zilog

Embedded in Life

An IXYS Company

## ZNEO32! Family of Microcontrollers

# Z32F384 MCU

## Product Specification

PS034602-0316

PRELIMINARY

ZNEO32!

32 Bit Microcontrollers



www.zilog.com

**Warning:** DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

## LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

### As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

### Document Disclaimer

 Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

ZNEO32! is a trademark or registered trademark of Zilog, Inc. All other product or service names are the property of their respective owners.

# Revision History

Each instance in this document's revision history reflects a change from its previous edition. For more details, refer to the corresponding page(s) or appropriate links furnished in the table below.

| Date | Revision Level | Description | Page |
|---|---|---|---|
| Mar 2016 | 02 | Updated to reflect new part, revision B (0x0002);<br>Added timing information for most of the peripherals; corrected typos. | All |
| Dec 2015 | 01 | Original issue. | |

# 1. Overview

## Introduction

Zilog's Z32F384 MCU, a member of the ZNEO32! Family of microcontrollers, is a cost-effective and high-performance 32-bit microcontroller. The Z32F384 MCU provides 3-phase PWM generator units which are suitable for inverter bridges, including motor drive systems. Two built-in channels of these generators control two inverter bridges simultaneously.

Two 12-bit high speed ADC units with 16-channel analog multiplexed inputs support feedback retrieval from the inverter bridge. The Z32F384 MCU can control up to two inverter motors or one inverter motor and the Power Factor Correction (PFC) function simultaneously.

Figure 1.1 shows a block diagram of the Z32F384 MCU.

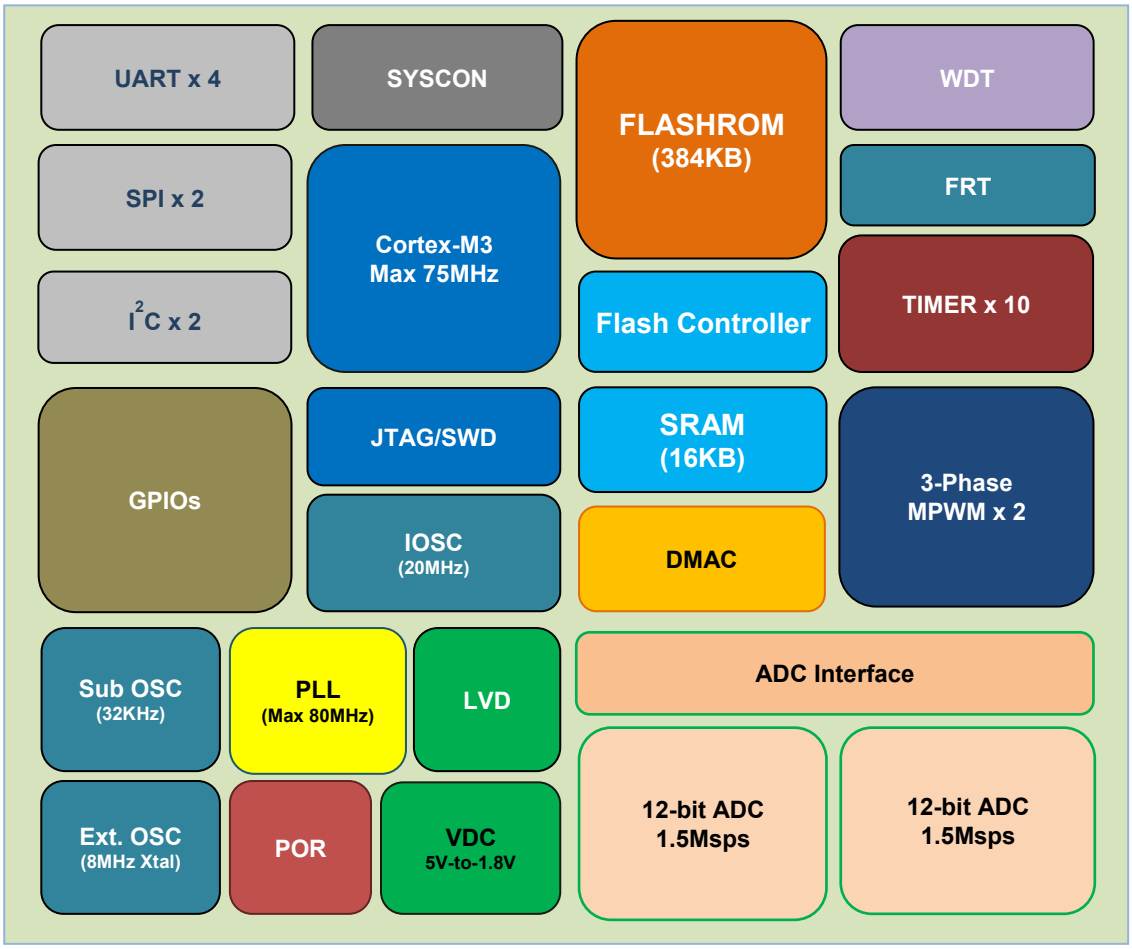



**Figure 1.1. Block Diagram**

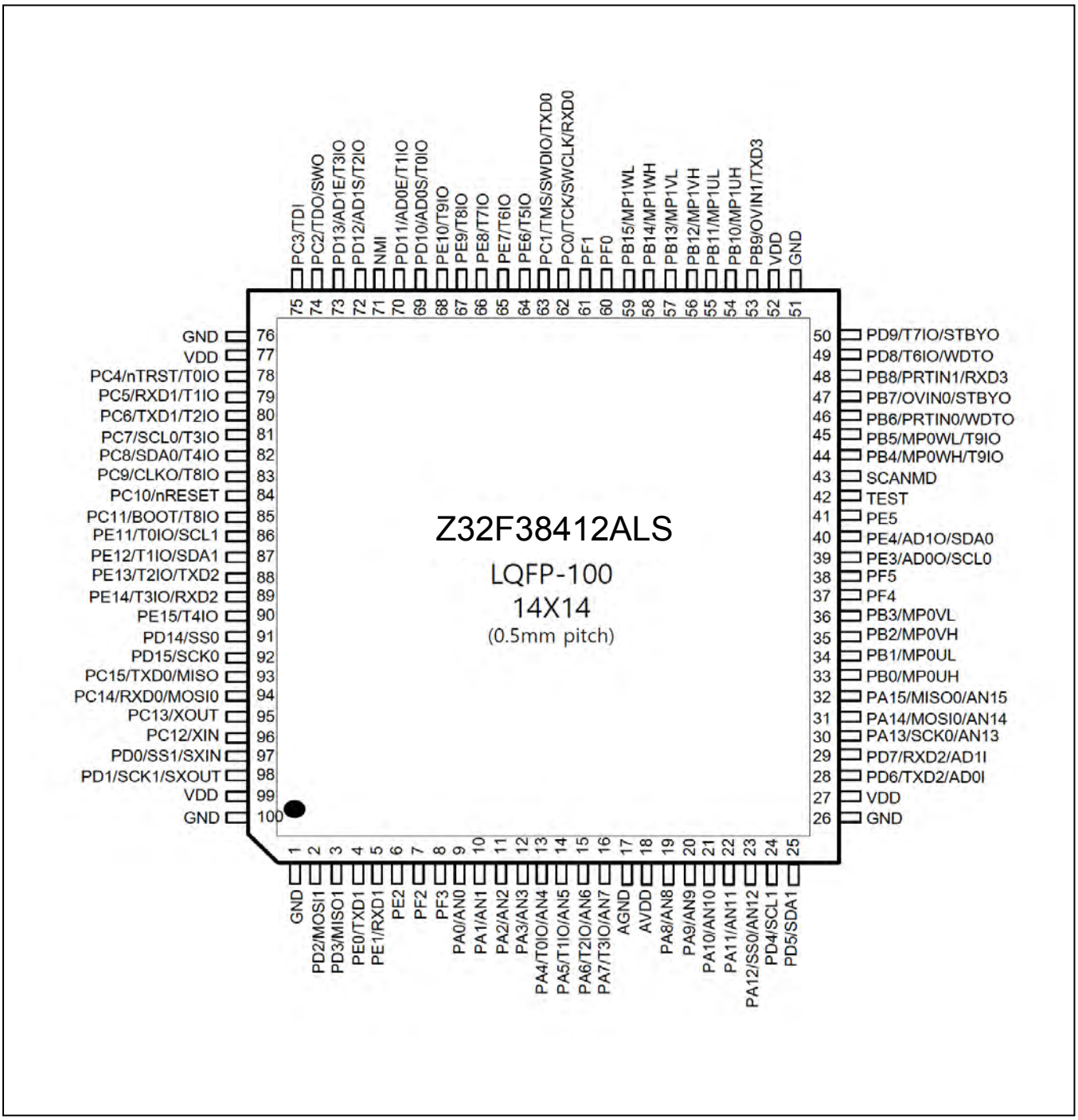



**Figure 1.2. Pin Layout (LQFP-100)**

# Product Features

The Z32F384 MCU includes the following features:

- High performance low-power Cortex-M3 core
- 384KB code Flash memory with cache function
- 16 KB SRAM
- 3-Phase PWM with ADC triggering function
  - 2 Channels
- 1.5 MSPS high-speed ADC with sequential conversion
  - 2 units with 16 Channel input
- System fail-safe function by clock monitoring
  - XTAL OSC fail monitoring function
  - System clock Fail monitoring function
- Internal clock sources
  - Internal ring oscillator (1 MHz ±50%)
  - Internal oscillator clock (20 MHz ±3%)
  - Internal Phase Lock Loop (PLL) up to 80Mhz
- External clock sources
  - External crystal oscillator (4~16 MHz)
  - External sub oscillator (32 kHz)
- Watchdog timer
- 10 general purpose timer channels
  - Timer/capture/PWM mode
- Free run timer
- Various external communication ports:
  - 4 UARTs
  - 2 $I^2C$s
  - 2 SPIs
- High current driving port for UART photo couplers
- Direct Memory Access (DMA) controller with 8 channels
- Debug and emergency stop function
- JTAG and SWD debugger
- Package: LQFP-100 (0.5mm pitch)
- Industrial grade operating temperature (– 40 ~ +85°C)

# Architecture

## Block Diagram

Figure 1.3 shows the Z32F384 MCU's internal block diagram.

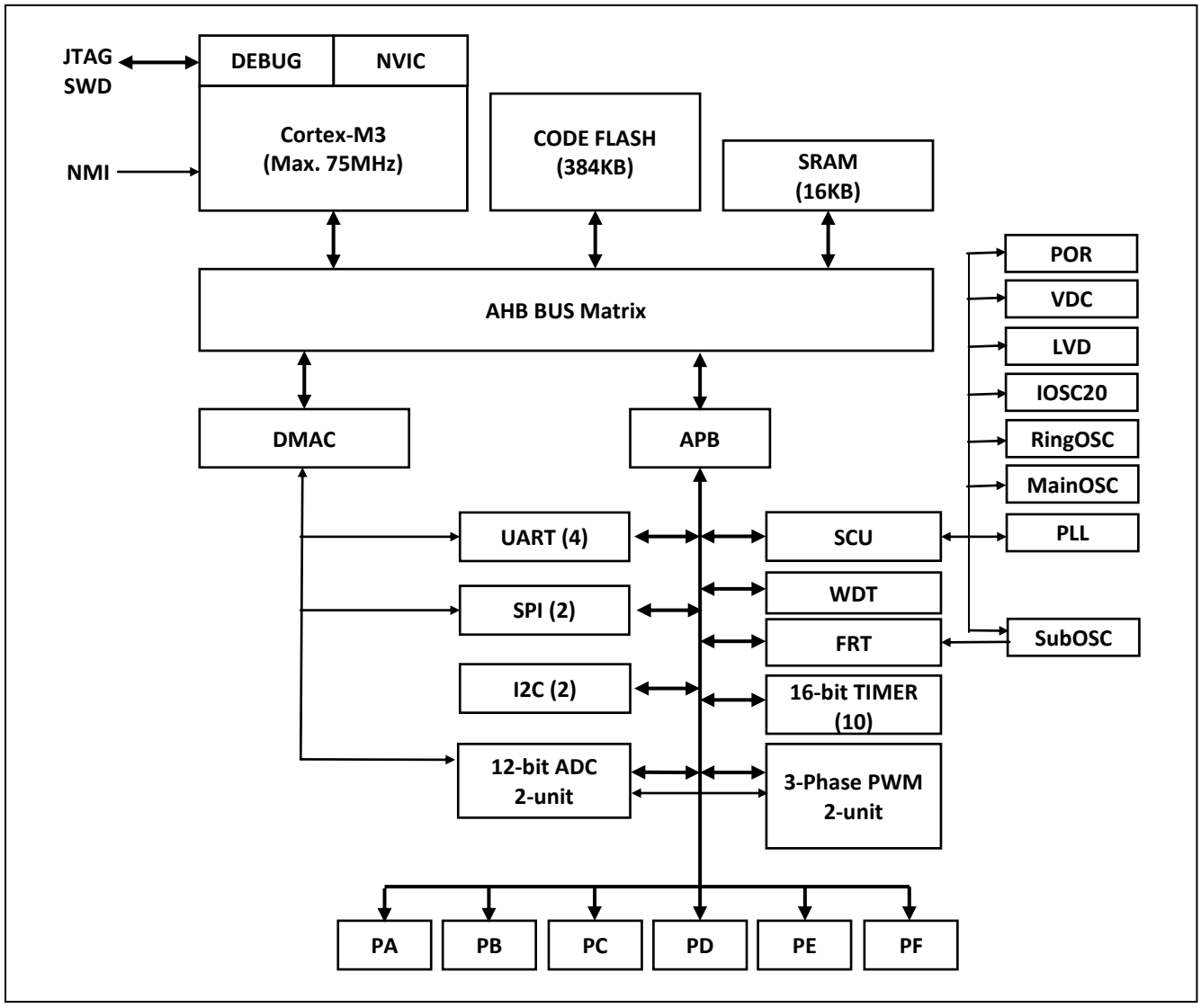



**Figure1.3. Internal Block Diagram**

# Functional Description

The following section provides an overview of the features of the Z32F384 microcontroller.

### ARM Cortex-M3

The ARM-powered Cortex-M3 Core based on v7M architecture is optimized for small size and low power system. An on core system timer (SYSTICK) provides a simple 24-bit timer, that enables easy management of the system operation. The thumb-compatible Thumb-2 only instruction set processor core makes code high-density. Hardware division and single-cycle multiplication is present. Integrated Nested Vectored Interrupt Controller (NVIC) provides deterministic interrupt handling. Full featured debug solutions are provided – JTAG and SWD, FPB, DWT, ITM and TPIU. It includes a maximum 72 MHz operating frequency with zero wait execution.

### Nested Vector-Interrupt Controller (NVIC)

The ARM Nested Vectored Interrupt Controller (NVIC) on the ARM Cortex-M3 core is included, which handles all internal and external exceptions. When an interrupt condition is detected, the processor state is automatically stored to the stack and automatically restored from the stack at the end of the interrupt service routine. The vector is fetched in parallel to the state saving, which enables efficient interrupt entry. The processor supports tail-chaining, which allows back-to-back interrupts to be performed without the overhead of state saving and restoring.

### 384 KB Internal Code Flash Memory

The Z32F384 MCU provides internal 384 KB code Flash memory and its controller. This is enough to program the motor algorithm and generally control the system. Self-programming is available and ISP and JTAG programming is also supported in boot or debugging mode.

Instruction and data cache buffer are present and overcome the low bandwidth Flash memory. The CPU can execute from Flash memory with zero wait state up to 72 MHz bus frequency.

### 16 KB Zero-wait Internal SRAM

On chip 16 KB zero-wait SRAM can be used for working memory space and program code can be loaded on this SRAM.

### Boot Logic

The smart boot logic supports Flash programming. The Z32F384 MCU can be accessed by an external boot pin and UART and SPI programming are available in boot mode.

### System Control Unit (SCU)

The SCU block manages internal power, clock, reset, and operation mode. It also controls analog blocks (INTOSC, VDC and BOD).

### 32-bit Watchdog Timer (WDT)

The watchdog timer performs the system monitoring function. It generates an internal reset or interrupt when it notices abnormal status of the system.

### Multi-purpose 16-bit Timer

10 16-bit general purpose timers channels support the following functions:

- Periodic timer mode
- Counter mode
- PWM mode
- Capture mode

### Free Run Timer

The 32-bit Free run timer has multiple clock sources (XTAL/16, IOSC/16, SXTAL).

### Motor PWM Generator

Two channels of the 3-phase PWM generator are implemented. A 16-bit up/down counter with prescaler

supports both the triangular and saw tooth waveform.

The PWM generates an internal ADC trigger signal to measure the signal on time. Dead time insertion and emergency stop functionality ensure that the chip and system operate under safe conditions.

**Serial Peripheral Interface (SPI)**

Synchronous serial communication is provided with the SPI block. The Z32F384 MCU has 2-channel SPI modules. It includes a DMA function supported by the DMA controller. Transfer data is moved to/from the memory area without CPU operation. Boot mode uses this SPI block to download the Flash program.

**Inter-Integrated Circuit Interface ($I^2C$)**

The Z32F384 MCU has a 2-channel $I^2C$ block and it supports up to 400 KHz $I^2C$ communication. Master and slave modes are supported.

**Universal Asynchronous Receiver/Transmitter (UART)**

The Z32F384 MCU has a 4-channel UART block. For accurate baud rate control, the fractional baud rate generator is provided. It includes a DMA function supported by the DMA controller. Transfer data is moved to/from the memory area without CPU operation.

**General PORT I/Os**

16-bit PA, PB, PC, PD, PE ports and 6-bit PF are available and provide multiple functionality:

- General I/O port
- Independent bit set/clear function
- External interrupt input port
- Pull-up/open-drain
- On chip debounce filter

**12-bit Analog-to-Digital Converter (ADC)**

2 built-in ADCs can convert analog signals up to 1usec conversion rate. A 16-channel analog mux provides various combinations from external analog signals.

# Pin Description

The pin configurations are shown in Table 1.1. 16 pins are reserved for power/ground pair and dedicated pins.

**Table1.1. Pin Description**

| Pin No | Pin Name | Type | Description | Remark |
|---|---|---|---|---|
| 1 | GND | P | Ground | |
| 2 | PD2 | IOUS | PORT D Bit 2 Input/Output | |
| | MOSI1 | I/O | SPI Channel 1 Master Out / Slave In | |
| 3 | PD3* | IOUS | PORT D Bit 3 Input/Output | |
| | MISO1 | I/O | SPI Channel 1 Master In / Slave Out | |
| 4 | PE0 | IOUS | PORT E Bit 0 input/Output | |
| | TXD1 | Output | UART Channel 1 TXD output | 2nd function |
| 5 | PE1 | IOUS | PORT E Bit 1 input/Output | |
| | RXD1 | Input | UART Channel 1 RXD input | 2nd function |
| 6 | PE2 | IOUS | PORT E Bit 2 input/Output | |
| | T4I | I/O | Timer 4 Input | 2nd function |
| | T3O | I/O | Timer 3 Output | 2nd function |
| 7 | PF2 | IOUS | PORT F Bit 2 input/Output | |
| | AN20 | IA | Analog Input 20 | |
| 8 | PF3 | IOUS | PORT F Bit 3 input/Output | |
| | AN21 | IA | Analog Input 21 | |
| 9 | PA0* | IOUS | PORT A Bit 0 Input/Output | |
| | AN0 | IA | Analog Input 0 | |
| 10 | PA1* | IOUS | PORT A Bit 1 Input/Output | |
| | AN1 | IA | Analog Input1 | |
| 11 | PA2* | IOUS | PORT A Bit 2 Input/Output | |
| | AN2 | IA | Analog Input 2 | |
| 12 | PA3* | IOUS | PORT A Bit 3 Input/Output | |
| | AN3 | IA | Analog Input 3 | |
| 13 | PA4* | IOUS | PORT A Bit 4 Input/Output | |
| | T0IO | IO | Timer 0 Input/Output | 3rd function |
| | AN4 | IA | Analog Input 4 | |
| 14 | PA5* | IOUS | PORT A Bit 5 Input/Output | |
| | T1IO | IO | Timer 1 Input/Output | 3rd function |
| | AN5 | IA | Analog Input 5 | |
| 15 | PA6* | IOUS | PORT A Bit 6 Input/Output | |
| | T2IO | IO | Timer 2 Input/Output | 3rd function |
| | AN6 | IA | Analog Input 6 | |
| 16 | PA7* | IOUS | PORT A Bit 7 Input/Output | |
| | T3IO | IO | Timer 3 Input/Output | 3rd function |
| | AN7 | IA | Analog Input 7 | |
| 17 | AGND | P | Analog Ground | |
| 18 | AVDD | P | Analog VDD | |
| 19 | PA8* | IOUS | PORT A Bit 8 Input/Output | |
| | AN8 | IA | Analog Input 8 | |
| 20 | PA9* | IOUS | PORT A Bit 9 Input/Output | |
| | AN9 | IA | Analog Input 9 | |

| | | | | |
|---|---|---|---|---|
| 21 | PA10* | IOUS | PORT A Bit 10 Input/Output | |
| | AN10 | IA | Analog Input 10 | |
| 22 | PA11* | IOUS | PORT A Bit 10 Input/Output | |
| | AN11 | IA | Analog Input 11 | |
| 23 | PA12* | IOUS | PORT A Bit 12 Input/Output | |
| | SS0 | I/O | SPI Channel 0 Slave Select signal | |
| | AN12 | IA | Analog Input 12 | |
| 24 | PD4 | IOUS | PORT D Bit 4 Input/Output | |
| | SCL1 | Output | I2C Channel 1 SCL In/Out | Open-drain |
| | AN16 | IA | Analog Input 16 | |
| 25 | PD5 | IOUS | PORT D Bit 5 Input/Ouput | |
| | SDA1 | Output | I2C Channel 1 SDA In/Out | Open-drain |
| | AN17 | IA | Analog Input 17 | |
| 26 | GND | P | Ground | |
| 27 | VDD | P | VDD | |
| 28 | PD6* | IOUS | PORT D Bit 6 Input/Ouput | |
| | TXD2 | Output | UART Channel 2 TxD Output | |
| | AN18 | IA | Analog Input 18 | |
| 29 | PD7* | IOUS | PORT D Bit 7 Input/Ouput | |
| | RXD2 | Input | UART Channel 2 RxD Input | |
| | AN19 | IA | Analog Input 19 | |
| 30 | PA13* | IOUS | PORT A Bit 13 Input/Output | |
| | SCK0 | I/O | SPI Channel 0 Clock Input/Output | |
| | AN13 | IA | Analog Input 13 | |
| 31 | PA14* | IOUS | PORT A Bit 14 Input/Output | |
| | MOSI0 | I/O | SPI Channel 0 Output(M)/Input(S) Data signal | |
| | AN14 | IA | Analog Input 14 | |
| 32 | PA15* | IOUS | PORT A Bit 15 Input/Output | |
| | MISO0 | I/O | SPI Channel 0 Input(M)/Output(S) Data signal | |
| | AN15 | IA | Analog Input 15 | |
| 33 | PB0 | IOUS | PORT B Bit 0 Input/Output | |
| | MP0UH | Output | PWM0 UH Output | |
| 34 | PB1 | IOUS | PORT B Bit 1 Input/Output | |
| | MP0UL | Output | PWM0 UL Output | |
| 35 | PB2 | IOUS | PORT B Bit 0 Input/Output | |
| | MP0VH | Output | PWM0 VH Output | |
| 36 | PB3 | IOUS | PORT B Bit 1 Input/Output | |
| | MP0VL | Output | PWM0 VL Output | |
| 37 | PF4 | IOUS | PORT F Bit 4 Input/Output | |
| 38 | PF5 | IOUS | PORT F Bit 5 Input/Output | |
| 39 | PE3 | IOUS | PORT E Bit 3 Input/Output | |
| | SCL0 | Output | I2C Channel 0 SCL In/Out | Open-drain<br>2nd function |
| 40 | PE4 | IOUS | PORT E Bit 4 Input/Output | |
| | SDA0 | Output | I2C Channel 0 SDA In/Out | Open-drain<br>2nd function |
| 41 | PE5 | IOUS | PORT E Bit 5 Input/Output | |
| | T5IO | I/O | Timer 5 Input/Output | |
| 42 | TEST | Input | Test-mode Input (Always 'L') | Pull-down |
| 43 | SCANMD | Input | Scan-mode Input (Always 'L') | Pull-down |

