This bit controls the feedback resistance value for 32 KHz crystal circuit
Writing '0' to this bit adds 20 MOhm feedback resistance
Writing '1' to this bit adds 30 MOhm Feedback resistance

Bits 14:13 – EFUSE_LDO_IBIAS_CTRL[1:0]
These are INTERNAL bits and not recommended to change

Bits 12:10 – EFUSE_LDO_VOUT_CTRL[2:0]
These are INTERNAL bits and not recommended to change

Bit 9 – EFUSE_LDO_BYP
This is an INTERNAL bit and not recommended to change

Bit 8 – LPD_CLK_INJECT_EN
This is an INTERNAL bit and not recommended to change

Bit 7 – BOD_EN
This is an INTERNAL bit and not recommended to change

Bits 6:4 – SADC_REF_SEL[2:0]
These bits select the reference signal for ADC measurement

SADC_REF_SEL[2:0]	Description
0	Selects internal buffered voltage as reference. It is possible to select different voltage levels using SADC_BIAS_RES_CTRL[3:0] bits.
1	Selects internal voltage as reference. It is possible to select different voltage levels using SADC_BIAS_RES_CTRL[3:0] bits.
2	Selects $V_{batt}/2$ as reference
3	Selects signal on GPIO_MS1 as reference
4	Selects signal on GPIO_MS2 as reference
5	Selects signal on GPIO_MS3 as reference
6	Selects signal on GPIO_MS4 as reference
7	Selects V_{batt} as reference

Bits 3:0 – SADC_BIAS_RES_CTRL[3:0]
These bits select the voltage level when internal buffer or internal reference is selected as reference signal for ADC measurement. The voltage level ranges from 0.5V to 2.0V in steps of 0.5V
The value of 0x00 equals 0.5V and value of 0xF equals 2.0V

22.7.3 ADC Input Clock Divider

Name: SENS_ADC_CLK_CTRL
Reset: 0x001A00

Absolute Address: 0x4000B1C0

This register is a part of LPMCU_MISC_REGS0 registers. This register sets the divide ratio which is used to generate the input clock for ADC.

Bit	23	22	21	20	19	18	17	16
				INVERT	INT_PART[11:8]			
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	INT_PART[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	0	1	0

Bit	7	6	5	4	3	2	1	0
	FRAC_PART[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 20 – INVERT
Writing '1' to this bit inverts the ADC input clock used for sampling

Bits 19:8 – INT_PART[11:0]
These bits are integer part of clock divider. For more details, see Conversion Timing and Sampling Rate.

Bits 7:0 – FRAC_PART[7:0]
These bits are fractional part of clock divider. For more details, see Conversion Timing and Sampling Rate.

22.7.4 ADC Conversion Status

Name: SENS_ADC_RAW_STATUS
Reset: 0x000000

Absolute Address: 0x4000B1C4

This register is a part of LPMCU_MISC_REGS0 registers. This register provides the status of ADC conversion.

Bit	23	22	21	20	19	18	17	16
								ADC_DONE
Access								R
Reset								0

Bit	15	14	13	12	11	10	9	8
			ADC_CH[1:0]			ADC_OUT[10:8]		
Access			R	R		R	R	R
Reset			0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
	ADC_OUT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 16 – ADC_DONE
Reading '1' indicates ADC conversion is completed and the result is available in the SENS_ADC_CHn_DATA register

Bits 13:12 – ADC_CH[1:0]
These bits provide the channel number of conversion data available in ADC_OUT[10:0] bits

Bits 10:0 – ADC_OUT[10:0]
These bits hold the converted output data
The data is also available on specific channel of the SENS_ADC_CHn_DATA register

22.7.5 ADC Result Register 0

Name: SENS_ADC_CH0_DATA
Reset: 0x0000

Absolute Address: 0x4000B1C8

This register is a part of LPMCU_MISC_REGS0 registers. This register holds the converted output data for Channel 1 or Channel 5.

Bit	15	14	13	12	11	10	9	8
						SENS_ADC_CH0_DATA[10:8]		
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	SENS_ADC_CH0_DATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 10:0 – SENS_ADC_CH0_DATA[10:0]
These bits hold the converted output data for Channel 1 or Channel 5

22.7.6 ADC Result Register 1

Name: SENS_ADC_CH1_DATA
Reset: 0x0000

Absolute Address: 0x4000B1CC

This register is a part of LPMCU_MISC_REGS0 registers. This register holds the converted output data for Channel 2 or Channel 6.

Bit	15	14	13	12	11	10	9	8
						SENS_ADC_CH1_DATA[10:8]		
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	SENS_ADC_CH1_DATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 10:0 – SENS_ADC_CH1_DATA[10:0]
These bits hold the converted output data for Channel 2 or Channel 6

22.7.7 ADC Result Register 2

Name: SENS_ADC_CH2_DATA
Reset: 0x0000

Absolute Address: 0x4000B1D0

This register is a part of LPMCU_MISC_REGS0 registers. This register holds the converted output data for Channel 3 or Channel 7.

Bit	15	14	13	12	11	10	9	8
						SENS_ADC_CH2_DATA[10:8]		
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	SENS_ADC_CH2_DATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 10:0 – SENS_ADC_CH2_DATA[10:0]
These bits hold the converted output data for Channel 3 or Channel 7

22.7.8 ADC Result Register 3

Name: SENS_ADC_CH3_DATA
Reset: 0x0000

Absolute Address: 0x4000B1D4

This register is a part of LPMCU_MISC_REGS0 registers. This register holds the converted output data for Channel 4 or Channel 8.

Bit	15	14	13	12	11	10	9	8
						SENS_ADC_CH3_DATA[10:8]		
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	SENS_ADC_CH3_DATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 10:0 – SENS_ADC_CH3_DATA[10:0]
These bits hold the converted output data for Channel 4 or Channel 8

23. Direct Memory Access Controller

The Direct Memory Access (DMA) Controller allows certain hardware subsystems to access main system memory independent of the Cortex-M0 processor. The DMA is used to replace two CPU functions, such as memory copy and peripheral control (slow peripheral devices such as SPI, UART, etc.) and thus off-load these tasks from the CPU. It enables high data transfer rates with minimum CPU intervention, and frees up CPU time.

23.1 Features

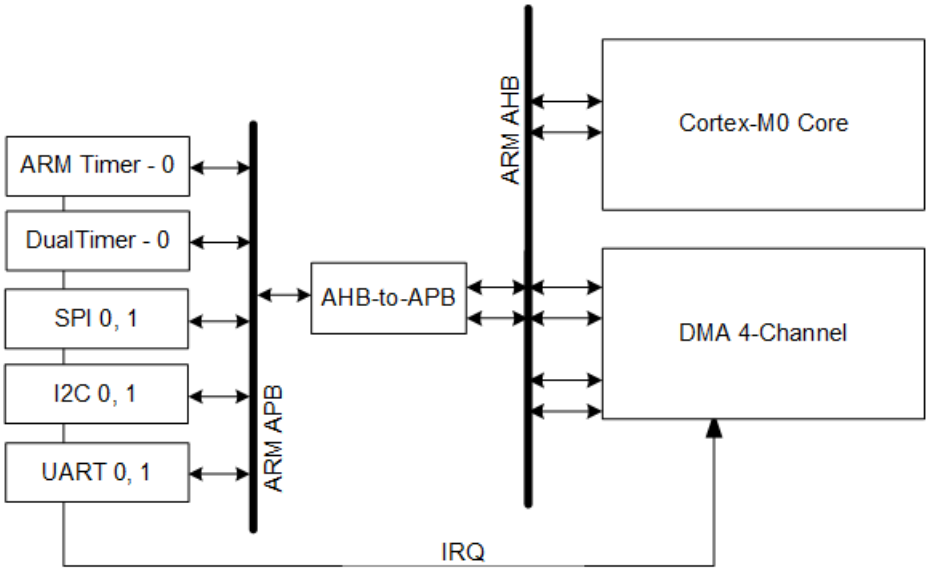
The DMA features and benefits are:

- Supports any address alignment
- Supports any buffer size alignment
- The following modes are supported:
  - Peripheral to Peripheral Transfer
  - Memory to Memory
  - Memory to Peripheral
  - Peripheral to Memory
  - Register to Memory
- RAM-based transfer descriptors:
  - Single transfer using one descriptor
  - Multi-Buffer or Circular Buffer modes by linking multiple descriptors (command list)
- Upto 4-channel operation
- Interrupts for both TX done and RX done in memory and peripheral mode
- Endianness byte swapping
- 32-bit data width
- AHB mux (on read and write AHB buses) core features:
  - Independent mode support
  - Command lists support
  - Usage of tokens
  - Time-outs on AHB buses
  - Watchdog timer
  - Peripheral control

23.2 Block Diagram

The DMA Engine consists of 4-channels, each capable of performing a DMA command independently of the other channels.

Figure 23-1. DMA Controller Block Diagram



23.3 Operation Modes

The operation mode determines how the channels use the AHB read and write buses. It can be configured to work in the Independent or Joint mode.

When using the Joint mode, each channel can work in the Normal or Joint mode.

23.3.1 Independent Mode - Normal Channel Mode

When using the Independent mode, DMA core uses an independent arbiter for read operations and write operations. The read operations on the AHB bus is issued without considering the write operations. This mode is configured in the CORE_JOINT_MODE register by writing ‘0’.

Each channel works in the following manner:

1. The channel calculates the next read and write burst sizes according to AHB address and software restrictions.
2. The channel issues read bursts as long as the data buffer can hold the data.
3. The channel issues write bursts as long as the write data is present in the data buffer.

This mode is most efficient when:

- Read and write bursts channels are of different sizes (like when transferring data from peripheral devices to DRAM)
- AHB slaves have unpredicted response times or long response times
- Multiple channels work simultaneously

The channel does not start a write burst until the read data is arrived, this causes long latency and in order to achieve maximum throughput the data buffer must be enlarged.

23.3.1.1 Burst Flow

To achieve maximum throughput regardless to alignments, the DMA controller starts with single AHB command until reaching a round address (multiples of burst size) from where full strobe bursts are possible.

The calculation of burst length considers the following:

- Does not exceed the value set in BURST_MAX_SIZE
- Does not exceed FIFO size
- Burst width must be aligned to burst address

The following is the data flow example:

FIFO_SIZE = 64 bytes

RD_START_ADDR = 0x30000001

RD_BURST_MAX_SIZE = 64 bytes

WR_START_ADDR = 0x40000017

WR_BURST_MAX_SIZE = 64 bytes

BUFFER_SIZE = 128 bytes

Table 23-1. Burst Flow Example – Independent Mode

Read					Write			
Burst Address	Burst Type	Burst Size	Buffer Remain		Burst Address	Burst Type	Burst Size	Buffer Remain
0x30000001	Single	1	128		0x40000017	Single	1	128
0x30000002	Single	2	127		0x40000018	Single	4	127
0x30000004	Single	4	125		0x4000001C	Single	4	123
0x30000008	Single	4	121		0x40000020	INCR8	32	119
0x3000000C	Single	4	117		0x40000040	INCR16	64	8
0x30000010	INCR4	16	113		0x40000080	INCR4	16	23
0x30000020	INCR8	32	97		0x40000090	Single	4	7
0x30000040	INCR16	64	65		0x40000094	Single	2	3
0x30000080	Single	1	1		0x40000096	Single	1	1

23.3.2 Joint Mode

In the Joint mode, DMA core uses a single arbiter for both read and write operations. This mode is used for channels that work at the same pace for their read and write operations. The current channel locks both read and write AHB buses and transfers the read data directly to the write data. When working in this mode the channel's configuration is done in the read registers only and they affect both read and write operations that are simultaneous.

This mode is configured in the CORE_JOINT_MODE register by writing ‘1’.

Each channel works as per the following process:

1. The channel reads single bursts until it reaches an aligned address (multiples of burst size) from which it can perform its requested bursts.
2. The channel writes single bursts until it reaches an aligned address (multiples of burst size) from which it can perform its requested bursts.
3. The channel moves into its joint stage, during this stage the channel reads and writes simultaneously on both AHB buses.

4. The channel performs single bursts to finish the remaining last bytes.

This mode is most efficient when:

- The read and write slaves work in same size bursts.
- The slave does not have many wait cycles (wait cycles on the read bus stall the write bus and vice versa).
- Working with large transfer buffers.

23.3.2.1 Disadvantages

- It is possible to use channels working in the Joint mode and Normal mode but these do not work well together as there is a flush stage when transferring from a 'joint' working channel to a 'normal' working channel.
- The 'normal' working channels in a 'joint' working core can use only a single arbiter.

23.3.2.2 Restrictions

- Cannot work with peripherals.
- When using 16 bytes data buffer, the read start address and the write start address must be aligned (multiples of burst size) to the burst size. A 32 bytes data buffer or larger does not restrict the Joint mode.

23.3.2.3 Burst Flow

In the Joint mode, each channel issues single read and write bursts until it is possible to issue bursts that are equal in length to the BURST_MAX_SIZE value. Then, the channel moves into the joint phase, performing simultaneous read and write bursts. At the end of the buffer, the channel flushes the last bytes by issuing smaller bursts again.

The following is the data flow example:

FIFO_SIZE = 32 bytes

RD_START_ADDR = 0x30000031

RD_BURST_MAX_SIZE = 64 bytes (affects WR_BURST_MAX_SIZE as well)

WR_START_ADDR = 0x40000037

BUFFER_SIZE = 256 bytes

Table 23-2. Burst Flow Example – Joint Mode

Read				Write		
Burst Address	Burst Size	Buffer Remain		Burst Address	Burst Size	Buffer Remain
0x30000039	1 (Single)	256		0x400000BF	1 (Single)	256
0x3000003A	2 (Single)	255		Ready for joint - wait for read		
0x3000003C	4 (Single)	253		Ready for joint - wait for read		
Going into the Joint mode				Going into the Joint mode		
0x30000040	64 (INCR16)	249	joint	0x400000C0	64 (INCR16)	255
0x30000080	64 (INCR16)	185	joint	0x40000100	64 (INCR16)	191

.....continued

Read				Write		
Burst Address	Burst Size	Buffer Remain		Burst Address	Burst Size	Buffer Remain
0x300000C0	64 (INCR16)	121	joint	0x40000140	64 (INCR16)	127
Going back to the Normal mode				Going back to the Normal mode		
0x30000100	32 (INCR8)	57		0x40000180	32 (INCR8)	63
0x30000120	16 (INCR4)	25		0x400001A0	16 (INCR4)	31
0x30000130	4 (Single)	9		0x400001B0	4 (Single)	15
0x30000134	4 (Single)	5		0x400001B4	4 (Single)	11
0x30000138	1 (Single)	1		0x400001B8	4 (Single)	7
-	-	-		0x400001BC	2 (Single)	3
-	-	-		0x400001BE	1 (Single)	1

23.4 Functional Description

DMA command is constructed of four 32-bit fields. Each channel holds its current command in its CHn_CMD_REG0 to CHn_CMD_REG3 (Channel number n=0,1,2,3) registers.

23.4.1 Initialization

- Configure the CHn_STATIC_REG0 (n=0,1,2,3) register:
  - Set read max burst size in RD_BURST_MAX_SIZE[6:0] bits.
  - Write ‘1’ to RD_INCR bit for memory transfer.
  - Write ‘0’ to RD_INCR bit for peripheral transfer.
  - Write number of AHB read commands to be issued before the channel is released in RD_TOKENS[5:0] bits.
- Configure the CHn_STATIC_REG1 (n=0,1,2,3) register:
  - Set write max burst size in WR_BURST_MAX_SIZE[6:0] bits.
  - Write ‘1’ to WR_INCR bit for memory transfer.
  - Write ‘0’ to WR_INCR bit for peripheral transfer.
  - Write number of AHB write commands to be issued before the channel is released in WR_TOKENS[5:0] bits.
- Configure the CHn_STATIC_REG2 (n=0,1,2,3) register:
  - Write ‘1’ in JOINT bit for using the Joint mode, else write ‘0’ for Independent mode. For Joint mode, the CORE_JOINT_MODE register should also be written ‘1’.
- Configure the CHn_STATIC_REG4 (n=0,1,2,3) register:
  - Write ‘0’ to RD_PERIPH_NUM[4:0] bits if the source address is memory, else the peripheral number as in the Peripheral Number.
  - Write the number of cycles to wait for peripheral read request based on specific peripheral latency to RD_PERIPH_DELAY[2:0] bits. This is required only if the source address is peripheral.

  - Write ‘0’ to WR_PERIPH_NUM[4:0] bits if the destination address is memory, else the peripheral number as in the Peripheral Number.
  - Write the number of cycles to wait for peripheral write request based on specific peripheral latency to WR_PERIPH_DELAY[2:0] bits. This is required only if the destination address is peripheral.
- If command lists are not required then write the CHn_CMD_REG3 (n=0,1,2,3) register with 0x03 and perform the following steps. If more than one command is to be executed then follow the steps in DMA Command Lists.
  - Write the source start address in RD_START_ADDR[31:0] bits of the CHn_CMD_REG0 (n=0,1,2,3) register.
  - Write the destination start address in WR_START_ADDR[31:0] bits of the CHn_CMD_REG1 (n=0,1,2,3) register.
  - Write the length of data to be transferred from source to destination in BUFFER_SIZE[12:0] bits of the CHn_CMD_REG2 (n=0,1,2,3) register.
- If interrupt generation is required on end of transfer then configure NVIC to enable the interrupt. The DMA is not mapped to default interrupt sources in vector table. Therefore, it is required to use muxable interrupt option to map DMA as interrupt source. To generate an interrupt, the appropriate bits in the interrupt mask registers must be set as follows:
  - Select the IRQ number from ATSAMB11 Interrupt Vector Table which is not used in application.
  - Configure the DMA to this IRQ number. For more details, see Muxable Interrupt chapter.
  - Register the DMA ISR function in the interrupt vector table.
  - Enable the DMA interrupt in the NVIC interrupt controller registers. For more details, see Muxable Interrupt chapter.
  - Enable DMA end of conversion interrupt by writing ‘1’ to CH_END bit of the CHn_INT_ENABLE_REG (n=0,1,2,3) register.
    Note: Only one ISR index is allocated in interrupt vector table for all four DMA channels.

23.4.2 Start DMA Transfer

To start the DMA transfer, the specific channel must be enabled by writing ‘1’ to CH_ENABLE bit of the CHn_CH_ENABLE_REG (n=0,1,2,3) register. Start the DMA transfer by writing ‘1’ to CH_START bit of the CHn_CH_START_REG (n=0,1,2,3) register. End of transfer is indicated by ‘1’ in CH_END bit of the CHn_INT_STATUS_REG (n=0,1,2,3) register. If interrupt is enabled then DMA ISR handler is called.

23.4.3 Stop DMA Transfer

A channel works until it completes its last command and then it stops by itself. After stopping the status bits, CH_RD_ACTIVE and CH_WR_ACTIVE (CHn_CH_ACTIVE_REG (n=0,1,2,3)) will be 0.

A channel can be stopped by clearing the CH_ENABLE bit of the CHn_CH_ENABLE_REG (n=0,1,2,3) register.

23.4.4 Pause and Resume a Channel

A channel can be paused by clearing the CH_ENABLE of the CHn_CH_ENABLE_REG (n=0,1,2,3) register, and the channel can be resumed by setting the bit.

23.4.5 Restart a Channel

Perform the following steps to restart a channel:

1. Stop the channel by clearing CH_ENABLE of the CHn_CH_ENABLE_REG (n=0,1,2,3) register.
2. Restart the channel by setting CH_START of the CHn_CH_START_REG (n=0,1,2,3) register.

3. Enable the channel by setting CH_ENABLE of the CHn_CH_ENABLE_REG (n=0,1,2,3) register.

23.4.6 Power-Down DMA

Perform the following steps to power-down the DMA:

1. Clear CH_ENABLE bit in all active channels (this stops the channel at the completion of the current pending transactions) or set CMD_LAST bit (CHn_CMD_REG3 (n=0,1,2,3)) in all active channels (this stops the channel at the completion of the current buffer).
2. Wait for CORE_IDLE bit to set. The DMA is ready for power-down.

23.4.7 Example

The following is the sample example to transmit 512 bytes of data from buffer to UART0 (LP_GPIO_2 (RXD), LP_GPIO_3 (TXD) without flow control with 115200 baud rate, odd parity, 1 stop bit) using DMA CH0:

```
/*Initialise UART0 as in 19.6.3.2 Example before "Check for Tx buffer has space" comment.
Disable any UART interrupt enabled previously*/
/* DMA need the UART TX FIFO empty interrupt to indicate the DMA transfer is complete*/
UART0->TX_INTERRUPT_MASK.bit.TX_FIFO_EMPTY_MASK = 1;
/* As there is no default ISR mapping for DMA interrupt in Table 6-2. ATSAMB11 Interrupt
Vector Table, use muxable interrupt option as explained in 11.1 Example and register ISR
handler */

/* Static 0 register configuration for source (Memory)
RD_BURST_MAX_SIZE = 1, RD_TOKENS= 1, RD_INCR = 1 (For Memory transfer as the data is
available in RAM buffer) */
 PROV_DMA_CTRL0->CH0_STATIC_REG0.reg = (1<<0) | (1<<16) | (1<<31);
/* Static 1 register configuration for destination (UART)
WR_BURST_MAX_SIZE = 1, WR_TOKENS= 1, WR_INCR = 0 (For Peripheral transfer as the data to be
transmitted on UART peripheral) */
 PROV_DMA_CTRL0->CH0_STATIC_REG1.reg = (1<<0) | (1<<16) | (0<<31);
/* Static 2 register configuration
JOINT = 0 (Independant mode), END_SWAP= 0 (No swap), WR_INCR = 0 (For Peripheral transfer as
the data to be transmitted on UART peripheral) */
 PROV_DMA_CTRL0->CH0_STATIC_REG2.reg = (0<<28) | (0<<16);
/* Static 4 register configuration
RD_PERIPH_NUM= 0 (source is Memory), RD_PERIPH_DELAY= 0, WR_PERIPH_NUM= 2 (destination is
UART0 Transmit Interrupt from Table Peripheral Number), WR_PERIPH_DELAY= 0, */
 PROV_DMA_CTRL0->CH0_STATIC_REG4.reg = (0<<0) | (0<<8) |(2<<16) | (0<<24);
/* Command 0 register configuration RD_START_ADDR = &read_buf[0]; address of buffer where
data is present */
PROV_DMA_CTRL0->CH0_CMD_REG0.reg = &read_buf[0];
/* Command 1 register configuration WR_START_ADDR = &UART0->TRANSMIT_DATA.reg address of
UART0 Trasmit data register where data to be transmitted*/
PROV_DMA_CTRL0->CH0_CMD_REG1.reg = &UART0->TRANSMIT_DATA.reg;
/* Command 2 register configuration BUFFER_SIZE = 512, size of data to be transmitted */
PROV_DMA_CTRL0->CH0_CMD_REG2.reg = 512;
/* Command 3 register configuration CMD_LAST = 1, CMD_SET_INT =1, CMD_NEXT_ADDR = 0 as there
is only one command to be executed*/
PROV_DMA_CTRL0->CH0_CMD_REG3.reg= 0x3;
/*Enable DMA interrupt CH_END=1*/
PROV_DMA_CTRL0->CH0_INT_ENABLE_REG.reg= (1<<0);
/* Enable the transfer channel */
PROV_DMA_CTRL0->CH0_CH_ENABLE_REG = 1;
/* Start the transfer channel */
PROV_DMA_CTRL0->CH0_CH_START_REG.reg = 1;
```

23.5 Additional Features

23.5.1 DMA Command Lists

The DMA command lists are linked lists of the DMA commands that can be placed anywhere in the RAM memory space. When the channel completes its current command, and if the CMD_LAST field of the CHn_CMD_REG3 (n=0,1,2,3) register is 0, the channel control reads the next command on the AHB bus

from the address specified in the CMD_NEXT_ADDR field of the CHn_CMD_REG3 (n=0,1,2,3) register. If CMD_LAST is 1, the channel stops.