| | | | | |
|---|---|---|---|---|
| 44 | PB4 | IOUS | PORT B Bit 4 Input/Output | |
| | MP0WH | Output | PWM0 WH Output | |
| | T9IO | I/O | Timer 9 Input/Output | |
| 45 | PB5 | IOUS | PORT B Bit 5 Input/Output | |
| | MP0WL | Output | PWM0 WL Output | |
| | T9IO | I/O | Timer 9 Input/Output | 2nd function |
| 46 | PB6 | IOUS | PORT B Bit 6 Input/Output | |
| | PRTIN0 | Input | PWM0 Protection Input signal 0 | |
| | WDTO | Output | WDT Output | |
| 47 | PB7 | IOUS | PORT B Bit 7 Input/Output | |
| | OVIN0 | Input | PWM0 Over-Current Input signal 1 | |
| | STBYO | Output | Power-down mode indication signal | |
| 48 | PB8 | IOUS | PORT B Bit 8 Input/Output | |
| | PRTIN1 | Input | PWM1 Protection Input signal 0 | |
| | RXD3 | Output | UART Channel 3 RXD Input | |
| 49 | PD8 | IOUS | PORT D Bit 8 Input/Output | |
| | WDTO | Output | WDT Output | 2nd function |
| | T6IO | I/O | Timer 6 Input/Output | |
| 50 | PD9 | IOUS | PORT D Bit 9 Input/Output | |
| | STBYO | Output | Power-down mode indication signal | 2nd function |
| | T7IO | I/O | Timer 7 Input/Output | |
| 51 | GND | P | Ground | |
| 52 | VDD | P | VDD | |
| 53 | PB9 | IOUS | PORT B Bit 9 Input/Output | |
| | OVIN1 | Input | PWM1 Over-Current Input signal 1 | |
| | TXD3 | Output | UART Channel 3 TXD Output | |
| 54 | PB10 | IOUS | PORT B Bit 10 Input/Output | |
| | MP1UH | Output | PWM Channel 1 UH Output | |
| 55 | PB11 | IOUS | PORT B Bit 11 Input/Output | |
| | MP1UL | Output | PWM Channel 1 UL Output | |
| 56 | PB12 | IOUS | PORT B Bit 12 Input/Output | |
| | MP1VH | Output | PWM Channel 1 VH Output | |
| 57 | PB13 | IOUS | PORT B Bit 13 Input/Output | |
| | MP1VL | Output | PWM Channel 1 VL Output | |
| 58 | PB14 | IOUS | PORT B Bit 14 Input/Output | |
| | MP1WH | Output | PWM Channel 1 WH Output | |
| 59 | PB15 | IOUS | PORT B Bit 15 Input/Output | |
| | MP1WL | Output | PWM Channel 1 WL Output | |
| 60 | PF0 | IOUS | PORT F Bit 0 Input/Output | |
| 61 | PF1 | IOUS | PORT F Bit 1 Input/Output | |
| 62 | PC0 | IOUS | PORT C Bit 0 Input/Output | |
| | TCK/SWCLK | Input | JTAG TCK, SWD Clock Input | |
| | RXD0 | Input | UART Channel 0 RXD Input | 2nd function |
| 63 | PC1 | IOUS | PORT C Bit 1 Input/Output | |
| | TMS/SWDIO | I/O | JTAG TMS, SWD Data Input/Output | |
| | TXD0 | Input | UART Channel 0 TXD Output | 2nd function |
| 64 | PE6 | IOUS | PORT E Bit 6 Input/Output | |
| | T5IO | I/O | Timer 5 Input/Output | 2nd function |
| 65 | PE7 | IOUS | PORT E Bit 7 Input/Output | |

| | | | | |
|---|---|---|---|---|
| | T6IO | I/O | Timer 6 Input/Output | 2nd function |
| 66 | PE8 | IOUS | PORT E Bit 8 Input/Output | |
| | T7IO | I/O | Timer 7 Input/Output | 2nd function |
| 67 | PE9 | IOUS | PORT E Bit 9 Input/Output | |
| | T8IO | I/O | Timer 8 Input/Output | 3rd function |
| 68 | PE10 | IOUS | PORT E Bit 10 Input/Output | |
| | T9IO | I/O | Timer 9 Input/Output | 3rd function |
| 69 | PD10 | IOUS | PORT D Bit 10 Input/Output | |
| | AD0SOC | Output | ADC0 Start-of-Conversion | |
| | T0IO | IO | Timer 0 Input/Output | 3rd function |
| 70 | PD11 | IOUS | PORT D Bit 10 Input/Output | |
| | AD0EOC | Output | ADC0 End-of-Conversion | |
| | T1IO | IO | Timer 1 Input/Output | 3rd function |
| 71 | NMI | Input | Non-maskable Interrupt Input | |
| 72 | PD12 | IOUS | PORT D Bit 12 Input/Output | |
| | AD1SOC | Output | ADC1 Start-of-Conversion | |
| | T2IO | IO | Timer 2 Input/Output | 3rd function |
| 73 | PD13 | IOUS | PORT D Bit 13 Input/Output | |
| | AD1EOC | Output | ADC1 End-of-Conversion | |
| | T3IO | IO | Timer 3 Input/Output | 3rd function |
| 74 | PC2 | IOUS | PORT C Bit 2 Input/Output | |
| | TDO/SWO | Output | JTAG TDO, SWO Output | |
| 75 | PC3 | IOUS | PORT C Bit 3 Input/Output | |
| | TDI | Input | JTAG TDI Input | |
| 76 | GND | P | Ground | |
| 77 | VDD | P | VDD | |
| 78 | PC4 | IOUS | PORT C Bit 4Input/Output | |
| | nTRST | Input | JTAG nTRST Input | |
| | T0IO | IO | Timer 0 Input/Output | 2nd function |
| 79 | PC5 | IOUS | PORT C Bit 5Input/Output | |
| | RXD1 | Input | UART Channel 1 RXD Input | |
| | T1IO | IO | Timer 1 Input/Output | 2nd function |
| 80 | PC6 | IOUS | PORT C Bit 6Input/Output | |
| | TXD1 | Output | UART Channel 1 TXD Output | |
| | T2IO | IO | Timer 2 Input/Output | 2nd function |
| 81 | PC7 | IOUS | PORT C Bit 7Input/Output | |
| | SCL0 | IO | I2C Channel 0 SCL In/Out | |
| | T3IO | IO | Timer 3 Input/Output | 2nd function |
| 82 | PC8 | IOUS | PORT C Bit 8 Input/Output | |
| | SDA0 | IO | I2C Channel 0 SDA In/Out | |
| | T4IO | IO | Timer 4 Input/Output | |
| 83 | PC9 | IOUS | PORT C Bit 9 Input/Output | |
| | CLKO | Output | System Clock Output | |
| | T8IO | IO | Timer 8 Input/Output | |
| 84 | PC10 | IOUS | PORT C Bit 10 Input/Output | |
| | nRESET | Input | External Reset Input | Pull-up |
| 85 | PC11 | IOUS | PORT C Bit 11 Input/Output | |
| | BOOT | Input | Boot mode Selection Input | |
| | T8IO | Input | Timer 8 Input/Output | 2nd function |
| 86 | PE11 | IOUS | PORT E Bit 11 Input/Output | |
| | T0IO | IO | Timer 0 Input/Output/Phase-A Input | 4st function |

| | | | | |
|---|---|---|---|---|
| | SCL1 | IO | I2C Channel 1 SCL Input/Output | 2nd function |
| 87 | PE12 | IOUS | PORT E Bit 12 Input/Output | |
| | T1IO | IO | Timer 1 Input/Output/Phase-B Input | 4st function |
| | SDA1 | IO | I2C Channel 1 SDA input/Output | 2nd function |
| 88 | PE13 | IOUS | PORT E Bit 13 Input/Output | |
| | T2IO | IO | Timer 2 Input/Output/Phase-Z Input | 4st function |
| | TXD2 | Output | UART Channel 2 TXD Output | 2nd function |
| 89 | PE14 | IOUS | PORT E Bit 14 Input/Output | |
| | T3IO | IO | Timer 3 Input/Output | 4st function |
| | RXD2 | Input | UART Channel 2 RXD Input | 2nd function |
| 90 | PE15 | IOUS | PORT E Bit 15 Input/Output | |
| | T4IO | IO | Timer 4 Input/Output | 2nd function |
| 91 | PD14 | IOUS | PORT D Bit 14 Input/Output | |
| | SS0 | IO | SPI Channel 0 Slave Select signal | 2nd function |
| 92 | PD15 | IOUS | PORT D Bit 15 Input/Output | |
| | SCK0 | IO | SPI Channel 0 Clock Input/Output | 2nd function |
| 93 | PC15 | IOUS | PORT C Bit 14 Input/Output | |
| | TXD0 | Output | UART Channel 0 TXD Output | |
| | MISO0 | I/O | SPI Channel 0 Input(M)/Output(S) | 2nd function |
| 94 | PC14 | IOUS | PORT C Bit 14 Input/Output | |
| | RXD0 | Input | UART0 RXD Input | |
| | MOSI0 | I/O | SPI Channel 0 Output(M)/Input(S) | 2nd function |
| | VMARGIN | OA | Not used. (test purpose) | |
| 95 | PC13 | IOUS | PORT C Bit 13 Input/Output | |
| | XOUT | OA | External Crystal Oscillator Output | |
| 96 | PC12 | IOUS | PORT C Bit 12 Input/Output | |
| | XIN | IA | External Crystal Oscillator Input | |
| 97 | PD0 | IOUS | PORT D Bit 0 Input/Output | |
| | SS1 | I/O | SPI Channel 1 Slave Select signal | |
| | SXIN | IA | Sub Crystal Oscillator Input | |
| 98 | PD1 | IOUS | PORT D Bit 1 Input/Output | |
| | SCK1 | I/O | SPI Channel 1 Clock Input/Output | |
| | SXOUT | OA | Sub Crystal Oscillator Output | |
| 99 | VDD | P | VDD | |
| 100 | GND | P | Ground | |

*Notation: I=Input, O=Output, U=Pull-up, D=Pull-down,
S=Schmitt-Trigger Input Type, C=CMOS Input Type, A=Analog, P=Power
(*) Selected pin function after reset condition
Pin order may be changed with revision notice

# Memory Map

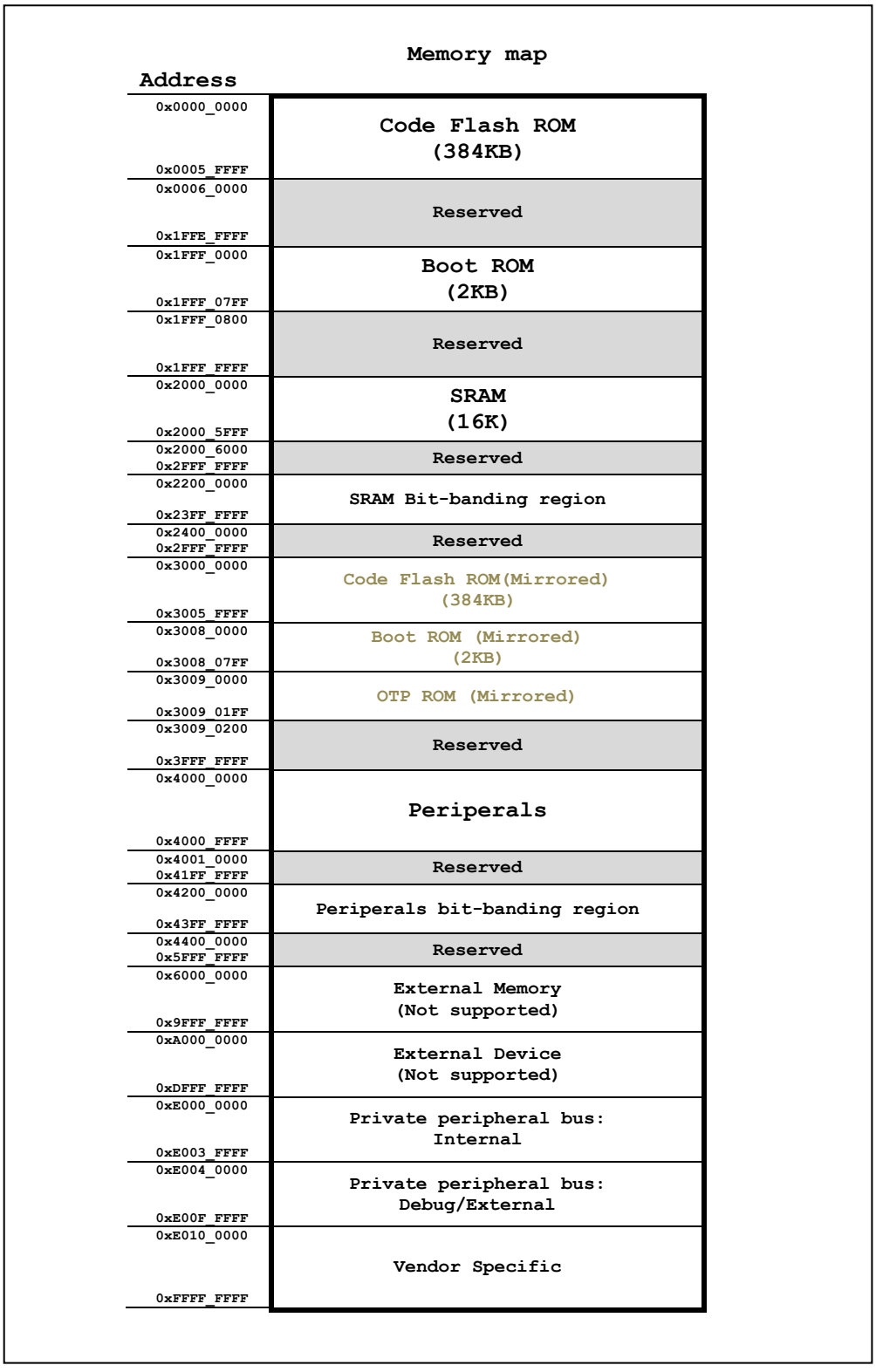



**Figure1.4. Main Memory Map**

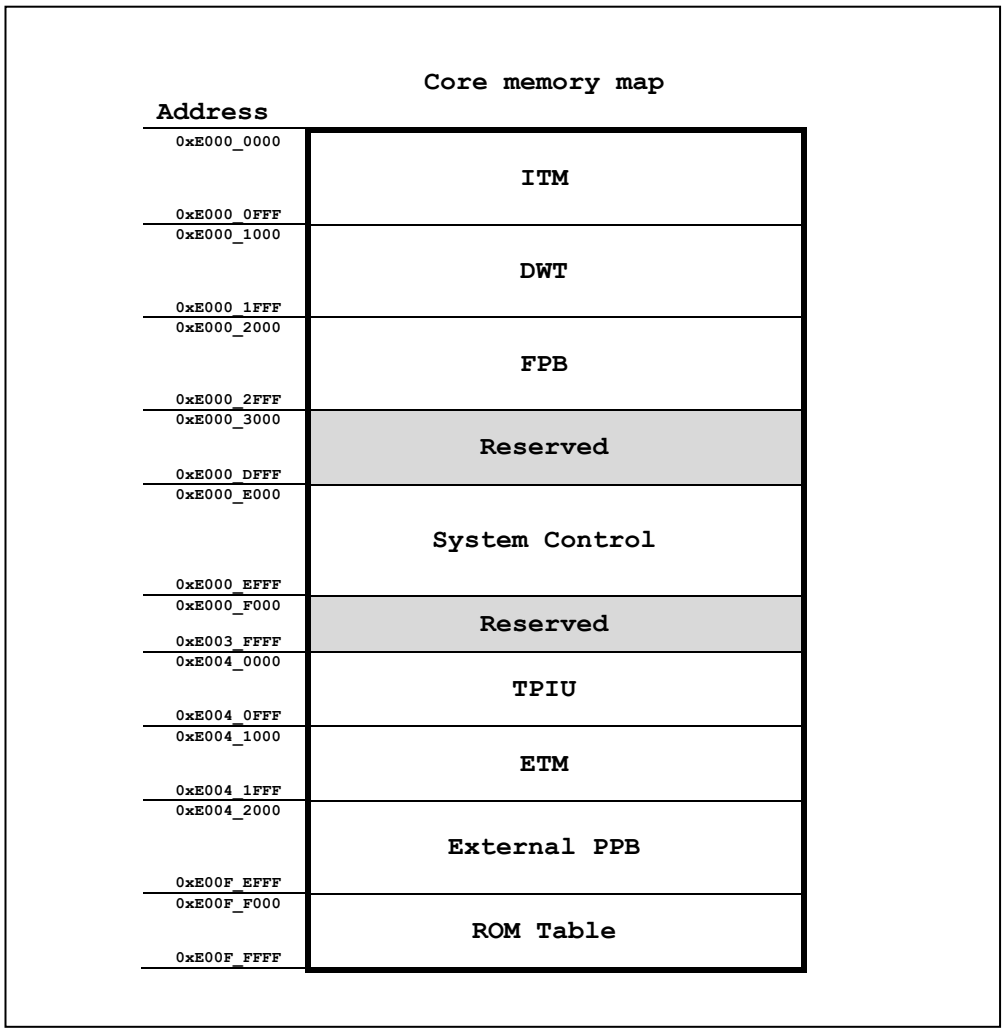



**Figure1.5. Cortex-M3 Private Memory Map**

**Note:** Please see document number DDI337 from ARM for more information about the Cortex-M3 memory map.

| Address | Peripheral map |
|---|---|
| 0x4000_0000 | SCU |
| 0x4000_0100 | FMC |
| 0x4000_0200 | WDT |
| 0x4000_0300 | Reserved |
| 0x4000_0400 | DMAC(8) |
| 0x4000_0500 | Reserved |
| 0x4000_0600 | FRT |
| 0x4000_1000 | PCU |
| 0x4000_2000 | GPIO(A,B,C,D,E,F) |
| 0x4000_3000 | TIMER |
| 0x4000_4000 | MPWM0 |
| 0x4000_5000 | MPWM1 |
| 0x4000_6000 | Reserved |
| 0x4000_8000 | UART0 |
| 0x4000_8100 | UART1 |
| 0x4000_8200 | UART2 |
| 0x4000_8300 | UART3 |
| 0x4000_8600 | Reserved |
| 0x4000_9000 | SPI0 |
| 0x4000_9100 | SPI1 |
| 0x4000_9200 | Reserved |
| 0x4000_A000 | $I^2C0$ |
| 0x4000_A100 | $I^2C1$ |
| 0x4000_A200 | Reserved |
| 0z4000_B000 | ADC0 |
| 0x4000_B100 | ADC1 |
| 0x4000_B200 | Reserved |
| 0x4000_B300 | Reserved |
| 0x4000_B400<br>0x4000_FFFF | Reserved |

Figure1.6. Peripheral Memory Map

# 2. CPU

## Cortex-M3 Core

The CPU core is supported from the ARM Cortex-M3 processor which provides a high-performance, low-cost platform.

Document DDI337 from ARM provides more information about the Cortex-M3.

## System Timer

The System Timer (SYSTICK) is a 24-bit timer and is part of the Cortex-M3 core. The system timer can be configured either through the registers (see the Cortex-M3 Technical Reference Manual) or through the provided functions defined in `core_cm3.h`. There is an interrupt vector for the system timer. To configure the system timer, call `SysTickConfig()` with the number of system clocks in between Interrupt intervals (up to maximum of 24 bits).

## Interrupt Controller

The Nested Vectored Interrupt Controller is part of the core Cortex-M3 MCU. The NVIC controls the system exceptions and peripheral interrupts and is closely coupled with the core to provide low latency and efficient processing of late arriving interrupts. The NVIC maintains knowledge of the nested interrupts to enable tail-chaining of interrupts.

The Z32F128 MCU supports 64 peripheral interrupts (although 25 are not used) and 16 system interrupts. The NVIC also allows for setting software interrupts as well as resetting the system.

Interrupts can be assigned a Priority Group (common interrupts with the same priorities) as well as individual priorities. Eight priority levels are available. For an interrupt to be active, you must enable it in the peripheral and the NVIC registers. For more information on NVIC, see the Cortex M3 Technical Reference Manual.

The system includes functions to set the NVIC registers, which are defined in the `core_cm3.h`.

Table2.1. Interrupt Vector Map

| Priority | Vector Address | Interrupt Source |
|---|---|---|
| -16 | 0x0000_0000 | Stack Pointer |
| -15 | 0x0000_0004 | Reset Address |
| -14 | 0x0000_0008 | NMI Handler |
| -13 | 0x0000_000C | Hard Fault Handler |
| -12 | 0x0000_0010 | MPU Fault Handler |
| -11 | 0x0000_0014 | BUS Fault Handler |
| -10 | 0x0000_0018 | Usage Fault Handler |
| -9 | 0x0000_001C | Reserved |
| -8 | 0x0000_0020 | Reserved |
| -7 | 0x0000_0024 | Reserved |
| -6 | 0x0000_0028 | Reserved |
| -5 | 0x0000_002C | SVCall Handler |
| -4 | 0x0000_0030 | Debug Monitor Handler |
| -3 | 0x0000_0034 | Reserved |
| -2 | 0x0000_0038 | PenSV Handler |
| -1 | 0x0000_003C | SysTick Handler |
| 0 | 0x0000_0040 | LVDDETECT |
| 1 | 0x0000_0044 | SYSCLKFAIL |
| 2 | 0x0000_0048 | XOSCFAIL |
| 3 | 0x0000_004C | WDT |
| 4 | 0x0000_0050 | FRT |
| 5 | 0x0000_0054 | TIMER0 |
| 6 | 0x0000_0058 | TIMER1 |
| 7 | 0x0000_005C | TIMER2 |
| 8 | 0x0000_0060 | TIMER3 |
| 9 | 0x0000_0064 | TIMER4 |
| 10 | 0x0000_0068 | TIMER5 |
| 11 | 0x0000_006C | TIMER6 |
| 12 | 0x0000_0070 | TIMER7 |
| 13 | 0x0000_0074 | TIMER8 |
| 14 | 0x0000_0078 | TIMER9 |
| 15 | 0x0000_007C | Reserved |
| 16 | 0x0000_0080 | GPIOAE |
| 17 | 0x0000_0084 | GPIOAO |
| 18 | 0x0000_0088 | GPIOBE |
| 19 | 0x0000_008C | GPIOBO |
| 20 | 0x0000_0090 | GPIOCE |
| 21 | 0x0000_0094 | GPIOCO |
| 22 | 0x0000_0098 | GPIODE |
| 23 | 0x0000_009C | GPIODO |
| 24 | 0x0000_00A0 | MPWM0 |
| 25 | 0x0000_00A4 | MPWM0PROT |

| | | |
|---|---|---|
| **26** | 0x0000_00A8 | MPWM0OVV |
| **27** | 0x0000_00AC | MPWM1 |
| **28** | 0x0000_00B0 | MPWM1PROT |
| **29** | 0x0000_00B4 | MPWM1OVV |
| **30** | 0x0000_00B8 | Reserved |
| **31** | 0x0000_00BC | Reserved |
| **32** | 0x0000_00C0 | SPI0 |
| **33** | 0x0000_00C4 | SPI1 |
| **34** | 0x0000_00C8 | Reserved |
| **35** | 0x0000_00CC | Reserved |
| **36** | 0x0000_00D0 | I2C0 |
| **37** | 0x0000_00D4 | I2C1 |
| **38** | 0x0000_00D8 | UART0 |
| **39** | 0x0000_00DC | UART1 |
| **40** | 0x0000_00E0 | UART2 |
| **41** | 0x0000_00E4 | UART3 |
| **42** | 0x0000_00E8 | Reserved |
| **43** | 0x0000_00EC | ADC0 |
| **44** | 0x0000_00F0 | ADC1 |
| **45** | 0x0000_00F4 | Reserved |
| **46** | 0x0000_00F8 | Reserved |
| **47** | 0x0000_00FC | Reserved |
| **48** | 0x0000_0100 | Reserved |
| **49** | 0x0000_0104 | Reserved |
| **50** | 0x0000_0108 | GPIOEE |
| **51** | 0x0000_010C | GPIOEO |
| **52** | 0x0000_0110 | GPIOFE |
| **53** | 0x0000_0114 | GPIOFO |
| **54** | 0x0000_0118 | Reserved |
| **55** | 0x0000_011C | Reserved |
| **56** | 0x0000_0120 | Reserved |
| **57** | 0x0000_0124 | Reserved |
| **58** | 0x0000_0128 | Reserved |
| **59** | 0x0000_012C | Reserved |
| **60** | 0x0000_0130 | Reserved |
| **61** | 0x0000_0134 | Reserved |
| **62** | 0x0000_0138 | Reserved |
| **63** | 0x0000_013C | Reserved |

# 3. Boot Mode

## Boot Mode Pins

The Z32F384 MCU has a boot mode option to program internal Flash memory. When the BOOT pin is pulled low, the system starts up in the BOOT area (0x1FFF_0000) instead of the default Flash area (0x0000_0000). This provides the ability to flash the part using either UART or SPI interfaces. The BOOT pin has an internal pull up resistor; therefore, when the BOOT pin is not connected, it rides high (normal state).

The boot mode uses the UART0 port and the SPI0 ports for the interface. JTAG and SW interfaces can also be used, which provides the ability to recover from a bad flash update that prevents the JTAG or SW debugger from attaching.

The pins for boot mode are listed in Table 3.1.

**Table3.1. Boot Mode Pin List**

| Block | Pin Name | Dir | Description |
|---|---|---|---|
| SYSTEM | nRESET/PC10 | I | Reset Input signal |
| | BOOT/PC11 | I | '0' to enter Boot mode |
| UART0 | RXD0/PC14 | I | UART Boot Receive Data |
| | TXD0/PC15 | O | UART Boot Transmit Data |
| SPI0 | SS0/PA12 | I | SPI Boot Slave Select |
| | SCK0/PA13 | I | SPI Boot Clock Input |
| | MOSI0/PA14 | I | SPI Boot Data Input |
| | MISO0/PA15 | O | SPI Boot Data Output |

## Boot Mode Connections

Users can design the target board using either of the boot mode ports – UART or SPI.

Figures 3.1 and 3.2 show sample boot mode connection diagrams.

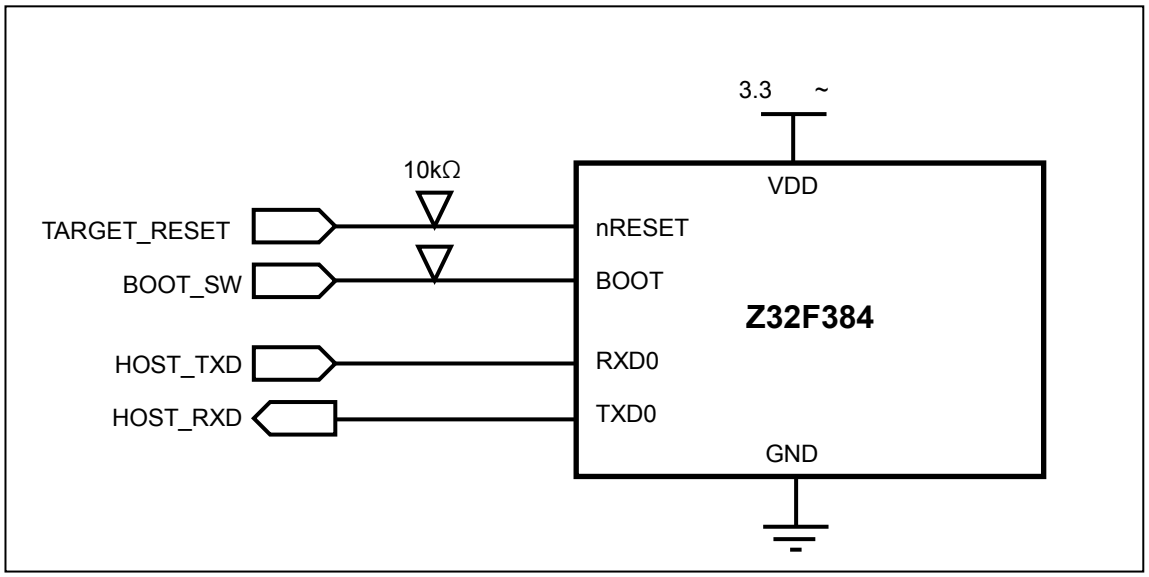



**Figure3.1. UART Boot Connection Diagram**

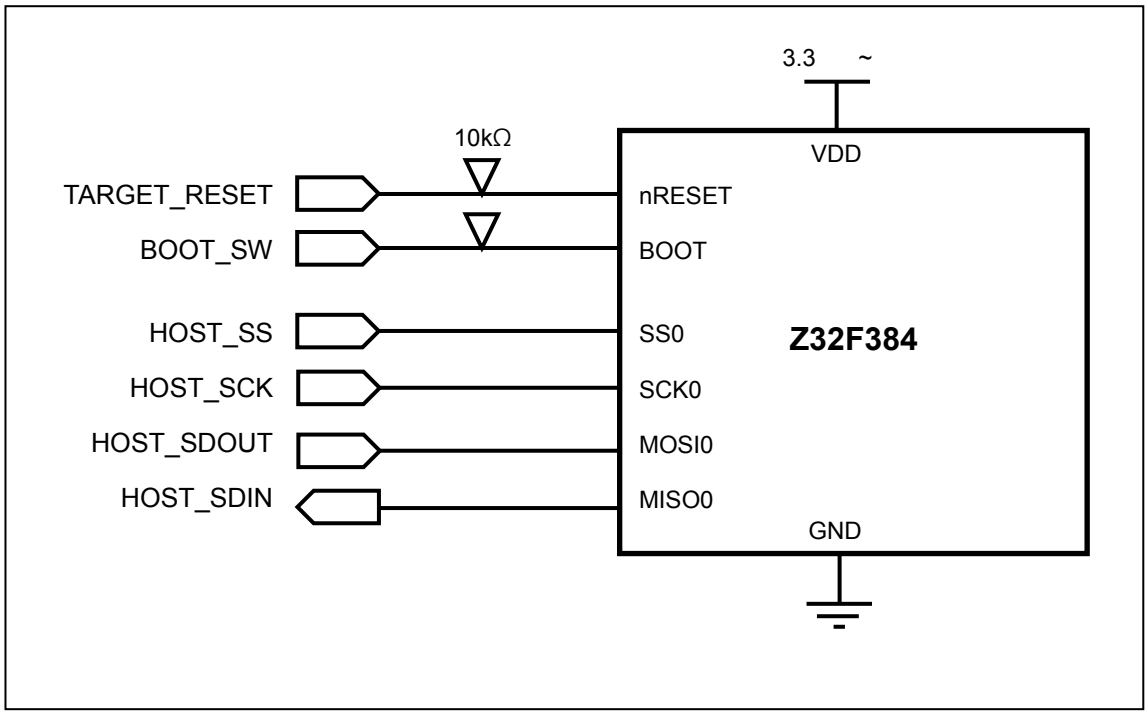



**Figure3.2. SPI Boot Connection Diagram**

# 4. System Control Unit

## Overview

The Z32F384 MCU has a built-in intelligent power control block which manages system analog blocks and operating modes. Internal reset and clock signals are controlled by SCU block to maintain optimize system performance and power dissipation.

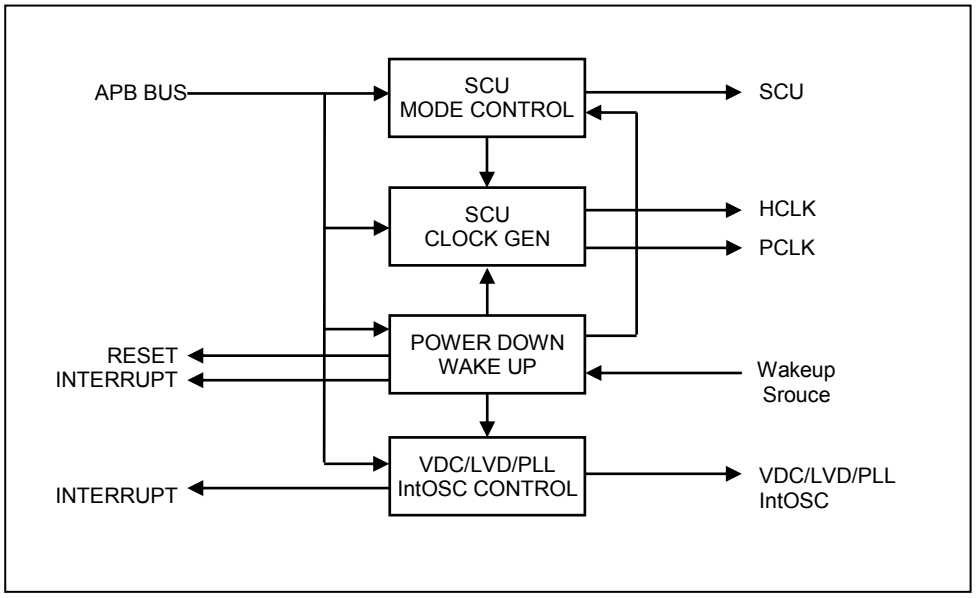



**Figure 4.1. SCU Block Diagram**

## Clock System

The Z32F384 MCU has two main clock systems. One is MCLK which supplies the clock to the HCLK_Free, CPU and AHB bus system. The PCLK clock is for the Peripheral clock and is supplied from MCLK. Some peripherals have the option to derive their clock from other clocks or the PCLK. User can control the clock system variation by software. Figure 4.2 shows the clock system of the chip, Table 4.1 lists the clock source descriptions.

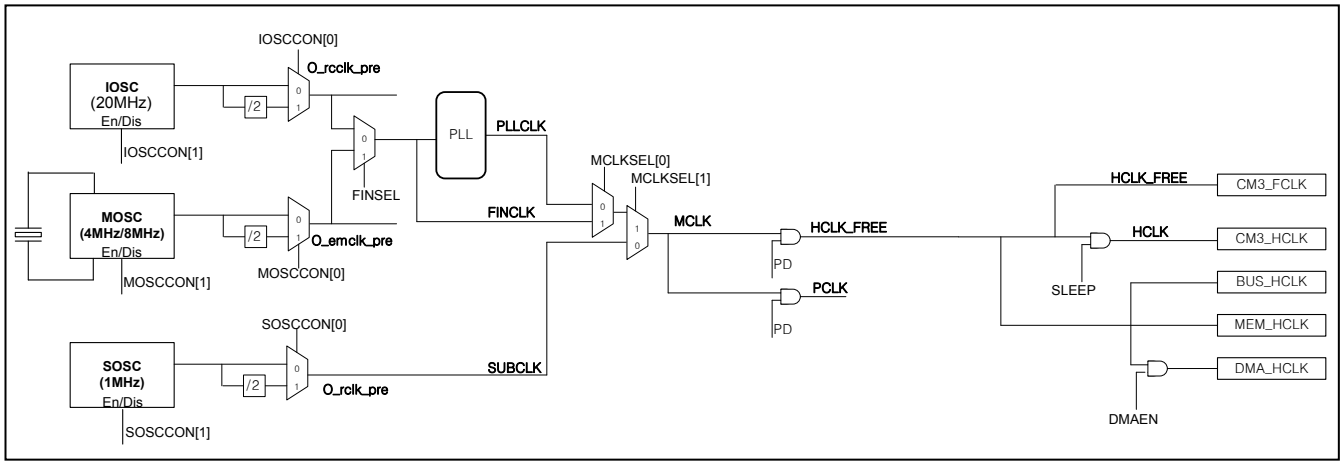



**Figure 4.2. System Clock Configuration**

Each of the mux to switch clock sources has a glitch-free circuit; therefore, the clock can be switched without a risk of glitches.

**Table 4.1. Clock Sources**

| Clock name | Frequency | Description |
|---|---|---|
| IOSC20 | 20MHz | Internal OSC |
| Sub OSC | Sub X-TAL (32.768KHz) | Sub External Crystal OSC |
| MainOSC | X-TAL(4MHz~16MHz) | External Crystal OSC |
| PLL Clock | 8MHz ~ 80MHz | On Chip PLL |
| ROSC | 1MHz | Internal RING OSC |

The PLL can synthesize the PLLCLK clock up to 80 MHz with either the Internal Oscillator or the External Crystal Oscillator reference clocks. It also has an internal pre-divider and post-divider.

## HCLK Clock Domain

The HCLK clock feeds the clock to the CPU and AHB bus. The Cortex-M3 CPU requires two clocks related to the HCLK clock, FCLK and HCLK. FCLK is a free running clock and is always running except in power down mode. HCLK can be stopped in idle mode.

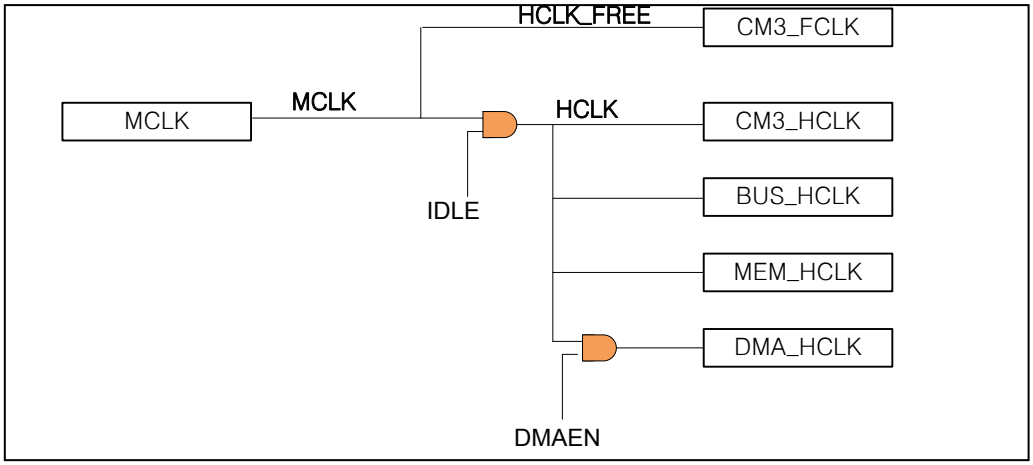



**Figure 4.3. System Clock Configuration**

# Miscellaneous Clock Domain for Cortex-M3

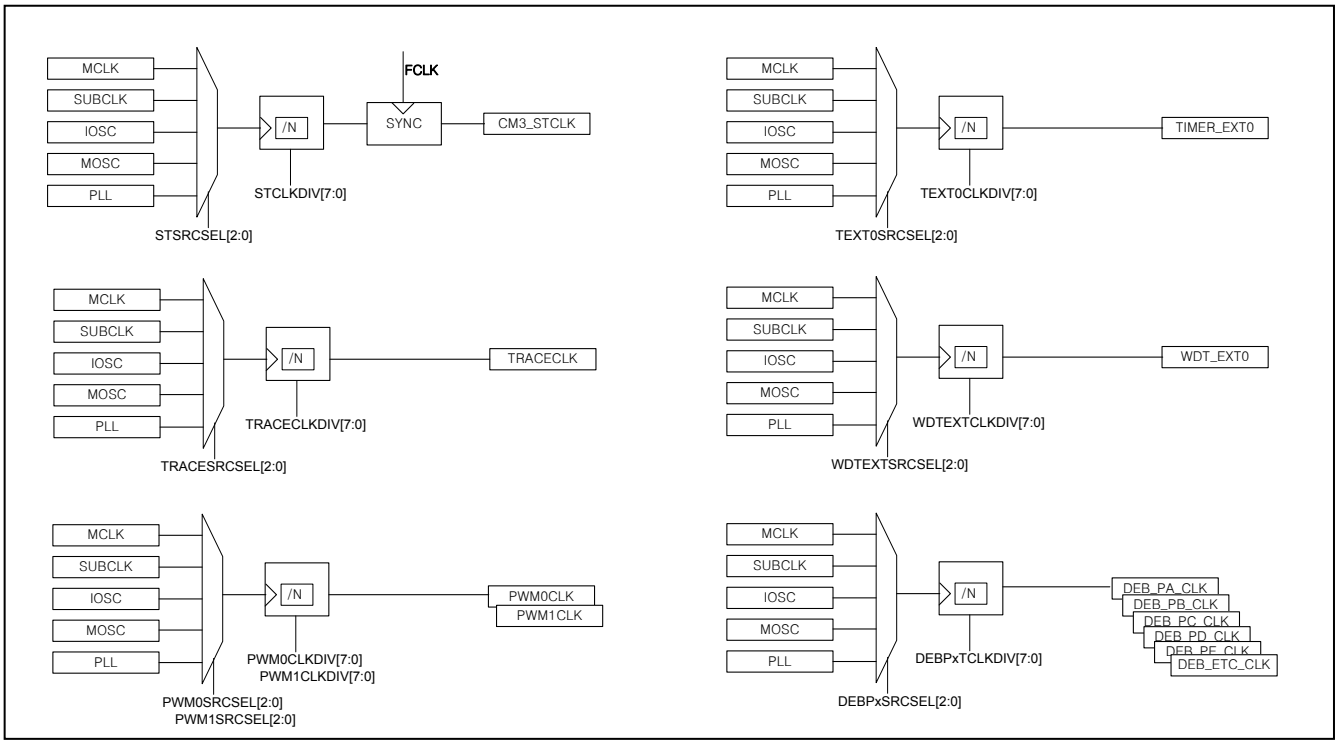



**Figure 4.4. Miscellaneous Clock Configuration**

## PCLK Clock Domain

PCLK is the master clock of all the peripherals. It can be stopped in power down mode. Each peripheral clock is generated by the PCER register set.

## Clock Configuration Procedure

After power up, the default system clock is fed by the RINGOSC (1 MHz) clock. RINGOSC is enabled by default during the power up sequence. The other clock sources are enabled by user controls with the RINGOSC system clock.

The MOSC clock can be enabled by the CSCR register. Prior to enabling the MOSC block, the pin mux configuration should be set for the XIN, XOUT function. PC12 and PC13 pins are shared with the MOSC's XIN and XOUT function; the PCCMR and PCCCR registers should be correctly configured. After enabling the MOSC block, you must wait for more than 1 msec to ensure stable crystal oscillation operation.

The PLL clock can be enabled by the PLLCON register. After enabling the PLL block, you must wait for the PLL lock, before you can select the PLL clock as the MCLK. Before changing the system clock, Flash access wait should be set to the maximum value. After the system clock is changed, you can set the desired Flash access wait value.

Figure 4.5 shows a sample flow chart for configuring the system clock.

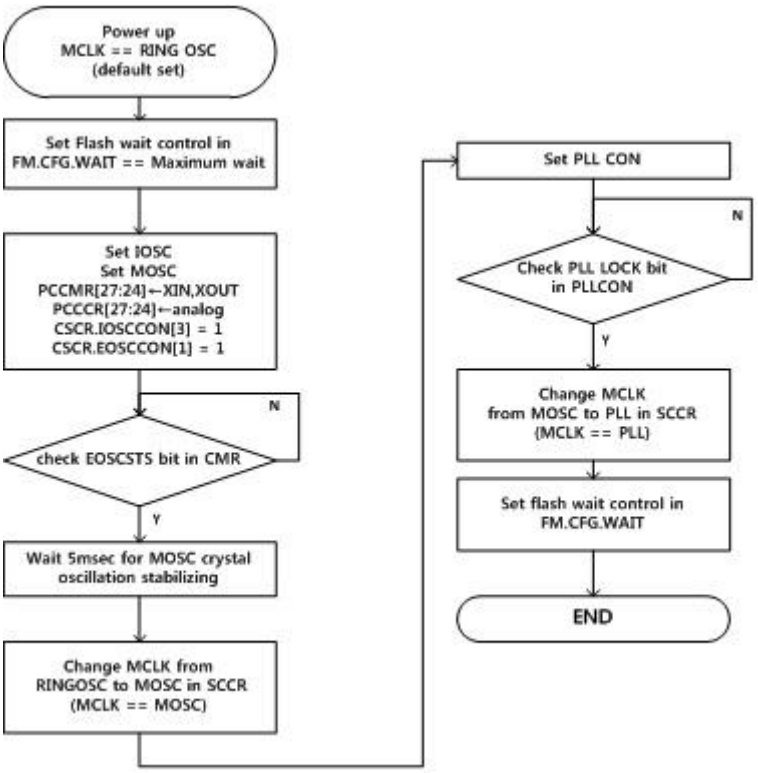



**Figure 4.5 Clock Configuration Flow Chart**

When you speed up the system clock to the maximum operating frequency, check the Flash wait control configuration. Flash read access time is a limiting factor for optimum performance. The wait control recommendation is provided in Table 4.1.

**Table 4.1. Flash Wait Control Recommendation**

| FM.CFG.WAIT | Flash Access Wait | Available Max System Clock Frequency |
|---|---|---|
| 000 | 0 wait | ~25MHz |
| 001 | 1 wait | ~50MHz |
| 010 | 2 wait | ~75MHz |
| 011 | 3 wait | ~75MHz |
| 100 | 4 wait | ~75MHz |
| 101 | 5 wait | ~75MHz |

## Cold Reset

Cold reset is an important feature of the chip when power is up. This characteristic globally affects the system boot. Internal VDC is enabled when VDD power is turned on. The internal VDD level slope is followed by the external VDD power slope. The internal PoR trigger level is 1.4 V of internal VDC voltage out level. At this time, boot operation is started. The RINGOSC clock is enabled and counts to 4 msec for internal VDC level stabilizing. During this time, the external VDD voltage level should be greater than the initial LVD level (2.3 V). After waiting 4 msec, the CPU reset is released and the operation is started.