Write the entire command list to RAM memory and set in the current command (CMD registers) an empty buffer that points to the beginning of the list.

Perform the following configuration sequence:

1. Write command list to RAM memory as descriptor as shown in the following tables and example pseudo code.
2. Set RD_START_ADDR[31:0] as 0 in the CHn_CMD_REG0 (n=0,1,2,3) register.
3. Set WR_START_ADDR[31:0] as 0 in the CHn_CMD_REG1 (n=0,1,2,3) register.
4. Set BUFFER_SIZE[12:0] as 0 in the CHn_CMD_REG2 (n=0,1,2,3) register.
5. Set CMD_SET_INT as 0 in the CHn_CMD_REG3 (n=0,1,2,3) register.
6. Set CMD_LAST as 0 in the CHn_CMD_REG3 (n=0,1,2,3) register.
7. Set CMD_NEXT_ADDR as the address of first command in memory of the CHn_CMD_REG3 (n=0,1,2,3) register.

The following is an example pseudo code for command list descriptor:

```
dma_descriptor {
    /** Start address of read buffer */
    uint32_t read_start_addr;
    /** Start address of write buffer */
    uint32_t write_start_addr;
    /** Size (in bytes) of buffer to transfer */
    uint32_t buffer_size;
    union {
        struct {
            /** Active high interrupt enable once buffer has been transferred */
            uint32_t set_interrupt:1;
            /** If set, channel stops when buffer done, otherwise load from cmd_next_addr */
            uint32_t last:1;
            /** Address of next command if cmd_last is not set */
            uint32_t next_addr:30;
        } cmd;
        uint32_t next;
    };
```

The command lists can be used for the following two purposes:

1. Scatter – List. When system allocates large memory blocks, the memory is continuous in the virtual address space but non-continuous in the physical address space. The list of address-size pairs of the allocation process can be written to memory as a list of DMA commands allowing the DMA to transfer all these chunks of data without CPU intervention.
   Example: A scatter list describing a 20 KB memory block fragmented into five 4 KB pages. After the last page is written, the list is indicated to issue a completion interrupt to the CPU.

Table 23-3. Example Scatter List

Command's Address in Memory	Command Number	RD Start Address	WR Start Address	Buffer Size	CMD SET INT	CMD Last	CMD Next Address
0x30000000	0 (first)	0x40001000	0x50001000	0x1000	0	0	0x30000010/4
0x30000010	1	0x40002000	0x50008000	0x1000	0	0	0x30000020/4

.....continued							
Command's Address in Memory	Command Number	RD Start Address	WR Start Address	Buffer Size	CMD SET INT	CMD Last	CMD Next Address
0x30000020	2	0x40003000	0x50015000	0x1000	0	0	0x30000030/4
0x30000030	3	0x40004000	0x50017000	0x1000	0	0	0x30000040/4
0x30000040	4 (last)	0x40005000	0x50025000	0x1000	1	1	0

2. Cyclic buffers for peripheral control. When servicing peripheral devices, it is good practice to use at least double buffers to hold the peripheral RX or TX data. Cyclic buffers can be easily configured by setting a cyclic command list.
   Example: A cyclic double buffer at address 0x30000000, services RX peripheral client at address 0xBE000000, buffer sits at 0x50001000. Notice that the list is cyclic and endless; this is because most peripherals such as SPI, UART etc. require endless servicing.

Table 23-4. Example Cyclic List

Command's Address in Memory	Command Number	RD Start Address	WR Start Address	Buffer Size	CMD SET INT	CMD Last	CMD Next Address
0x30000000	0 (first)	0xBE000000	0x50001000	0x1000	1	0	0x30000010/4
0x30000010	1	0xBE000000	0x50002000	0x1000	1	0	0x30000000/4

23.5.2 Tokens

When a DMA channel starts working, it transfers a maximum number of bursts according to the value set in the RD_TOKENS (CHn_STATIC_REG0 (n=0,1,2,3)) or WR_TOKENS (CHn_STATIC_REG1 (n=0,1,2,3)) bits. Using many tokens improves the overall performance of the channel but can result in lengthening other channels' latency.

23.5.3 Endianness Byte Swapping

Each channel can manipulate the data written to support little-/big-endian devices. Byte swapping is configured in the END_SWAP [1:0] bits of the CHn_STATIC_REG2 (n=0,1,2,3) register. This device supports byte swapping within 32-bit data.

23.5.4 AHB Bus Time-out

If the AHB bus does not respond in 1024 cycles, a time-out interrupt is issued by the corresponding channel. In this way not only the interrupt indicates that the slave is not responding, but also indicates which channel issued the request. The value '1' in TIMEOUT_RD, TIMEOUT_WR status bits of the CHn_INT_STATUS_REG (n=0,1,2,3)/CHn_INT_RAWSTAT_REG (n=0,1,2,3) registers indicate this time-out.

23.5.5 Watchdog Timer

A watchdog timer is present inside the DMA core. The watchdog timer checks each active channel (enabled and not ended) in a round robin basis. If the checked channel does not start working in 2048

cycles, a time-out interrupt is issued by the checked channel. The value '1' in WDT status bit of the CHn_INT_STATUS_REG (n=0,1,2,3)/CHn_INT_RAWSTAT_REG (n=0,1,2,3) register indicates this watchdog time-out.

23.6 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40002000 (CH0), 0x40002100 (CH1), 0x40002200 (CH2), 0x40002300 (CH3)	PROV_DMA_CTRL0	CHn_CMD_REG0 (n=0,1,2,3)	7:0	RD_START_ADDR[7:0]							
			15:8	RD_START_ADDR[15:8]							
			23:16	RD_START_ADDR[23:16]							
			31:24	RD_START_ADDR[31:24]							
0x40002004 (CH0), 0x40002104 (CH1), 0x40002204 (CH2), 0x40002304 (CH3)	PROV_DMA_CTRL0	CHn_CMD_REG1 (n=0,1,2,3)	7:0	WR_START_ADDR[7:0]							
			15:8	WR_START_ADDR[15:8]							
			23:16	WR_START_ADDR[23:16]							
			31:24	WR_START_ADDR[31:24]							
0x40002008 (CH0), 0x40002108 (CH1), 0x40002208 (CH2), 0x40002308 (CH3)	PROV_DMA_CTRL0	CHn_CMD_REG2 (n=0,1,2,3)	7:0	BUFFER_SIZE[7:0]							
			15:8				BUFFER_SIZE[12:8]				
0x4000200C (CH0), 0x4000210C (CH1), 0x4000220C (CH2), 0x4000230C (CH3)	PROV_DMA_CTRL0	CHn_CMD_REG3 (n=0,1,2,3)	7:0	CMD_NEXT_ADDR[5:0]						CMD_LAST	CMD_NEXT_ADDR
			15:8	CMD_NEXT_ADDR[13:6]							
			23:16	CMD_NEXT_ADDR[21:14]							
			31:24	CMD_NEXT_ADDR[29:22]							
0x40002010 (CH0), 0x40002110 (CH1), 0x40002210 (CH2), 0x40002310 (CH3)	PROV_DMA_CTRL0	CHn_STATIC_REG0 (n=0,1,2,3)	7:0		RD_BURST_MAX_SIZE[6:0]						
			15:8								
			23:16			RD_TOKENS[5:0]					
			31:24	RD_INCR							
0x40002014 (CH0), 0x40002114 (CH1), 0x40002214 (CH2), 0x40002314 (CH3)	PROV_DMA_CTRL0	CHn_STATIC_REG1 (n=0,1,2,3)	7:0		WR_BURST_MAX_SIZE[6:0]						
			15:8								
			23:16			WR_TOKENS[5:0]					
			31:24	WR_INCR							
0x40002018 (CH0), 0x40002118 (CH1), 0x40002218 (CH2), 0x40002318 (CH3)	PROV_DMA_CTRL0	CHn_STATIC_REG2 (n=0,1,2,3)	7:0								
			15:8								
			23:16								JOINT
			31:24			END_SWAP[1:0]					
0x40002020 (CH0), 0x40002120 (CH1), 0x40002220 (CH2), 0x40002320 (CH3)	PROV_DMA_CTRL0	CHn_STATIC_REG4 (n=0,1,2,3)	7:0		RD_PERIPH_NUM[6:0]						
			15:8						RD_PERIPH_DELAY[2:0]		
			23:16				WR_PERIPH_NUM[4:0]				
			31:24						WR_PERIPH_DELAY[2:0]		
0x4000202C (CH0), 0x4000212C (CH1), 0x4000222C (CH2), 0x4000232C (CH3)	PROV_DMA_CTRL0	CHn_RESTRICT_REG (n=0,1,2,3)	7:0				ALLOW_JOINT_BURST	ALLOW_FULL_BURST	ALLOW_FULL_FIFO	WR_ALLOW_FULL_FIFO	RD_ALLOW_FULL_FIFO
			15:8								SIMPLE_MEM

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x40002038 (CH0), 0x40002138 (CH1), 0x40002238 (CH2), 0x40002338 (CH3)	PROV_DMA_CTRL0	CHn_FIFO_FULLNESS_REG (n=0, 1,2,3)	7:0	RD_GAP[7:0]							
			15:8				RD_GAP[12:8]				
			23:16	WR_FULLNESS[7:0]							
			31:24				WR_FULLNESS[12:8]				
0x40002040 (CH0), 0x40002140 (CH1), 0x40002240 (CH2), 0x40002340 (CH3)	PROV_DMA_CTRL0	CHn_CH_ENABLE_REG (n=0,1,2,3)	7:0								CH_ENABLE
0x40002044 (CH0), 0x40002144 (CH1), 0x40002244 (CH2), 0x40002344 (CH3)	PROV_DMA_CTRL0	CHn_CH_START_REG (n=0,1,2,3)	7:0								CH_ENABLE
0x40002048 (CH0), 0x40002148 (CH1), 0x40002248 (CH2), 0x40002348 (CH3)	PROV_DMA_CTRL0	CHn_CH_ACTIVE_REG (n=0,1,2,3)	7:0							CH_WR_ACTIVE	CH_RD_ACTIVE
0x40002050 (CH0), 0x40002150 (CH1), 0x40002250 (CH2), 0x40002350 (CH3)	PROV_DMA_CTRL0	CHn_COUNT_REG (n=0,1,2,3)	7:0	BUFF_COUNT[7:0]							
			15:8					BUFF_COUNT[11:8]			
			23:16					INT_COUNT[3:0]			
0x400020A0 (CH0), 0x400021A0 (CH1), 0x400022A0 (CH2), 0x400023A0 (CH3)	PROV_DMA_CTRL0	CHn_INT_RAWSTAT_REG (n=0,1,2,3)	7:0	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
0x400020A4 (CH0), 0x400021A4 (CH1), 0x400022A4 (CH2), 0x400023A4 (CH3)	PROV_DMA_CTRL0	CHn_INT_CLEAR_REG (n=0,1,2,3)	7:0	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
0x400020A8 (CH0), 0x400021A8 (CH1), 0x400022A8 (CH2), 0x400023A8 (CH3)	PROV_DMA_CTRL0	CHn_INT_ENABLE_REG (n=0,1,2,3)	7:0	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
0x400020AC (CH0), 0x400021AC (CH1), 0x400022AC (CH2), 0x400023AC (CH3)	PROV_DMA_CTRL0	CHn_INT_STATUS_REG (n=0,1,2,3)	7:0	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
0x40002800	PROV_DMA_CTRL0	CORE_INT_STATUS	7:0					CHANNEL_3	CHANNEL_2	CHANNEL_1	CHANNEL_0
0x40002830	PROV_DMA_CTRL0	CORE_JOINT_MODE	7:0								CORE_JOINT_MODE
0x40002848	PROV_DMA_CTRL0	CORE_CH_START	7:0					CH_3	CH_2	CH_1	CH_0

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x40002850	PROV_DMA_CTRL0	PERIPH_RX_CTRL	7:0	RX_REQ[6:0]							
			15:8	RX_REQ[14:7]							
			23:16	RX_REQ[22:15]							
			31:24	RX_REQ[30:23]							
0x40002854	PROV_DMA_CTRL0	PERIPH_TX_CTRL	7:0	TX_REQ[6:0]							
			15:8	TX_REQ[14:7]							
			23:16	TX_REQ[22:15]							
			31:24	TX_REQ[30:23]							
0x400028D0	PROV_DMA_CTRL0	CORE_IDLE	7:0								CORE_IDLE
0x400028E0	PROV_DMA_CTRL0	USER_DEF_STATUS	7:0	IC_DUAL_PORT	IC	DUAL_CORE		INT_NUM[3:0]			
			15:8						PORT1_MUX	PORT0_MUX	CLKGATE
			23:16								
			31:24	PROJ							
0x400028F0	PROV_DMA_CTRL0	CORE_DEF_STATUS0	7:0	FIFO_SIZE[3:0]				CH_NUM[3:0]			
			15:8	RCMD_DEPTH[3:0]				WCMD_DEPTH[3:0]			
			23:16		AHB_BUS_32	ADDR_BITS[5:0]					
			31:24				BUFF_BITS[4:0]				
0x400028F4	PROV_DMA_CTRL0	CORE_DEF_STATUS1	7:0	JOINT	BLOCK	WAIT	OUTS	PRIO	TOKENS	AHB_TIMEOUT	WDT
			15:8				CLKDIV	END	LISTS	PERIPH	INDEPENDENT
0x4000B0C0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_0	7:0				MUX_0[4:0]				
			15:8				MUX_1[4:0]				
			23:16				MUX_2[4:0]				
			31:24				MUX_3[4:0]				
0x4000B0C4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_1	7:0				MUX_4[4:0]				
			15:8				MUX_5[4:0]				
			23:16				MUX_6[4:0]				
			31:24				MUX_7[4:0]				
0x4000B0C8	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_2	7:0				MUX_8[4:0]				
			15:8				MUX_9[4:0]				
			23:16				MUX_10[4:0]				
			31:24				MUX_11[4:0]				
0x4000B0CC	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_3	7:0				MUX_12[4:0]				
			15:8				MUX_13[4:0]				
			23:16				MUX_14[4:0]				
			31:24				MUX_15[4:0]				
0x4000B0D0	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_4	7:0				MUX_16[4:0]				
			15:8				MUX_17[4:0]				
			23:16				MUX_18[4:0]				
			31:24				MUX_19[4:0]				
0x4000B0D4	LPMCU_MISC_REGS0	IRQ_MUX_IO_SEL_5	7:0				MUX_20[4:0]				

23.7 Register Description

23.7.1 DMA Command Register 0

Name: CHn_CMD_REG0 (n=0,1,2,3)
Reset: 0x00000000

Absolute Address: 0x40002000 (CH0), 0x40002100 (CH1), 0x40002200 (CH2), 0x40002300 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is a DMA command register which holds the start address of read buffer. When using the command lists options, this register is overwritten by the next command by the DMA controller. There are four DMA channels, therefore it has four CHn_CMD_REG0 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
	RD_START_ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RD_START_ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RD_START_ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD_START_ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RD_START_ADDR[31:0]
These bits hold the start address of read buffer

23.7.2 DMA Command Register 1

Name: CHn_CMD_REG1 (n=0,1,2,3)
Reset: 0x00000000

Absolute Address: 0x40002004 (CH0), 0x40002104 (CH1), 0x40002204 (CH2), 0x40002304 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is a DMA command register and holds the start address of write buffer. When using command lists options, this register is overwritten by the next command by the DMA controller. There are four DMA channels, therefore it has four CHn_CMD_REG1 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
	WR_START_ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WR_START_ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WR_START_ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WR_START_ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – WR_START_ADDR[31:0]
These bits hold the start address of write buffer

23.7.3 DMA Command Register 2

Name: CHn_CMD_REG2 (n=0,1,2,3)
Reset: 0x0000

Absolute Address: 0x40002008 (CH0), 0x40002108 (CH1), 0x40002208 (CH2), 0x40002308 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is a DMA command register and holds the size of buffer to transfer. When using command lists options, this register is overwritten by the next command by the DMA controller. There are four DMA channels, therefore it has four CHn_CMD_REG2 (n=0,1,2,3) registers.

Bit	15	14	13	12	11	10	9	8
				BUFFER_SIZE[12:8]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BUFFER_SIZE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 12:0 – BUFFER_SIZE[12:0]
These bits hold the size of buffer to transfer

23.7.4 DMA Command Register 3

Name: CHn_CMD_REG3 (n=0,1,2,3)
Reset: 0x00000000

Absolute Address: 0x4000200C (CH0), 0x4000210C (CH1), 0x4000220C (CH2), 0x4000230C (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is DMA command register. When using command lists options, this register is overwritten by the next command by the DMA controller. There are four DMA channels, therefore it has four CHn_CMD_REG3 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
	CMD_NEXT_ADDR[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CMD_NEXT_ADDR[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CMD_NEXT_ADDR[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CMD_NEXT_ADDR[5:0]						CMD_LAST	CMD_SET_INT
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:2 – CMD_NEXT_ADDR[29:0]
These bits hold the address of next command. The next command is read on AHB bus if the CMD_LAST field is not set.

Bit 1 – CMD_LAST
If this bit is set as '1', the channel stops once the entire buffer is transferred
If this bit is set as '0', the next command is loaded from the address specified in the CMD_NEXT_ADD[29:0]

Bit 0 – CMD_SET_INT
Writing '1' to this bit, the channel issues an interrupt once the entire buffer is transferred

23.7.5 DMA Static Configuration Register 0

Name: CHn_STATIC_REG0 (n=0,1,2,3)
Reset: 0x80010000

Absolute Address: 0x40002010 (CH0), 0x40002110 (CH1), 0x40002210 (CH2), 0x40002310 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is DMA configuration register. The parameters set in this register should not be changed when the channel is active. There are four DMA channels, therefore it has four CHn_STATIC_REG0 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
	RD_INCR							
Access	R/W							
Reset	1							

Bit	23	22	21	20	19	18	17	16
			RD_TOKENS[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		RD_BURST_MAX_SIZE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 31 – RD_INCR
If this bit is set the DMA controller increments the next burst address
This bit must be set for all memory copy channels. This bit must be cleared for all peripheral clients that use a static address FIFO.

Bits 21:16 – RD_TOKENS[5:0]
These bits specify the number of AHB read commands to issue before the channel is released

Bits 6:0 – RD_BURST_MAX_SIZE[6:0]
These bits hold the maximum number of bytes of an AHB read burst
Possible values are 1, 2, 4, 4*N (N=4,8,16)
If the channel is reading from a peripheral, the number is set according to the peripheral FIFO

23.7.6 DMA Static Configuration Register 1

Name: CHn_STATIC_REG1 (n=0,1,2,3)
Reset: 0x80010000

Absolute Address: 0x40002014 (CH0), 0x40002114 (CH1), 0x40002214 (CH2), 0x40002314 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is DMA configuration register. The parameters set in this register should not be changed when the channel is active. There are four DMA channels, therefore it has four CHn_STATIC_REG1 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
	WR_INCR							
Access	R/W							
Reset	1							

Bit	23	22	21	20	19	18	17	16
			WR_TOKENS[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		WR_BURST_MAX_SIZE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 31 – WR_INCR
If this bit is set the DMA controller increments the next burst address
This bit must be set for all memory copy channels. This bit must be cleared for all peripheral clients that use a static address FIFO.

Bits 21:16 – WR_TOKENS[5:0]
These bits specify the number of AHB write commands to issue before the channel is released

Bits 6:0 – WR_BURST_MAX_SIZE[6:0]
These bits hold the maximum number of bytes of an AHB write burst
Possible values are 1, 2, 4, 4*N (N=4,8,16)
If the channel is writing to a peripheral, the number is set according to the peripheral FIFO

23.7.7 DMA Static Configuration Register 2

Name: CHn_STATIC_REG2 (n=0,1,2,3)
Reset: 0x00010000

Absolute Address: 0x40002018 (CH0), 0x40002118 (CH1), 0x40002218 (CH2), 0x40002318 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is DMA configuration register. The parameters set in this register should not be changed when the channel is active. There are four DMA channels, therefore it has four CHn_STATIC_REG2 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
			END_SWAP[1:0]					
Access			R/W	R/W				
Reset			0	0				

Bit	23	22	21	20	19	18	17	16
								JOINT
Access								R/W
Reset								1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 16 – JOINT
If this bit is set as '1', the channel works in the Joint mode, and has effect only if CORE_JOINT_MODE is set
If this bit is set as '0', the channel works in the Normal mode

Bits 29:28 – END_SWAP[1:0]
These bits perform the endianness byte swapping of data

END_SWAP[1:0]	Description
0	No swapping (data_out = data_in)
1	Swap bytes within 16 bits (data_out = {data_in[23:16], data_in[31:24], data_in[7:0], data_in[8:15]})
2	Swap bytes within 32 bits (data_out = {data_in[7:0], data_in[15:8], data_in[23:16], data_in[31:24]})
3	Swap bytes within 64 bits (data_out = {data_in[7:0], data_in[15:8], data_in[23:16], data_in[31:24]})

23.7.8 DMA Static Configuration Register 4

Name: CHn_STATIC_REG4 (n=0,1,2,3)
Reset: 0x00000000

Absolute Address: 0x40002020 (CH0), 0x40002120 (CH1), 0x40002220 (CH2), 0x40002320 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register is DMA configuration register. The parameters set in this register should not be changed when the channel is active. There are four DMA channels, therefore it has four CHn_STATIC_REG4 (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
						WR_PERIPH_DELAY[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	1

Bit	23	22	21	20	19	18	17	16
				WR_PERIPH_NUM[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
						RD_PERIPH_DELAY[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
				RD_PERIPH_NUM[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 26:24 – WR_PERIPH_DELAY[2:0]
These bits hold the number of cycles to wait for the peripheral write request signal to update after issuing the write clear signal. This is determined by the peripheral latency.

Bits 20:16 – WR_PERIPH_NUM[4:0]
These bits hold the peripheral number to write to.
If this bit is set as '0', the channel writes to a memory. If this bit is set as non-zero value, the channel writes to a peripheral that does not use peripheral flow control.
The following table shows the mapping peripheral number and its corresponding peripheral in the ATSAMB11.

Table 23-5. Peripheral Number

WR_PERIPH_NUM[4:0]/ RD_PERIPH_NUM[4:0]	Description
0x0	Memory write/read
0x1	UART0 Receive interrupt
0x2	UART0 Transmit interrupt
0x3	UART1 Receive interrupt

.....continued	
WR_PERIPH_NUM[4:0]/ RD_PERIPH_NUM[4:0]	**Description**
0x4	UART1 Transmit interrupt
0x5	SPI0 Receive interrupt
0x6	SPI0 Transmit interrupt
0x7	SPI1 Receive interrupt
0x8	SPI1 Transmit interrupt
0x9	I2C0 Receive interrupt
0xA	I2C0 Transmit interrupt
0xB	I2C1 Receive interrupt
0xC	I2C1 Transmit interrupt
0xD	Reserved
0xE	Reserved
0xF	Dual Timer
0x10	ARM Timer

Bits 10:8 – RD_PERIPH_DELAY[2:0]
These bits hold the number of cycles to wait for the peripheral read request signal to update after issuing the read clear signal. This is determined by the peripheral latency.

Bits 4:0 – RD_PERIPH_NUM[4:0]
These bits hold the peripheral number to read from.
If this bit is set as '0', the channel reads from a memory, and if non-zero value then read from a peripheral that does not use peripheral flow control
The Peripheral Number in WR_PERIPH_NUM bit description shows the mapping peripheral number and its corresponding peripheral in the ATSAMB11.