Figure 4.6 shows the power up sequence and internal reset waveform.

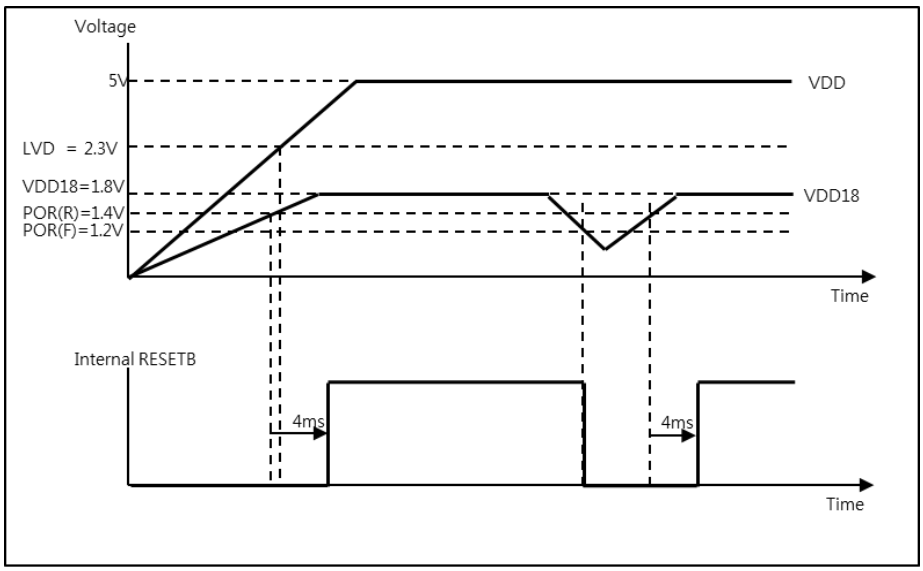



**Figure 4.6. Power Up POR Sequence**

The RSSR register shows the POR reset status. The last reset comes from POR; RSSR.PORST is set to “1”. After power up, this bit is always “1”. If an abnormal internal voltage drop occurs during normal operation, the system will be reset and this bit is also set to “1”.

When cold reset is applied, the chip returns to its initial state.

## Warm Reset

The warm reset event has several reset sources. Some parts of the chip return to initial state when a warm reset condition occurs.

The warm reset source is controlled by the RSER register and the status appears in the RSSR register. The reset for each peripheral block is controlled by the PRER register. The reset can be masked independently.

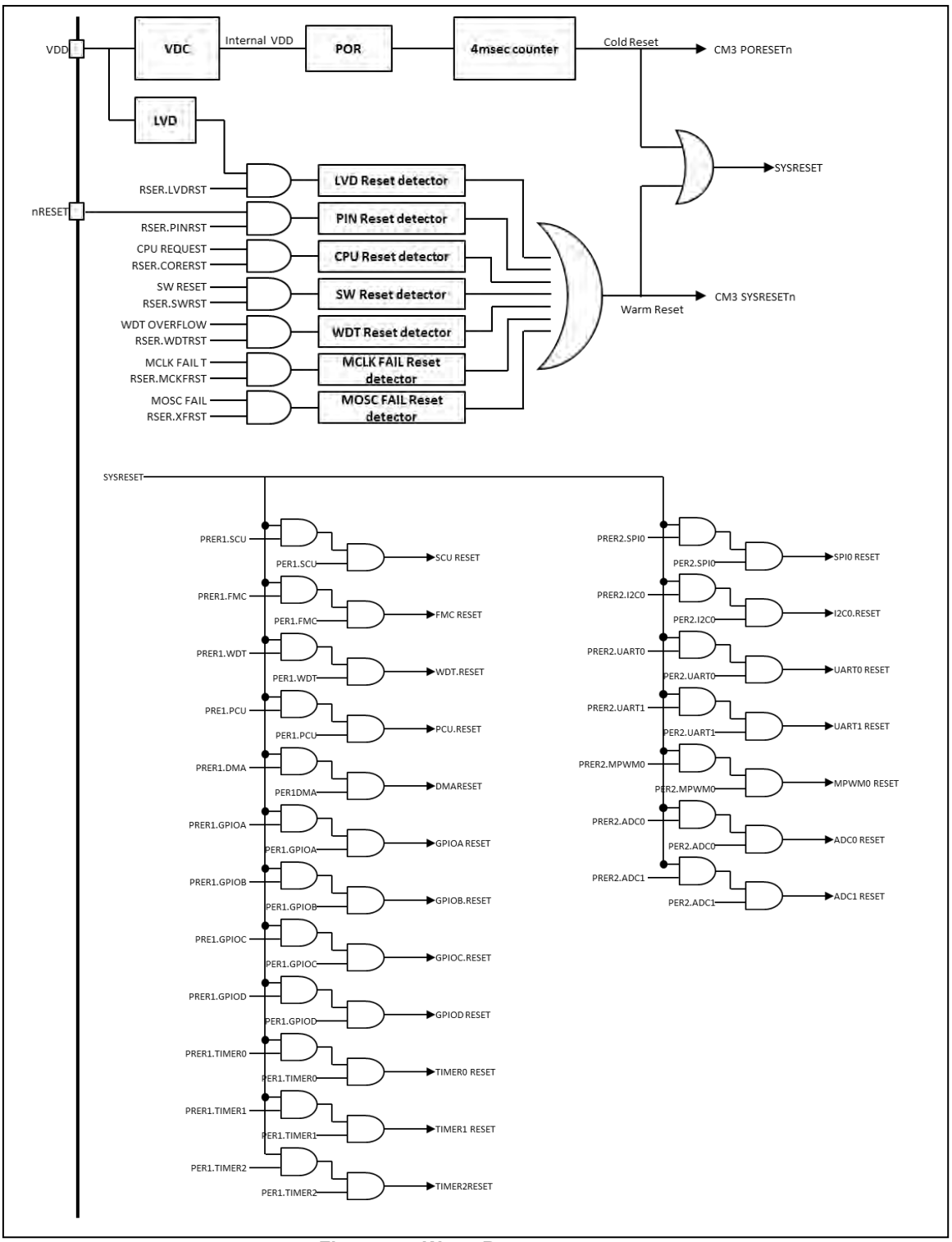



**Figure 4.7. Warm Reset**

The CM3_SYSRESETn signal resets the processor, excluding debug logic in the processor.

# Operation Mode

The INIT mode is the initial state of the chip when reset is asserted. The RUN mode is for maximum performance of the CPU with a high-speed clock system. The SLEEP and POWER DOWN modes can be used as low- power consumption modes. Low-power consumption is achieved by halting the processor core and unused peripherals.

Figure 4.5 shows the operating mode transition diagram.

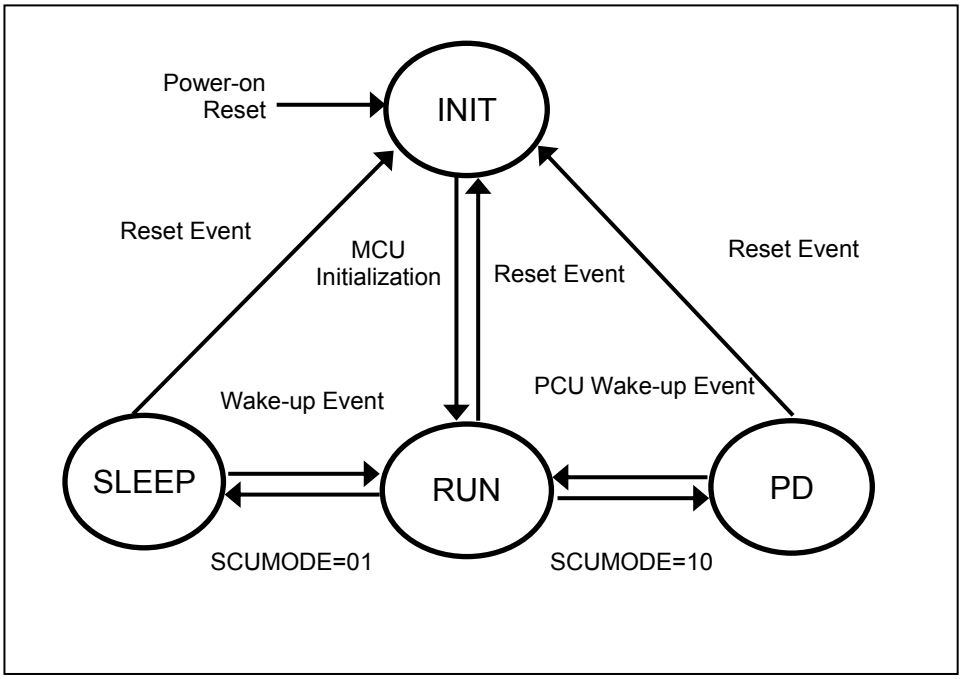



**Figure4.5. Operating Mode**

## RUN Mode

In RUN mode, the CPU core and the peripheral hardware is operated by using the high-speed clock. After reset, followed by the INIT state, the chip enters RUN mode.

## SLEEP Mode

In SLEEP mode, only the CPU is stopped. Each peripheral function can be enabled by the function enable and clock enable bit in the PER and PCER register.

## POWER DOWN Mode

In POWER DOWN mode, all internal circuits enter the stop state. Power down operation has a special power off sequence, as shown in Figure 4.6.

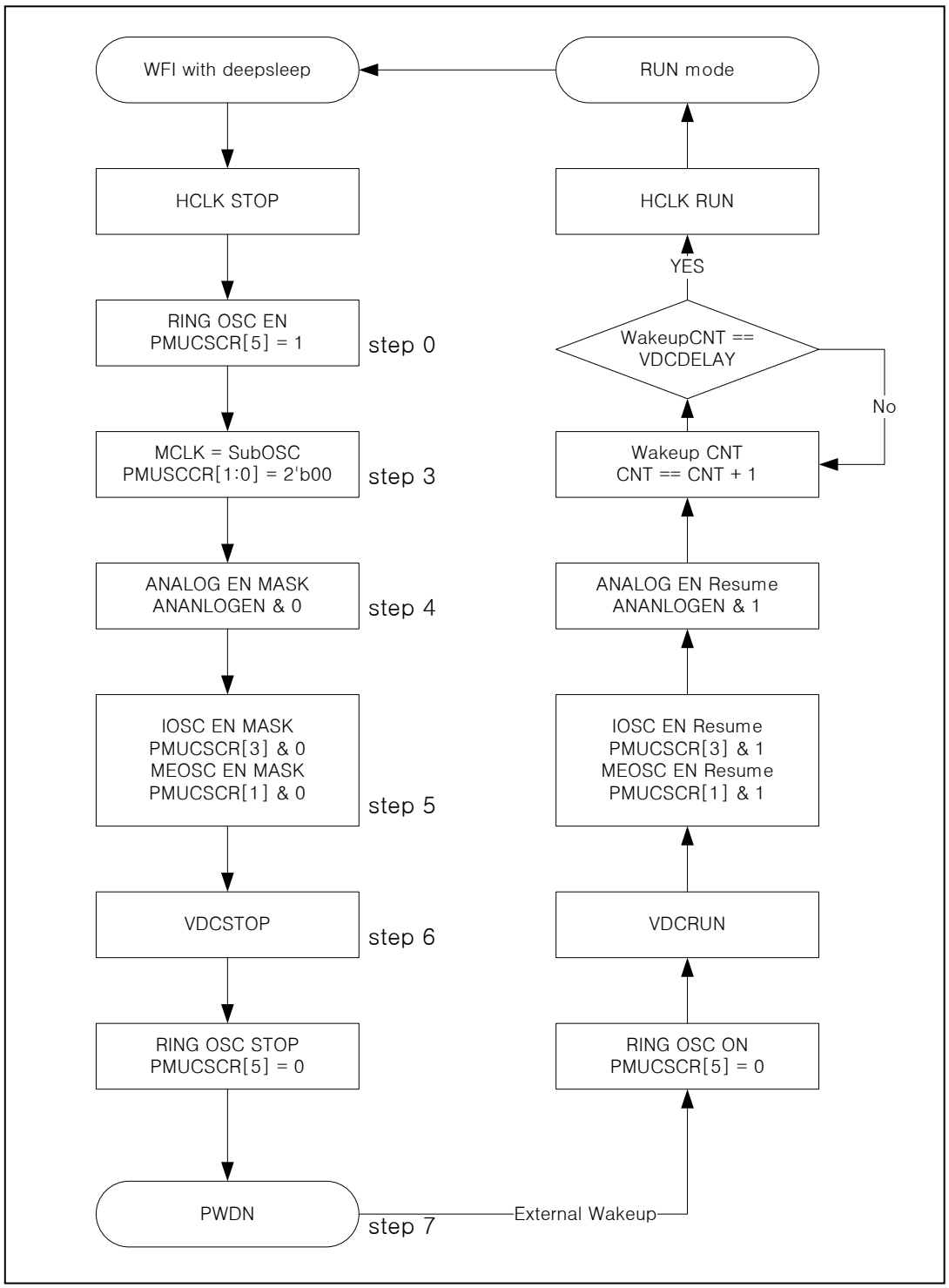



**Figure 4.6. Power-down and Wake-up Procedure**

# Pin Description

Table4.2. SCU and PLL Pins

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **nRESET** | I | External Reset Input |
| **XIN/XOUT** | OSC | External Crystal Oscillator |
| **SXIN/SXOUT** | OSC | External Sub Crystal Oscillator |
| **STBYO** | O | Stand-by Output Signal |
| **CLKO** | O | Clock Output Monitoring Signal |

# Registers

The base address of SCU is `0x4000_0000` and the register map is described in Table.4.3

**Table 4.3. SCU Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **CIDR** | 0x0000 | R | CHIP ID Register | AC33_0384 |
| **SMR** | 0x0004 | R/W | System Mode Register | 0000_0000 |
| **SRCR** | 0x0008 | R/W | System Reset Control Register | 0000_0000 |
| **CIDR2** | 0x000C | R/W | CHIP Revision ID Register | 0000_0000 |
| **WUER** | 0x0010 | R/W | Wake up source enable register | 0000_0000 |
| **WUSR** | 0x0014 | R/W | Wake up source status register | 0000_0000 |
| **RSER** | 0x0018 | R/W | Reset source enable register | 0000_0049 |
| **RSSR** | 0x001C | R/W | Reset source status register | 0000_0080* |
| **PRER1** | 0x0020 | R/W | Peripheral reset enable register 1 | 03FF_1F1F* |
| **PRER2** | 0x0024 | R/W | Peripheral reset enable register 2 | 00F3_0F33* |
| **PER1** | 0x0028 | R/W | Peripheral enable register 1 | 0000_000F* |
| **PER2** | 0x002C | R/W | Peripheral enable register 2 | 0000_0101* |
| **PCER1** | 0x0030 | R/W | Peripheral clock enable register 1 | 0000_000F* |
| **PCER2** | 0x0034 | R/W | Peripheral clock enable register 2 | 0000_0101* |
| **CSCR** | 0x0040 | R/W | Clock Source Control register | 0000_0020 |
| **SCCR** | 0x0044 | R/W | System Clock Control register | 0000_0000 |
| **CMR** | 0x0048 | R/W | Clock Monitoring register | 0000_0003 |
| **NMIR** | 0x004C | R/W | NMI control register | 0000_0000 |
| **COR** | 0x0050 | R/W | Clock Output Control register | 0000_000F |
| | 0x0054 | - | Reserved | |
| **PLLCON** | 0x0060 | R/W | PLL Control register | 0000_1000 |
| **VDCCON** | 0x0064 | R/W | VDC Control register | 0000_000F |
| **LVDCON** | 0x0068 | R/W | LVD Control register | 0000_0001 |
| **IOSCTRIM** | 0x006C | R/W | Internal RC OSC Control register | 0000_0000 |
| | 0x0070 | - | Reserved | 0000_0000 |
| | 0x0074 | - | Reserved | 0000_0000 |
| | 0x0078 | - | Reserved | 0000_0000 |
| | 0x007C | - | Reserved | 0000_0000 |
| **EOSCR** | 0x0080 | R/W | External Oscillator control register | 0000_0000 |
| **EMODR** | 0x0084 | R/W | External mode pin read register | 0000_000X |
| **DBCLK1** | 0x009C | R/W | Debounce Clock for PA, PB Pins | 0000_0000 |
| **DBCLK2** | 0x00A0 | R/W | Debounce Clock for PC, PD Pins | 0000_0000 |
| **DBCLK3** | 0x00A4 | R/W | Debounce Clock for PE, PF pins | 0000_0001 |
| **MCCR1** | 0x0090 | R/W | Trace and SysClock Clock Control | 0404_0001 |
| **MCCR2** | 0x0094 | R/W | MPWM0 and MPWM1 Clock Control | 0000_0000 |
| **MCCR3** | 0x0098 | R/W | TEXT0 and WDT clock control | 0000_0001 |
| **MCCR4** | 0x00A8 | R/W | ADC and NMI Debounce Clock control | 0000_0001 |

## CIDR Chip ID Register

The Chip ID register shows chip identification information. This register is a 32-bit read-only register.

**CIDR=0x4000_0000**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CHIPID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AC33_0384 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Read Only | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31 | CHIP ID | Device ID |
| 0 | | AC33_0384 |

**CIDR2=0x4000_000C**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REVISION ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0x0000_0002 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Read Only | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31 | REVISION ID | Device Revision ID |
| 0 | | 0x0000_0002 |

## SMR System Mode Register

The current operating mode is shown in this SCU mode register and the operation mode can be changed by writing new mode in this register. The previous operating mode is saved in this register after a reset event.

The System Mode register is a 16-bit register.

**SMR=0x4000_0004**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | VDCAON | | | PREVMODE | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 | | 0 | 0 | 0 | 0 |
| | | | | | | | R/W | | | R | | | | | |

| | | | |
|---|---|---|---|
| 8 | VDCAON | VDC Always on | |
| | | 0 | VDC will be off when Power down mode |
| | | 1 | VDC always on even in power down mode |
| 5 | PREVMODE | Previous operating mode before current reset event. | |
| 4 | | 00 | Previous operating mode was RUN mode |
| | | 01 | Previous operating mode was SLEEP mode |
| | | 10 | Previous operating mode was PowerDown mode |
| | | 11 | Previous operating mode was INIT mode |

# SRCR System Reset Control Register

The Ssystem Reset Control register is an 8-bit register.

SCR=0x4000_0008

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | STBYOP | | | | SWRST |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | RW | | | | W |

| | | |
|---|---|---|
| 5 | STBYOP | STBYO pin output polarity select bit |
| | | 0 Low active when chip is in Power Down |
| | | 1 High active when chip is in PowerDown |
| 1 | SWRST | Internal soft reset activation bit |
| | | 0 Normal operation |
| | | 1 Internal soft reset is applied and auto cleared |

# WUER Wakeup Source Enable Register

The Wakeup Source Enable register enables the wakeup source when the chip is in Power Down mode. Wakeup sources which will be used as the source of chip wakeup should be enabled in each bit field. If the source will be used as the wakeup source, write '1' to its enable bit. If the source will not be used as the wakeup source, write 0 to its enable bit.

This register is a 16-bit register.

**WUER=-0x4000_0010**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | GPIOFWUE | GPIOEWUE | GPIODWUE | GPIOCWUE | GPIOBWUE | GPIOAWUE | | | | | | FRTWUE | WDTWUE | LVDWUE |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | RW | RW | RW | RW | RW | RW | | | | | | RW | RW | RW |

| | | |
|---|---|---|
| 13 | GPIOFWUE | Enable wakeup source of GPIOF port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 12 | GPIOEWUE | Enable wakeup source of GPIOE port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 11 | GPIODWUE | Enable wakeup source of GPIOD port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 10 | GPIOCWUE | Enable wakeup source of GPIOC port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 9 | GPIOBWUE | Enable wakeup source of GPIOB port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 8 | GPIOAWUE | Enable wakeup source of GPIOA port pin change event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 2 | FRTWUE | Enable wakeup source of free run timer event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 1 | WDTWUE | Enable wakeup source of watchdog timer event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |
| 0 | LVDWUE | Enable wakeup source of LVD event |
| | | 0 Not used for wakeup source |
| | | 1 Enable the wakeup event generation |

# WUSR Wakeup Source Status Register

When the system is woken up by any wakeup source, the wakeup source is identified by reading the Wakeup Source Status register. When the bit is set to 1, the related wakeup source issues the wakeup to the SCU. The bit will be cleared when the event is cleared by the software.

**WUSR=0x4000_0014**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | GPIOFWU | GPIOEWU | GPIODWU | GPIOCWU | GPIOBWU | GPIOAWU | | | | | | FRTWU | WDTWU | LVDWU |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | R | R | R | R | R | R | | | | | | R | R | R |

| | | |
|---|---|---|
| 13 | GPIOFWU | Status of wakeup source of GPIOF port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 12 | GPIOEWU | Status of wakeup source of GPIOE port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 11 | GPIODWU | Status of wakeup source of GPIOD port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 10 | GPIOCWU | Status of wakeup source of GPIOC port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 9 | GPIOBWU | Status of wakeup source of GPIOB port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 8 | GPIOAWU | Status of wakeup source of GPIOA port pin change event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 2 | FRTWU | Status of wakeup source of free run timer event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 1 | WDTWU | Status of wakeup source of watchdog timer event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |
| 0 | LVDWU | Status of wakeup source of LVD event |
| | | 0 No wakeup event |
| | | 1 Wakeup event was generated |

# RSER Reset Source Enable Register

The reset source which will generate the reset event can be selected by the Reset Source Enable register. When writing 1 to the bit field of each reset source, the reset source event will be transferred to the reset generator. When writing 0 to the bit field of each reset source, the reset source event will be masked and will not generate a reset event.

**RSER=0x4000_0018**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | PINRST | CORERST | SWRST | WDTRST | MCKFRST | XFRST | LVDRST |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| | RW | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 6 | PINRST | External pin reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 5 | CPURST | CPU request reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 4 | SWRST | Software reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 3 | WDTRST | Watchdog Timer reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 2 | MCKFRST | MCLK Clock fail reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 1 | XFRST | External OSC Clock fail reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |
| 0 | LVDRST | LVD reset enable bit |
| | | 0 Reset from this event is masked |
| | | 1 Reset from this event is enabled |

# RSSR Reset Source Status Register

The Reset Source Status register shows the reset source information when a reset event occurs. “1” indicates that a reset event exists and “0” indicates that a reset event does not exist for a reset source. When the reset source is found, writing “1” to the corresponding bit will clear the reset status.

This register is an 8-bit register.