23.7.9 DMA Channel Restriction Status

Name: CHn_RESTRICT_REG (n=0,1,2,3)
Reset: 0x0007

Absolute Address: 0x4000202C (CH0), 0x4000212C (CH1), 0x4000222C (CH2), 0x4000232C (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides DMA channels' restriction status. There are four DMA channels and therefore it has four CHn_RESTRICT_REG (n=0,1,2,3) registers.

Bit	15	14	13	12	11	10	9	8
								SIMPLE_MEM
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
				ALLOW_JOINT_BURST	ALLOW_FULL_BURST	ALLOW_FULL_FIFO	WR_ALLOW_FULL_FIFO	RD_ALLOW_FULL_FIFO
Access				R	R	R	R	R
Reset				0	0	1	1	1

Bit 8 – SIMPLE_MEM
Reading '1' to this bit indicates configuration is aligned and peripherals are not used for DMA transfer

Bit 4 – ALLOW_JOINT_BURST
Reading '1' to this bit indicates joint bursts are currently active

Bit 3 – ALLOW_FULL_BURST
The logic of this bit is ALLOW_JOINT_BURST & ALLOW_FULL_FIFO. Reading '1' to this bit indicates the pre-selected maximum number of bytes of an AHB read burst is 64 (maximum allowed burst size in ATSAMB11), and it compares with the value of RD_BURST_MAX_SIZE bit of the CHn_STATIC_REG0 (n=0,1,2,3) register. Then the smaller of these two is selected.

Bit 2 – ALLOW_FULL_FIFO
The logic of this bit is RD_ALLOW_FULL_FIFO and WR_ALLOW_FULL_FIFO

Bit 1 – WR_ALLOW_FULL_FIFO
This bit indicates the write start address restriction burst size. This bit is set when the WR_START_ADDRESS[4:0] is 0.

Bit 0 – RD_ALLOW_FULL_FIFO
This bit indicates the read start address restriction on burst size. This bit is set when the RD_START_ADDRESS[4:0] is 0.

23.7.10 DMA FIFO Fullness Status

Name: CHn_FIFO_FULLNESS_REG (n=0,1,2,3)
Reset: 0x00000020

Absolute Address: 0x40002038 (CH0), 0x40002138 (CH1), 0x40002238 (CH2), 0x40002338 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides DMA channels' FIFO fullness status. There are four DMA channels, therefore it has four CHn_FIFO_FULLNESS_REG (n=0,1,2,3) registers.

Bit	31	30	29	28	27	26	25	24
				WR_FULLNESS[12:8]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WR_FULLNESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				RD_GAP[12:8]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD_GAP[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	0	0	0	0

Bits 28:16 – WR_FULLNESS[12:0]
These bits indicate the occupied space in channel's FIFO by write data (value is in bytes). The value is decremented when write command is issued and incremented after data is read

Bits 12:0 – RD_GAP[12:0]
These bits provide the remaining space in channel's FIFO for read data (value is in bytes). The value is decremented when the read command is issued and incremented after data is written.
The Reset value is equal to FIFO size (0x20).

23.7.11 DMA Channel Enable

Name: CHn_CH_ENABLE_REG (n=0,1,2,3)
Reset: 0x01

Absolute Address: 0x40002040 (CH0), 0x40002140 (CH1), 0x40002240 (CH2), 0x40002340 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to enable the specific DMA channel. There are four DMA channels, therefore it has four CHn_CH_ENABLE_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
								CH_ENABLE
Access								R/W
Reset								1

Bit 0 – CH_ENABLE
Writing '1' to this bit enables the DMA channel
Writing '0' to this bit disables the DMA channel
This bit is also used for pause and resume operation

23.7.12 DMA Channel Start

Name: CHn_CH_START_REG (n=0,1,2,3)
Reset: 0x00

Absolute Address: 0x40002044 (CH0), 0x40002144 (CH1), 0x40002244 (CH2), 0x40002344 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to start the specific DMA channel. There are four DMA channels and therefore it has four CHn_CH_START_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
								CH_START
Access								W
Reset								0

Bit 0 – CH_START
Writing '1' to this bit starts the DMA channel

23.7.13 DMA Channel Active Status

Name: CHn_CH_ACTIVE_REG (n=0,1,2,3)
Reset: 0x00

Absolute Address: 0x40002048 (CH0), 0x40002148 (CH1), 0x40002248 (CH2), 0x40002348 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides the active status of the DMA channels. There are four DMA channels, therefore it has four CHn_CH_ACTIVE_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
							CH_WR_ACTIVE	CH_RD_ACTIVE
Access							R	R
Reset							0	0

Bit 1 – CH_WR_ACTIVE
This bit value is set if the channel is enabled and all write data is transferred

Bit 0 – CH_RD_ACTIVE
This bit value is set if the channel is enabled and all read data is received

23.7.14 DMA Buffer Counter Status

Name: CHn_COUNT_REG (n=0,1,2,3)
Reset: 0x000000

Absolute Address: 0x40002050 (CH0), 0x40002150 (CH1), 0x40002250 (CH2), 0x40002350 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA channel buffer counter status. There are four DMA channels and therefore it has four CHn_COUNT_REG (n=0,1,2,3) registers.

Bit	23	22	21	20	19	18	17	16
					INT_COUNT[3:0]			
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
					BUFF_COUNT[11:8]			
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BUFF_COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 19:16 – INT_COUNT[3:0]
These bits provide the number of unserviced end interrupts
The bit value is incremented each time when an end interrupt (CH_END) is issued and is decremented when the CH_END bit in the CHn_INT_CLEAR_REG (n=0,1,2,3) register is written

Bits 11:0 – BUFF_COUNT[11:0]
These bits provide the number of buffers transferred by channel since started
When using a command list this status indicates how many DMA commands are completed

23.7.15 DMA Raw Interrupt Status

Name: CHn_INT_RAWSTAT_REG (n=0,1,2,3)
Reset: 0x00

Absolute Address: 0x400020A0 (CH0), 0x400021A0 (CH1), 0x400022A0 (CH2), 0x400023A0 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA channel raw interrupt status. There are four DMA channels and therefore it has four CHn_INT_RAWSTAT_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – WDT
Reading '1' to this bit indicates that the channel is active but did not start a burst for 2048 cycles
Writing '1' to this bit issues an interrupt

Bit 6 – TIMEOUT_WR
Reading '1' to this bit indicates that a write issued by this channel caused a time-out on the AHB bus
Time-out value is fixed to 1024 cycles
Writing '1' to this bit issues an interrupt

Bit 5 – TIMEOUT_RD
Reading '1' to this bit indicates that a read issued by this channel caused a time-out on the AHB bus
Time-out value is fixed to 1024 cycles
Writing '1' to this bit issues an interrupt

Bit 4 – FIFO_UNDERFLOW
Reading '1' to this bit indicates that the data FIFO is under-flown
Writing '1' to this bit issues an interrupt

Bit 3 – FIFO_OVERFLOW
Reading '1' to this bit indicates that the data FIFO is over-flown
Writing '1' to this bit issues an interrupt

Bit 2 – WR_SLVERR
Reading '1' to this bit indicates that a write issued by this channel caused an AHB write slave error
Writing '1' to this bit issues an interrupt

Bit 1 – RD_SLVERR
Reading '1' to this bit indicates that a read issued by this channel caused an AHB read slave error
Writing '1' to this bit issues an interrupt

Bit 0 – CH_END
Reading '1' to this bit indicates an unserviced channel end interrupt. The total number of unserviced end interrupts can be read in the INT_COUNT[3:0].
Writing '1' to this bit issues an interrupt

23.7.16 DMA Interrupt Status Clear

Name: CHn_INT_CLEAR_REG (n=0,1,2,3)
Reset: 0x00

Absolute Address: 0x400020A4 (CH0), 0x400021A4 (CH1), 0x400022A4 (CH2), 0x400023A4 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to clear the interrupt status. There are four DMA channels, therefore it has four CHn_INT_CLEAR_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – WDT
Writing '1' to this bit clears WDT status bit

Bit 6 – TIMEOUT_WR
Writing '1' to this bit clears TIMEOUT_WR status bit

Bit 5 – TIMEOUT_RD
Writing '1' to this bit clears TIMEOUT_RD status bit

Bit 4 – FIFO_UNDERFLOW
Writing '1' to this bit clears FIFO_UNDERFLOW status bit

Bit 3 – FIFO_OVERFLOW
Writing '1' to this bit clears FIFO_OVERFLOW status bit

Bit 2 – WR_SLVERR
Writing '1' to this bit clears WR_SLVERR status bit

Bit 1 – RD_SLVERR
Writing '1' to this bit clears RD_SLVERR status bit

Bit 0 – CH_END
Writing '1' to this bit clears CH_END status bit

23.7.17 DMA Interrupt Enable

Name: CHn_INT_ENABLE_REG (n=0,1,2,3)
Reset: 0xFF

Absolute Address: 0x400020A8 (CH0), 0x400021A8 (CH1), 0x400022A8 (CH2), 0x400023A8 (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to enable the DMA interrupt. There are four DMA channels, therefore it has four CHn_INT_ENABLE_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 7 – WDT
Writing '1' to this bit enables WDT interrupt

Bit 6 – TIMEOUT_WR
Writing '1' to this bit enables TIMEOUT_WR interrupt

Bit 5 – TIMEOUT_RD
Writing '1' to this bit enables TIMEOUT_RD interrupt

Bit 4 – FIFO_UNDERFLOW
Writing '1' to this bit enables FIFO_UNDERFLOW interrupt

Bit 3 – FIFO_OVERFLOW
Writing '1' to this bit enables FIFO_OVERFLOW interrupt

Bit 2 – WR_SLVERR
Writing '1' to this bit enables WR_SLVERR interrupt

Bit 1 – RD_SLVERR
Writing '1' to this bit enables RD_SLVERR interrupt

Bit 0 – CH_END
Writing '1' to this bit enables CH_END interrupt

23.7.18 DMA Interrupt Status

Name: CHn_INT_STATUS_REG (n=0,1,2,3)
Reset: 0x00

Absolute Address: 0x400020AC (CH0), 0x400021AC (CH1), 0x400022AC (CH2), 0x400023AC (CH3)

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA interrupt status. The bits in this register are set only when the corresponding interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register. There are four DMA channels and therefore it has four CHn_INT_STATUS_REG (n=0,1,2,3) registers.

Bit	7	6	5	4	3	2	1	0
	WDT	TIMEOUT_WR	TIMEOUT_RD	FIFO_UNDERFLOW	FIFO_OVERFLOW	WR_SLVERR	RD_SLVERR	CH_END
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – WDT
Reading '1' to this bit indicates WDT interrupt is triggered. This bit is set only when WDT interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 6 – TIMEOUT_WR
Reading '1' to this bit indicates TIMEOUT_WR interrupt is triggered. This bit is set only when TIMEOUT_WR interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 5 – TIMEOUT_RD
Reading '1' to this bit indicates TIMEOUT_RD interrupt is triggered. This bit is set only when TIMEOUT_RD interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 4 – FIFO_UNDERFLOW
Reading '1' to this bit indicates FIFO_UNDERFLOW interrupt is triggered. This bit is set only when FIFO_UNDERFLOW interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 3 – FIFO_OVERFLOW
Reading '1' to this bit indicates FIFO_OVERFLOW interrupt is triggered. This bit is set only when FIFO_OVERFLOW interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 2 – WR_SLVERR
Reading '1' to this bit indicates WR_SLVERR interrupt is triggered. This bit is set only when WR_SLVERR interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 1 – RD_SLVERR
Reading '1' to this bit indicates RD_SLVERR interrupt is triggered. This bit is set only when RD_SLVERR interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

Bit 0 – CH_END
Reading '1' to this bit indicates CH_END interrupt is triggered. This bit is set only when CH_END interrupt is enabled in the CHn_INT_ENABLE_REG (n=0,1,2,3) register.

23.7.19 DMA Channel Interrupt Status

Name: CORE_INT_STATUS
Reset: 0x00

Absolute Address: 0x40002800

This register is a part of PROV_DMA_CTRL0 registers. This register provides which DMA channel caused the interrupt.

Bit	7	6	5	4	3	2	1	0
					CHANNEL_3	CHANNEL_2	CHANNEL_1	CHANNEL_0
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – CHANNEL_3
Reading '1' to this bit indicates that the interrupt is caused by channel 3

Bit 2 – CHANNEL_2
Reading '1' to this bit indicates that the interrupt is caused by channel 2

Bit 1 – CHANNEL_1
Reading '1' to this bit indicates that the interrupt is caused by channel 1

Bit 0 – CHANNEL_0
Reading '1' to this bit indicates that the interrupt is caused by channel 0

23.7.20 DMA Joint Mode Enable

Name: CORE_JOINT_MODE
Reset: 0x00

Absolute Address: 0x40002830

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to enable the Joint mode.

Bit	7	6	5	4	3	2	1	0
								CORE_JOINT_MODE
Access								R/W
Reset								0

Bit 0 – CORE_JOINT_MODE
Writing '1' to this bit enable the DMA to work in the Joint mode otherwise it works in the Independent mode

23.7.21 DMA Multiple Channel Start

Name: CORE_CH_START
Reset: 0x00

Absolute Address: 0x40002848

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to start multiple channels simultaneously.

Bit	7	6	5	4	3	2	1	0
					CH_3	CH_2	CH_1	CH_0
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – CH_3
Writing '1' to this bit starts channel 3

Bit 2 – CH_2
Writing '1' to this bit starts channel 2

Bit 1 – CH_1
Writing '1' to this bit starts channel 1

Bit 0 – CH_0
Writing '1' to this bit starts channel 0

23.7.22 Direct Control of Peripheral Rx Request

Name: PERIPH_RX_CTRL
Reset: 0x00000000

Absolute Address: 0x40002850

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to directly control peripheral receive requests.

Bit	31	30	29	28	27	26	25	24
	RX_REQ[30:23]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RX_REQ[22:15]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RX_REQ[14:7]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RX_REQ[6:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	

Bits 31:1 – RX_REQ[30:0]
Allows direct control of the peripheral RX request bus. This is useful when the application reads results without waiting for them to arrive.

23.7.23 Direct Control of Peripheral Tx Request

Name: PERIPH_TX_CTRL
Reset: 0x00000000

Absolute Address: 0x40002854

This register is a part of PROV_DMA_CTRL0 registers. This register allows the user to directly control peripheral transmit requests.

Bit	31	30	29	28	27	26	25	24
	TX_REQ[30:23]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TX_REQ[22:15]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TX_REQ[14:7]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TX_REQ[6:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	

Bits 31:1 – TX_REQ[30:0]
Allows direct control of the peripheral TX request bus. This is useful when the application writes configuration registers to a slow device.

23.7.24 DMA Core Idle Status

Name: CORE_IDLE
Reset: 0x01

Absolute Address: 0x400028D0

This register is a part of PROV_DMA_CTRL0 registers. This register indicates the DMA core Idle status.

Bit	7	6	5	4	3	2	1	0
								CORE_IDLE
Access								R
Reset								1

Bit 0 – CORE_IDLE
Reading '1' to this bit indicates that all channels have stopped working and all bus transactions are complete

23.7.25 DMA Core Internal User Configuration

Name: USER_DEF_STATUS
Reset: 0x80000001

Absolute Address: 0x400028E0

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA core user defined internal configuration.

Bit	31	30	29	28	27	26	25	24
	PROJ							
Access	R							
Reset	1							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						PORT1_MUX	PORT0_MUX	CLKGATE
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
	IC_DUAL_PORT	IC	DUAL_CORE		INT_NUM[3:0]			
Access	R	R	R		R	R	R	R
Reset	0	0	0		0	0	0	1

Bit 31 – PROJ
Reading this bit always returns '1'

Bit 10 – PORT1_MUX
The AHB port 1 uses an AHB mux. This is not used in ATSAMB11

Bit 9 – PORT0_MUX
The AHB port 0 uses an AHB mux. This is not used in ATSAMB11

Bit 8 – CLKGATE
If this bit is set, the design will contain functional clock gates. This is not used in ATSAMB11.

Bit 7 – IC_DUAL_PORT
If this bit is set, the AHB matrix will have two output ports, otherwise it has a single port. ATSAMB11 has single port only.

Bit 6 – IC
If this bit is set, an AHB matrix is used. This is not used in ATSAMB11.

Bit 5 – DUAL_CORE
If this bit is set, the design has two cores else the design has a single core. ATSAMB11 has single DMA core.

Bits 3:0 – INT_NUM[3:0]
This bit indicates the number of bits in interrupt bus INT. When reading, this bit always returns '1'.

23.7.26 DMA Core Internal Configuration 0

Name: CORE_DEF_STATUS0
Reset: 0x0A601151

Absolute Address: 0x400028F0

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA core internal configuration.

Bit	31	30	29	28	27	26	25	24
				BUFF_BITS[4:0]				
Access				R	R	R	R	R
Reset				0	1	0	1	0

Bit	23	22	21	20	19	18	17	16
		AHB_BUS_32	ADDR_BITS[5:0]					
Access		R	R	R	R	R	R	R
Reset		1	1	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RCMD_DEPTH[3:0]				WCMD_DEPTH[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	FIFO_SIZE[3:0]				CH_NUM[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	1	0	1	0	0	0	1

Bit 22 – AHB_BUS_32
If this bit is set, the core AHB bus is set as 32-bit, otherwise 64-bit. This bit is set in ATSAMB11 as it uses 32-bit AHB bus.

Bits 28:24 – BUFF_BITS[4:0]
These bits indicate the number of bits in core BUFFER_SIZE. When read, 0x0A is returned always.

Bits 21:16 – ADDR_BITS[5:0]
These bits indicate the number of bits in address buses. ATSAMB11 uses 32-bit address bits and returns 0x20 when read.

Bits 15:12 – RCMD_DEPTH[3:0]
These bits indicate the maximum number of pending read commands. When read, 0x01 is returned always.

Bits 11:8 – WCMD_DEPTH[3:0]
These bits indicate the maximum number of pending write commands. When read, 0x01 is returned always.

Bits 7:4 – FIFO_SIZE[3:0]
These bits indicate the FIFO size per channel. When read, 0x05 is returned always.

Bits 3:0 – CH_NUM[3:0]
When reading, this bit always returns '1'.

23.7.27 DMA Core Internal Configuration 1

Name: CORE_DEF_STATUS1
Reset: 0x0F87

Absolute Address: 0x400028F4

This register is a part of PROV_DMA_CTRL0 registers. This register provides the DMA core internal configuration.

Bit	15	14	13	12	11	10	9	8
				CLKDIV	END	LISTS	PERIPH	INDEPENDENT
Access				R	R	R	R	R
Reset				0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	JOINT	BLOCK	WAIT	OUTS	PRIO	TOKENS	AHB_TIMEOUT	WDT
Access	R	R	R	R	R	R	R	R
Reset	1	0	0	0	0	1	1	1

Bit 12 – CLKDIV
Clock division is not supported. This bit is read as '0'.

Bit 11 – END
Endianness swapping is supported. This bit is read as '1'.

Bit 10 – LISTS
Command list is supported. This bit is read as '1'.

Bit 9 – PERIPH
Peripheral transfer is supported. This bit is read as '1'.

Bit 8 – INDEPENDENT
Independent mode is supported. This bit is read as '1'.

Bit 7 – JOINT
Joint mode is supported. This bit is read as '1'.

Bit 6 – BLOCK
Block transfer is not supported. This bit is read as '0'.

Bit 5 – WAIT
Scheduled channels are not supported. This bit is read as '0'.

Bit 4 – OUTS
Outstanding mode is not supported. This bit is read as '0'.

Bit 3 – PRIO
Priority mode is not supported. This bit is read as '0'.

Bit 2 – TOKENS
Tokens are supported. This bit is read as '1'.

Bit 1 – AHB_TIMEOUT
Time-outs on AHB read and write buses are supported. This bit is read as '1'.

Bit 0 – WDT
Watchdog timer is supported. This bit is read as '1'.

24. Watchdog Timer

The Watchdog Timer (WDT) is a system function for monitoring the program operation. It helps to recover from error such as, runaway or deadlocked code. The WDT is configured to a predefined time-out period, and constantly runs when enabled. If the WDT is not cleared within the time-out period, it issues a system Reset. An early-warning interrupt is available to indicate an upcoming Watchdog Time-out condition.

24.1 Features

The following are the WDT features:

- Two independent watchdog blocks
- Issues a system Reset if the watchdog timer is not cleared before its time-out period
- Generates early warning interrupt
- Selectable time-out periods

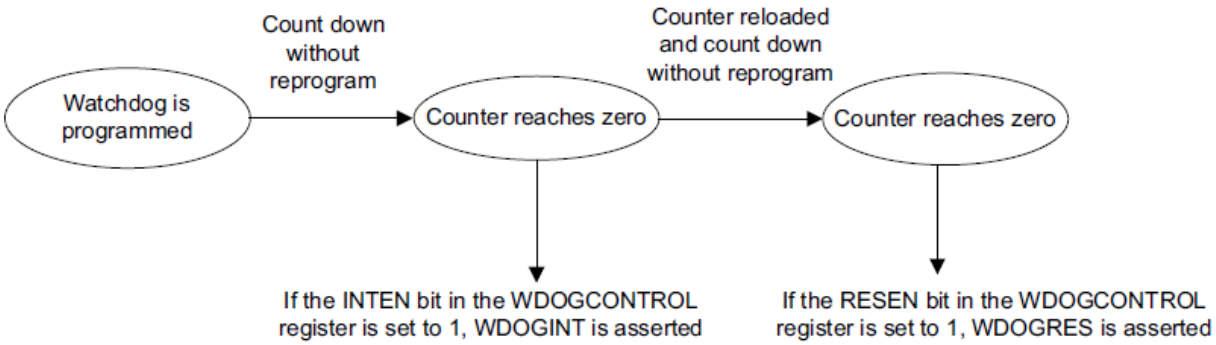
24.2 Flow Diagram

The watchdog is based on a 32-bit down-counter that is initialized from the Reload register, WDOGLOAD. The watchdog module generates a regular interrupt, WDOGINT, depending on a programmed value. The counter decrements by one on each positive clock edge of WDOGCLK.

The watchdog monitors the interrupt and asserts a Reset request WDOGRES signal if the interrupt is not cleared before the counter reaches 0 and the counter is stopped. If the interrupt is cleared before it reaches zero, then on the next enabled WDOGCLK clock edge, the counter is reloaded from the WDOGLOAD register and the countdown sequence continues.

The watchdog module applies a Reset to a system in the event of a software failure, providing a way to recover from software crashes.

Figure 24-1. WDT Flow Diagram



24.3 Clock Configuration

The clock for WDT timer is enabled by setting WDT0_CLK_EN, and WDT1_CLK_EN bits in the LPMCU_CLOCK_ENABLES_0 register. The MCU free running clock (26 MHz) is the input clock for watchdog module. For more details on configuration, see Peripheral Clock Configuration.

24.4 Functional Description

24.4.1 Initialization

The initialization procedures are identical for both WDT0 and WDT1 modules.

- To generate an interrupt on WDT time-out before it resets the device, set the appropriate bit in the interrupt mask registers. Both the WDT modules are mapped to Non-Maskable Interrupt (NMI) (see ATSAMB11 Interrupt Vector Table).
  - Register the WDT ISR function in interrupt vector table.
  - Enable the NMI interrupt in NVIC interrupt controller registers.

Note:

- Only one ISR index is allocated in interrupt vector table for both WDT0 and WDT1 modules.
- Though the ISR index 28, and 29 in ATSAMB11 Interrupt Vector Table are mapped to WDT0 and WDT1, only the ISR index 2 (NMI) is used for both the WDT modules. Since NMI is non-maskable interrupt, the ISR index 28, and 29 are overridden. Therefore, ISR index 28, and 29 can be used for other peripherals through muxable interrupt configuration.
- WDT0 is a reserved resource, being used by the BLE stack. The application must refrain from using the WDT0.

24.4.2 Operation

- The WDOGLOCK register disables write accesses to all the WDT registers. This prevents software from disabling the watchdog functionality. Writing a value of 0x1ACCE551 enables write access to registers.
  Before writing to the WDT registers, unlock the register by writing 0x1ACCE551 to the WDOGLOCK register.
- The WDT timer is decremented for each P_CLK (26 MHz). Calculate the counter value for desired interval (us) using the following formula:
  WDOGLOAD= Interval (us) * P_CLK (MHz)

  The interval is loaded by writing to WDOGLOAD[31:0] bits of the WDOGLOAD register.
- The loaded counter value is reloaded on current counter (WDOGVALUE) register.
- Write ‘1’ to INTEN bit of the WDOGCONTROL register to enable watchdog and early warning interrupt. Enabling the NVIC interrupt (as mentioned in Initialization) triggers an interrupt when the WDT counter (WDOGVALUE) reaches zero. The status of interrupt is indicated by setting WDOGMIS bit in the WDOGMIS register. If the WDOGINTCLR register is written ‘1’ in the ISR routine, then WDOGMIS bit is cleared. If RESEN bit is ‘0’, then watchdog timer is reloaded with WDOGLOAD value and continues to decrement.
- Writing RESEN bit to ‘1’ along with writing ‘1’ to INTEN bit, triggers the WDT reset and resets the ARM MCU power domain.
  Note: This is not recommended since this resets only ARM MCU power domain but not the complete system.
  Recommended solution is to enable only INTEN bit and to do the global reset on the interrupt (see Global (Chip) Reset). This resets the complete system.
- Once all the configuration is done, lock the registers by writing any value other than 0x1ACCE551 to the WDOGLOCK register.

24.4.3 Reload Count

If WDT current counter is not reloaded with initial value then when it reaches zero, the WDT triggers an early warning interrupt. Therefore, it is required to reload the WDOGLOAD register with the same or new value before the interrupt is triggered. To access the register, WDOGLOCK should be written with 0x1ACCE551 to unlock.

Note: Application task is one of the tasks in RTOS based BLE stack running on ATSAMB11. Enabling WDT from the application also requires reloading before it gets expired. Therefore, the recommended solution is to start another periodic timer (lesser than WDT timeout) and reload the WDT when the timer is triggered from the application task.

24.5 Power Management

If the system goes to the Ultra-Low Power mode, the watchdog peripheral shuts down. The watchdog timer configuration registers lose their content, and are not restored when powered- up again. User must reconfigure the watchdog peripheral at power-up to ensure it is in a well-defined state before use. For details on reconfiguration, refer to ATSAMB11 BluSDK Smart Interrupts and ULP Architecture and Usage User Guide. This document also explains on how sleep and wake-up are controlled.

24.6 Register Summary

This is the summary of all the registers used in this chapter.

Absolute Address	Register Group	Name	Bit Pos.								
0x40008000(WDT0), 0x40009000(WDT1)	WDT	WDOGLOAD	7:0	WDOGLOAD[7:0]							
			15:8	WDOGLOAD[15:8]							
			23:16	WDOGLOAD[23:16]							
			31:24	WDOGLOAD[31:24]							
0x40008004(WDT0), 0x40009004(WDT1)	WDT	WDOGVALUE	7:0	WDOGVALUE[7:0]							
			15:8	WDOGVALUE[15:8]							
			23:16	WDOGVALUE[23:16]							
			31:24	WDOGVALUE[31:24]							
0x40008008(WDT0), 0x40009008(WDT1)	WDT	WDOGCONTROL	7:0							RESEN	INTEN
0x4000800C(WDT0), 0x4000900C(WDT1)	WDT	WDOGINTCLR	7:0								WDOGINTCLR
0x40008010(WDT0), 0x40009010(WDT1)	WDT	WDOGRIS	7:0								WDOGRIS
0x40008014(WDT0), 0x40009014(WDT1)	WDT	WDOGMIS	7:0								WDOGMIS
0x40008C00(WDT0), 0x40009C00(WDT1)	WDT	WDOGLOCK	7:0	ENABLE_REGISTER_WRITES[7:0]							
			15:8	ENABLE_REGISTER_WRITES[14:7]							
			23:16	ENABLE_REGISTER_WRITES[22:15]							
			31:24	ENABLE_REGISTER_WRITES[30:23]							
0x40008F00(WDT0), 0x40009F00(WDT1)	WDT	WDOGITCR	7:0								ENABLE

.....continued

Absolute Address	Register Group	Name	Bit Pos.								
0x40008F04(WDT0), 0x40009F04(WDT1)	WDT	WDOGITOP	7:0							WDOGINT_VALUE	WDOGRES_VALUE
0x40008FD0(WDT0), 0x40009FD0(WDT1)	WDT	WDOGPERIPHID4	7:0	BLOCK_COUNT[3:0]							JEP106_C_CODE
0x40008FD4(WDT0), 0x40009FD4(WDT1)	WDT	WDOGPERIPHID5	7:0								
0x40008FD8(WDT0), 0x40009FD8(WDT1)	WDT	WDOGPERIPHID6	7:0								
0x40008FDC(WDT0), 0x40009FDC(WDT1)	WDT	WDOGPERIPHID7	7:0								
0x40008FE0(WDT0), 0x40009FE0(WDT1)	WDT	WDOGPERIPHID0	7:0	PART_NUMBER[7:0]							
0x40008FE4(WDT0), 0x40009FE4(WDT1)	WDT	WDOGPERIPHID1	7:0	JEP106_ID_3_0[3:0]				PART_NUMBER[3:0]			
0x40008FE8(WDT0), 0x40009FE8(WDT1)	WDT	WDOGPERIPHID2	7:0	REVISION[3:0]				JEDEC_USED	JEP106_ID_6_4[2:0]		
0x40008FEC(WDT0), 0x40009FEC(WDT1)	WDT	WDOGPERIPHID3	7:0	ECO_REV_NUMER[3:0]				CUSTOMER_MOD_NUMBER[3:0]			
0x40008FF0(WDT0), 0x40009FF0(WDT1)	WDT	WDOGPCELLID0	7:0	WDOGPCELLID0[7:0]							
0x40008FF4(WDT0), 0x40009FF4(WDT1)	WDT	WDOGPCELLID1	7:0	WDOGPCELLID1[7:0]							
0x40008FF8(WDT0), 0x40009FF8(WDT1)	WDT	WDOGPCELLID2	7:0	WDOGPCELLID2[7:0]							
0x40008FFC(WDT0), 0x40009FFC(WDT1)	WDT	WDOGPCELLID3	7:0	WDOGPCELLID3[7:0]							

24.7 Register Description

24.7.1 WDT Load

Name: WDOGLOAD
Reset: 0xFFFFFFFF

Absolute Address: 0x40008000(WDT0), 0x40009000(WDT1)

This register is a part of WDT registers. This register contains the value from which the WDT counter decrements.

Bit	31	30	29	28	27	26	25	24
	WDOGLOAD[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	WDOGLOAD[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	WDOGLOAD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	WDOGLOAD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 31:0 – WDOGLOAD[31:0]
These bits hold the counter value equivalent to timer interval that is loaded into the WDOGVALUE register. For more details on formula, see Operation.

24.7.2 WDT value

Name: WDOGVALUE
Reset: 0xFFFFFFFF

Absolute Address: 0x40008004(WDT0), 0x40009004(WDT1)

This register is a part of WDT registers. This register contains the value from which the WDT counter decrements.

Bit	31	30	29	28	27	26	25	24
	WDOGVALUE[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	WDOGVALUE[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	WDOGVALUE[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	WDOGVALUE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1

Bits 31:0 – WDOGVALUE[31:0]
These bits contain the current counter value

24.7.3 WDT Control

Name: WDOGCONTROL
Reset: 0x00

Absolute Address: 0x40008008(WDT0), 0x40009008(WDT1)

This register is a part of WDT registers. This register allows the user to configure the watchdog timer.

Bit	7	6	5	4	3	2	1	0
							RESEN	INTEN
Access							R/W	R/W
Reset							0	0

Bit 1 – RESEN
Writing '0' to this bit disables watchdog reset
Writing '1' to this bit enables watchdog reset output

Bit 0 – INTEN
Writing '0' to this bit disables watchdog and watchdog early warning interrupt
Writing '1' to this bit enables watchdog and watchdog early warning interrupt

24.7.4 WDT Interrupt Clear

Name: WDOGINTCLR
Reset: 0x00

Absolute Address: 0x4000800C(WDT0), 0x4000900C(WDT1)

This register is a part of WDT registers. This register allows the user to clear the WDT interrupt request and reloads the counter value in the WDOGLOAD register.

Bit	7	6	5	4	3	2	1	0
								WDOGINTCLR
Access								W
Reset								0

Bit 0 – WDOGINTCLR
Writing '0' or '1' to this bit clears the WDT interrupt request and reloads the counter from the value of the WDOGLOAD register

24.7.5 WDT Raw Interrupt Status

Name: WDOGRIS
Reset: 0x00

Absolute Address: 0x40008010(WDT0), 0x40009010(WDT1)

This register is a part of WDT registers. This register provides the raw interrupt status of WDT.

Bit	7	6	5	4	3	2	1	0
								WDOGRIS
Access								R
Reset								0

Bit 0 – WDOGRIS
This bit indicates the raw interrupt status of the WDT timer. When the counter value decrements and reaches zero, this bit is set to indicate that the timer is triggered.

24.7.6 WDT Masked Interrupt Status

Name: WDOGMIS
Reset: 0x00

Absolute Address: 0x40008014(WDT0), 0x40009014(WDT1)

This register is a part of WDT registers. This register provides the masked interrupt status of WDT.

Bit	7	6	5	4	3	2	1	0
								WDOGMIS
Access								R
Reset								0

Bit 0 – WDOGMIS
This bit indicates the masked interrupt status of the watchdog timer. When the counter value decrements and reaches zero, this bit is set to indicate that the timer is triggered. This bit is set only when the INTEN bit of the WDOGCONTROL register is enabled.

24.7.7 WDT Write Access

Name: WDOGLOCK
Reset: 0x00000000

Absolute Address: 0x40008C00(WDT0), 0x40009C00(WDT1)

This register is a part of WDT registers. This register locks/unlocks all the watchdog related configuration registers.

Bit	31	30	29	28	27	26	25	24
	ENABLE_REGISTER_WRITES[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ENABLE_REGISTER_WRITES[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ENABLE_REGISTER_WRITES[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ENABLE_REGISTER_WRITES[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ENABLE_REGISTER_WRITES[31:0]
Enable write access to all other registers by writing 0x1ACCE551
Write access is disabled when writing value other than 0x1ACCE551
While reading, only bit 0 is visible, bits [31:1] are all '0'.
Reading '0' in bit 0 indicates the write access to all other registers are enabled. This is the default status.
Reading '1' in bit 0 indicates the write access to all other registers is disabled

24.7.8 WDT Integrated Test Control

Name: WDOGITCR
Reset: 0x00

Absolute Address: 0x40008F00(WDT0), 0x40009F00(WDT1)

This register is a part of WDT registers. This register allows the Test mode when integrating the WDT peripheral with ARM processor through APB bus. This is not usable for user application. When the Integration Test mode is enabled through this register, the WDOGITOP register directly controls the output of the WDT masked interrupt.

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R/W
Reset								0

Bit 0 – ENABLE Integrated Test Control Enable
Writing '0' to this bit disables the Integrated Test mode
Writing '1' to this bit enables the Integrated Test mode

24.7.9 WDT Integrated Test Output Set

Name: WDOGITOP
Reset: 0x00

Absolute Address: 0x40008F04(WDT0), 0x40009F04(WDT1)

This register is a part of WDT registers. This register allows the Test mode, which is used when integrating the WDT peripheral with ARM processor through APB bus. This is not usable for user application. When Integration Test mode is enabled through the WDOGITCR register, the values in this register directly drive the WDT outputs.

Bit	7	6	5	4	3	2	1	0
							WDOGINT_VALUE	WDOGRES_VALUE
Access							R/W	R/W
Reset							0	0

Bit 1 – WDOGINT_VALUE
Writing '0' to this bit outputs '0' on WDOGINT when in Integration Test Mode
Writing '1' to this bit outputs '1' on WDOGINT when in Integration Test Mode

Bit 0 – WDOGRES_VALUE
Writing '0' to this bit outputs '0' on WDOGRES when in Integration Test Mode
Writing '1' to this bit outputs '1' on WDOGRES when in Integration Test Mode

24.7.10 WDT Peripheral ID 4

Name: WDOGPERIPHID4
Reset: 0x04

Absolute Address: 0x40008FD0(WDT0), 0x40009FD0(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	BLOCK_COUNT[3:0]				JEP106_C_CODE[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	0

Bits 7:4 – BLOCK_COUNT[3:0]
These bits always return zero when read

Bits 3:0 – JEP106_C_CODE[0:0]
These bits always return 0x04 when read

24.7.11 WDT Peripheral ID 5

Name: WDOGPERIPHID5
Reset: 0x00

Absolute Address: 0x40008FD4(WDT0), 0x40009FD4(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.
Note: This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

24.7.12 WDT Peripheral ID 6

Name: WDOGPERIPHID6
Reset: 0x00

Absolute Address: 0x40008FD8(WDT0), 0x40009FD8(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.
Note: This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

24.7.13 WDT Peripheral ID 7

Name: WDOGPERIPHID7
Reset: 0x00

Absolute Address: 0x40008FDC(WDT0), 0x40009FDC(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.
Note: This register is not used.

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

24.7.14 WDT Peripheral ID 0

Name: WDOGPERIPHID0
Reset: 0x24

Absolute Address: 0x40008FE0(WDT0), 0x40009FE0(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	PART_NUMBER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	0	1	0	0

Bits 7:0 – PART_NUMBER[7:0] Part number low
These bits hold lower bits of the part number. These bits always return 0x24 when read.

24.7.15 WDT Peripheral ID 1

Name: WDOGPERIPHID1
Reset: 0xB8

Absolute Address: 0x40008FE4(WDT0), 0x40009FE4(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	JEP106_ID_3_0[3:0]				PART_NUMBER[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	1	0	0	0

Bits 7:4 – JEP106_ID_3_0[3:0] Lower part of the JEP-106 Identity Code
These bits hold the lower part of JEP-106 identity code. These bits always return 0x0B when read (JEP-106 identity code is 0x3B).

Bits 3:0 – PART_NUMBER[3:0] Part number high
These bits hold higher bits of the part number. These bits always return 0x08 when read.

24.7.16 WDT Peripheral ID 2

Name: WDOGPERIPHID2
Reset: 0x1B

Absolute Address: 0x40008FE8(WDT0), 0x40009FE8(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	REVISION[3:0]				JEDEC_USED	JEP106_ID_6_4[2:0]		
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	1	0	1	1

Bits 7:4 – REVISION[3:0]
These bits indicate the revision of the peripheral. The bit value starts at 0x0 and increments by one at both major and minor revisions. These bits always return 0x01 when read.

Bit 3 – JEDEC_USED
This bit always returns one when read, indicating that JEP-106 code is used

Bits 2:0 – JEP106_ID_6_4[2:0] High part of the JEP-106 Identity Code
These bits hold the higher part of JEP-106 identity code. These bits always return 0x03 when read (JEP-106 identity code is 0x3B).

24.7.17 WDT Peripheral ID 3

Name: WDOGPERIPHID3
Reset: 0x00

Absolute Address: 0x40008FEC(WDT0), 0x40009FEC(WDT1)

This register is a part of WDT registers. This is the WDT peripheral identification read-only register.

Bit	7	6	5	4	3	2	1	0
	ECO_REV_NUMER[3:0]				CUSTOMER_MOD_NUMBER[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:4 – ECO_REV_NUMER[3:0] Revision Number
These bits always return 0x0 when read

Bits 3:0 – CUSTOMER_MOD_NUMBER[3:0] Part number high
These bits always return 0x0 when read

24.7.18 WDT Component ID 0

Name: WDOGPCELLID0
Reset: 0x0D

Absolute Address: 0x40008FF0(WDT0), 0x40009FF0(WDT1)

This register is a part of WDT registers. This is the WDT component identification read-only register

Bit	7	6	5	4	3	2	1	0
	WDOGPCELLID0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	1	0	1

Bits 7:0 – WDOGPCELLID0[7:0]
These bits always return 0x0D when read

24.7.19 WDT Component ID 1

Name: WDOGPCELLID1
Reset: 0xF0

Absolute Address: 0x40008FF4(WDT0), 0x40009FF4(WDT1)

This register is a part of WDT registers. This is the WDT component identification read-only register

Bit	7	6	5	4	3	2	1	0
	WDOGPCELLID1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	1	0	0	0	0

Bits 7:0 – WDOGPCELLID1[7:0]
These bits always return 0xF0 when read

24.7.20 WDT Component ID 2

Name: WDOGPCELLID2
Reset: 0x05

Absolute Address: 0x40008FF8(WDT0), 0x40009FF8(WDT1)

This register is a part of WDT registers. This is the WDT component identification read-only register

Bit	7	6	5	4	3	2	1	0
	WDOGPCELLID2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	1

Bits 7:0 – WDOGPCELLID2[7:0]
These bits always return 0x05 when read

24.7.21 WDT Component ID 3

Name: WDOGPCELLID3
Reset: 0xB1

Absolute Address: 0x40008FFC(WDT0), 0x40009FFC(WDT1)

This register is a part of WDT registers. This is the WDT component identification read-only register

Bit	7	6	5	4	3	2	1	0
	WDOGPCELLID3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	1	0	0	0	1

Bits 7:0 – WDOGPCELLID3[7:0]
These bits always return 0xB1 when read

25. Electrical Characteristics

There are voltage ranges where different VDDIO levels apply. This separation is for the IO drivers whose drive strength is directly proportional to the IO supply voltage. In the ATSAMB11 products, there is a large gap in the IO supply voltage range (2.3V to 3.6V). A guarantee on drive strength across this voltage range would be intolerable to most vendors who only use a subsection of the IO supply range. As such, these voltages are segmented into two manageable sections referenced as VDDIOM, and VDDIOH in tables listed in this document.

25.1 Absolute Maximum Ratings

The values listed in this section are ratings that can be peaked by the device, but not sustained without causing irreparable damage to the device.

Table 25-1. Absolute Maximum Ratings

Symbol	Characteristics	Min.	Max.	Unit
VDDIO	I/O Supply Voltage	-0.3	4.2	V
VBAT	Battery Supply Voltage	-0.3	5.0	
V_{IN} (1)	Digital Input Voltage	-0.3	VDDIO	
V_{AIN} (2)	Analog Input Voltage	-0.3	1.5	
T_A	Storage Temperature	-65	150	°C

Note:

1. V_{IN} corresponds to all the digital pins.
2. V_{AIN} corresponds to all the analog pins, RFIO, XO_N, XO_P, TPP, RTC_CLK_N and RTC_CLK_P.

25.2 Recommended Operating Conditions

Table 25-2. Recommended Operating Conditions

Symbol	Characteristic	Min.	Typ.	Max.	Unit
$VDDIO_M$	I/O Supply Voltage Mid-Range	2.3	2.50	3.00	V
$VDDIO_H$	I/O Supply Voltage High Range	3.00	3.30	3.60	
VBAT	Battery Supply Voltage (1)	2.3	3.6	4.3	
	Operating Temperature	-40		85	°C

Note:

1. VBAT must not be less than VDDIO.
2. VBAT is the supply pin for DC/DC Convertor and VDDIO is the supply pin for I/Os. VDDIO and VBAT can either rise together or can be tied together. User can either tie it together and make sure the maximum specification of VDDIO is honored or can have VDDIO and VBAT powered separately and maintain the voltage levels within individual maximum levels specified for VDDIO and VBAT.

25.3 DC Characteristics

The Table 25-3 provides the DC characteristics for the digital pads.

Table 25-3. DC Electrical Characteristics

VDDIO Condition	Characteristic	Min.	Typ.	Max.	Unit
$VDDIO_M$	Input Low Voltage V_{IL}	-0.30		0.63	V
	Input High Voltage V_{IH}	VDDIO-0.60		VDDIO+0.30	
	Output Low Voltage V_{OL}			0.45	
	Output High Voltage V_{OH}	VDDIO-0.50			
$VDDIO_H$	Input Low Voltage V_{IL}	-0.30		0.65	
	Input High Voltage V_{IH}	VDDIO-0.60		VDDIO+0.30 (up to 3.60)	
	Output Low Voltage V_{OL}			0.45	
	Output High Voltage V_{OH}	VDDIO-0.50			
All	Output Loading			20	pF
	Digital Input Load			6	
$VDDIO_M$	Pad drive strength (regular pads (1))	3.4	6.6		mA
$VDDIO_H$	Pad drive strength (regular pads (1))	10.5	14		
$VDDIO_M$	Pad drive strength (high-drive pads (1))	6.8	13.2		
$VDDIO_H$	Pad drive strength (high-drive pads (1))	21	28		

Note:

1. The following GPIO pads are high-drive pads: GPIO_8, GPIO_9; all other pads are regular pads.

25.4 Receiver Performance

Table 25-4. BLE Receiver Performance

Parameter	Minimum	Typical	Maximum	Unit
Frequency	2,402		2,480	MHz
Sensitivity with on-chip DC/DC(1)	-92.7	-91.9		dBm
Maximum receive signal level		+5		

.....continued

Parameter	Minimum	Typical	Maximum	Unit
CCI		12.5		dB
ACI (N±1)		0		
N+2 Blocker (Image)		-20		
N-2 Blocker		-38		
N+3 Blocker (Adj. Image)		-35		
N-3 Blocker		-43		
N±4 or greater		-45		dB
Intermod (N+3, N+6)		-32		dBm
OOB (2 GHz<f<2.399 GHz)	-15			
OOB (f<2 GHz or f>2.5 GHz)	-10			

All measurements are taken after the RF input matching network. Refer to the reference schematic of Figure 27-1.

All measurements are performed at VBAT - 3.3V; VDDIO-3.3V and 25°C, with tests following the Bluetooth standard tests.

Note:

1. Typical receiver sensitivity is average across 40 channels.

25.5 Transmitter Performance

The transmitter contains fine step power control with P_{out} variable in <3 dB steps below 0 dBm and in <0.5 dB steps above 0 dBm.

Table 25-5. BLE Transmitter Performance

Parameter	Minimum	Typical	Maximum	Unit
Frequency	2,402		2,480	MHz
Maximum output power		3.5		dBm
In-band Spurious (N±2)		-45		
In-band Spurious (N±3)		-50		
2nd harmonic P_{out}	-41			
3rd harmonic P_{out}	-41			
4th harmonic P_{out}	-41			
5th harmonic P_{out}	-41			
Frequency deviation		±250		kHz

All measurements are taken after the RF input matching network. Refer to the reference schematic Figure 27-1.

All measurements are performed at VBAT - 3.3V; VDDIO - 3.3V and 25°C, with tests following the Bluetooth standard tests.

Note:

1. At 0 dBm TX output power.
2. With respect to TX power, different (higher/lower) RF output power settings may be used for specific antennas and/or enclosures, in which case recertification may be required.
3. Country specific settings (as per the Module Certifications) should be programmed at the Host product factory to match the intended Destination. Regulatory bodies prohibit exposing the settings to the end user. This requirement needs to be taken care of via Host implementation.