**RSSR=0x4000_001C**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PORST | PINRST | CORERST | SWRST | WDTRST | MCKFRST | XFRST | LVDRST |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RC1 | RC1 | RC1 | RC1 | RC1 | RC1 | RC1 | RC1 |

| | | |
|---|---|---|
| 7 | PORST | Power on reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 6 | PINRST | External pin reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 5 | CPURST | CPU request reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 4 | SWRST | Software reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 3 | WDTRST | Watchdog Timer reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 1 | XFRST | Clock fail reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |
| 0 | LVDRST | LVD reset status bit |
| | | 0 Read : Reset from this event was not exist<br>Write : no effect |
| | | 1 Read :Reset from this event was occurred<br>Write : Clear the status |

# PRER1 Peripheral Reset Enable Register 1

The reset of each peripheral by an event reset can be masked by a user setting. The Peripheral Reset Enable register controls enabling of the event reset. If the corresponding bit is '1', the peripheral corresponds with this bit and accepts the reset event. Otherwise, the peripheral is protected from the reset event and maintains its current operation.

When a reset is issued (enabled by the RSER register), you can configure each peripheral to either reset the registers to the default settings or ignore the reset. This applies to all resets except for removal of power.

**Caution**: If you disable the SCU reset response, you may not be able to connect via the debugger without a power off and on reset. Caution should also be applied with the GPIO/PCU peripherals because the debugger uses these as well.

**PRER1=0x4000_0020**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TIMER9 | TIMER8 | TIMER7 | TIMER6 | TIMER5 | TIMER4 | TIMER3 | TIMER2 | TIMER1 | TIMER0 | | | GPIOF | GPIOE | GPIOD | GPIOC | GPIOB | GPIOA | FRT | | | DMA | PCU | WDT | FMC | SCU |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| | | | | | | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | | | RW | RW | RW | RW | RW | RW | RW | | | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 25 | TIMER9 | TIMER9 reset mask |
| 24 | TIMER8 | TIMER8 reset mask |
| 23 | TIMER7 | TIMER3 reset mask |
| 22 | TIMER6 | TIMER2 reset mask |
| 21 | TIMER5 | TIMER1 reset mask |
| 20 | TIMER4 | TIMER0 reset mask |
| 19 | TIMER3 | TIMER3 reset mask |
| 18 | TIMER2 | TIMER2 reset mask |
| 17 | TIMER1 | TIMER1 reset mask |
| 16 | TIMER0 | TIMER0 reset mask |
| 13 | GPIOF | GPIOF reset mask |
| 12 | GPIOE | GPIOE reset mask |
| 11 | GPIOD | GPIOE reset mask |
| 10 | GPIOC | GPIOE reset mask |
| 9 | GPIOB | GPIOE reset mask |
| 8 | GPIOA | GPIOA reset mask |
| 7 | FRT | FRT reset mask |
| 4 | DMA | DMA reset mask |
| 3 | PCU | Port Control Unit reset mask |
| 2 | WDT | Watchdog Timer reset mask |
| 1 | FMC | Flash memory controller reset mask |
| 0 | SCU | System Control Unit reset mask |

# PRER2 Peripheral Reset Enable Register 2

Peripheral Reset Enable Register 2 is a 32-bit register (See PRER1 for a full explaination of this register).

**PRER2=0x4000_0024**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | ADC1 | ADC0 | | | MPWM1 | MWPM0 | | | | | UART3 | UART2 | UART1 | UART0 | | | I2C1 | I2C0 | | | SPI1 | SPI0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| | | | | | | | | RW | | RW | RW | | | RW | RW | | | | | RW | RW | RW | RW | | | RW | RW | | | RW | RW |

| | | |
|---|---|---|
| 21 | ADC1 | ADC1 reset enable |
| 20 | ADC0 | ADC0 reset enable |
| 17 | MPWM1 | MPWM1 reset enable |
| 16 | MPWM0 | MPWM0 reset enable |
| 11 | UART3 | UART3 reset enable |
| 10 | UART2 | UART2 reset enable |
| 9 | UART1 | UART1 reset enable |
| 8 | UART0 | UART0 reset enable |
| 5 | I2C1 | I²C1 reset enable |
| 4 | I2C0 | I²C0 reset enable |
| 1 | SPI1 | SPI1 reset enable |
| 0 | SPI0 | SPI0 reset enable |

# PER1 Peripheral Enable Register 1

All the peripherals are disabled by default except SPI0 and UART0.
To use the peripheral unit, activate it by writing "1" to the corresponding bit in the PER0/1 register. When the bit for the peripheral unit is cleared ("0"), the peripheral will stay in a reset state.

**PER1=0x4000_0028**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TIMER9 | TIMER8 | TIMER7 | TIMER6 | TIMER5 | TIMER4 | TIMER3 | TIMER2 | TIMER1 | TIMER0 | | | GPIOF | GPIOE | GPIOD | GPIOC | GPIOB | GPIOA | FRT | | | DMA | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| | | | | | | RW | RW | RW | | | | RW | RW | RW | RW | | | RW | RW | RW | RW | RW | RW | | | | RW | R | R | R | R |

| | | |
|---|---|---|
| 25 | TIMER9 | TIMER9 function enable |
| 24 | TIMER8 | TIMER8 function enable |
| 23 | TIMER7 | TIMER7 function enable |
| 22 | TIMER6 | TIMER6 function enable |
| 21 | TIMER5 | TIMER5 function enable |
| 20 | TIMER4 | TIMER4 function enable |
| 19 | TIMER3 | TIMER3 function enable |
| 18 | TIMER2 | TIMER2 function enable |
| 17 | TIMER1 | TIMER1 function enable |
| 16 | TIMER0 | TIMER0 function enable |
| 13 | GPIOF | GPIOF function enable |
| 12 | GPIOE | GPIOE function enable |
| 11 | GPIOD | GPIOD function enable |
| 10 | GPIOC | GPIOC function enable |
| 9 | GPIOB | GPIOB function enable |
| 8 | GPIOA | GPIOA function enable |
| 7 | FRT | FRT function enable |
| 4 | DMA | DMA function enable |
| 3 | | Reserved |
| 2 | | |
| 1 | | |
| 0 | | |

# PER2 Peripheral Enable Register 2

Peripheral Enable Register 2 is a 32-bit register.

PER2=0x4000_002C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | ADC1 | ADC0 | | | MPWM1 | MWPM0 | | | | | UART3 | UART2 | UART1 | UART0 | | | I2C1 | I2C0 | | | SPI1 | SPI0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | | | | | RW | | RW | RW | | | RW | RW | | | | | RW | RW | RW | RW | | | RW | RW | | | RW | RW |

| | | |
|---|---|---|
| 21 | ADC1 | ADC1 function enable |
| 20 | ADC0 | ADC0 function enable |
| 17 | MPWM1 | MPWM1 function enable |
| 16 | MPWM0 | MPWM0 function enable |
| 11 | UART3 | UART3 function enable |
| 10 | UART2 | UART2 function enable |
| 9 | UART1 | UART1 function enable |
| 8 | UART0 | UART0 function enable |
| 5 | I2C1 | $I^2C1$ function enable |
| 4 | I2C0 | $I^2C0$ function enable |
| 1 | SPI1 | SPI1 function enable |
| 0 | SPI0 | SPI0 function enable |

# PCER1 Peripheral Clock Enable Register 1

To use the peripheral unit, its clock should be activated by writing ‘1’ to the corresponding bit in the PCER0/1 register. Before enabling its clock, the peripheral will not operate correctly.

To stop the clock of the peripheral unit, write ‘0’ to the corresponding bit in the PCER0/1 register, after which the clock of the peripheral is stopped.

**PCER1=0x4000_0030**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TIMER9 | TIMER8 | TIMER7 | TIMER6 | TIMER5 | TIMER4 | TIMER3 | TIMER2 | TIMER1 | TIMER0 | | | GPIOF | GPIOE | GPIOD | GPIOC | GPIOB | GPIOA | FRT | | | DMA | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| | | | | | | RW | RW | RW | | | | RW | RW | RW | RW | | | RW | RW | RW | RW | RW | RW | | | | RW | R | R | R | R |

| | | |
|---|---|---|
| 25 | TIMER9 | TIMER9 clock enable |
| 24 | TIMER8 | TIMER8 clock enable |
| 23 | TIMER7 | TIMER7 clock enable |
| 22 | TIMER6 | TIMER6 clock enable |
| 21 | TIMER5 | TIMER5 clock enable |
| 20 | TIMER4 | TIMER4 clock enable |
| 19 | TIMER3 | TIMER3 clock enable |
| 18 | TIMER2 | TIMER2 clock enable |
| 17 | TIMER1 | TIMER1 clock enable |
| 16 | TIMER0 | TIMER0 clock enable |
| 13 | GPIOF | GPIOF clock enable |
| 12 | GPIOE | GPIOE clock enable |
| 11 | GPIOD | GPIOD clock enable |
| 10 | GPIOC | GPIOC clock enable |
| 9 | GPIOB | GPIOB clock enable |
| 8 | GPIOA | GPIOA clock enable |
| 7 | FRT | FRT clock enable |
| 4 | DMA | DMA clock enable |
| 3 | | Reserved |
| 2 | | |
| 1 | | |
| 0 | | |

# PCER2 Peripheral Clock Enable Register 2

To use the peripheral unit, its clock should be activated by writing ‘1’ to the corresponding bit.

**PCER2=0x4000_0034**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | ADC1 | ADC0 | | | MPWM1 | MWPM0 | | | | | UART3 | UART2 | UART1 | UART0 | | | I2C1 | I2C0 | | | SPI1 | SPI0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | | | | | | | RW | RW | | | RW | RW | | | | | RW | RW | RW | RW | | | RW | RW | | | RW | RW |

| | | |
|---|---|---|
| 21 | ADC1 | ADC1 clock enable |
| 20 | ADC0 | ADC0 clock enable |
| 17 | MPWM1 | MPWM1 clock enable |
| 16 | MPWM0 | MPWM0 clock enable |
| 11 | UART3 | UART3 clock enable |
| 10 | UART2 | UART2 clock enable |
| 9 | UART1 | UART1 clock enable |
| 8 | UART0 | UART0 clock enable |
| 5 | I2C1 | $I^2C1$ clock enable |
| 4 | I2C0 | $I^2C0$ clock enable |
| 1 | SPI1 | SPI1 clock enable |
| 0 | SPI0 | SPI0 clock enable |

# CSCR Clock Source Control Register

The Z32F384 MCU has multiple clock sources to generate internal operating clocks. Each clock source can be enabled or disabled by the CSCR register.

This register is an 8-bit register.

**CSCR=0x4000_0040**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SXOSCEN | | RINGOSCCON | | IOSCCON | | EOSCCON | |
| 0 | 0 | 10 | | 00 | | 00 | |
| RW | | RW | | RW | | RW | |

| | | |
|---|---|---|
| 7 | SXOSCEN | External Sub Oscillator Enable |
| | | 0 Disable Sub Oscillator |
| | | 1 Enable Sub Oscillator |
| 5<br>4 | RINGOSCCON | Internal ring oscillator control |
| | | 0X Stop internal sub oscillator |
| | | 10 Enable internal sub oscillator |
| | | 11 Enable internal sub oscillator divide by 2 |
| 3<br>2 | IOSCCON | Internal oscillator control |
| | | 0X Stop internal oscillator |
| | | 10 Enable internal oscillator |
| | | 11 Enable internal oscillator divide by 2 |
| 1<br>0 | EOSCON | External crystal oscillator control |
| | | 0X Stop internal oscillator |
| | | 10 Enable internal oscillator |
| | | 11 Enable internal oscillator divide by 2 |

# SCCR System Clock Control Register

The System Clock Control register is the source for the PLL system and clock selection.

FINSEL selects either the IOSC or the External OSC as the input to the PLL system.The System clock select selects internal sub osciallator (Ring), External OSC, Internal OSC or PLL.

When changing FINSEL to MOSC ("1"), both internal OSC and external OSC should be alive, otherwise the chip will malfunction.

**SCCR=0x4000_0044**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | | | | | FINSEL | MCLKSEL | |
| 0000 | | | | | 0 | 00 | |
| R | | | | | RW | RW | |

| | | |
|---|---|---|
| 2 | FINSEL | PLL input source FIN select register |
| | | 0 IOSC clock is used as FIN clock |
| | | 1 MOSC clock is used as FIN clock |
| 1<br>0 | MCLKSEL | System clock select register |
| | | 0X Internal sub oscillator |
| | | 10 PLL bypassed clock |
| | | 11 PLL output clock |

# CMR Clock Monitoring Register

You can monitor the internal clock and external osciallators. To enable monitoring, the MCLKMNT/EOSCMNTSXOSCMNT bits must be set before the MCLKSTS, EOSCSTS, and SXOSCSTS bits are valid.

**Note:** The EOSCSTS bit only checks for the EOSCSTS oscillation and does not check for stability. When the system detects an MCLKFAIL interrupt, the MCLKREC bit determines if the system just dies or will auto recover using the ROSC. In most cases, the system should auto recover to keep running. The Clock Monitoring register is a 16-bit register.

**Note:** Oscillator clock statuses only refer to oscilliation and are not necessarly stable. After enabling a clock, check the status for oscillation, then wait for stability before using the clock.

**MR=0x4000_0048**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MCLKREC | | | | SXOSCMNT | SXOSCIE | SXOSCFAIL | SXOSCSTS | MCLKMNT | MCLKIE | MCLKFAIL | MCLKSTS | EOSCMNT | EOSCIE | EOSCFAIL | EOSCSTS |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| R | | | | RW | RW | WC1 | WC1 | RW | RW | WC1 | WC1 | RW | RW | WC1 | WC1 |

| | | |
|---|---|---|
| 15 | MCLKREC | MCLK fail auto recovery |
| | | 0 MCLK is changed to RINGOSC by default when MCLKFAIL issued |
| | | 1 MCLK auto recovery is disabled |
| 11 | SXOSCMNT | Sub Oscillator monitoring enable |
| | | 0 Sub Oscillator monitoring disabled |
| | | 1 Sub Oscillator monitoring enabled |
| 10 | SXOSCIE | Sub Oscillator fail interrupt enable |
| | | 0 Sub Oscillator fail interrupt disabled |
| | | 1 Sub Oscillator fail interrupt enabled |
| 9 | SXOSCFAIL | Sub Oscillator fail interrupt |
| | | 0 Sub Oscillator fail interrupt not occurred |
| | | 1 Read : Sub Oscillator fail interrupt is pending<br>Write : Clear pending interrupt |
| 8 | SXOSCSTS | Sub Oscillator clock status |
| | | 0 Not oscillate |
| | | 1 Sub oscillator is working normally |
| 7 | MCLKMNT | MCLK monitoring enable |
| | | 0 MCLK monitoring disabled |
| | | 1 MCLK monitoring enabled |
| 6 | MCLKIE | MCLK fail interrupt enable |
| | | 0 MCLK fail interrupt disabled |
| | | 1 MCLK fail interrupt enabled |
| 5 | MCLKFAIL | MCLK fail interrupt |
| | | 0 MCLK fail interrupt not occurred |
| | | 1 Read : MCLK fail interrupt is pending<br>Write : Clear pending interrupt |
| 4 | MCLKSTS | MCLK clock status |
| | | 0 No clock is present on MCLK |
| | | 1 Clock is present on MCLK |
| 3 | EOSCMNT | External oscillator monitoring enable |
| | | 0 External oscillator monitoring disabled |
| | | 1 External oscillator monitoring enabled |
| 2 | EOSCIE | External oscillator fail interrupt enable |
| | | 0 External oscillator fail interrupt disabled |

| | | | |
|---|---|---|---|
| | | 1 | External oscillator fail interrupt enabled |
| 1 | EOSCFAIL | External oscillator fail interrupt | |
| | | 0 | External oscillator fail interrupt not occurred |
| | | 1 | Read : External oscillator fail interrupt is pending<br>Write : Clear pending interrupt |
| 0 | EOSCSTS | External oscillator status | |
| | | 0 | Not oscillate |
| | | 1 | External oscillator is working normally |

# NMIR NMI Control Register

The NMI Control register provides control and status for Non-Maskable Interrupt. There are 6 available NMI sources. Write access key is required 0xA32 on NMIR[31:20] when writing to this register.

**NMIR = 0x4000_004C**

| 31 30 29 28 | 27 26 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REG ('hA) | ACCESS ('h3) | CODE ('h2) | NMIPINSTS | NMIINT | NMIPINDBEN | | NMIINTSTS | PROT1STS | OVP1STS | PROT0STS | OVP0STS | WDTSTS | MCLKFAILSTS | BODSTS | NIMPINEN | PROT1EN | OVP1EN | PROT0EN | OVP0EN | WDTEN | MCLKFAILEN | LVDEN |
| | | | 1 | 0 | 1 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | R | WC1 | RW | | R | R | R | R | R | R | R | R | RW | RW | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 19 | NMIPINSTS | NMI pin status bit |
| | 0 | NMI pin is low. |
| | 1 | NMI pin is high. |
| 18 | NMIINT | NMI interrupt bit, Write 1 to clear |
| | 0 | NMI interrupt is not pending |
| | 1 | NMI interrupt is pending |
| 17 | NMIPINDBEN | NMI pin debounce enable |
| | 0 | Disable |
| | 1 | Enable |
| 15 | NMIINTSTS | NMI pin interrupt status bit |
| | 0 | NMI interrupt is not pending |
| | 1 | NMI interrupt is pending |
| 14 | PROT1STS | MPWM1's Protection interrupt status bit. |
| | 0 | No PROT1 interrupt |
| | 1 | PROT1 interrupt occurred |
| 13 | OVP1STS | MPWM1's Over Voltage Protection interrupt status bit |
| | 0 | No OVP1 interrupt |
| | 1 | OVP1 interrupt occurred |
| 12 | PROT0STS | MPWM0's Protection interrupt status bit. |
| | 0 | No PROT0 interrupt |
| | 1 | PROT interrupt occurred |
| 11 | OVP0STS | MPWM0's Over Voltage Protection interrupt status bit |
| | 0 | No OVP0 interrupt |
| | 1 | OVP0 interrupt occurred |
| 10 | WDTINTSTS | WDT Interrrupt status bit |
| | 0 | No WDT interrupt |
| | 1 | WDT interrupt occurred |
| 9 | MCLKFAILSTS | MCLK Fail interrupt status bit |
| | 0 | No MCLK fail interrupt |
| | 1 | MCLK fail interrupt occurred |
| 8 | BODSTS | BOD interrupt status bit |
| | 0 | No BOD interrupt |
| | 1 | BOD interrupt occurred |

| | | |
|---|---|---|
| 7 | NMIPINEN | NMI pin interrupt Enable<br>Write permission is required by PCU write enable sequence |
| | | 0 Disable |
| | | 1 Enable |
| 6 | PROT1EN | MPWM1's Protection interrupt enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 5 | OVP1EN | MPWM1's Over Voltage Protection interrupt enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 4 | PROT0EN | MPWM0's Protection interrupt enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 3 | OVP0EN | MPWM0's Over Voltage Protection interrupt enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 2 | WDTINTEN | WDT Interrrupt condition enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 1 | MCLKFAILEN | MCLK Fail condition enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |
| 0 | LVDEN | LVD Detect condition enable for NMI interrupt |
| | | 0 Disable |
| | | 1 Enable |

# COR Clock Output Register

The Clock Output register controls enabling/disabling and provides a divider for the clock output. To output the clock signal, you must enable the clock out function pin (See Chapter 5, Port Control Unit).

**COR=0x4000_0050**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | | | CLKOEN | CLKODIV | | | |
| 000 | | | 0 | 1111 | | | |
| R | | | RW | RW | | | |

| | | | |
|---|---|---|---|
| 4 | CLKOEN | Clock output enable | |
| | | 0 | CLKO is disabled and stay “L” output |
| | | 1 | CLKO Is enabled |
| 3 0 | CLKODIV | Clock output divider value<br>CLKO = MCLK (CLKODIV = 0)<br>$\mathbf{CLKO} = \frac{\boldsymbol{MCLK}}{\mathbf{2} * (\mathbf{CLKODIV} + \mathbf{1})}$ $(\boldsymbol{CLKODIV} > 0)$ | |

# PLLCON PLL Control Register

Integrated PLL can synthesize the high speed clock for extremely high performance of the CPU from either the internal oscillator (IOSC) or the external oscillator (MOSC). The PLL Control register provides the configuration for the PLL system. By default, the PLL system is in reset mode and disabled. You must negate the reset and enable the PLL to operate (bits 14 and 15 must be set). The Bypass bit must be set to output the PLL clock. The active clock is defined in SCCR bit 2 (FIN).

To calculate the PLL output:

```
PLL Out = ( (Active clock / PREDIV) * FBCTRL) / POSTDI
```

**Note:** (Active Clock/PREDIV) * FBCTRL) must be below 224 MHz else PLL will not lock.

**PLLCON=0x4000_0060**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PLLRSTB | PLLEN | BYPASS | LOCKSTS | | | | PREDIV | FBCTRL | | | | POSTDIV | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0000 | | | | 0000 | | | |
| RW | RW | RW | R | | | | RW | RW | | | | RW | | | |

| | | | |
|---|---|---|---|
| 15 | PLLRSTB | PLL reset | |
| | | 0 PLL reset is asserted | |
| | | 1 PLL reset is negated | |
| 14 | PLLEN | PLL enable | |
| | | 0 PLL is disabled | |
| | | 1 PLL is enabled | |
| 13 | BYPASS | FIN bypass | |
| | | 0 FOUT is bypassed as FIN | |
| | | 1 FOUT is PLL output | |
| 12 | LOCK | LOCK status | |
| | | 0 PLL is not locked | |
| | | 1 PLL is locked | |
| 8 | PREDIV | FIN predivider | |
| | | 0 FIN divided by 1 | |
| | | 1 FIN divided by 2 | |
| 7 4 | FBCTRL | Feedback control | |
| | | 0000 M = 8 | 1000 M = 30 |
| | | 0001 M = 12 | 1001 M =32 |
| | | 0010 M = 14 | 1010 M = 38 |
| | | 0011 M = 16 | 1011 M = 40 |
| | | 0100 M = 18 | 1100 Not available |
| | | 0101 M = 20 | 1101 |
| | | 0110 M = 22 | 1110 |
| | | 0111 M = 28 | 1111 |
| 3 0 | POSTDIV | Post divider control | |
| | | 000 N = 1 | |
| | | 001 N = 2 | |
| | | 010 N = 3 | |
| | | 011 N =4 | |
| | | 100 N = 6 | |
| | | 101 N = 8 | |
| | | 110 N = 12 | |
| | | 111 N =16 | |

# VDCCON VDC Control Register

The on chip VDC control register is shown below.

VDCTRIM is used for the trim value of VDC output. To modify the VDCTRIM bit, VDCTE should write “1”simultaneously. VDCWDLY value can be written with writing “1” to VDCDE bit simultaneously.

**VDCCON=0x4000_0064**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BMRTE | | | | | BMRTRIM | | | VDCTE | | | | VDCTRIM | | | | | | | | | | | VDCDE | VDCWDLY | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 00 | | | 0 | 0 | 0 | 0 | 0000 | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x0F | | | | | | | |
| W | | | | | RW | | | W | | | | RW | | | | | | | | | | | W | RW | | | | | | | |

| | | |
|---|---|---|
| 31 | BMRTE | Reference BGR trim write enable.<br>0 BMRTRIM field is not updated by writing<br>1 BMRTRIM filed can be updated by writing |
| 26<br>24 | BMRTRIM | Reference BGR output voltage trim value |
| 23 | VDCTE | VDCTRIM value write enable. Write only with VDCTRIM value.<br>0 VDCTRIM field is not updated by writing<br>1 VDCTRIM filed can be updated by writing |
| 19<br>16 | VDCTRIM | VDC output voltage trim value |
| 8 | VDCDE | VDCWDLY value write enable. Write only with VDCWDLY value<br>0 VDCWLDLY field is not updated by writing<br>0 VDCWLDLY field can be updated by writing |
| 7<br>0 | VDCWDLY | VDC warm-up delay count value.<br>When SCU is waked up from powerdown mode, the warm-up delay is inserted for VDC output being stabilized.<br>The amount of delay can be defined with this register value<br>7F : 2msec |

# LVDCON LVD Control Register

The on chip brown-out detector control register is a 32-bit register.