25.6 Current Consumption in Various Device States

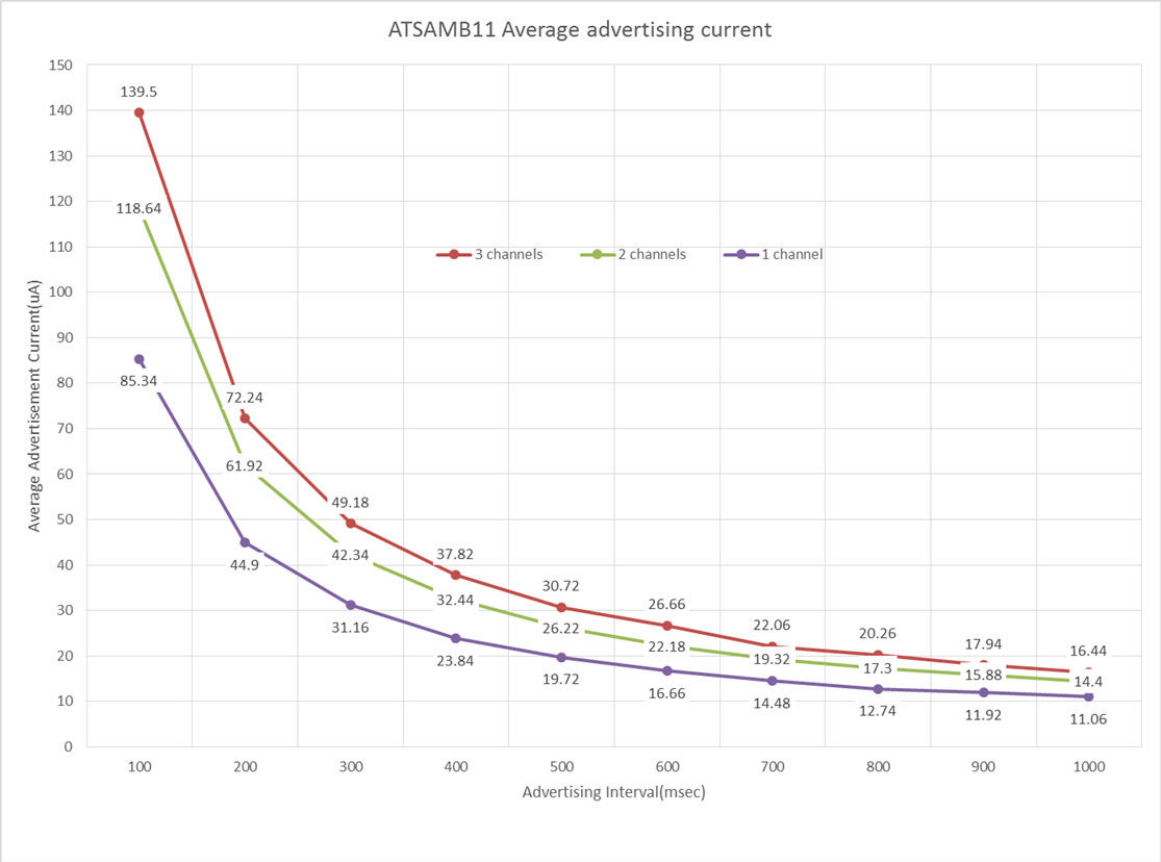
Table 25-6. Device State Current Consumption

Device State	C_EN	VDDIO	I_{VBAT}+I_{VDDIO} (typical) (2)
Power_Down	Off	On	0.03 µA
Ultra_Low_Power with BLE timer, with RTC (1)	On	On	2.03 µA
MCU_Only, idle	On	On	1.35 mA
BLE_On_Receive @channel 37 (2402 MHz)	On	On	5.26 mA
BLE_On_Transmit, 0 dBm output power @Channel 37 (2402 MHz)	On	On	4.18 mA
BLE_On_Transmit, 0 dBm output power @Channel 39 (2480 MHz)	On	On	3.71 mA
BLE_On_Transmit, 3 dBm output power @Channel 37 (2480 MHz)	On	On	5.69 mA
BLE_On_Transmit, 3 dBm output power @Channel 39 (2480 MHz)	On	On	4.65 mA

Note:

1. Sleep clock derived from external 32.768 kHz crystal specified for CL = 7 pF, using the default on-chip capacitance only, without using external capacitance.
2. Measurement conditions:
   - 2.1. VBAT=3.3V
   - 2.2. VDDIO=3.3V
   - 2.3. Temperature=25°C
   - 2.4. These measurements are taken with FW BluSDK Smart V6.1.6991

Figure 25-1. Average Advertising Current



Note:

1. The Average advertising current is measured at VBAT = 3.3 V, VDDIO = 3.3 V, TX output power = 0 dBm. Temperature - 25°C.
2. Advertisement data payload size - 31 octets.
3. Advertising event type - Connectable Undirected.
4. Advertising channels used in 2 channel : 37 and 38.
5. Advertising channels used in 1 channel: 37.

25.7 ADC Characteristics

Table 25-7. Static Performance of SAR ADC

Parameter	Condition	Min.	Typ.	Max.	Unit
Input voltage range		0		VBAT	V
Resolution			11		bits
Sample rate			100	1000	KSPS
Input offset	Internal VREF	-10		+10	mV
Gain error	Internal VREF	-4		+4	%

.....continued

Parameter	Condition	Min.	Typ.	Max.	Unit
DNL[2]	100 KSPS. Internal VREF=1.6V. Same result for external VREF.	-0.75		+1.75	LSB
INL[2]	100 KSPS. Internal VREF=1.6V. Same result for external VREF.	-2		+2.5	LSB
THD	1 kHz sine input at 100 KSPS		73		dB
SINAD	1 kHz sine input at 100 KSPS		62.5		dB
SFDR	1 kHz sine input at 100 KSPS		73.7		dB
Conversion time			13		cycles
Current consumption	Using external VREF, at 100 KSPS		13.5		µA
	Using internal VREF, at 100 KSPS		25.0		µA
	Using external VREF, at 1 MSPS		94		µA
	Using internal VREF, at 1MSPS		150		µA
	Using internal VREF, during VBAT monitoring		100		µA
	Using internal VREF, during temperature monitoring		50		µA
Internal reference voltage	Mean value using VBAT = 2.5V		1.026 [1]		V
	Standard deviation across parts		10.5		mV
VBAT Sensor Accuracy	Without calibration	-55		+55	mV
	With offset and gain calibration	-17		+17	mV
Temperature Sensor Accuracy	Without calibration	-9		+9	ºC
	With offset calibration	-4		+4	ºC

Note:

1. Effective VREF is 2xInternal Reference Voltage.
2. These values are characterized for 0x4000F404<28:29>: 0x03.

25.8 ADC Typical Characteristics

T_C = 25ºC and V_{BAT} = 3.0V, unless otherwise noted.

Figure 25-2. INL of SAR ADC

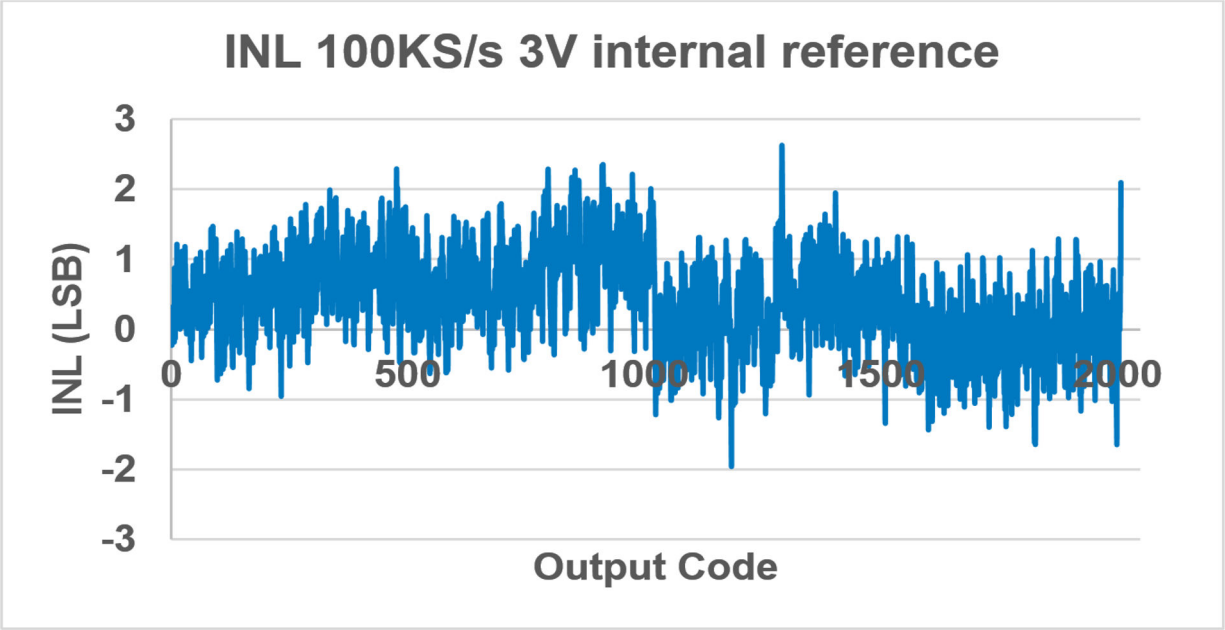


Figure 25-3. DNL of SAR ADC

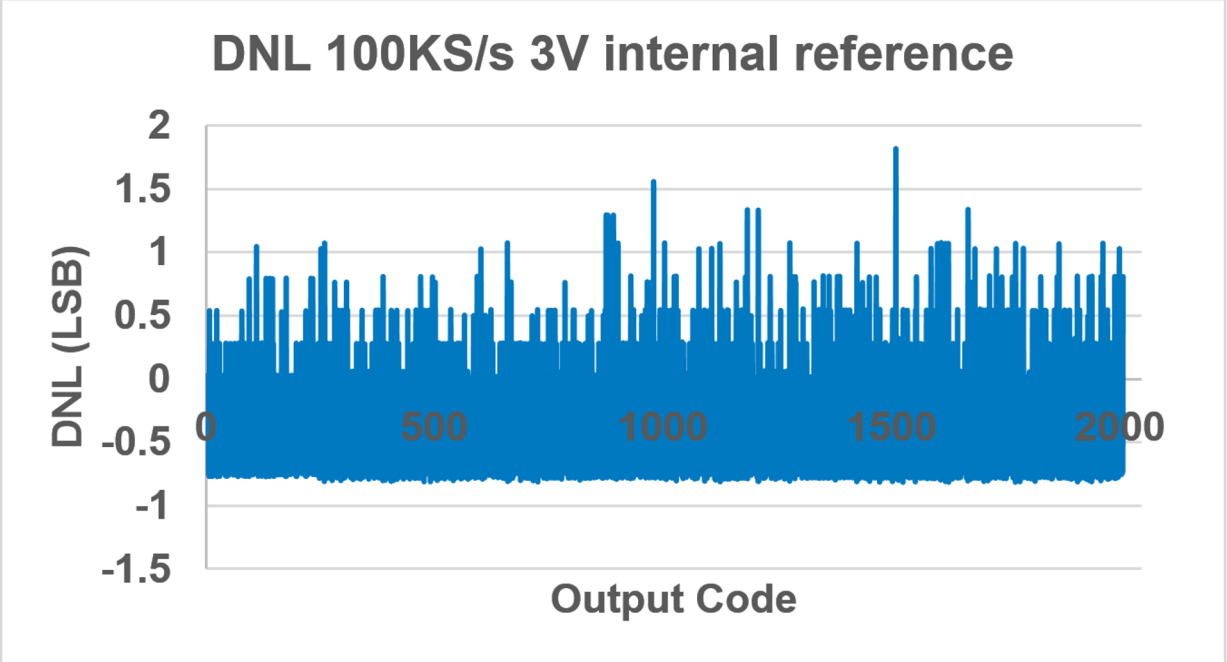
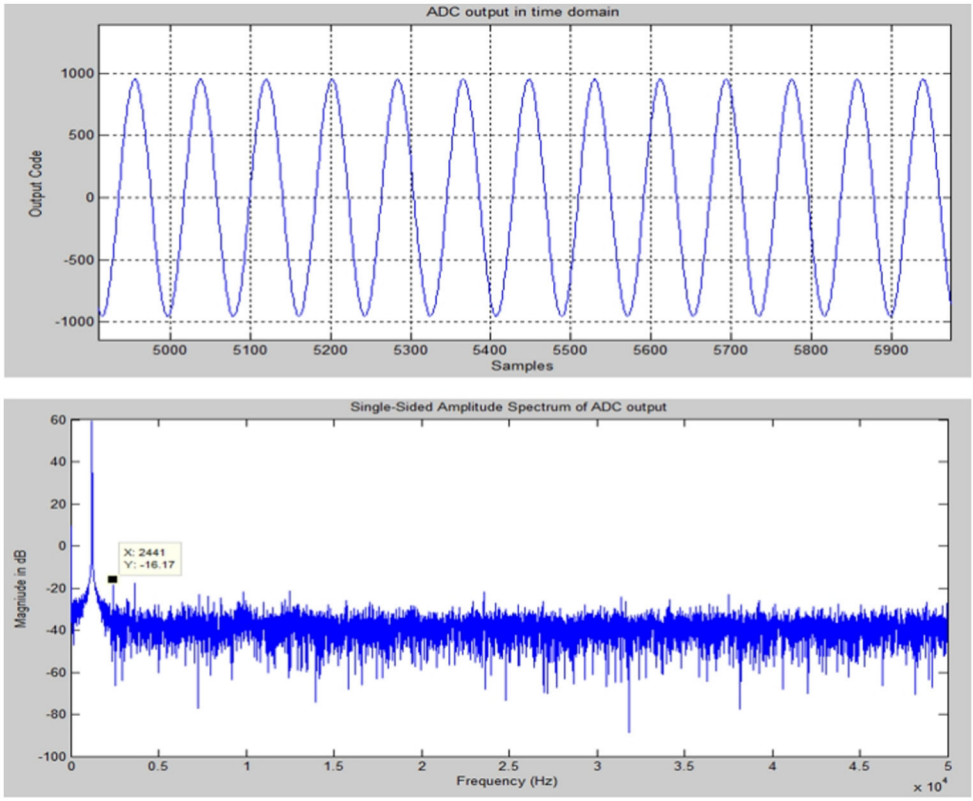


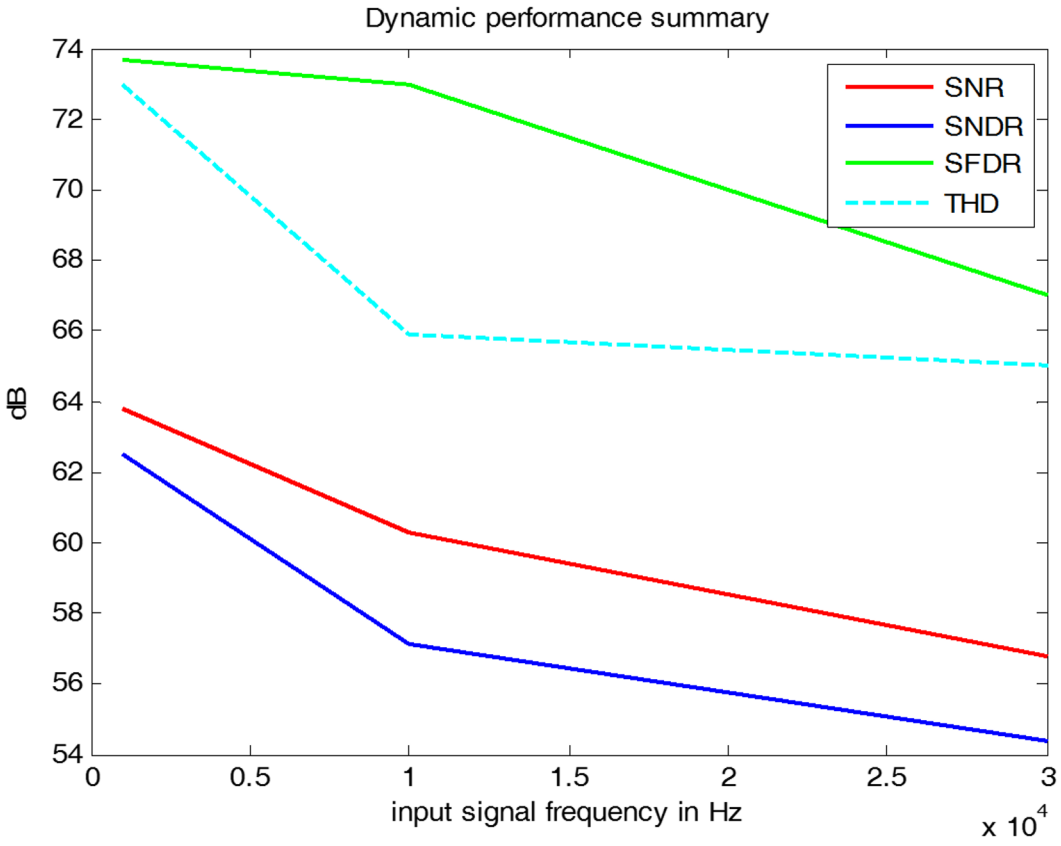
Figure 25-4. Sensor ADC Dynamic Measurement with Sinusoidal Input



Note:

1. 25ºC, 3.6V VBAT, and 100 kS/s
   . Input signal: 1 kHz sine wave, 3Vp-p amplitude.
2. SNDR = 62.5 dB
   , SFDR = 73.7 dB, and
   THD = 73.0 dB.

Figure 25-5. Sensor ADC Dynamic Performance Summary at 100 KSPS



25.9 Timing Characteristics

25.9.1 I²C Interface Timing

The I²C Interface timing (common to both Slave and Master) is provided in Figure 25-6. The timing parameters for Slave and Master modes are specified in tables Table 25-8 and Table 25-9 respectively.

Figure 25-6. I²C Slave Timing Diagram

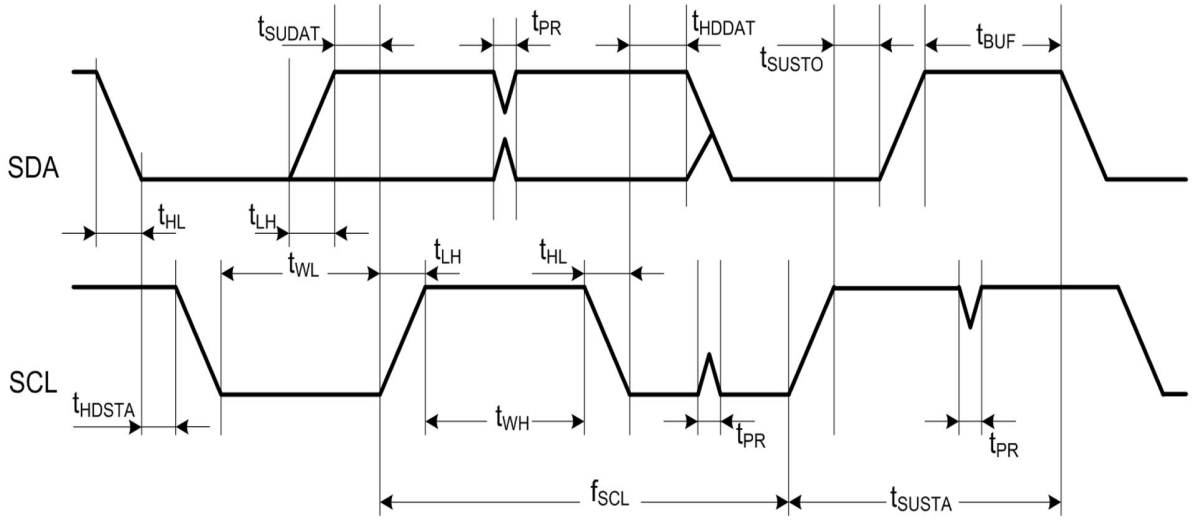


Table 25-8. I²C Slave Timing Parameters

Parameter	Symbol	Min.	Max.	Units	Remarks
SCL Clock Frequency	f_{SCL}	0	400	kHz	
SCL Low Pulse Width	t_{WL}	1.3		µs	
SCL High Pulse Width	t_{WH}	0.6			
SCL, SDA Fall Time	t_{HL}		300	ns	
SCL, SDA Rise Time	t_{LH}		300		This is dictated by external components
START Setup Time	t_{SUSTA}	0.6		µs	
START Hold Time	t_{HDSTA}	0.6			
SDA Setup Time	t_{SUDAT}	100		ns	
SDA Hold Time	t_{HDDAT}	0 40			Slave and Master Default Master Programming Option
STOP Setup time	t_{SUSTO}	0.6		µs	
Bus Free Time between STOP and START	t_{BUF}	1.3			
Glitch Pulse Reject	t_{PR}	0	50	ns	

Table 25-9. I²C Master Timing Parameters

Parameter	Symbol	Standard Mode		Fast Mode		High-speed Mode		Units
		Min.	Max.	Min.	Max.	Min.	Max.	
SCL Clock Frequency	f_{SCL}	0	100	0	400	0	3400	kHz

.....continued

Parameter	Symbol	Standard Mode		Fast Mode		High-speed Mode		Units
		Min.	Max.	Min.	Max.	Min.	Max.	
SCL Low Pulse Width	t_{WL}	4.7		1.3		0.16		µs
SCL High Pulse Width	t_{WH}	4		0.6		0.06		
SCL Fall Time	t_{HLSCL}		300		300	10	40	ns
SDA Fall Time	t_{HLSDA}		300		300	10	80	
SCL Rise Time	t_{LHSCL}		1000		300	10	40	
SDA Rise Time	t_{LHSDA}		1000		300	10	80	
START Setup Time	t_{SUSTA}	4.7		0.6		0.16		µs
START Hold Time	t_{HDSTA}	4		0.6		0.16		
SDA Setup Time	t_{SUDAT}	250		100		10		ns
SDA Hold Time	t_{HDDAT}	5		40		0	70	
STOP Setup time	t_{SUSTO}	4		0.6		0.16		µs
Bus Free Time between STOP and START	t_{BUF}	4.7		1.3				
Glitch Pulse Reject	t_{PR}			0	50			ns

25.9.2 SPI Slave Timing

The SPI Slave timing is provided in the following figure and tables.

Figure 25-7. SPI Slave Timing Diagram

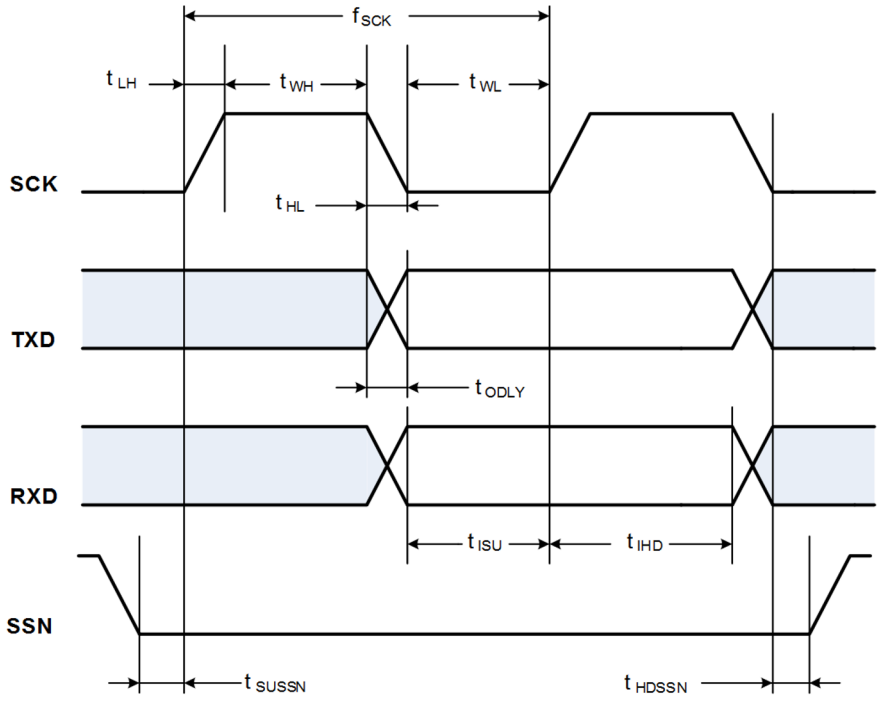


Table 25-10. SPI Slave Timing Parameters [1]

Parameter	Symbol	Min.	Max.	Units
Clock Input Frequency [2]	f_{SCK}		2	MHz
Clock Low Pulse Width	t_{WL}	55		ns
Clock High Pulse Width	t_{WH}	55		
Clock Rise Time	t_{LH}	0	7	
Clock Fall Time	t_{HL}	0	7	
TXD Output Delay[3]	t_{ODLY}	7	28	
RXD Input Setup Time	t_{ISU}	5		
RXD Input Hold Time	t_{IHD}	10		
SSN Input Setup Time	t_{SUSSN}	5		
SSN Input Hold Time	t_{HDSSN}	10		

Note:

1. Timing is applicable to all SPI modes.
2. Maximum clock frequency specified is limited by the SPI Slave interface internal design. Actual maximum clock frequency can be lower and depends on the specific PCB layout.
3. Timing based on 15 pF output loading.

25.9.3 SPI Master Timing

The SPI Master Timing is provided in the following figure and table.

Figure 25-8. SPI Master Timing Diagram

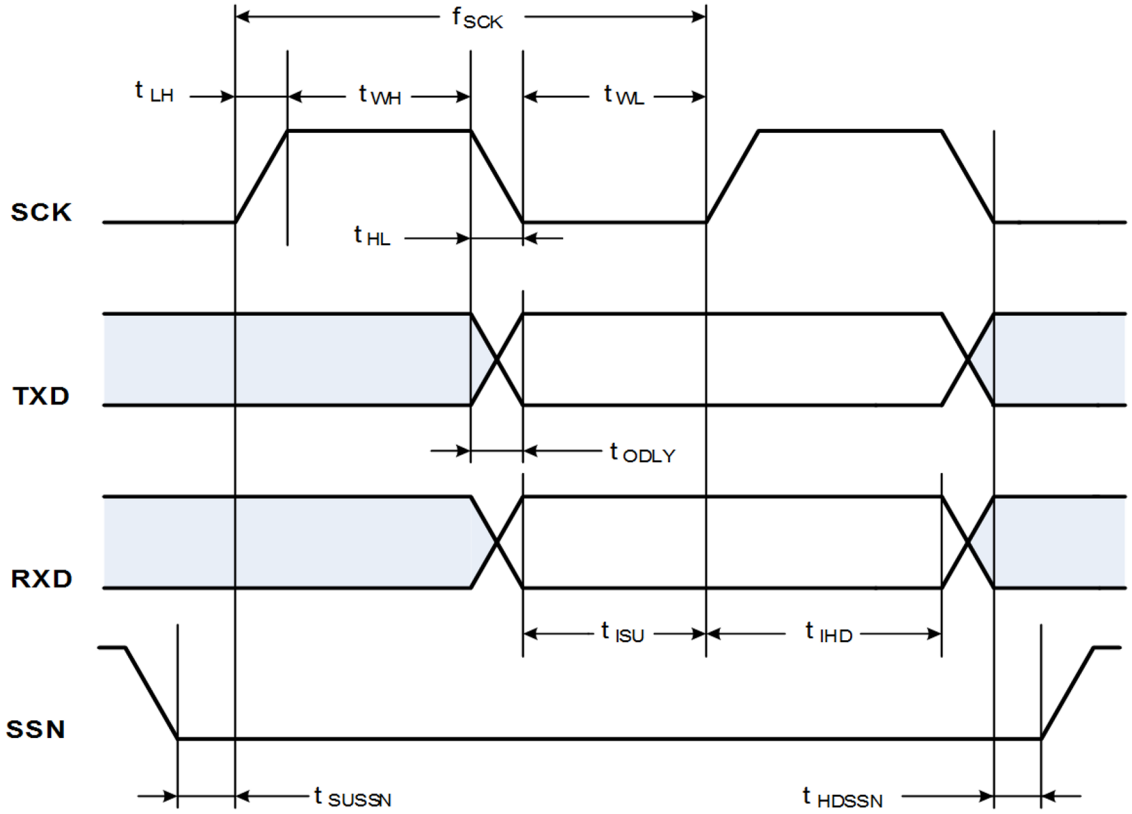


Table 25-11. SPI Master Timing Parameters(1)

Parameter	Symbol	Min.	Units	Max.
Clock Output Frequency(2)	f_{SCK}		4	MHz
Clock Low Pulse Width	t_{WL}	30		ns
Clock High Pulse Width	t_{WH}	32		
Clock Rise Time(3)	t_{LH}		7	
Clock Fall Time(3)	t_{HL}		7	
RXD Input Setup Time	t_{ISU}	23		
RXD Input Hold Time	t_{IHD}	0		
SSN/TXD Output Delay(3)	t_{ODLY}	0	12	

Note:

1. Timing is applicable to all SPI modes
2. Maximum clock frequency specified is limited by the SPI Master interface internal design. The actual maximum clock frequency can be lower and depends on the specific PCB layout
3. Timing based on 15pF output loading

25.9.4 SPI Flash Master Timing

The SPI Master Timing is provided in the following figure and table.

Figure 25-9. SPI Flash Master Timing Diagram

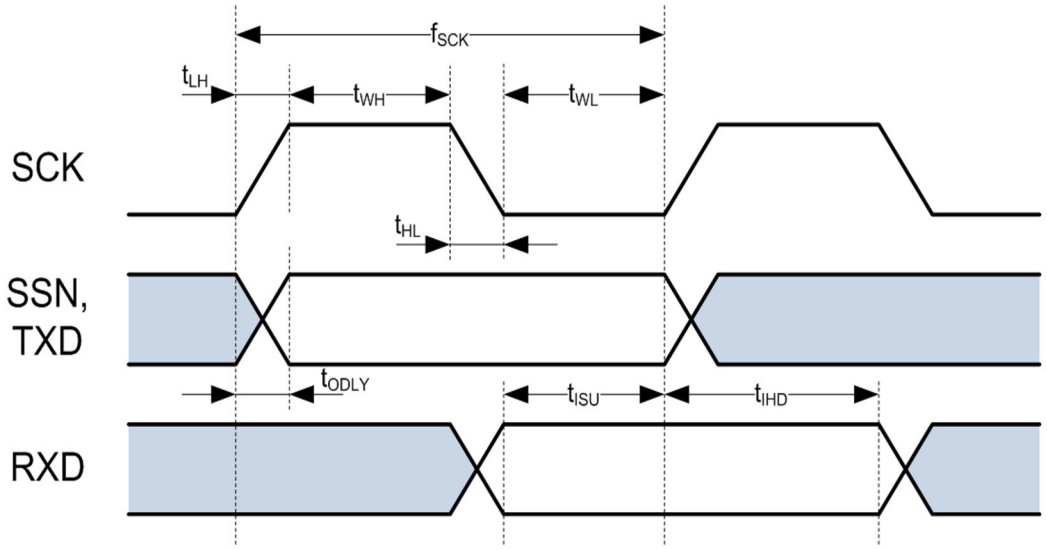


Table 25-12. SPI Flash Master Timing Parameters(1)

Parameter	Symbol	Min.	Max.	Units
Clock Output Frequency(2)	f_{SCK}		13	MHz
Clock Low Pulse Width	t_{WL}	25		ns
Clock High Pulse Width	t_{WH}	27		
Clock Rise Time(3)	t_{LH}		11	
Clock Fall Time(3)	t_{HL}		10	
RXD Input Setup Time	t_{ISU}	19		
RXD Input Hold Time	t_{IHD}	0		
SSN/TXD Output Delay(3)	t_{ODLY}	1	7	

Note:

1. Timing is applicable to all SPI modes.
2. Maximum clock frequency specified is limited by the SPI Master interface internal design. Actual maximum clock frequency can be lower and depends on the specific PCB layout.
3. Timing based on 15 pF output loading.

26. Package Outline Drawings

26.1 Package Outline Drawing

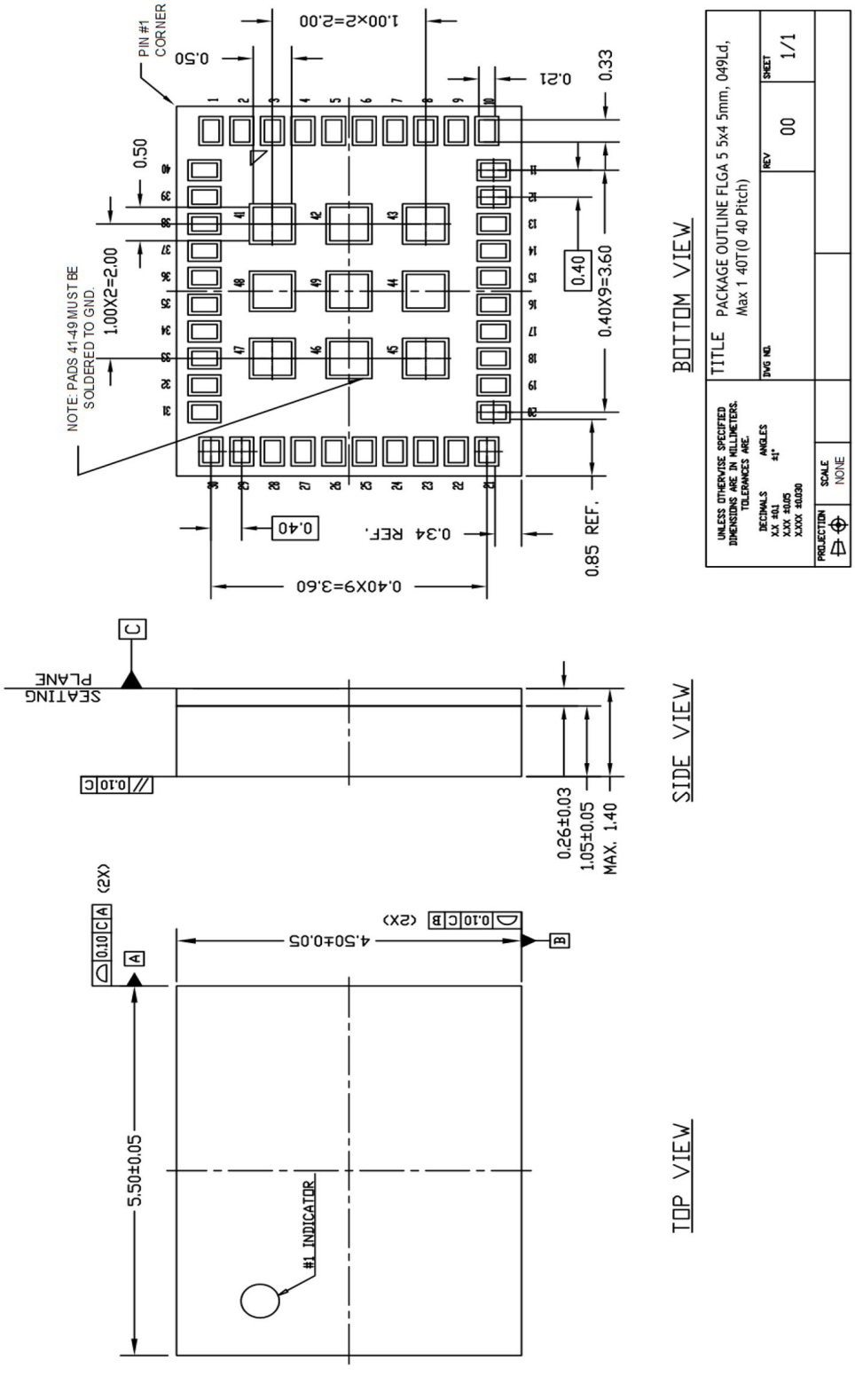
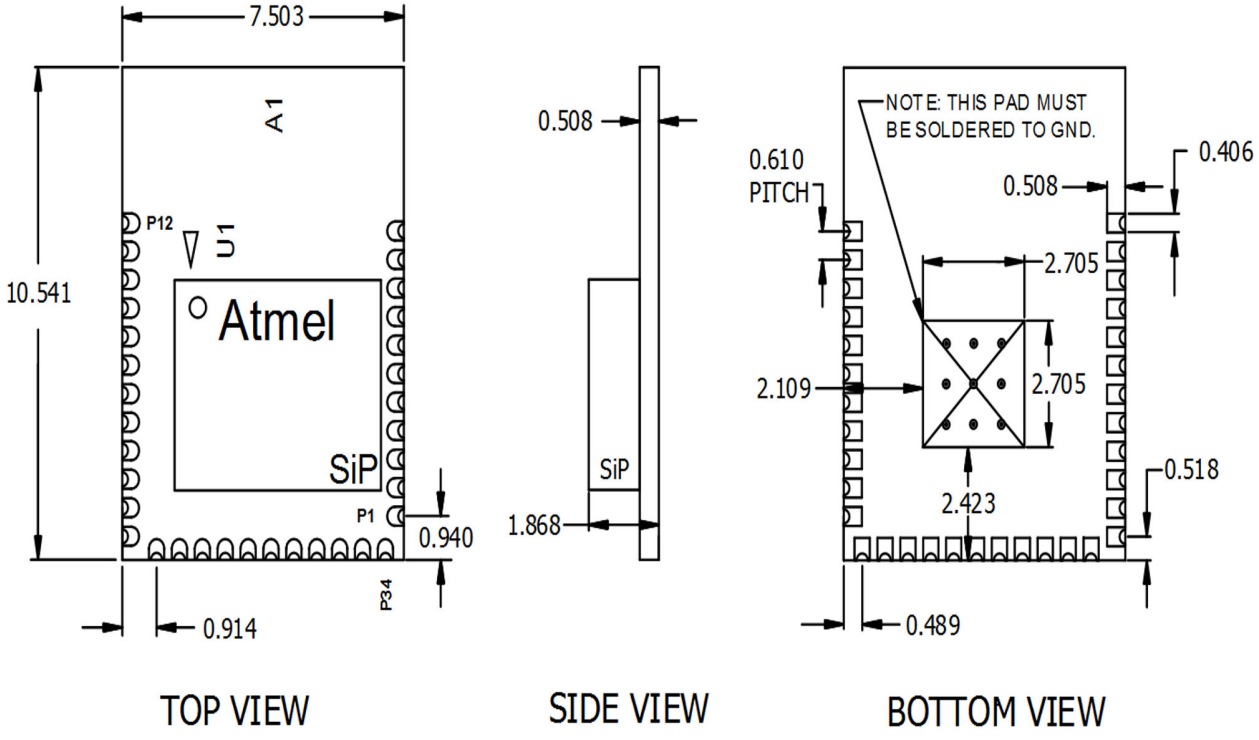


Figure 26-1. ATSAMB11-XR2100A Package Outline Drawing

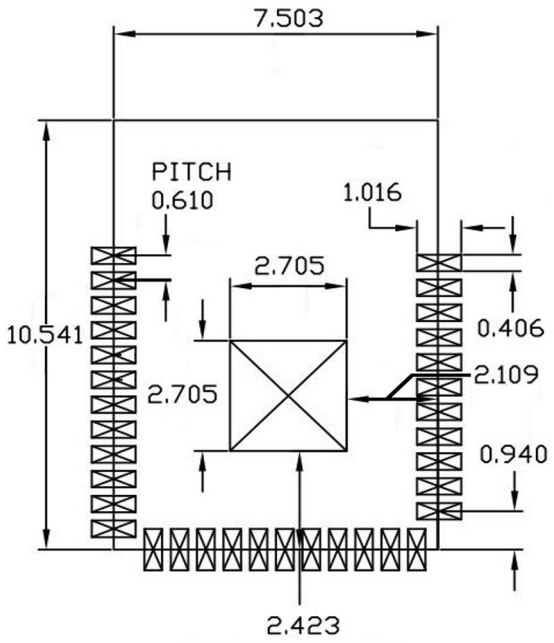
26.2 Module PCB Package Outline Drawing

Figure 26-2. ATSAMB11-ZR210CA Module Package Outline Drawing



ATSAMB11-ZR210CA
Dimension units: mm
Untoleranced dimensions
Drawing not to scale

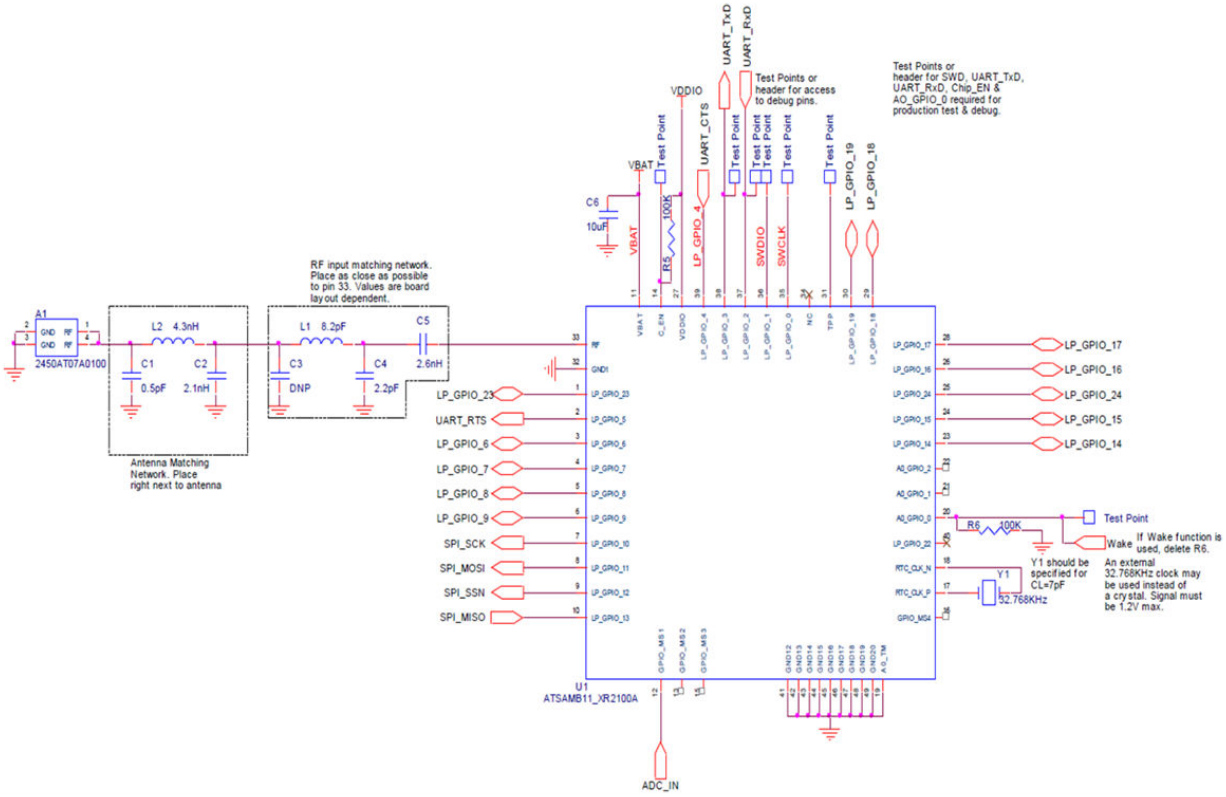
Figure 26-3. Customer PCB Top View Footprint



27. Module Reference Schematics

27.1 Reference Schematic

Figure 27-1. ATSAMB11-XR2100A Reference Schematic



27.2 Reference Schematic Bill of Materials (BOM)

Table 27-1. ATSAMB11-XR2100A Reference Schematic Bill of Materials (BOM)

Item	Qty	Reference	Value	Description	Manufacturer	Part Number	Footprint
1	1	A1	2450AT07A0100	1x0.5 mm Ceramic Chip Antenna	Johanson Dielectrics	2450AT07A0100	
2	1	C1	0.5 pF	CAP, CER, 0.5 pF, +/-0.1 pF, NPO, 0201, 25V, -55-125 C	Johanson Dielectrics	250R05L0R5BV4T	0201

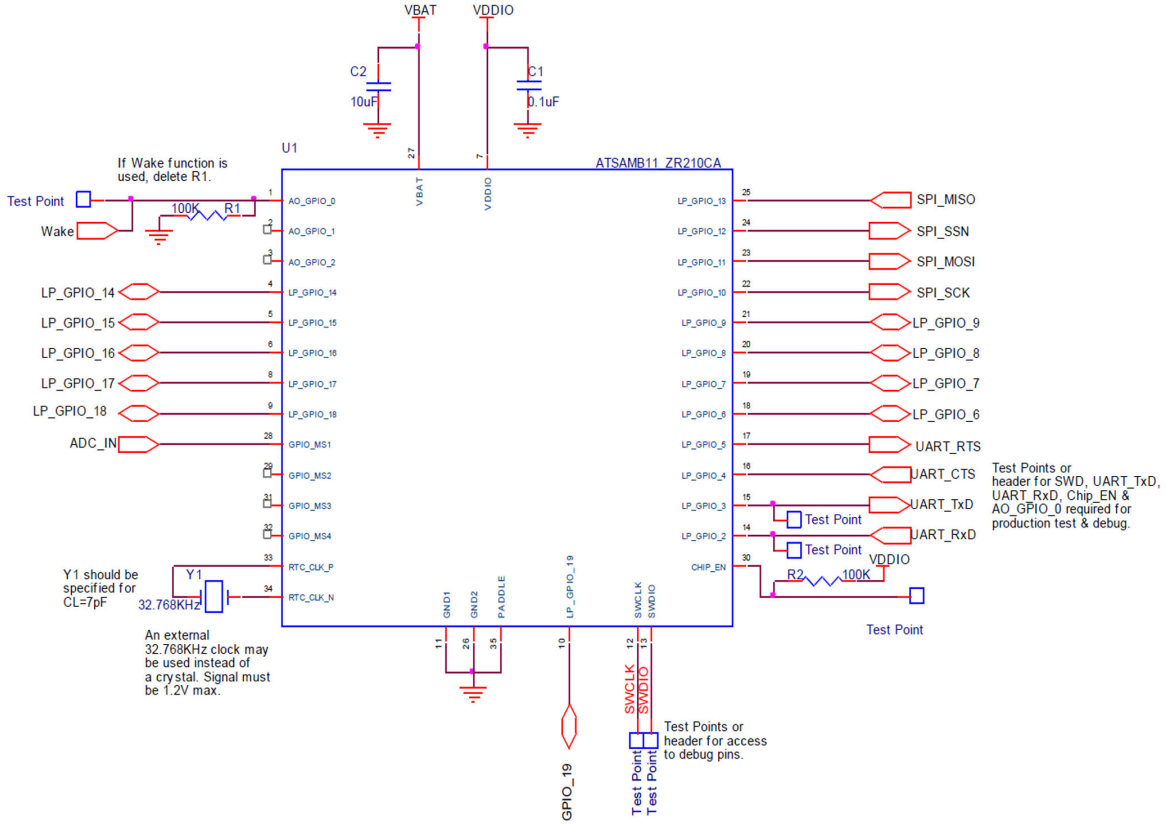
.....continued

Item	Qty	Reference	Value	Description	Manufacturer	Part Number	Footprint
3	1	C2	2.1 nH	Inductor,2.1 nH, +/-0.1 nH, Q=14@500 MHz, SRF=11 GHz, 0201, -55-125 C	Murata Americas	LQP03TN2N1B02D	0201
4	1	C3	DNP	CAP, CER, 2.2 pF, +/-0.1 pF, NPO, 0201, 25V, -55-125C	Johanson Dielectrics	250R05L2R2BV4T	0201
5	1	C4	2.2 pF	CAP, CER, 2.2 pF, +/-0.1 pF, NPO, 0201, 25V, -55-125 C	Johanson Dielectrics	250R05L2R2BV4T	0201
6	1	C5	2.6 nH	Inductor, 2.6 nH, +/-0.1 nH, Q=13@500 MHz, SRF=6 GHz, 0201,-55-125 C	Murata Americas	LQP03TG2N6B02D	0201
7	1	C6	10 uF	CAP, CER, 10 uF, 20%, X5R, 0603, 6.3V	AVX Corporation	06036D106MAT2A	0603
8	1	L1	8.2 pF	CAP, CER, 8.2 pF, +/-0.1 pF, NPO, 0201, 25V, -55-125 C	Johanson Dielectrics	250R05L8R2BV4T	0201
9	1	L2	4.3 nH	Inductor, 4.3 nH, +/-3%, Q=13@500 MHz, SRF=6 GHz, 0201, -55-125 C	Murata Americas	LQP03TG4N3H02D	0201
10	2	R5,R6	100K	RESISTOR, Thick Film, 100 kOhm, 0201	Panasonic®	ERJ-1GEF1003C	0201
11	7	TP1,TP2, TP4,TP5, TP6,TP7, TP8	Non-Component	Test Point,Surface Mount, 0.040"sq w/0.25"hole		40X40_SM_TEST_POINT	0.04"SQx 0.025"H
12	1	U1	ATSAMB11-XR2100A	ATSAMB11-XR2100A BLE SIP	Microchip Technology Inc	ATSAMB11-XR2100A	ATSAMB11-XR2100A