**LVDCON=0x4000_0068**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BODTE | | | | | | BODTRIM | | SELEN | | | | | | BODSEL | | | | | | | | BODLVL | BODEN |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 | | 0 | 0 | 0 | 0 | 0 | 0 | 00 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | | | | | RW | | | | | | RW | | RW | | | | | | RW | | | | | | | | RW | RW |

| | | |
|---|---|---|
| 23 | BODTE | BODTRIM value write enable. Write only with BODTRIM value. |
| | | 0 BODTRIM field is not updated by writing |
| | | 1 BODTRIM filed can be updated by writing |
| 17<br>16 | BODTRIM | BOD voltage level trim value<br>It can writable when trim enable mode in FMC |

| | | |
|---|---|---|
| 9<br>8 | BODSEL | BOD detect level select |
| | | 00 BOD detect level is 1.8V- 50mV |
| | | 01 BOD detect level is 2.2V – 50mV |
| | | 10 BOD detect level is 2.7V -50mV |
| | | 11 BOD detect level is 4.3V – 50mV |
| 0 | BODEN | BOD Function enable |
| | | 0 BOD is not enabled |
| | | 1 BOD is enabled |

## IOSCTRIM Internal OSC Trim Register

The Internal Oscillator Frequency Trim register. Is a 32-bit register. All trim bits are writable when trim mode in FMC is enabled.

**IOSCTRIM=0x4000_006C**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TSLEN | | | | | TSL | | | LTEN | | LT | | | | | LTM | | UDCEN | | | | UDCH | | UDCL | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 000 | | | 0 | 0 | 0000 | | | | | 00 | 0 | 0 | 0 | 00 | | 000 | | |
| | | | | | | | | W | | | | | RW | | | W | | RW | | | | | RW | W | | | RW | | RW | | |

| | | |
|---|---|---|
| 23 | TSLEN | TSL trim value write enable. Write only with TSL trim value. |
| | | 0 TSL field is not updated by writing |
| | | 1 TSL filed can be updated by writing |
| 18 16 | TSL[2:0] | TSL trim value |
| 15 | LTEN | LTM/LT value write enable. Write only with LTM/LT value |
| | | 0 LT field is not updated by writing |
| | | 1 LT filed can be updated by writing |
| 13 8 | LTM/LT | Internal oscillator LT trim value |
| 7 | UDCEN | UDCH/UDCL value write enable. Write only with UDC value |
| | | 0 UDC field is not updated by writing |
| | | 1 UDC filed can be updated by writing |
| 4 0 | UDCH/UDCL | Internal oscillator UDC trim value |

## EOSCR External Oscillator Control Register

The External Oscillator Control register is a 16-bit register. The external main crystal oscillator has two characteristics. For noise immunity, NMOS amp type is recommended and for the low power characteristic, INV amp type is recommended.

**EOSCR=0x4000_0080**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ISELEN | | | | | | ISEL | | NCEN | | | | | | NCSEL | |
| 0 | 0 | 0 | 0 | 0 | 0 | 00 | | 0 | 0 | 0 | 0 | 0 | 0 | 10 | |
| W | | | | | | RW | | W | | | | | | RW | |

| | | |
|---|---|---|
| 15 | ISELEN | Write enable of bit field ISEL. |
| | | 0 Write access of ISEL field is masked |
| | | 1 Write access of ISEL field is accepted |
| 9 8 | ISEL | Select current. Default 0x0 |
| | | 00 Minimum current driving option |
| | | 01 Low current driving option |

| | | |
|---|---|---|
| | | 10 High current driving option |
| | | 11 Maximum current driving option |
| 7 | NCEN | Write enable of bit field NCSEL |
| | | 0 Write access of NCSEL field is masked |
| | | 1 Write access of NCSEL field is accepted |
| 1 | NCSEL | Select noise cancel delay , default 0x2 |
| 0 | | 00 10ns |
| | | 01 15ns |
| | | 10 20ns |
| | | 11 25ns |

| Freq. (MHz) | ISEL_I<1:0> | NCSEL_I<1:0> | NC DELAY (ns) |
|---|---|---|---|
| 4 | <00> | <11> | 25 |
| 8 | <00> | <10> | 20 |
| 12 | <01> | <01> | 15 |
| 16 | <11> | <00> | 10 |

# EMODR External Mode Status Register

The External Mode Status register shows external mode pin status while booting. This register is an 8-bit register.

**EMODR=0x4000_0084**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | SCANMD | TEST | BOOT |
| | | 0x0 | | | 0 | 0 | - |
| | | R | | | R | R | R |

| | | |
|---|---|---|
| 2 | SCANMD | SCANMD pin level |
| | | 0 SCANMD pin is low |
| | | 1 SCANMD pin is high |
| 1 | TEST | TEST pin level |
| | | 0 TEST pin is low |
| | | 1 TEST pin is high |
| 0 | BOOT | BOOT pin level |
| | | 0 BOOT(PC11) pin is low |
| | | 1 BOOT(PC11) pin is high |

# DBCLK1 Debounce Clock Control Register 1

The Debounce Clock Control Register 1 is a register for PA and PB port pins.

MCCR4=0x4000_009C

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | PBDCSEL | | | PBDDIV | | | | | | | | | | | | | | PADCSEL | | | PADDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 26<br>24 | PBDCSEL | Debounce Clock for Port B source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 23<br>16 | PBDDIV | PORT B Debounce Clock N divider |
| 10<br>8 | PADCSEL | Debounce Clock for Port A source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | PADDIV | PORT A Debounce Clock N divider |

# DBCLK2 Debounce Clock Control Register 2

The Debounce Clock Control Register 2 is a register for PC and PD port pins.

MCCR5=0x4000_00A0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | PDDCSEL | | | PDDDIV | | | | | | | | | | | | | | PCDCSEL | | | PCDDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 26<br>24 | PDDCSEL | Debounce Clock for PORT D source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 23<br>16 | PDDDIV | PORT D Debounce Clock N divider |
| 10<br>8 | PCDCSEL | Debounce Clock for PORT C source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | PCDDIV | PORT C Debounce Clock N divider |

# DBCLK3 Debounce Clock Control Register 3

The Debounce Clock Control Register 3 is a register for PE and PF port pins.

0x4000_00A4

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | PFDCSEL | | | PFDDIV | | | | | | | | | | | | | | PEDCSEL | | | PEDDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x00 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 26 | PFDCSEL | Debounce Clock for PORT F source select bit |
| 24 | | 0xx RING OSC 1MHz |
| | | 100 MCLK (bus clock) |
| | | 101 INT OSC 20MHz |
| | | 110 External Main OSC |
| | | 111 PLL Clock |
| 23 | PFDDIV | PORT F Debounce Clock N divider |
| 16 | | |
| 10 | PEDCSEL | Debounce Clock for PORT E source select bit |
| 8 | | 0xx RING OSC 1MHz |
| | | 100 MCLK (bus clock) |
| | | 101 INT OSC 20MHz |
| | | 110 External Main OSC |
| | | 111 PLL Clock |
| 7 | PEDDIV | PORT E Debounce Clock N divider |
| 0 | | |

# MCCR1 Miscellaneous Clock Control Register 1

The Miscellaneous Clock Control Register 1 sets Trace and SysTick clock sources and dividers.

MCCR1=0x4000_0090

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TRCPOL | | | | | TRCSEL | | | TRACEDIV | | | | | | | | | | | | | STCSEL | | | STDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 100 | | | 0x04 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| W | | | | | RW | | | RW | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 10<br>8 | TRCSEL | TRACE Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | TRACEDIV | TRACE Clock N divider |
| 10<br>8 | STCSEL | SYSTIC Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | STDIV | SYSTIC Clock N divider |

# MCCR2 Miscellaneous Clock Control Register 2

The Miscellaneous Clock Control Register 2 is the clock source and divider register for the MPWM generator units.

MCCR2=0x4000_0094

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | PWM1CSEL | | | PWM1DIV | | | | | | | | | | | | | | PWM0CSEL | | | PWM0DIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x00 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x00 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 10<br>8 | PWM1CSEL | PWM1 Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | PWM1DIV | PWM1 Clock N divider |
| 10<br>8 | PWM0CSEL | PWM0 Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | PWM0DIV | PWM0 Clock N divider |

# MCCR3 Miscellaneous Clock Control Register 3

The Miscellaneous Clock Control Register 3 is the Timer EXT0 Clock and Watch Dog Timer clock control register.

**MCCR3=0x4000_0098**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TEXT0CSEL | | | TEXT0DIV | | | | | | | | | | | | | | WDTCSEL | | | WDTDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | RW | | | | RW | | | | | | | |

| | | |
|---|---|---|
| 10<br>8 | TEXT0CSEL | TEXT0 Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | TEXT0DIV | TEXT0 Clock N divider |
| 10<br>8 | WDTCSEL | WDT Clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | WDTDIV | WDT Clock N divider |

# MCCR4 Miscellaneous Clock Control Register 4

The Miscellaneous Clock Control Register 4 is the alterntative ADC and NMI Debounce Clock Control register.

0x4000_00A8

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | ADCCSEL | | | ADCCDIV | | | | | | | | | | | | | | NMICSEL | | | NMIDDIV | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | | 0x00 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 000 | | | 0x01 | | | | | | | |
| | | | | | RW | | | RW | | | | | | | | | | | | | RW | | | RW | | | | | | | |

| | | |
|---|---|---|
| 26<br>24 | ADCCSEL | ADC clock source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 23<br>16 | ADCCDIV | ADC Clock N divider |
| 10<br>8 | NMIDCSEL | Debounce Clock for NMI source select bit<br>0xx RING OSC 1MHz<br>100 MCLK (bus clock)<br>101 INT OSC 20MHz<br>110 External Main OSC<br>111 PLL Clock |
| 7<br>0 | NMIDDIV | NMI Debounce Clock N divider |

# Functional Description

**System Clock Setup Procedure Example for the Internal Clock with PLL**

- Configure the FM.CFG register to the maximum wait
- Enable the internal clock IOSC in the CSCR register.
- Write 0x02 to the SCCR register (system clock control register) to select the IOSC as the PLL source (FIN) with bypassing the PLL output
- In the PLLCON register, Set bits 14,15 to enable PLL, clear bit 13 to bypass PLL output and configure bits 0-8 to the PREDIV/FBCTRL/POSTDIV for desired PLL output. For full speed, the PLLCON register would be set to 0xC100
- Wait for the PLL to be locked by monitoring the LOCK bit (bit 12) in the PLLCON register.
- Set bit 13 of the PLLCON register to enable the PLL output
- Set bit 0 of SCCR to enable the PLL for the system clock
- Set FM.CFG for the appropriate Flash Wait states for the speed selected.

**System Clock Setup Procedure Example for the External Clock with PLL**

- Enable the Port C peripheral and clock in the SCU PER1 and PCER1 registers
- Unlock the Port Controller using the PORTEN register as defined in PORT CONTROL UNIT (PCU)
- Enable the Alternative function 11b for pins 12 and 13 on PORT C through the PCC_MR register
- Set the Pin type for pins 12 and 13 on PORT C to analog (11b)
- Lock the Port Controller by writing any value to PORTEN register
- Configure the FM.CFG register to the maximum wait
- If not already enabled, enable Internal oscillator in CSCR
- Set bit 3 in the CMR register to monitor External Oscillator
- Enable External Oscillator in CSCR register
- Wait for bit 0 of the CMR register to be set. Note: if the external oscillator does not start, this bit will never be set.
- Wait for an additional time (more than 1 ms) to allow the oscillator to stabilize
- Write 0x06 to the SCCR register (system clock control register) to select the External Oscillator as the PLL source (FIN)
- Set PLLCON high byte (8-15) to 0xC and low byte (0-7) to the FBCTRL/POSTDIV for desired PLL output.
- Wait until bit 12 of PLLCON is set. Note: if the PLL does not lock, this bit will never be set.
- Set bit 13 to enable the PLL output
- Set bit 0 of SCCR to enable the PLL for the system clock
- Set FM.CFG for the appropriate Flash Wait states for the speed selected.

**To Enable Clock Out for monitoring actual clock output**

- Enable the Port C peripheral and clock in the SCU PER1 and PCER1 registers
- Unlock the Port Controller using the PORTEN register as defined in PORT CONTROL UNIT (PCU)
- Enable the Alternative function 01b for pin 9 on PORT C through the PCC_MR register
- Set the Pin type for pin 9 on PORT C to output (00b)
- Lock the Port Controller by writing any value to PORTEN register
- Set bit 4 of the Clock Output Register (COR) to enable the output
- Configure the CLKODIV to the desired output divider

# 5. Port Control Unit

## Overview

Port Control Unit (PCU) controls the external I/Os as follows:

- Set the multiplex state of each pin (for alternative functions)
- Set external signal type (Analog / Push-Pull output /Open Drain output /Input)
- Set enable/monitor/trigger type for interrupts for each pin
- Set internal pull-up register control for each pin
- Set debounce for each pin

**Note:** You must enable both the Port Peripheral and the Port Peripheral CLOCK in PER1/PCER1 to use the pins of the port.

Figure 5.1 shows a block diagram of the PCU. Figures 2.2 and 2.3 show I/O Port Block diagrams.

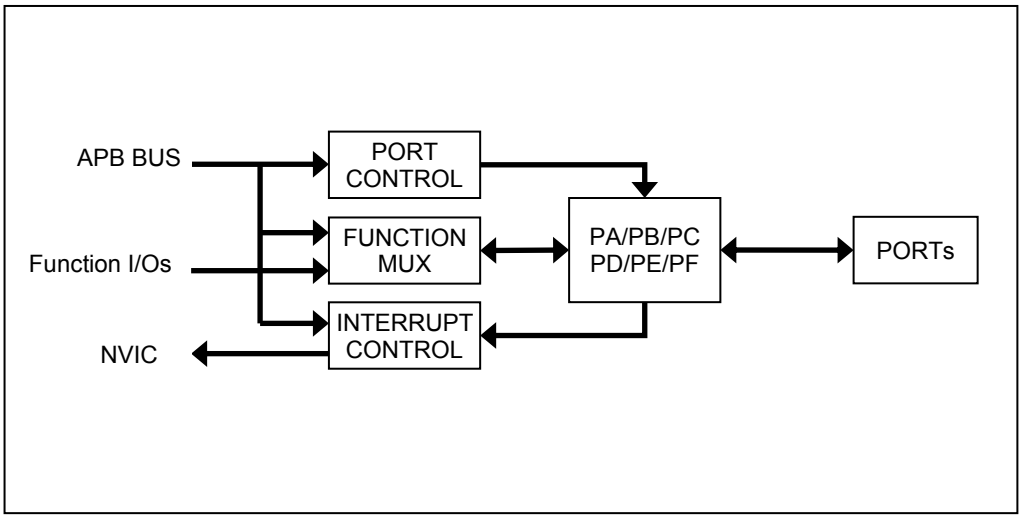



**Figure 5.1. Block Diagram**

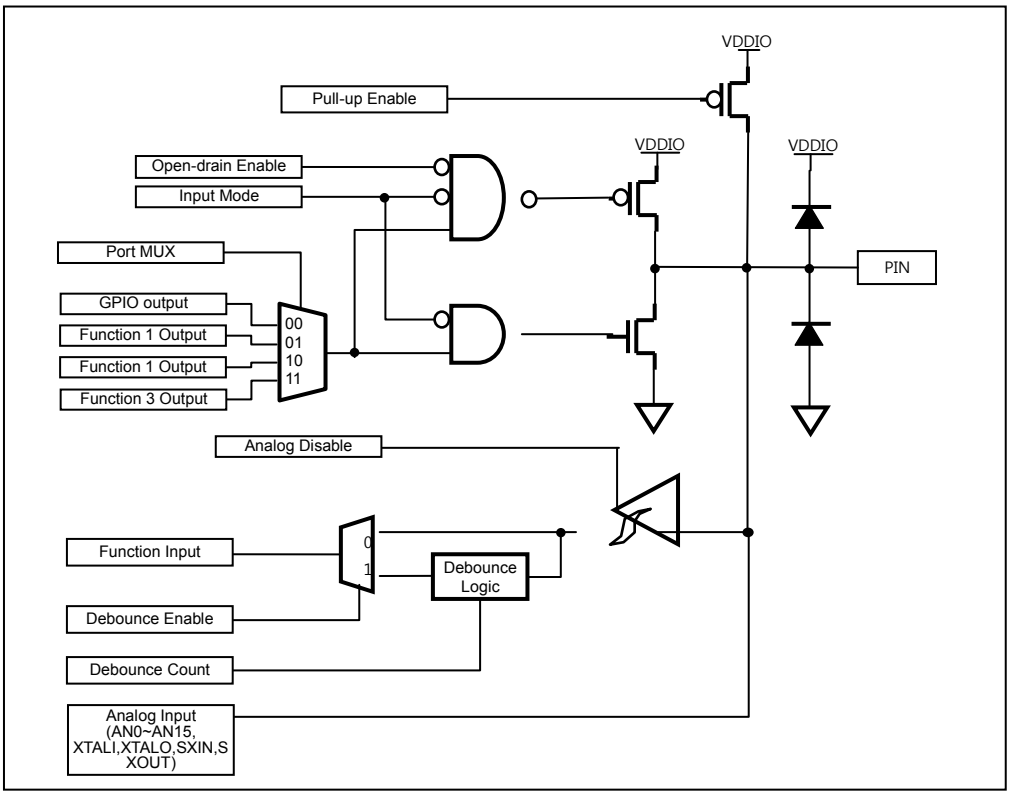



**Figure 5.2. I/O Port Block Diagram (ADC and External Oscillator Pins)**

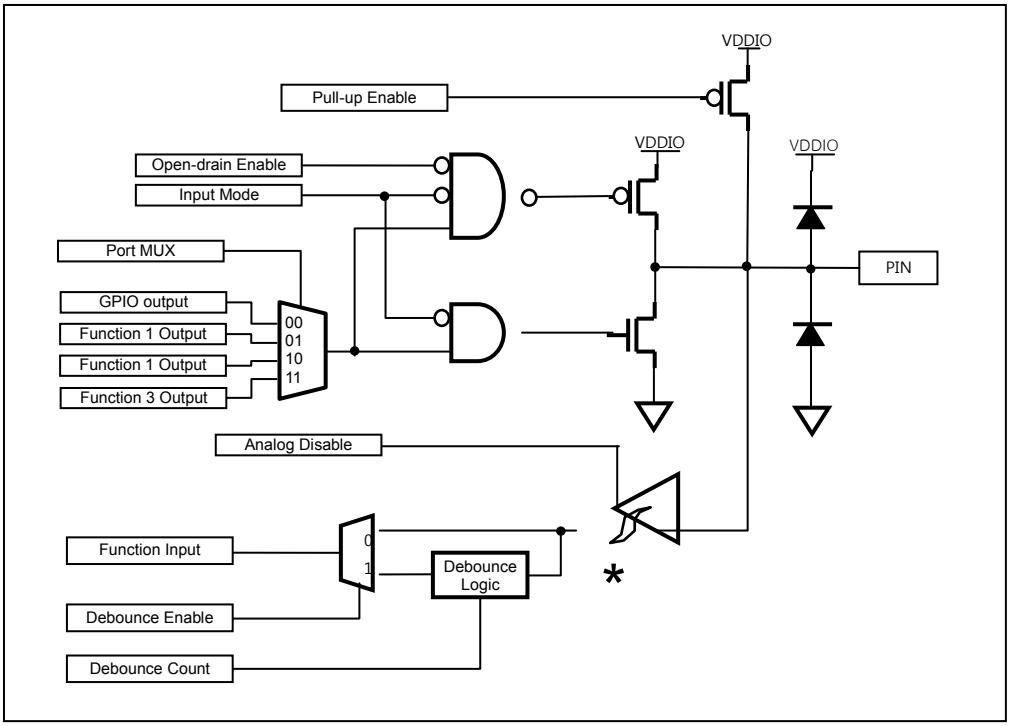



**Figure 5.3. I/O Port Block Diagram (General I/O Pins)**

# Pin Multiplexing

GPIO pins have alternative function pins. Table 5.1 lists the pin multiplexing information.