.....continued							
Item	Qty	Reference	Value	Description	Manufacturer	Part Number	Footprint
13	1	Y1	32.768 KHz	Crystal, 32.768 KHz, +/-20 ppm, -40-+85C, CL=7 pF, 2 lead, SMD	ECS, Inc. International	ECS-. 327-7-34B-TR	

27.3 Reference Schematic

Figure 27-2. ATSAMB11-ZR210CA Reference Schematic



27.4 Reference Bill of Materials(BOM)

Table 27-2. ATSAMB11-ZR210CA Reference Schematic Bill of Materials (BOM)

Item	Qty	Reference	Value	Description	Manufacturer	Part Number	Footprint
1	1	C1	0.1 uF	CAP, CER, 0.1 UF 6.3V +/-10% X5R 0201	AVX Corporation	02016D104KAT2A	0201

.....continued

Item	Qty	Reference	Value	Description	Manufacturer	Part Number	Footprint
2	1	C2	10 uF	CAP, CER, 10 uF, 20%, X5R, 0603, 6.3V	AVX Corporation	06036D106MAT2A	0603
3	2	R1, R2	100 K	RESISTOR, Thick Film, 100 kOhm, 0201	Panasonic	ERJ-1GEF1003C	0201
4	1	U1	ATSAMB11-ZR210CA	ATSAMB11-ZR210CA BLE Module	Microchip Technology Inc.	ATSAMB11-ZR210CA	ATSAMB11-ZR210CA
5	1	Y1	32.768 KHz	Crystal, 32.768 KHz, +/-20 ppm, -40-+85 C, CL=7 pF, 2 lead, SMD	ECS, Inc. International	ECS-.327-7-34B-TR	

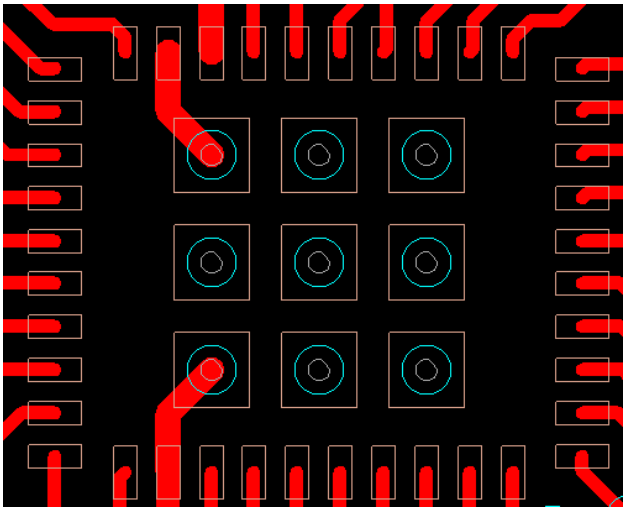
28. ATSAMB11-XR2100A Design Considerations

The ATSAMB11-XR2100A is offered in a shielded Land Grid Array (LGA) package with organic laminate substrates. The LGA package makes the second level interconnect (from package to the customer PCB) with an array of solderable surfaces. This may consist of a layout similar to a BGA with no solder spheres. However, it may also have an arbitrary arrangement of solderable surfaces that typically includes large planes for grounding or thermal dissipation, smaller lands for signals or shielding grounds, and in some cases, mechanical reinforcement features for mechanical durability.

28.1 Layout Recommendation

Referring to the SiP footprint dimensions in Figure 26-1, it is recommended to use solder mask defined with PCB pads 0.22 mm wide that have a 0.4 mm pitch. A Sample PCB pad layout in following figure shows the required vias for the center ground paddle.

Figure 28-1. PCB Footprint For ATSAMB11-XR2100A



The land design on the customer PCB should follow the following rules:

1. The solderable area on the customer PCB should match the nominal solderable area on the LGA package 1:1.
2. The solderable area should be finished with organic surface protectant (OSP), NiAu, or a solder cladding.
3. The decision on whether to have a solder mask defined (SMD) land or a non-solder mask defined (NSMD) land depends on the application space.
   - SMD: If field reliability is at risk due to impact failures such as dropping a hand-held portable application, then the SMD land is recommended to optimize mechanical durability.
   - NSMD: If field reliability is at risk due to a solder fatigue failure (temperature cycle related open circuits), then the NSMD land is recommended to maximize solder joint life.

28.1.1 Power and Ground

Proper grounding is essential for correct operation of the SiP and peak performance. Figure 26-1 shows the bottom view of the ATSAMB11-XR2100A SiP with exposed ground pads. The SiP exposed ground pads must be soldered to customer PCB ground plane. A solid inner layer ground plane should be provided. The center ground paddle of the SiP must have a grid of ground vias solidly connecting the pad to the inner layer ground plane (one via per exposed center ground pads J41 to J49).

Dedicate one layer as a ground plane, preferably the second layer from the top. Make sure that this ground plane does not get broken up by routes. Power can route on all layers except the ground layer. Power supply routes should be heavy copper fill planes to ensure the lowest possible inductance. The power pins of the ATSAMB11-XR2100A should have a via directly to the power plane as close to the pin as possible. Decoupling capacitors should have a via right next to the capacitor pin and this via should go directly down to the power plane – that is to say, the capacitor should not route to the power plane through a long trace. The ground side of the decoupling capacitor should have a via right next to the pad which goes directly down to the ground plane. Each decoupling capacitor should have its own via directly to the ground plane and directly to the power plane right next to the pad. The decoupling capacitors should be placed as close to the pin that it is filtering as possible.

28.1.2 Antenna

When designing the ATSAMB11-XR2100A, it is important to pay attention to the following recommendations for antenna placement:

1. Make sure to choose an antenna that covers the proper frequency band; 2.400 GHz to 2.500 GHz.
2. Assure that the antenna is designed matched to 50 Ohm input impedance.
3. Talk to the antenna vendor and make sure it is understood that the full frequency range must be covered by the antenna.
4. Be sure to follow the antenna vendors best practice layout recommendations, while placing the antenna in the customer PCB design.
5. The customer PCB pad that the antenna is connected to must be properly designed for 50 Ohm impedance.
6. Make sure that the trace from the RF pin on the ATSAMB11-XR2100A to the antenna matching circuitry has a 50 Ohm impedance.
7. Do not enclose the antenna within a metal shield.
8. Keep any components that may radiate noise or signals within the 2.4 GHz to 2.5 GHz frequency band far away from the antenna and RF traces or better yet, shield the noisy components. Any noise radiated from the customer PCB in this frequency band will degrade the sensitivity of the ATSAMB11-XR2100A device.

28.2 SWD Interface

For programming and/or debugging the ATSAMB11XR/ZR, the device must be connected using the Serial Wire Debug (SWD) interface. Currently, the SWD interface is supported by Microchip programmers and debuggers SAM-ICE and ATMEL-ICE.

For ATMEL-ICE, which supports Cortex Debug Connector (10-pin) interface, the signals must be connected, as shown in Figure 28-2 with details described in Table 28-1.

Figure 28-2. Cortex Debug Connector (10-pin)

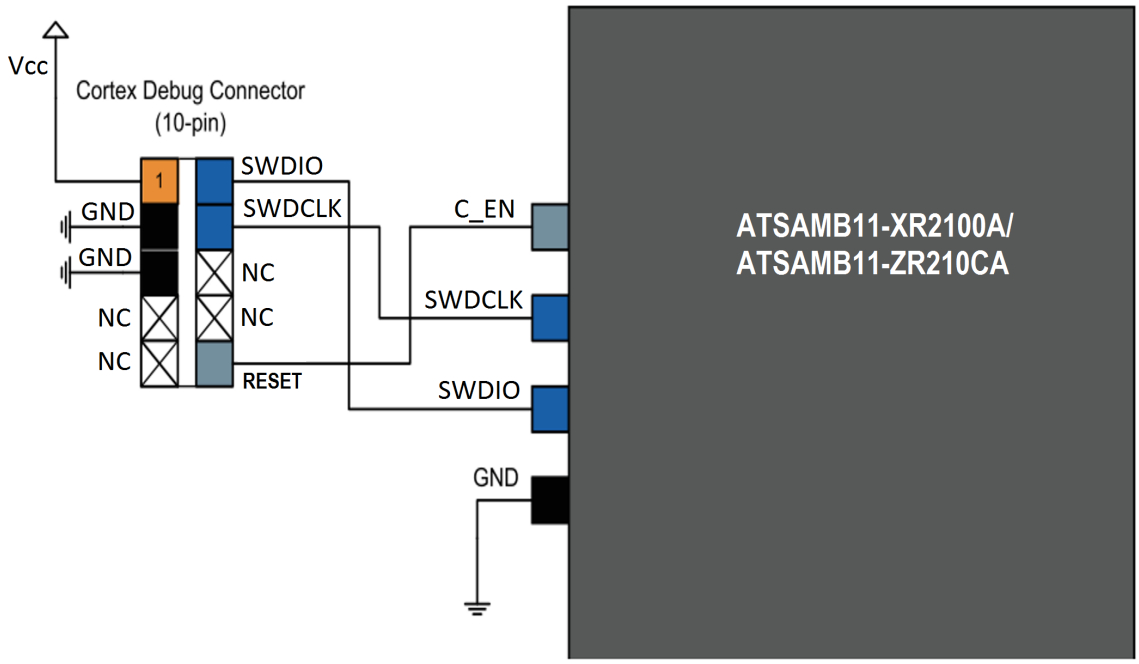


Table 28-1. Cortex Debug Connector (10-pin)

Header	Signal Name Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
C_EN	Target device reset pin, active-low
Vcc	Target voltage
GND	Ground

For SAM-ICE which support the 20-pin IDC JTAG Connector, the signals from ATSAMB11-XR2100A/ ATSAMB11-ZR210CA must be connected, as shown in Figure 28-3 with details described in Table 28-2.

Figure 28-3. 20-pin IDC JTAG Connector

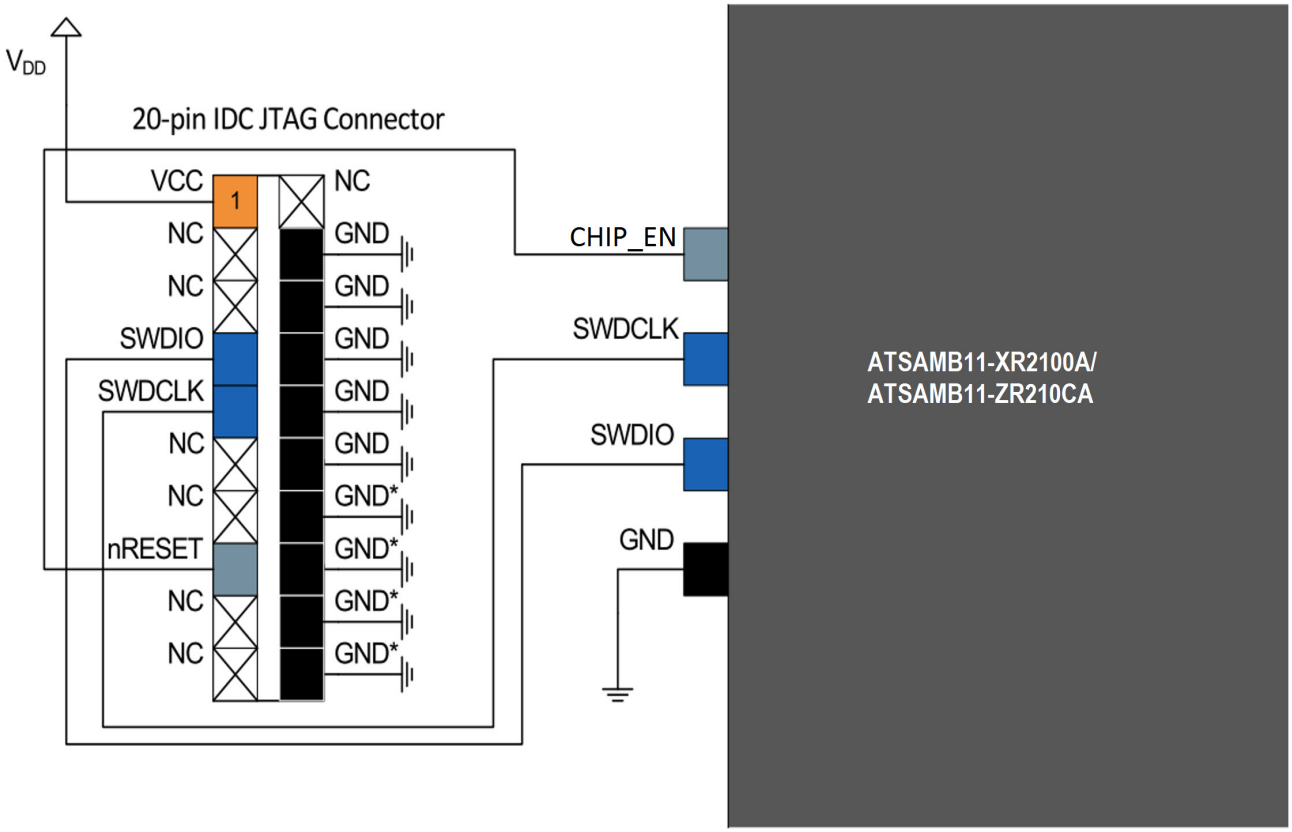


Table 28-2. 20-pin IDC JTAG Connector

Header	Signal Name Description
SWDCLK	Serial wire clock pin
SWDIO	Serial wire bidirectional data pin
nRESET	Target device reset pin, active-low
Vcc	Target voltage
GND	Ground
GND*	These pins are reserved for firmware extension purposes. They can be left open or connected to GND in normal debug environment. They are not essential for SWD in general.

28.3 Unused or Unconnected Pins

Internal pull-down for unused pins must be enabled to acheive the lowest current leakage.

29. ATSAMB11-ZR210CA Design Considerations

29.1 Placement and Routing Guidelines

It is critical to follow the recommendations listed below to achieve the best RF performance for the ATSAMB11-ZR210CA module:

1. The module must be placed on the host board and the chip antenna area must not overlap with the host board. The portion of the module containing the antenna must not stick out over the edge of the host board. Figure 29-2 shows the best, poor and worst-case module placements in the host board.
2. Follow the module placement and mechanical cutout recommendation as shown in Figure 29-1.
   - Avoid routing any traces in the highlighted region on the top layer of the host board which will be directly below the module area.
   - Place GND polygon pour below the module with the recommended boundary in the top layer of the host board as shown in Figure 29-1. Do not have any breaks in this GND plane. We recommend having a minimum area of 40x20mm for GND polygon pour in the lop layer of the host board.
   - Place sufficient GND vias in the highlighted area below the module for better RF performance.
   - It is recommended to have a 3x3 grid of GND vias solidly connecting the exposed GND paddle of the module to the ground plane on the inner/other layers of the host board. The GND vias should have a minimum via hole size of 0.2mm.
3. Do not enclose the antenna within a metal shield. Keep large metal objects as far away as possible from the antenna, to avoid electromagnetic field blocking.`
4. Keep any components which may radiate noise or signals within the 2.4 GHz to 2.5 GHz frequency band away from the antenna and if possible, shield those components. Any noise radiated from the host board in this frequency band will degrade the sensitivity of the module.

Figure 29-1. PCB Keep Out Area

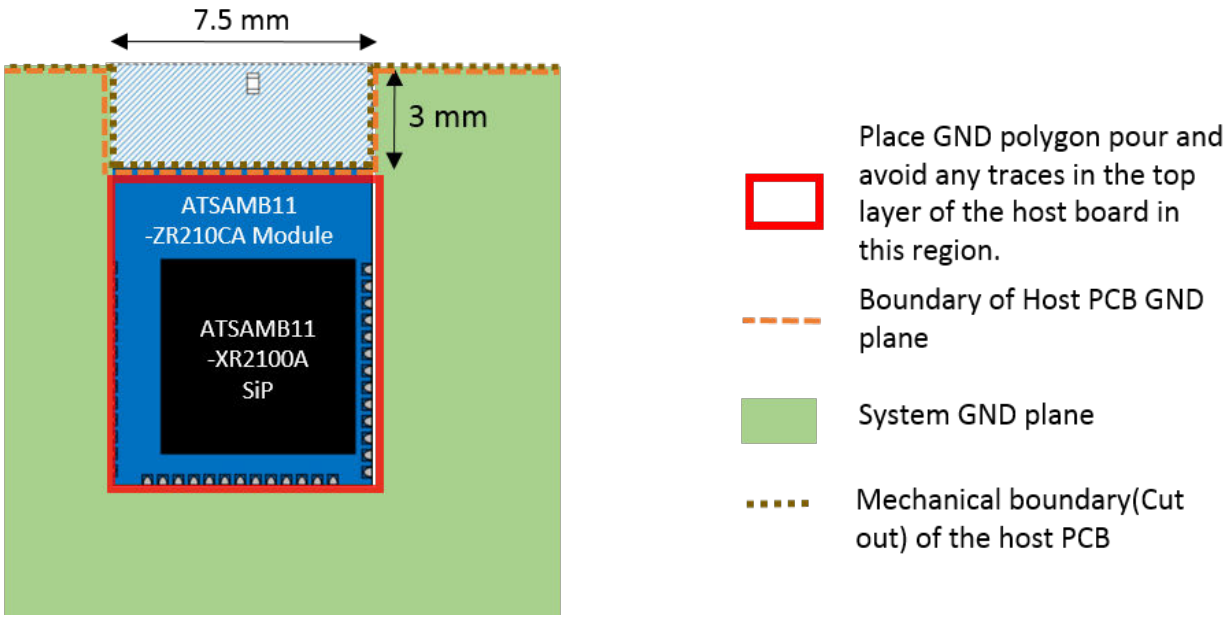
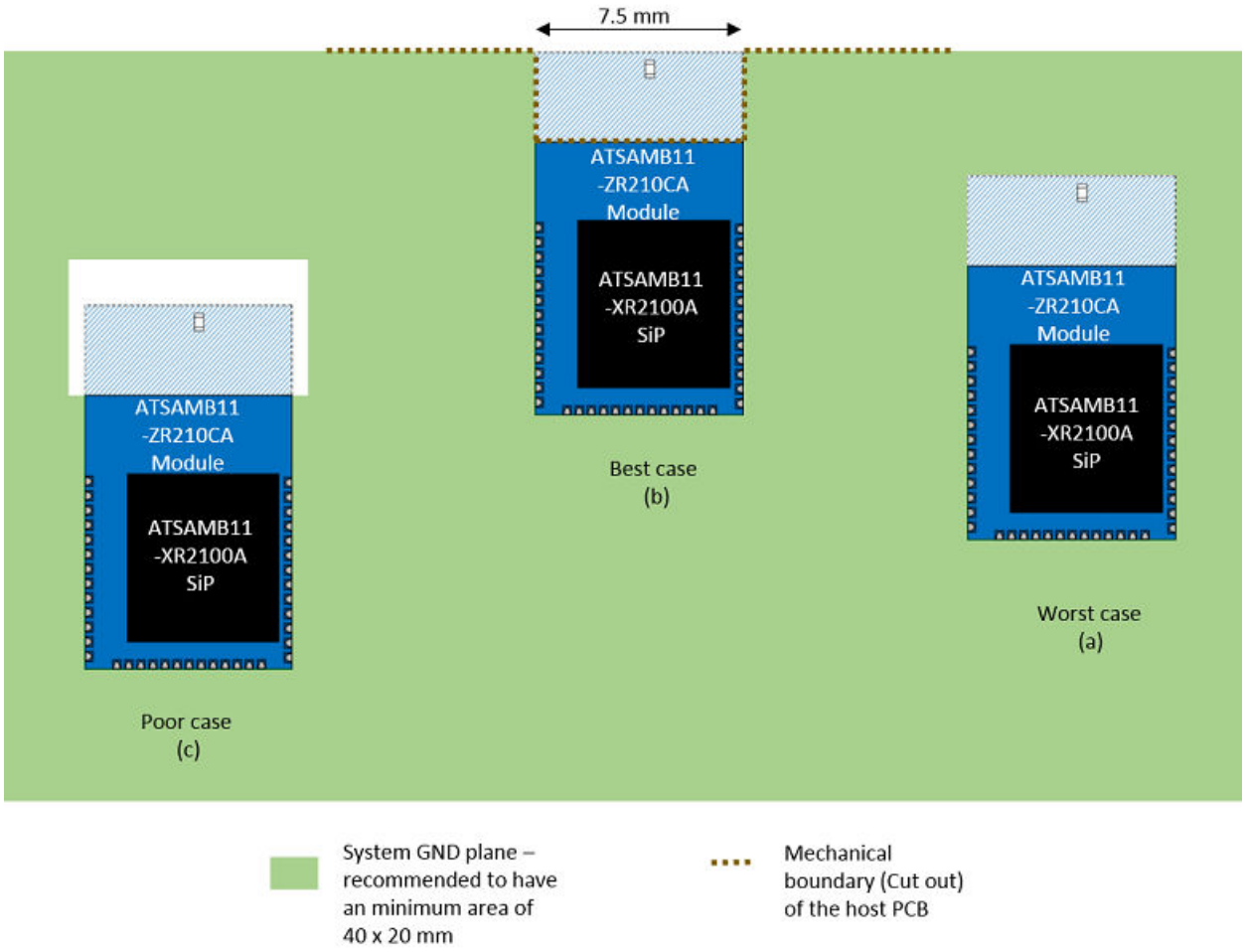


Figure 29-2. ATSAMB11-ZR210CA Placement Examples



29.2 Interferers

One of the biggest problems with RF devices is poor performance due to interferers on the board radiating noise into the antenna or coupling into the RF traces going to input LNA. Care must be taken to make sure that there no noisy circuitry is placed anywhere near the antenna or the RF traces. All noise generating circuits should also be shielded so they do not radiate noise that is picked up by the antenna. This applies to all layers. Even if there is a ground plane on a layer between the RF route and another signal, the ground return current will flow on the ground plane and couple into the RF traces.