**Table 5.1. GPIO Alternative Function**

| PORT | | FUNCTION | | | |
|---|---|---|---|---|---|
| | | 00 | 01 | 10 | 11 |
| PA | 0 | **PA0*** | | | AN0 |
| | 1 | **PA1*** | | | AN1 |
| | 2 | **PA2*** | | | AN2 |
| | 3 | **PA3*** | | | AN3 |
| | 4 | **PA4*** | | T0IO | AN4 |
| | 5 | **PA5*** | | T1IO | AN5 |
| | 6 | **PA6*** | | T2IO | AN6 |
| | 7 | **PA7*** | | T3IO | AN7 |
| | 8 | **PA8*** | | | AN8 |
| | 9 | **PA9*** | | | AN9 |
| | 10 | **PA10*** | | | AN10 |
| | 11 | **PA11*** | | | AN11 |
| | 12 | **PA12*** | SS0 | | AN12 |
| | 13 | **PA13*** | SCK0 | | AN13 |
| | 14 | **PA14*** | MOSI0 | | AN14 |
| | 15 | **PA15*** | MISO0 | | AN15 |
| PB | 0 | **PB0*** | MP0UH | | |
| | 1 | **PB1*** | MP0UL | | |
| | 2 | **PB2*** | MP0VH | | |
| | 3 | **PB3*** | MP0VL | | |
| | 4 | **PB4*** | MP0WH | | |
| | 5 | **PB5*** | MP0WL | | |
| | 6 | **PB6*** | PRTIN0 | WDTO(2) | |
| | 7 | **PB7*** | OVIN0 | STBYO(2) | |
| | 8 | **PB8*** | PRTIN1 | RXD3 | |
| | 9 | **PB9*** | OVIN1 | TXD3 | |
| | 10 | **PB10*** | MP1UH | | |
| | 11 | **PB11*** | MP1UL | | |
| | 12 | **PB12*** | MP1VH | | |
| | 13 | **PB13*** | MP1VL | | |
| | 14 | **PB14*** | MP1WH | | |
| | 15 | **PB15*** | MP1WL | | |

***(*) mark indicates default pin setting.***
***(2) mark indicates secondary port***

**Table 5.1. GPIO Alternative Function (Continued)**

| PORT | | FUNCTION | | | |
|---|---|---|---|---|---|
| | | 00 | 01 | 10 | 11 |
| PC | 0 | PC0 | **TCK/SWCLK*** | RXD0(2) | |
| | 1 | PC1 | **TMS/SWDIO*** | RXD0(2) | |
| | 2 | PC2 | **TDO/SWO*** | | |
| | 3 | PC3 | **TDI*** | | |
| | 4 | PC4 | **nTRST*** | T0IO(2) | |
| | 5 | **PC5*** | RXD1 | T1IO(2) | |
| | 6 | **PC6*** | TXD1 | T2IO(2) | |
| | 7 | **PC7*** | SCL0 | T3IO(2) | |
| | 8 | **PC8*** | SDA0 | T4IO(2) | |
| | 9 | **PC9*** | CLKO | T8IO | |
| | 10 | PC10 | **nRESET*** | | |
| | 11 | **PC11/BOOT*** | | T8IO(2) | |
| | 12 | **PC12*** | | | XIN |
| | 13 | **PC13*** | | | XOUT |
| | 14 | **PC14*** | RXD0 | MISO0(2) | |
| | 15 | **PC15*** | TXD0 | MOSI0(2) | |
| PD | 0 | **PD0*** | SS1 | | SXIN |
| | 1 | **PD1*** | SCK1 | | SXOUT |
| | 2 | **PD2*** | MOSI1 | | |
| | 3 | **PD3*** | MISO1 | | |
| | 4 | **PD4*** | SCL1 | | AN16 |
| | 5 | **PD5*** | SDA1 | | AN17 |
| | 6 | **PD6*** | TXD2 | I | AN18 |
| | 7 | **PD7*** | RXD2 | | AN19 |
| | 8 | **PD8*** | T6IO(2) | WDTO | |
| | 9 | **PD9*** | T7IO(2) | STBO | |
| | 10 | **PD10*** | AD0SOC | T0IO | |
| | 11 | **PD11*** | AD0EOC | T1IO | |
| | 12 | **PD12*** | AD1SOC | T2IO | |
| | 13 | **PD13*** | AD1EOC | T3IO | |
| | 14 | **PD14*** | | SS0* | |
| | 15 | **PD15*** | | SCK0* | |

***(*) mark indicates default pin setting.***
***(2) mark indicates secondary port***

**Table 5.1. GPIO Alternative Function (Continued)**

| PORT | | FUNCTION | | | |
|---|---|---|---|---|---|
| | | 00 | 01 | 10 | 11 |
| PE | 0 | **PE0** | | TXD1 | |
| | 1 | **PE1** | | RXD1 | |
| | 2 | **PE2** | | T4I(3)/T3O(5) | |
| | 3 | **PE3** | AD0O* | SCL0* | |
| | 4 | **PE4** | AD1O* | SDA0* | |
| | 5 | **PE5*** | | T5IO | |
| | 6 | **PE6*** | | T5IO(2) | |
| | 7 | **PE7*** | | T6IO(2) | |
| | 8 | **PE8*** | | T7IO(2) | |
| | 9 | **PE9*** | | T8IO(2) | |
| | 10 | **PE10** | | T9IO(2) | |
| | 11 | **PE11** | T0IO(2) | SCL1* | |
| | 12 | **PE12*** | T1IO(2) | SDA1* | |
| | 13 | **PE13*** | T2IO(2) | TXD2* | |
| | 14 | **PE14*** | T3IO(2) | RXD2* | |
| | 15 | **PE15*** | T4IO(2) | | |
| PF | 0 | **PF0*** | | | |
| | 1 | **PF1*** | | | |
| | 2 | **PF2*** | | | AN20 |
| | 3 | **PF3*** | | | AN21 |
| | 4 | **PF4*** | | | |
| | 5 | **PF5*** | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

***(*) mark indicates default pin setting.***
***(2) mark indicates secondary port***

# Registers

The base address of the PCU block is `0x4000_1000`.

**Table 5.2. Base Address of Port**

| PORT | ADDRESS |
|---|---|
| **PA** | 0x4000_1000 |
| **PB** | 0x4000_1100 |
| **PC** | 0x4000_1200 |
| **PD** | 0x4000_1300 |
| **PE** | 0x4000_1400 |
| **PF** | 0x4000_1500 |

**Table 5.3. PCU Register Map**

| Register | Offset | R/W | Description |
|---|---|---|---|
| **PC*n*.MR** | 0x--00 | R/W | Port *n* pin mux select register |
| **PC*n*.CR** | 0x--04 | R/W | Port *n* pin control register |
| **PC*n*.PCR** | 0x--08 | R/W | Port *n* internal pull-up control register |
| **PCn.DER** | 0x--0C | R/W | Port *n* debounce register |
| **PC*n*.IER** | 0x--10 | R/W | Port *n* interrupt enable register |
| **PC*n*.ISR** | 0x--14 | R/W | Port *n* interrupt status register |
| **PC*n*.ICR** | 0x--18 | R/W | Port *n* interrupt control register |
| | | | |
| **PORTEN** | 0x1FF0 | R/W | Port Access enable |

# PCA.MR Port A Pin Mux Register

The Port A Pin Mux register is the PA port mode select register. This register and the PERx and PCERx registers must be configured correctly before using the port to guarantee its functionality. PERx enables the port and PCERx enables the clock to the port.

**PCA.MR=0x4000_1000**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PA15 | PA14 | PA13 | PA12 | PA11 | PA10 | PA9 | PA8 | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PA0** | PA0 | | | AN0 |
| **PA1** | PA1 | | | AN1 |
| **PA2** | PA2 | | | AN2 |
| **PA3** | PA3 | | | AN3 |
| **PA4** | PA4 | | T0IO | AN4 |
| **PA5** | PA5 | | T1IO | AN5 |
| **PA6** | PA6 | | T2IO | AN6 |
| **PA7** | PA7 | | T3IO | AN7 |
| **PA8** | PA8 | | | AN8 |
| **PA9** | PA9 | | | AN9 |
| **PA10** | PA10 | | | AN10 |
| **PA11** | PA11 | | | AN11 |
| **PA12** | PA12 | SS0 | | AN12 |
| **PA13** | PA13 | SCK0 | | AN13 |
| **PA14** | PA14 | MOSI0 | | AN14 |
| **PA15** | PA15 | MISO0 | | AN15 |

* : 2nd function

# PCB.MR Port B Pin Mux Register

The Port B Pin Mux register is the PB port mode select register. This register and the PERx and PCERx registers must be configured correctly before using the port to guarantee its functionality. PERx enables the port and PCERx enables the clock to the port.

**PCB.MR=0x4000_1100**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PB15 | PB14 | PB13 | PB12 | PB11 | PB10 | PB9 | PB8 | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PB0** | PB0 | MP0UH | | |
| **PB1** | PB1 | MP0UL | | |
| **PB2** | PB2 | MP0VH | | |
| **PB3** | PB3 | MP0VL | | |
| **PB4** | PB4 | MP0WH | T9IO | |
| **PB5** | PB5 | MP0WL | T9IO(2) | |
| **PB6** | PB6 | PRTIN0 | WDTO(2) | |
| **PB7** | PB7 | OVIN0 | STBYO(2) | |
| **PB8** | PB8 | PRTIN1 | RXD3 | |
| **PB9** | PB9 | OVIN1 | TXD3 | |
| PB10 | PB10 | MP1UH | | |
| PB11 | PB11 | MP1UL | | |
| PB12 | PB12 | MP1VH | | |
| PB13 | PB13 | MP1VL | | |
| PB14 | PB14 | MP1WH | | |
| PB15 | PB15 | MP1WL | | |

# PCC.MR Port C Pin Mux Register

The Port C Pin Mux register is the PC port mode select register. This register and the PERx and PCERx registers must be configured correctly before using the port to guarantee its functionality. PERx enables the port and PCERx enables the clock to the port.

**PCC.MR=0x4000_1200**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PC15 | PC14 | PC13 | PC12 | PC11 | PC10 | PC9 | PC8 | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| 00 | 00 | 00 | 00 | 00 | 01 | 00 | 00 | 00 | 00 | 00 | 01 | 01 | 01 | 01 | 01 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PC0** | PC0 | **TCK/SWCLK*** | RXD0(2) | |
| **PC1** | PC1 | **TMS/SWDIO*** | TXD0(2) | |
| **PC2** | PC2 | **TDO/SWO*** | | |
| **PC3** | PC3 | **TDI*** | | |
| **PC4** | PC4 | **nTRST*** | T0IO(2) | |
| **PC5** | PC5 | RXD1 | T1IO(2) | |
| **PC6** | PC6 | TXD1 | T2IO(2) | |
| **PC7** | PC7 | SCL0 | T3IO(2) | |
| **PC8** | PC8 | SDA0 | T4IO(2) | |
| **PC9** | PC9 | CLKO | T8IO | |
| **PC10** | PC10 | **nRESET*** | | |
| **PC11** | PC11/BOOT* | | T8IO(2) | |
| **PC12** | PC12 | | | XIN |
| **PC13** | PC13 | | | XOUT |
| **PC14** | PC14 | RXD0 | MISO0(2) | |
| **PC15** | PC15 | TXD0 | MOSI0(2) | |

* : 2nd function

# PCD.MR Port D Pin Mux Register

The Port D Pin Mux register is the PD port mode select register. This register and the PERx and PCERx registers must be configured correctly before using the port to guarantee its functionality. The PERx enables the port and PCERx enables the clock to the port.

**PCD.MR=0x4000_1300**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PD15 | PD14 | PD13 | PD12 | PD11 | PD10 | PD9 | PD8 | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PD0** | PD0 | SS1 | | SXIN |
| **PD1** | PD1 | SCK1 | | SXOUT |
| **PD2** | PD2 | MOSI1 | | |
| **PD3** | PD3 | MISO1 | | |
| **PD4** | PD4 | SCL1 | | AN16 |
| **PD5** | PD5 | SDA1 | | AN17 |
| **PD6** | PD6 | TXD2 | | AN18 |
| **PD7** | PD7 | RXD2 | | AN19 |
| **PD8** | PD8 | T6IO | WDTO | |
| **PD9** | PD9 | T7IO | STBO | |
| **PD10** | PD10 | AD0SOC | T0IO | |
| **PD11** | PD11 | AD0EOC | T1IO | |
| **PD12** | PD12 | AD1SOC | T2IO | |
| **PD13** | PD13 | AD1EOC | T3IO | |
| **PD14** | PD14 | | SS0* | |
| **PD15** | PD15 | | SCK0* | |

* : 2nd function

# PCE.MR Port E Pin Mux Register

The Port E Pin Mux register is the PE port mode select register. This register and the PERx and PCERx registers must be configured properly before using the port to guarantee its functionality. The PERx enables the port and PCERx enables the clock to the port.

**PCE.MR=0x4000_1400**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PE15 | PE14 | PE13 | PE12 | PE11 | PE10 | PE9 | PE8 | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PE0** | PE0 | | TXD1 | |
| **PE1** | PE1 | | RXD1 | |
| **PE2** | PE2 | | T4I(3)/T3O(5) | |
| **PE3** | PE3 | | SCL0* | |
| **PE4** | PE4 | | SDA0* | |
| **PE5** | PE5 | | T5IO | |
| **PE6** | PE6 | | T5IO(2) | |
| **PE7** | PE7 | | T6IO(2) | |
| **PE8** | PE8 | | T7IO(2) | |
| **PE9** | PE9 | | T8IO(2) | |
| **PE10** | PE10 | | T9IO(2) | |
| **PE11** | PE11 | T0IO(2) | SCL1* | |
| **PE12** | PE12 | T1IO(2) | SDA1* | |
| **PE13** | PE13 | T2IO(2) | TXD2* | |
| **PE14** | PE14 | T3IO(2) | RXD2* | |
| **PE15** | PE15 | T4IO(2) | | |

* : 2nd function , ** : 3rd function

# PCF.MR Port F Pin Mux Register

The Port F Pin Mux register is the PF port mode select register. This register and the PERx and PCERx registers must be configured correctly before using the port to guarantee its functionality. The PERx enables the port and PCERx enables the clock to the port.

**PCF.MR=0x4000_1500**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | | | | PF5 | | PF4 | | PF3 | | PF2 | | PF1 | | PF0 | |
| 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | | 00 | |
| | | | | | | | | | | | | | | | | | | | | RW | | RW | | RW | | RW | | RW | | RW | |

| PORT | SELECTION BIT | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| **PF0** | PF0 | | | |
| **PF1** | PF1 | | | |
| **PF2** | PF2 | | | AN20 |
| **PF3** | PF3 | | | AN21 |
| **PF4** | PF4 | | | |
| **PF5** | PF5 | | | |

## PCn.CR Port n Pin Control Register (Except for PCC.CR)

The Port n Pin Control register handles the input or output control of each port pin. Each pin can be configured as input pin, output pin, or open-drain pin.

**PCA.CR=0x4000_1004, PCB.CR=0x4000_1104**
**PCD.CR=0x4000_1304, PCE.CR=0x4000_1404, PCF.CR=0x4000_1504**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| Pn | Port control |
|---|---|
| | 00 Push-pull output |
| | 01 Open-drain output |
| | 10 Input |
| | 11 Analog |

### 5.1.1 PCC.CR Port C Pin Control Register

The Port C Pin Control register handles the input or output control of each port pin. Each pin can be configured as input pin, output pin, or open-drain pin.

**PCC.CR=0x4000_1204**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| 11 | 11 | 11 | 11 | 10 | 10 | 11 | 11 | 11 | 11 | 11 | 10 | 10 | 00 | 10 | 10 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| Pn | Port control |
|---|---|
| | 00 Push-pull output |
| | 01 Open-drain output |
| | 10 Input |
| | 11 Analog |

## PCn.PCR Port n Pull-up Resistor Control Register

Every pin in the port has on-chip pull-up resistors which can be configured by the Port n Pull-up Resistor Control registers.

**PCA.PCR=0x4000_1008, PCB.PCR=0x4000_1108, PCC.PCR=0x4000_1208**
**PCD.PCR=0x4000_1308, PCE.PCR=0x4000_1408, PCF.PCR=0x4000_1508**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PUE15 | PUE14 | PUE13 | PUE12 | PUE11 | PUE10 | PUE9 | PUE8 | PUE7 | PUE6 | PUE5 | PUE4 | PUE3 | PUE2 | PUE1 | PUE0 |
| 0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| n | PUEn | Port pull-up control |
|---|---|---|

| | |
|---|---|
| 0 | Disable pull-up resistor |
| 1 | Enable pull-up resister |

## PCn.DER Port n Debounce Enable Register

Every pin in the port has a digital debounce filter which can be configured by the Port n Debounce Enable registers. The Debounce clock can be configured in the DBCLKx registers in the SCU.

**PCA.DER=0x4000_100C, PCB.DER=0x4000_110C, PCC.DER=0x4000_120C**
**PCD.DER=0x4000_130C, PCE.DER=0x4000_140C, PCF.DER=0x4000_150C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDE15 | PDE14 | PDE13 | PDE12 | PDE11 | PDE10 | PDE9 | PDE8 | PDE7 | PDE6 | PDE5 | PDE4 | PDE3 | PDE2 | PDE1 | PDE0 |
| 0000 | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| n | PDEn | Pin debounce enable |
| | 0 | Disable debounce filter |
| | 1 | Enable debounce filter |

## PCn.IER Port n Interrupt Enable Register

Each individual pin can be an external interrupt source. Edge trigger interrupt and level trigger interrupt are both supported. Interrupt mode can be configured by setting the Port n Interrupt Enable registers

**PCA.IER=0x4000_1010, PCB.IER=0x4000_1110, PCC.IER=0x4000_1210**
**PCD.IER=0x4000_1310, PCE.IER=0x4000_1410, PCF.IER=0x4000_1510**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIE15 | PIE14 | PIE13 | PIE12 | PIE11 | PIE10 | PIE9 | PIE8 | PIE7 | PIE6 | PIE5 | PIE4 | PIE3 | PIE2 | PIE1 | PIE0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PIEn | Pin interrupt enable |
|---|---|
| 00 | Interrupt disabled |
| 01 | Enable interrupt as level trigger mode |
| 10 | Reserved |
| 11 | Enable interrupt as edge trigger mode |

## PCn.ISR Port n Interrupt Status Register

When an interrupt is delivered to the CPU, the interrupt status can be detected by reading the Port n Interrupt Status register. This register reports a source pin of interrupt and a type of interrupt.

**PCA.ISR=0x4000_1014, PCB.ISR=0x4000_1114, PCC.ISR=0x4000_1214**
**PCD.ISR=0x4000_1314, PCE.ISR=0x4000_1414, PCF.ISR=0x4000_1514**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIS15 | PIS14 | PIS13 | PIS12 | PIS11 | PIS10 | PIS9 | PIS8 | PIS7 | PIS6 | PIS5 | PIS4 | PIS3 | PIS2 | PIS1 | PIS0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| PISn | Pin interrupt status |
|---|---|
| 00 | No interrupt event |
| 01 | Low level interrupt or Falling edge interrupt event is present |
| 10 | High level interrupt or rising edge interrupt event is present |
| 11 | Both of rising and falling edge interrupt event is present in edge trigger interrupt mode.<br>Not available in level trigger interrupt mode |

## PCn.ICR Port n Interrupt Control Register

The Port n Interrupt Control register is the interrupt mode control register. Edge interrupt produces a pulsed interrupt while a level interrupt maintains the interrupt as long as the pin is in the defined state (low or high).

**PCA.ICR=0x4000_1018, PCB.ICR=0x4000_1118, PCC.ICR=0x4000_1218**
**PCD.ICR=0x4000_1318, PCE.ICR=0x4000_1418, PCF.ICR=0x4000_1518**

| 31 30 | 29 28 | 27 26 | 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIC15 | PIC14 | PIC13 | PIC12 | PIC11 | PIC10 | PIC9 | PIC8 | PIC7 | PIC6 | PIC5 | PIC4 | PIC3 | PIC2 | PIC1 | PIC0 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| | PICn | Pin interrupt mode |
|---|---|---|
| | | 00 Prohibit external interrupt |
| | | 01 Low level interrupt or Falling edge interrupt mode |
| | | 10 High level interrupt or rising edge interrupt mode |
| | | 11 Both of rising and falling edge interrupt mode.<br>Not support for level trigger mode |

## PORTEN Port Access Enable

The Port Access Enable register enables register writing permission of all PCU registers.

**PORTEN=0x4000_1FF0**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PORTEN | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -- | | | | | | | |
| | | | | | | | | WO | | | | | | | |

| | | |
|---|---|---|
| 7<br>0 | PORTEN | Writing the sequence of 0x15 and 0x51 in this register enables writing to PCU registers, and writing other values protects all PCU registers from writing. |

# Functional Description

All the GPIO pins can be configured for different operations, inputs, outputs, triggered interrupts (both level and edge) through the PCU. The system is also able to disable ports by setting the PER1 and PCER1 registers in the SCU. By default, all pins are disabled (except for UART0/SPI0) so the developer must enable these to operate.

All configuration parameters are protected by the Port Access Enable register. You must write the sequence in order (`0x15`, `0x51`) to the PORTEN register to configure any pin(s). After the configuration is complete, write any other value to the PORTEN register to lock it.

**Note:** Do not read in between the sequence, that will prevent the configuration registers from being unlocked.

# 6. General Purpose I/O

## Overview

Most of the pins except the dedicated function pins can be used as general I/O ports. General input/output ports are controlled by the GPIO block.

- Output signal level (H/L) select

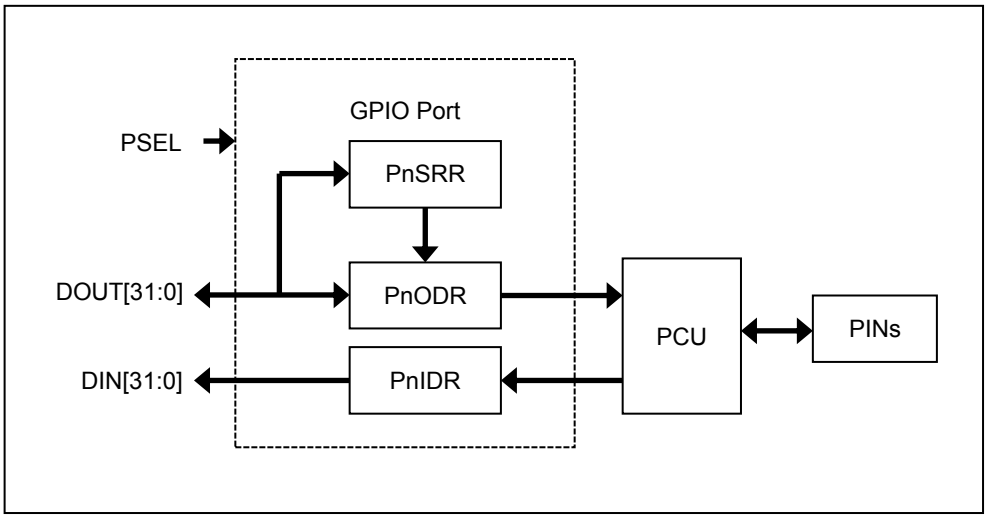



**Figure 6.1. Block Diagram**

# Pin Description

Table 6.1. External Signal

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **PA** | IO | PA0 - PA15 |
| **PB** | IO | PB0 - PB15 |
| **PC** | IO | PC0 - PC15 |
| **PD** | IO | PD0 - PD15 |
| **PE** | IO | PE0 – PE15 |
| **PF** | IO | PF0 – PF5 |

# Registers

The base address of GPIO is `0x4000_2000` and the register map is described in Tables 6.2 and 6.3.

Table 6.2. Base Address of Each Port

| PORT | Address |
|---|---|
| **PA PORT** | 0x4000_2000 |
| **PB PORT** | 0x4000_2100 |
| **PC PORT** | 0x4000_2200 |
| **PD PORT** | 0x4000_2300 |
| **PE PORT** | 0x4000_2400 |
| **PF PORT** | 0x4000_2500 |

Table 6.3. GPIO Register Map

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **Pn.ODR** | 0x--00 | R/W | Port *n* Output data register | 0x00000000 |
| **Pn.IDR** | 0x--04 | RO | Port *n* Input data register | 0x00000000 |
| **Pn.BSR** | 0x--08 | WO | Port *n* Pin set register | 0x00000000 |
| **Pn.BCR** | 0x—0C | WO | Port *n* Pin clear register | 0x00000000 |

## Pn.ODR Port n Output Data Register

When the pin is set to output and GPIO mode, the pin output level is defined by the Port n Output Data registers.

**PA.ODR=0x4000_2000, PB.ODR=0x4000_2100, PC.ODR=0x4000_2200**
**PD.ODR=0x4000_2300, PE.ODR=0x4000_2400, PF.ODR=0x4000_2500**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ODR | | | | | | | | | | | | | | | |
| 0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | |
|---|---|
| ODR | Pin output level<br>0 Output low level<br>1 Output high level |

## Pn.IDR Port n Input Data Register

Each pin level status can be read in the Port n Input Data register. Even if the pin is in alternative mode except analog mode, the pin level can be detected in this register.

**PA.IDR=0x4000_2004, PB.IDR=0x4000_2104, PC.IDR=0x4000_2204**
**PD.IDR=0x4000_2304, PE.IDR=0x4000_2404, PF.IDR=0x4000_2504**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PnIDR | | | | | | | | | | | | | | | |
| 0000 | | | | | | | | | | | | | | | |
| Read Only | | | | | | | | | | | | | | | |

| | |
|---|---|
| IDR | Pin current level<br>0 The pin is low level<br>1 The pin is high level |

## Pn.BSR Port n Bit Set Register

The Port n Bit Set register controls each bit of of the Port n Output Data register (PnODR). When you write “1” to a specific bit, the corresponding bit in the PnODR register is set.