30. Reflow Profile Information

This section provides guidelines for the reflow process in soldering the ATSAMB11-XR2100A or the ATSAMB11-ZR210CA to the customer's design. For more information on the reflow process guidelines, refer to the Solder Reflow Recommendation application note (DS00233D).

30.1 Storage Condition

30.1.1 Moisture Barrier Bag Before Opening

A moisture barrier bag must be stored in a temperature of less than 30°C with humidity under 85% RH.

The calculated shelf life for the dry-packed product shall be 12 months from the date the bag is sealed.

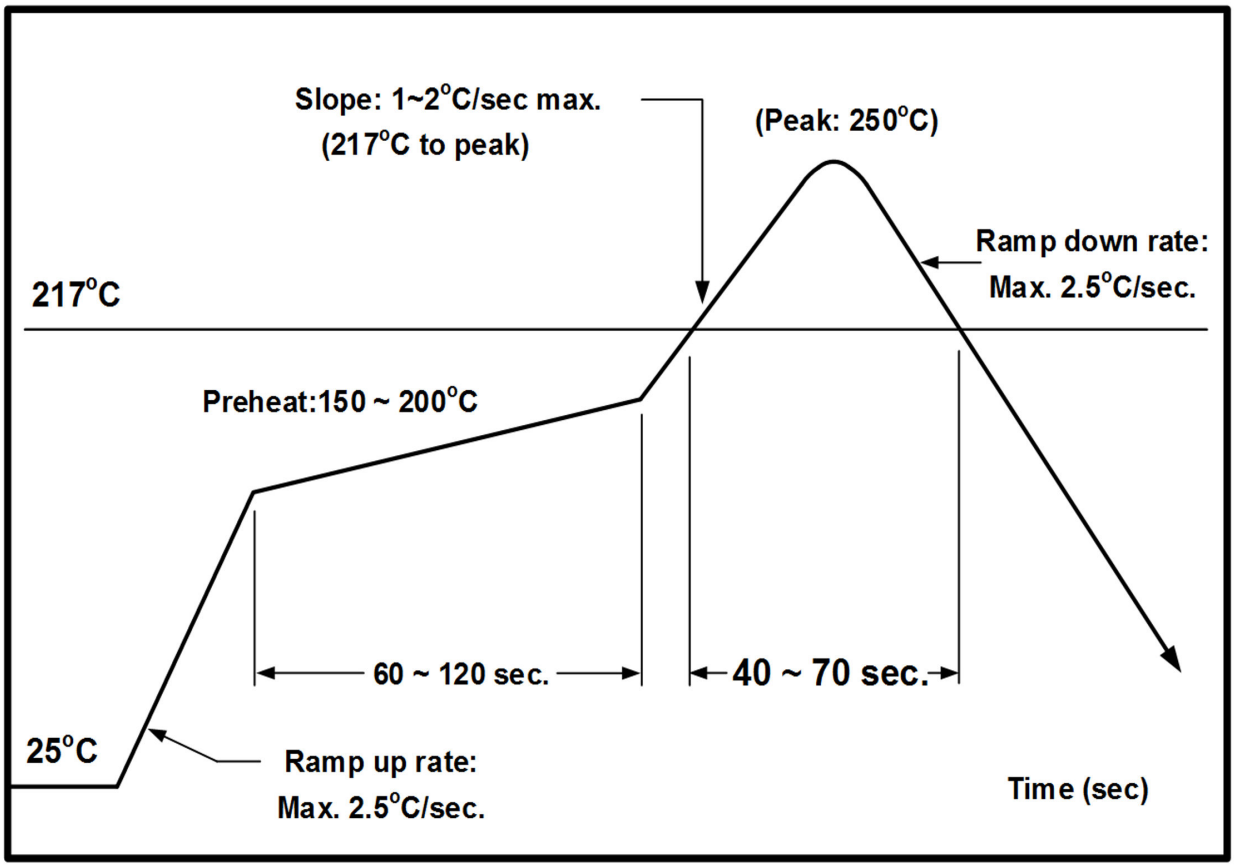
30.1.2 Moisture Barrier Bag Open

Humidity indicator cards must be blue, < 30%.

30.2 Soldering and Reflow Conditions

30.2.1 Solder Reflow Profile

Figure 30-1. Solder Reflow Profile



30.2.2 Reflow Oven

The following items should also be observed in the reflow process.

Allowable reflow soldering iterations:

- Three times based on the following reflow soldering profile (see Solder Reflow Profile).

Temperature profile:

- Reflow soldering shall be done according to the following temperature profile (see Solder Reflow Profile).
- Peak temperature: 250°C.

Cleaning:

The exposed ground paddle helps to self-align the module, avoiding pad misalignment. The use of no clean solder pastes is recommended. Full drying of no-clean paste fluxes as a result of the reflow process must be ensured. This may require longer reflow profiles and/or peak temperatures toward the high end of the process window as recommended by the solder paste vendor. It is believed that uncured flux residues could lead to corrosion and/or shorting in accelerated testing and possibly the field.
Note: Solutions like IPA and similar solvents can be used to clean the ATBTLC1000-ZR110CA module. However, cleaning solutions containing acid must never be used on the module.

Rework:

Rework is to remove the mounted SIP package and replace with a new unit. It is recommended that once an ATSAMB11-ZR210CA Module has been removed it should never be reused. During the rework process, the mounted module and PCB are heated partially, and the module is removed. It is recommended to pay attention to heat-proof the proximity of the mounted parts and junctions and use the best nozzle for rework that is suited to the module size.

30.3 Baking Conditions

This module is rated at MSL level 3. After sealed bag is opened, no baking is required within 168 hours so long as the devices are held at <= 30°C/60% RH or stored at <10% RH.

The module will require baking before mounting if:

- The sealed bag has been open for > 168 hours.
- Humidity Indicator Card reads >10%.
- SiPs need to be baked for 8 hours at 125°C.

30.4 Module Assembly Considerations

The Microchip ATSAMB11-ZR210CA modules are not intended for use with a conformal coating and the customer assumes all risks if a conformal coating is applied to the modules.

31. Regulatory Approval

Regulatory approvals received:

ATSAMB11-ZR210CA

- United States/FCC ID: 2ADHKBZR
- Canada/ISED
  - IC: 20266-SAMB11ZR
  - HVIN: ATSAMB11-ZR210CA
  - PMN: ATSAMB11-ZR210CA
- Europe - CE (RED)
- Japan/MIC: 005-101793
- Korea/KCC: R-CRM-mcp-SAMB11ZR210C
- Taiwan/NCC No: CCAN18LP0500T2

31.1 United States

The ATSAMB11-ZR210CA module has received Federal Communications Commission (FCC) CFR47 Telecommunications, Part 15 Subpart C “Intentional Radiators” single-modular approval in accordance with Part 15.212 Modular Transmitter approval. Single-modular transmitter approval is defined as a complete RF transmission sub-assembly, designed to be incorporated into another device, that must demonstrate compliance with FCC rules and policies independent of any host. A transmitter with a modular grant can be installed in different end-use products (referred to as a host, host product, or host device) by the grantee or other equipment manufacturer, then the host product may not require additional testing or equipment authorization for the transmitter function provided by that specific module or limited module device.

The user must comply with all of the instructions provided by the Grantee, which indicate installation and/or operating conditions necessary for compliance.

A host product itself is required to comply with all other applicable FCC equipment authorization regulations, requirements, and equipment functions that are not associated with the transmitter module portion. For example, compliance must be demonstrated: to regulations for other transmitter components within a host product; to requirements for unintentional radiators (Part 15 Subpart B), such as digital devices, computer peripherals, radio receivers, etc.; and to additional authorization requirements for the non-transmitter functions on the transmitter module (i.e., SDoC or certification) as appropriate (e.g., Bluetooth and Wi-Fi transmitter modules may also contain digital logic functions).

31.1.1 Labeling And User Information Requirements

Due to the limited module size of ATSAMB11-ZR210CA (7.503 mm x10.541 mm), FCC identifier is displayed only in the datasheet and packaging box label. FCC identifier cannot be displayed on the module. When the module is installed inside another device, then the outside of the finished product into which the module is installed must display a label referring to the enclosed module. This exterior label can use wording as follows:

For the ATSAMB11-ZR210CA:

Contains Transmitter Module FCC ID: 2ADHKBZR

or

Contains FCC ID: 2ADHKBZR

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation

A user's manual for the finished product should include the following statement:

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy, and if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help

Additional information on labeling and user information requirements for Part 15 devices can be found in KDB Publication 784748, which is available at the FCC Office of Engineering and Technology (OET) Laboratory Division Knowledge Database (KDB) https://apps.fcc.gov/oetcf/kdb/index.cfm

31.1.2 RF Exposure

All transmitters regulated by FCC must comply with RF exposure requirements. KDB 447498 General RF Exposure Guidance provides guidance in determining whether proposed or existing transmitting facilities, operations or devices comply with limits for human exposure to Radio Frequency (RF) fields adopted by the Federal Communications Commission (FCC).

From the FCC Grant: Output power is conducted.

Module is approved for use in mixed mobile-device and portable-device exposure host platforms. The antenna(s) used with this transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.

31.1.3 Helpful Web Sites

Federal Communications Commission (FCC): http://www.fcc.gov

FCC Office of Engineering and Technology (OET) Laboratory Division Knowledge Database (KDB): https://apps.fcc.gov/oetcf/kdb/index.cfm

31.2 Canada

The ATSAMB11-ZR210CA module has been certified for use in Canada under Innovation, Science and Economic Development Canada (ISED, formerly Industry Canada) Radio Standards Procedure (RSP) RSP-100, Radio Standards Specification (RSS) RSS-Gen and RSS-247. Modular approval permits the installation of a module in a host device without the need to recertify the device.

31.2.1 Labeling and User Information Requirements

Labeling Requirements (from RSP-100 Issue 11, Section 3): The host product shall be properly labeled to identify the module within the host device.

Due to limited size of the ATSAMB11-ZR210CA (7.503 mm x10.541 mm), the Innovation, Science and Economic Development Canada certification number is not displayed on the module. Therefore, the host device must be labeled to display the Innovation, Science and Economic Development Canada certification number of the module, preceded by the words “Contains”, or similar wording expressing the same meaning, as follows:

For the ATSAMB11-ZR210CA:

Contains IC: 20266-SAMB11ZR

User Manual Notice for License-Exempt Radio Apparatus (from Section 8.4 RSS-Gen, Issue 4, November 2014): User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both:

This device complies with Industry Canada’s license exempt RSS standard(s). Operation is subject to the following two conditions:

(1) This device may not cause interference, and

(2) This device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes:

(1) l'appareil ne doit pas produire de brouillage, et

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

Guidelines on Transmitter Antenna for License Exempt Radio Apparatus:

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'Industrie Canada, le présent émetteur radio peut fonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvé pour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillage radioélectrique à l'intention des autres utilisateurs, il faut choisir le type d'antenne et son gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépasse pas l'intensité nécessaire à l'établisse-ment d'une communication satisfaisante.

31.2.2 RF Exposure

All transmitters regulated by Innovation, Science and Economic Development Canada (ISED) must comply with RF exposure requirements listed in RSS-102 - Radio Frequency (RF) Exposure Compliance of Radio communication Apparatus (All Frequency Bands).

This transmitter is restricted for use with a specific antenna tested in this application for certification, and must not be co-located or operating in conjunction with any other antenna or transmitters within a host device, except in accordance with Canada multi-transmitter product procedures.

The device operates at an output power level which is within the ISED SAR test exemption limits at any user distance.

31.2.3 Helpful Web Sites

Innovation, Science and Economic Development Canada: http://www.ic.gc.ca/

31.3 Europe

The ATSAMB11-ZR210CA module is a Radio Equipment Directive (RED) assessed radio module that is CE marked and has been manufactured and tested with the intention of being integrated into a final product.

The ATSAMB11-ZR210CA module has been tested to RED 2014/53/EU Essential Requirements for Health and Safety (Article (3.1(a)), Electromagnetic Compatibility (EMC) (Article 3.1(b)), and Radio (Article 3.2), which is summarized in Table 31-1.

The ETSI provides guidance on modular devices in the "*Guide to the application of harmonised standards covering articles 3.1b and 3.2 of the RED 2014/53/EU (RED) to multi-radio and combined radio and non-radio equipment*" document available at http://www.etsi.org/deliver/etsi_eg/203300_203399/203367/01.01.01_60/eg_203367v010101p.pdf.

Note: To maintain conformance to the testing listed in Table 31-1, the module shall be installed in accordance with the installation instructions in this data sheet and shall not be modified. When integrating a radio module into a completed product, the integrator becomes the manufacturer of the final product and is therefore responsible for demonstrating compliance of the final product with the essential requirements against the RED.

31.3.1 Labeling and User Information Requirements

The label on the final product that contains the ATSAMB11-ZR210CA module must follow CE marking requirements.

Table 31-1. European Compliance Testing (ATSAMB11-ZR210CA)

Certification	Standards	Article	Laboratory	Report Number	Date
Safety	EN60950-1:2006/A11:2009/ A1:2010/ A12:2011/A2:2013	[3.1(a)]	TUV Rheinland, Taiwan	11062248 001	2017-08-18
Health	EN 300 328 V2.1.1/ EN 62479:2010			50126888 001	2018-03-21
EMC	EN 301 489-1 V2.1.1 EN 301 489-1 V2.2.0	[3.1(b)]		10062088 001	2017-09-22
	EN 301 489-17 V3.1.1 EN 301 489-17 V3.2.0				
Radio	EN 300 328 V2.1.1	(3.2)		50126888 001	2018-03-21

31.3.2 Conformity Assessment

From ETSI Guidance Note EG 203367, section 6.1, when non-radio products are combined with a radio product:

If the manufacturer of the combined equipment installs the radio product in a host non-radio product in equivalent assessment conditions (i.e. host equivalent to the one used for the assessment of the radio product) and according to the installation instructions for the radio product, then no additional assessment of the combined equipment against article 3.2 of the RED is required.

The European Compliance Testing listed in the Table 31-1 is performed using the integral chip antenna.

31.3.2.1 Simplified EU Declaration of Conformity

Hereby, Microchip Technology Inc. declares that the radio equipment type ATSAMB11-ZR210CA is in compliance with Directive 2014/53/EU.

The full text of the EU declaration of conformity for this product is available at http://www.microchip.com/design-centers/wireless-connectivity/.

31.3.3 Helpful Web Sites

A document that can be used as a starting point in understanding the use of Short Range Devices (SRD) in Europe is the European Radio Communications Committee (ERC) Recommendation 70-03 E, which can be downloaded from the European Communications Committee (ECC) at: http://www.ecodocdb.dk/. Additional helpful web sites are:

- Radio Equipment Directive (2014/53/EU): : https://ec.europa.eu/growth/single-market/european-standards/harmonised-standards/rtte_en
- European Conference of Postal and Telecommunications Administrations (CEPT): http://www.cept.org
- European Telecommunications Standards Institute (ETSI): http://www.etsi.org
- The Radio Equipment Directive Compliance Association (REDCA) (Previously known as R&TTE Compliance Association): http://www.redca.eu

31.4 Japan

The ATSAMB11-ZR210CA module has received type certification and is labeled with its own technical conformity mark and certification number as required to conform to the technical standards regulated by the Ministry of Internal Affairs and Communications (MIC) of Japan pursuant to the Radio Act of Japan. Integration of this module into a final product does not require additional radio certification provided installation instructions are followed and no modifications of the module are allowed. Additional testing may be required:

- If the host product is subject to electrical appliance safety (for example, powered from an AC mains), the host product may require Product Safety Electrical Appliance and Material (PSE) testing. The integrator should contact their conformance laboratory to determine if this testing is required
- There is an voluntary Electromagnetic Compatibility (EMC) test for the host product administered by VCCI: http://www.vcci.jp/vcci_e/index.html

31.4.1 Labeling and User Information Requirements

The label on the final product which contains the ATSAMB11-ZR210CA module must follow Japan marking requirements. The integrator of the module should refer to the labeling requirements for Japan available at the Ministry of Internal Affairs and Communications (MIC) website. For the ATSAMB11-ZR210CA module, due to a limited module size, the technical conformity logo and ID is displayed in the data sheet and/or packaging label and cannot be displayed on the module. The final product in which this module is being used must have a label referring to the type certified module inside:

31.4.2 Helpful Web Sites

- Ministry of Internal Affairs and Communications (MIC): http://www.tele.soumu.go.jp/e/index.htm
- Association of Radio Industries and Businesses (ARIB): http://www.arib.or.jp/english/

31.5 Korea

The ATSAMB11-ZR210CA module has received certification of conformity in accordance with the Radio Waves Act. Integration of this module into a final product does not require additional radio certification provided installation instructions are followed and no modifications of the module are allowed.

31.5.1 Labeling and User Information Requirements

The label on the final product which contains the ATSAMB11-ZR210CA module must follow KC marking requirements. The integrator of the module should refer to the labeling requirements for Korea available on the Korea Communications Commission (KCC) website.

On the ATSAMB11-ZR210CA module, due to the limited module size the KC mark and identifier is displayed in the data sheet and/or packaging label, and cannot be displayed on the module label. The final product requires display of the KC mark and certificate number of the module:

31.5.2 Helpful Web Sites

- Korea Communications Commission (KCC): http://www.kcc.go.kr
- National Radio Research Agency (RRA): http://rra.go.kr

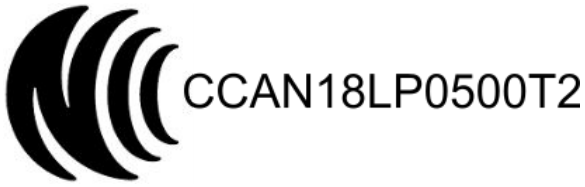
31.6 Taiwan

The ATSAMB11-ZR210CA module has received compliance approval in accordance with the Telecommunications Act. Customers seeking to use the compliance approval in their product should contact Microchip Technology Inc. sales or distribution partners to obtain a Letter of Authority.

Integration of this module into a final product does not require additional radio certification provided installation instructions are followed and no modifications of the module are allowed.

31.6.1 Labeling and User Information Requirements

For the ATSAMB11-ZR210CA module, due to the limited module size the NCC mark and ID are displayed in the data sheet only and cannot be displayed on the module:



The user's manual should contain the following warning (for RF device) in traditional Chinese:

注意！

依據 低功率電波輻射性電機管理辦法

第十二條 經型式認證合格之低功率射頻電機，非經許可，

公司、商號或使用者均不得擅自變更頻率、加大功率或變更原設計

之特性及功能。

第十四條 低功率射頻電機之使用不得影響飛航安全及干擾合法通信；

經發現有干擾現象時，應立即停用，並改善至無干擾時 方得繼續使用。

前項合法通信，指依電信規定作業之無線電信。

低功率射頻電機須忍受合法通信或工業、科學及醫療用 電波輻射性

電機設備之干擾。

31.6.2 Helpful Web Sites

- National Communications Commission (NCC): http://www.ncc.gov.tw

31.7 Other Regulatory Information

- For information on the other countries jurisdictions covered, refer to the http://www.microchip.com/design-centers/wireless-connectivity
- Should other regulatory jurisdiction certification be required by the customer, or the customer need to recertify the module for other reasons, contact Microchip for the required utilities and documentation

32. Reference Documents and Support

32.1 Reference Documents

Microchip offers a set of collateral documentation to ease integration and device ramp. The following table list documents available on Microchip website or integrated into development tools.

Table 32-1. Reference Documents

Title	Content
ATSAMB11 BluSDK Smart Release Package	This package contains the software development kit and all the necessary documentation including getting started guides for interacting with different hardware devices, tools and API user manual.
BluSDK Smart BLE API Software Development Guide	This user guide details the functional description of Bluetooth Low Energy (BLE) Application Peripheral Interface (API) programming model. This also provides the example code to configure an API for Generic Access Profile (GAP), Generic Attribute (GATT) Profile, and other services using the ATSAMB11.
ATSAMB11 BluSDK SMART OTAU Profile Getting Started Guide	This document describes how to set the evaluation board for the Bluetooth Low Energy Over-the-Air Upgrade (OTAU) application supported by the ASF.
ATSAMB11 BluSDK SMART Interrupts and ULP - Architecture and Usage User's Guide	This document details the design and usage scenarios for the Atmel® ATSAMB11 peripheral interrupts and ULP feature
ATSAMB11 BluSDK SMART Example Profiles Application User Guide	This document describes how to set up the evaluation boards for various example applications supported by the Advanced Software Framework (ASF).
Ultra Low Power BLE SiP/ Module Errata	Errata document capturing the known issues with the ATSAMB11-XR2100A SiP

For a complete listing of development support tools and documentation, visit http://www.microchip.com, or contact the nearest microchip field representative.

33. Document Revision History

Revision	Date	Section	Description
D	03/2019	Sections 7.5, 7.5.2	• Updated the bit map of Bank 5 Block 0. • Added description for bit 31 and 30 of Bank 5 Block 0. • Updated Bluetooth low energy version as 5.0
C	08/2018	Sections 9.8, 16, 17, 18, 19, 20, 21, 22, 23, 24 29, 30, 31	• Added detailed information and register descriptions for various peripherals such as, Dual Timer, ARM Timer, SPI, UART Interface, SPI Flash Controller, Three-axis Quadrature Decoder, ADC, DMA, Clock Output and WDT. • Updated Section 29.1 with enhanced guidelines for module placement and figures. • Updated Section 30 with additional reference and generic recommendations. • Updated Section 31 with regulatory certification IDs and notices for Japan, Korea and Taiwan

.....continued

Revision	Date	Section	Description
B	03/2018	Sections 9, 10, 11, 12, 13, 14 and 15	• Added register descriptions for BLE Clock. • Updated and added register descriptions for I/O Peripheral Multiplexing and MEGAMUXing. • Added detailed information and register descriptions for various peripherals such as, muxable interrupts, GPIO pin controller, Always-On sleep timer, Pulse Width Modulation, and I^2C interface.
A	09/2017	Document	• Updated from Atmel to Microchip template. • Assigned a new Microchip document number. Previous version is Atmel 42751 revision A. • ISBN number added.

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-4244-8

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-67-3636 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Bluetooth Modules - 802.15.1 ***category:***

Click to view products by Microchip ***manufacturer:***

Other Similar products are found below :

A2541R24A10GM CYBLE-212023-10 BM78SPP05NC2-0002AA BM78SPP05MC2-0002AA CYW20732S 968EMB0019 E73-2G4M08S1CX TB-03F-AT_Mesh TB-04 TB-04--AT_Mesh BT3L(jibu) BT5S(xoft) BT5S(4k43) BT5S(jcyv) 1327 RN42HID-I/RM ENW-89829C3KF BLE113-A-V1 BM70BLE01FC2-0B03AA ACN52832 A2541E24A10GM RN42-I/RM630 MOTG-BLUETOOTH ABBTM-2.4GHz-52-T ABBTM-2.4GHz-T ABBTM-2.4GHz-T2 ACN52840 AFERO-BL24-01 BLED112 BM62SPKS1MC2-0001AA BM78SPPS5MC2-0002AA PX0880/1 DAT12 BM833F BT680F PBA31309V1.00 S LK64 ISP1907-LL-ST ATSAMB11-MR510CA BM20SPKA1NBC-0001AA BM20SPKS1NBC-0001AA BM23SPKS1NB9-0B02AA BM64SPKS1MC2-0002AA BM70BLE01FC2-0B04AA BM70BLES1FC2-0B05BA BM70BLE01FC2-0B05BA BM70BLES1FC2-0B04AA BM77SPP03MC2-0007AA BM77SPP03MC2-0008AA BM78SPPS5NC2-0002AA DM164146