**PA.BSR=0x4000_2008, PB.BSR=0x4000_2108, PC.BSR=0x4000_2208**
**PD.BSR=0x4000_2308, PE.BSR=0x4000_2408, PF.BSR=0x4000_2508**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BSR | | | | | | | | | | | | | | | |
| 0000 | | | | | | | | | | | | | | | |
| Write Only | | | | | | | | | | | | | | | |

| | |
|---|---|
| BSR | Pin current level<br>0 Not effect<br>1 Set correspondent bit in PnODR register |

## Pn.BCR Port n Bit Clear Register

The Port n Bit Clear register controls each bit of the Port n Output Data register (PnODR). When you write “1” to a specific bit, the corresponding bit in the PnODR register is cleared.

**PA.BCR=0x4000_200C, PB.BCR=0x4000_210C, PC.BCR=0x4000_220C**
**PD.BCR=0x4000_230C, PE.BCR=0x4000_240C, PF.BCR=0x4000_250C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BCR | | | | | | | | | | | | | | | |
| 0000 | | | | | | | | | | | | | | | |
| Write Only | | | | | | | | | | | | | | | |

| BCR | Pin current level |
|---|---|
| 0 | Not effect |
| 1 | Clear correspondent bit in PnODR register |

# Functional Description

The GPIO registers provide the input/output condition of the GPIO pins. The input data registers give the states of the pins of the ports. The output data register is for setting the port pins. The Set and Clear registers control the pins at the individual level.

# 7. Flash Memory Controller

## Introduction

Flash Memory Controller is an internal Flash memory interface controller with the following features:

- 384KB Flash code memory
- 32-bit data bus width
- Code cache block for fast access mode
- 256-byte page size
- Supports page erase and macro erase
- 256-byte unit program

| Item | Description |
|---|---|
| **Size** | 384KB |
| **Start Address** | 0x0000_0000 |
| **End Address** | 0x0005_FFFF |
| **Page Size** | 256-byte |
| **Total Page Count** | 1,536 pages |
| **PGM Unit** | 256-byte |
| **Erase Unit** | 256-byte |

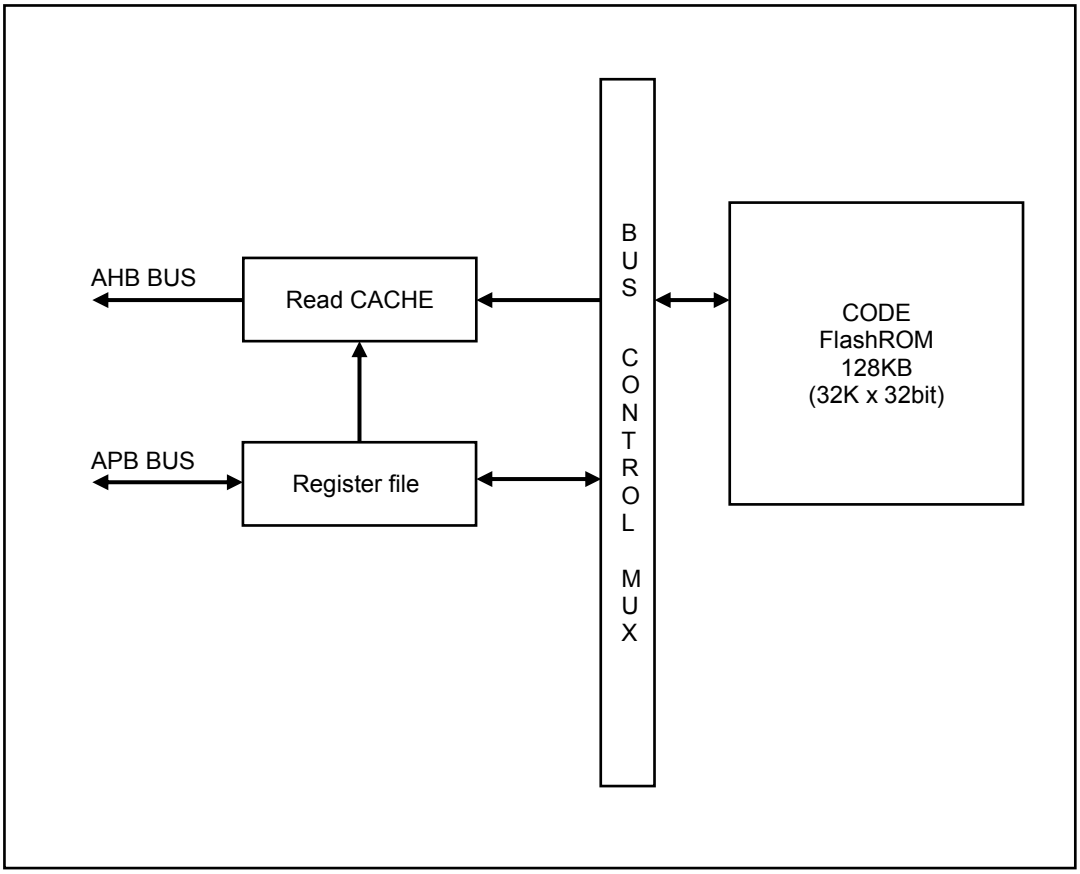



**Figure 7.1. Block Diagram**

# Pin Description

There are no external interface pins for this peripheral.

# Registers

The base address of the Flash Memory Controller is shown in Table 7.1.

**Table 7.1. Flash Memory Controller Base Address**

| | Address |
|---|---|
| **Flash Controller** | 0x4000_0100 |

Table 7.2 shows the register memory map.

**Table 7.2. Flash Memory Controller Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **FM.MR** | 0x0004 | R/W | Flash Memory Mode Select register | 0x01000000 |
| **FM.CR** | 0x0008 | R/W | Flash Memory Control register | 0x82000000 |
| **FM.AR** | 0x000C | R/W | Flash Memory Address register | 0x00000000 |
| **FM.DR** | 0x0010 | R/W | Flash Memory Data register | 0x00000000 |
| **FM.TMR** | 0x0014 | R/W | Flash Memory Timer register | 0x000000bb |
| **FM.DRTY** | 0x0018 | R/W | Flash Memory Dirty bit | |
| **FM.TICK** | 0x001C | RO | Flash Memory Tick Timer | 0x00000000 |
| **FM.CRC** | 0x0020 | RO | Flash Memory Read CRC Value | |
| **FM.CFG** | 0x0030 | R/W | Flash Memory config value register | 0x00000000 |
| **FM.OTPCR** | 0x0034 | R/W | Flash OTP control register | 0x00000000 |
| **FM.BOOTCR** | 0x0074 | R/W | Boot ROM Remap Clear register | 0x00000000 |
| **FM.PROT** | 0x0078 | R/W | Flash Page protection register | 0x00000000 |
| **FM.JTAGEN** | 0x007C | R/W | Jtag protection register | 0x00000001 |

# FM.MR Flash Memory Mode Register

The Internal Flash Memory Mode register is a 32-bit register.

**FM.MR=0x4000_0104**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BOOT | | | | | | | IDLE | VERIFY | AMBAEN | | | | | TRMEN | TRM | | | | | | | FEMOD | FMOD | ACODE | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x00 | | | | | | | |
| R | | | | | | | R | RW | RW | | | | | R | R | | | | | | | R | R | RW | | | | | | |

| | | | |
|---|---|---|---|
| 31 | BOOT | 0 | |
| | | 1 | Boot mode enable status(read only) |
| 24 | IDLE | 0 | |
| | | 1 | Boot mode enable status(read only) |
| 23 | VERIFY | 0 | |
| | | 1 | Flash Verify mode enable status(read only) |
| 22 | AMBAEN | 0 | AMBA mode disable |
| | | 1 | AMBA mode enable (can change wait state and etc) |
| 17 | TRMEN | 0 | |
| | | 1 | Trim mode entry status(read only) |
| 16 | TRM | 0 | |
| | | 1 | Trim mode status(read only) |
| 9 | FEMOD | 0 | |
| | | 1 | Flash mode entry status(read only) |
| 8 | FMOD | 0 | |
| | | 1 | Flash mode status(read only) |
| 7 | ACODE | 5A → A5 | Flash mode |
| 0 | | A5 → 5A | Trim mode |
| | | 81 → 28 | AMBA mode |

# FM.CR Flash Memory Control Register

The Internal Flash Memory Control register is shown below.

**FM.CR=0x4000_0108**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | TIMER | | | TEST1 | TEST0 | VPPOUT | EVER | PVER | RESERVED | RESERVED | RESERVED | PPGM | AE | | | PMOD | WE | PBLD | PGM | ERS | PBR |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | RW | RW | RW | RW | | RW | RW | RW | RW | | RW | | | RW | RW | R | RW | RW | | RW | RO | RW | RW | | | RW | RW | RW | RW | RW | RW |

| | | | |
|---|---|---|---|
| | | 1 | |
| 20 | TIMER | 0 | Program/Erase timer enable |
| | | 1 | (timer can be enable by PGM or ERS bit) |
| 17 | TEST[1:0] | 00 | Normal operation |
| 16 | | 01 | (read) Row voltage mode |
| | | 01 | (write) ODD Row program |
| | | 10 | Even Row program |
| | | 11 | All Row program |
| 15 | VPPOUT | | Enable charge-pump Vpp output |
| 14 | EVER | | Set erase verify mode |
| 13 | PVER | | Set program verify mode |
| 12 | RESERVED | | Reserved |
| 11 | RESERVED | | Reserved |
| 10 | RESERVED | | Reserved |
| 9 | PPGM | | Pre PGM enable<br>Page buffer set automatically |
| 8 | AE | | All erase enable |
| 5 | PMODE | | PMODE enable(Address path changing) |
| 4 | WE | | Write enable |
| 3 | PBLD | | Page buffer load (PMODE should be set) |
| 2 | PGM | | Program enable |
| 1 | ERS | 0 | Program mode enable |
| | | 1 | Erase mode enable |
| 0 | PBR | | Page buffer reset |

## FM.AR Flash Memory Address Register

The Flash Memory Address register is the internal Flash Memory program/erase address register.

**FM.AR=0x4000_010C**

| 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FADDR | | | | | | | | | | | | | | | | |
| 0 | 0x0000 | | | | | | | | | | | | | | | |
| | RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 16<br>0 | FADDR | 96K words address (one word = 4 bytes) |

## FM.DR Flash Memory Data Register

The Internal Flash Memory Data register is shown below.

**FM.DR=0x4000_0110**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FDATA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0x0000_0000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31<br>0 | FDATA | Flash PGM data (32-bit) |

## FM.TMR Flash Memory Timer Register

In the internal Flash Memory Timer value register (16-bit), the Erase/Program timer runs up to {TMR[15:0}.

**FM.TMR=0x4000_0114**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMR | | | | | | | | | | | | | | | |
| 0x09C4 | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15<br><br>0 | TMR | Erase/PGM timer (default, 0x09C4)<br>Timer counts up to TMR[15:0] by int. OSC clock or External OSC clock. It can be selected in TMRCK bit. |

## FM.DRTY Flash Memory Dirty Bit Register

The internal Flash Memory Dirty Bit clear register is shown below.

**FM.DRTY=0x4000_0118**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FDRTY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Write Only | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31 0 | FDRTY | Write any value here, cache line fill flag will be cleared. |

## FM.TICK Flash Memory Tick Timer Register

The Flash Memory Tick Timer register is the internal Flash Memory Burst Mode channel selection register.

**FM.TICK=0x4000_011C**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | FTICK | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x00000 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | RW | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 17 0 | FTICK | TICK goes to 0x3FFFF from written TICK value while TRM runs by PCLK clock |

## FM.CRC Flash Memory CRC Value Register

The Flash Memory CRC Value register shows the CRC value resulting from read accesses on internal Flash memory.

**FM.CRC=0x4000_0120**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | CRC | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0xFFFF | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | RO | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 0 | CRC | CRC16 value |

# FM.CFG Flash Memory Config Value Register

The Flash Memory Config Value register is the Flash Trim value register.

**FM.CFG=0x4000_0130**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WRITE KEY | | | | | | | | | | | | | | | | HRESPD | | | TMRCK | | WAIT | | | CRCINIT | CRCEN | | | TRIM | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | | | 0 | | 1 | 1 | 0 | 0 | | | 0 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | R/W | | | |

| | | | |
|---|---|---|---|
| 31<br>15 | WRITE KEY | | KEY Value : 0x7858 |
| 15 | HRESPD | | Disable HRESP(error response function) of Data or System bus<br>(HRESP is AMBA AHB signal) |
| 12 | TMRCK | 0<br>1 | PGM/ERASE timer source is 20MHz INTOSC<br>PRM/ERASE timer source is External Clock |
| 10<br>8 | WAIT | 000<br>001<br>010<br>011<br>100<br>101 | No wait access for flash memory<br>1-wait inserted for flash access<br>2-wait inserted for flash access<br>3-wait inserted for flash access<br>4-wait inserted for flash access<br>5-wait inserted for flash access |
| 7 | CRCINIT | 0<br>1 | CRC register winll be initialized. It should be reset again before read flash to generate CRC16 calculation<br>(Initial value of FMCRC is 0xFFFF) |
| 6 | CRCEN | 0<br>1 | CRC16 enable<br>CRC value will be calculated at every flash read timing |
| 3<br>0 | TRIM | | FLASH TRIM Value (trim_mode_entry) |

# FM.BOOTCR Boot ROM Remap Clear Register

The Boot ROM Remap Clear register is an 8-bit register.

**FM.BOOTCR=0x4000_0174**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | BOOTROM |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | | | | R |

| | | | |
|---|---|---|---|
| 0 | BOOTROM | | Boot Mode (only can be written in boot loader mode)<br>This bit is used to clear boot loader mode at end of boot code (when BOOTROM low, external BOOT pin signal is masked) |

## FM.JTAGEN JTAG Protection Control Register

The JTAG Protection Control register is the debug access control register.

**FM.JTAGEN=0x4000_017C**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WRITE_KEY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | JTAGEN |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | RW |

| | | | |
|---|---|---|---|
| 0 | JTAGEN | 0 | Debug access port is disabled, write access code is 0xC7 |
| | | 1 | Debug access port is enabled |

## FM.PROT Write Protection Control Register

The Write Protection Control register is the internal Flash memory control register. The PAS selects the area to protect and the WP bits specify the section within the area.

**FM.PROTECT=0x4000_0178**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WRITE_KEY | | | | | | | | APR | | | | | | PAS | | WP15 | WP14 | WP13 | WP12 | WP11 | WP10 | WP9 | WP8 | WP7 | WP6 | WP5 | WP4 | WP3 | WP2 | WP1 | WP0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x0 | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | | | RW | | | | | | RW | | R/w | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

| | | | |
|---|---|---|---|
| 23 | APR | | All protection removed, write_key is 0xA9 |
| | | | 0x0 : protection enabled (default)<br>0x1 : removed All protection , WP15~0 will be set as APR set |
| 18<br>16 | PAS | | Protection Area Selction, write_key is 0x98 |
| | | | 0x0 : protection area < 64KB<br>0x1 : 64KB < protection area < 128KB<br>0x2 : 128KB < protection area < 192KB<br>0x3 : 192KB < protection area < 256KB<br>0x4 : 256KB < protection area < 320KB<br>0x5 : 320KB < protection area < 384KB |
| 15 | WP15 | | 0xF000 ~ 0xFFFF, write_key is 0x87 or 0x98 |
| 14 | WP14 | | 0xE000 ~ 0xEFFF, write_key is 0x87 or 0x98 |
| 13 | WP13 | | 0xD000 ~ 0xDFFF, write_key is 0x87 or 0x98 |
| 12 | WP12 | | 0xC000 ~ 0xCFFF, write_key is 0x87 or 0x98 |
| 11 | WP11 | | 0xB000 ~ 0xBFFF, write_key is 0x87 or 0x98 |
| 10 | WP10 | | 0xA000 ~ 0xAFFF, write_key is 0x87 or 0x98 |
| 9 | WP9 | | 0x9000 ~ 0x9FFF, write_key is 0x87 or 0x98 |
| 8 | WP8 | | 0x8000 ~ 0x8FFF, write_key is 0x87 or 0x98 |
| 7 | WP7 | | 0x7000 ~ 0x7FFF, write_key is 0x87 or 0x98 |
| 6 | WP6 | | 0x6000 ~ 0x6FFF, write_key is 0x87 or 0x98 |
| 5 | WP5 | | 0x5000 ~ 0x5FFF, write_key is 0x87 or 0x98 |
| 4 | WP4 | | 0x4000 ~ 0x4FFF, write_key is 0x87 or 0x98 |

| | | |
|---|---|---|
| 3 | WP3 | 0x3000 ~ 0x3FFF, write_key is 0x87 or 0x98 |
| 2 | WP2 | 0x2000 ~ 0x2FFF, write_key is 0x87 or 0x98 |
| 1 | WP1 | 0x1000 ~ 0x1FFF, write_key is 0x87 or 0x98 |
| 0 | WP0 | 0x0000 ~ 0x0FFF, write_key is 0x97 or 0x98<br>0x0 : Protected (default)<br>0x1 : PGM/ERASE enabled |

# Functional Description

The Flash area can be read from directly via the memory address. Writing of Flash memory can be done through the Boot mode or In-application programming. The execution for the writing of Flash must occur from the RAM area (or from Boot ROM). The Flash controller cannot read Flash memory (including instructions) once the program bit has been set.

**Caution:** If the vector table is not placed in RAM, you MUST disable interrupts to prevent reading the interrupt service routine in Flash memory.

# 8. Internal SRAM

## Overview

The Z32F384 MCU implements zero-wait on the chip's SRAM. The size of the SRAM is 16 KB. The SRAM base address is 0x2000_0000.

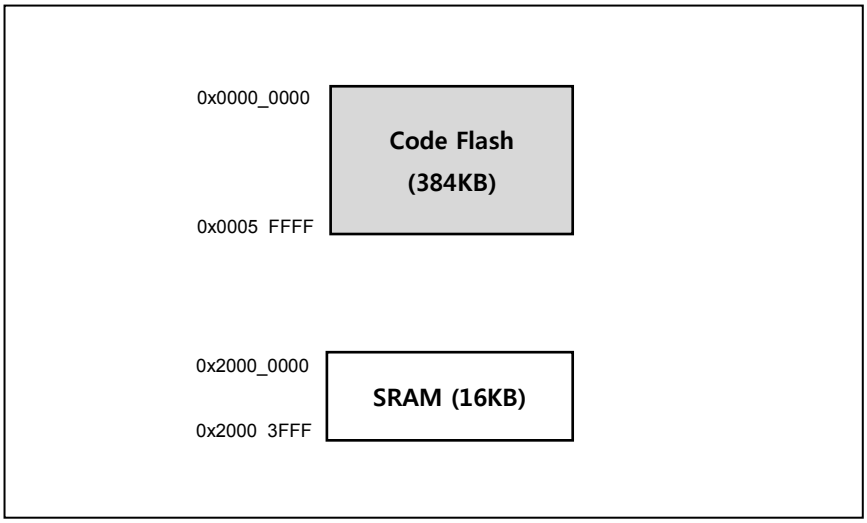



**Figure 8.1. SRAM Block Diagram**

# 9. Direct Memory Access Contoller

## Introduction

The Direct Memory Access (DMA) controller has the following features:

- 8 channels
- Single transfer only
- Supports 8-/16-/32-bit data size
- Supports multiple buffers with same size
- Interrupt condition is transferred through peripheral interrupt

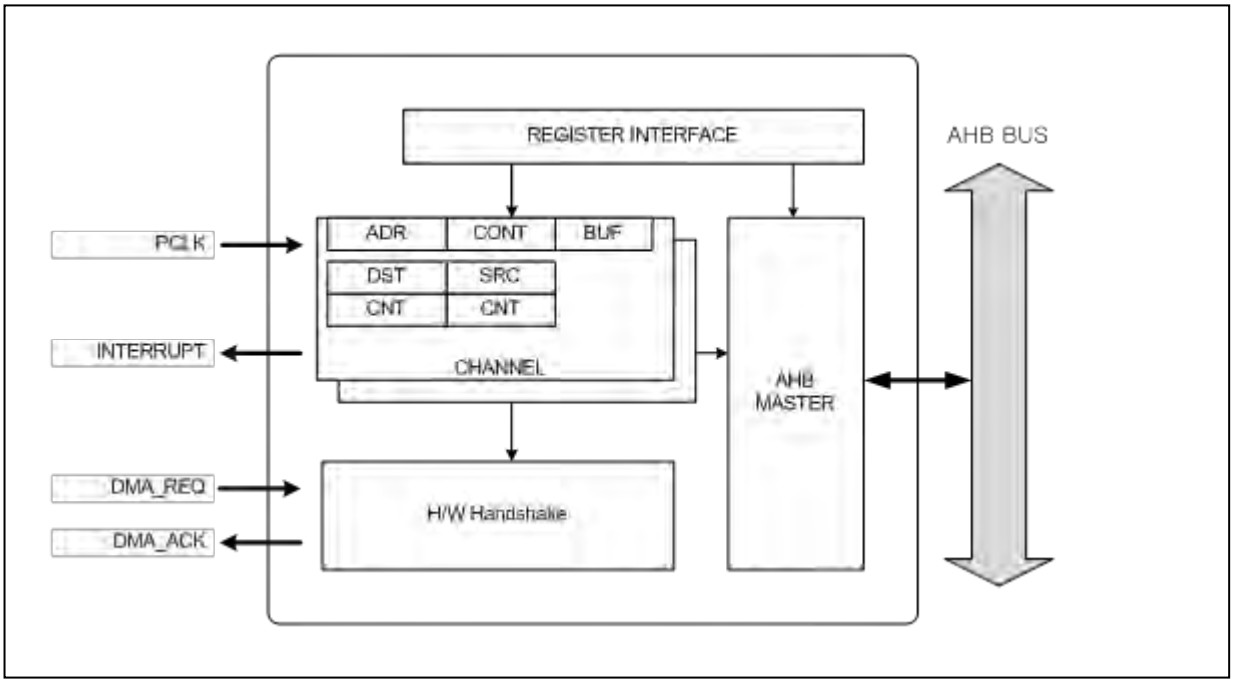



**Figure 9.1. Block Diagram**

# Pin Description

There are no external interface pins.

# Registers

The base addresses of the DMA controller are shown in Table 9.1.

**Table 9.1. DMA Controller Base Addresses**

| Ch. No. | Base Address | Assigned Peripheral |
|---|---|---|
| **DMACH0** | 0x4000_0400 | |
| **DMACH1** | 0x4000_0410 | |
| **DMACH2** | 0x4000_0420 | |
| **DMACH3** | 0x4000_0430 | |
| **DMACH4** | 0x4000_0440 | |
| **DMACH5** | 0x4000_0450 | |
| **DMACH6** | 0x4000_0460 | |
| **DMACH7** | 0x4000_0470 | |

Table 9.2 shows the register map of the DMA controller.

**Table 9.2. DMAC Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **DC*n*.CR** | 0x0000 | R/W | DMA Channel n Control Register | 0x0000_0000 |
| **DC*n*.SR** | 0x0004 | R/W | DMA Channel n Status Register | 0x0000_0000 |
| **DC*n*.PAR** | 0x0008 | R | DMA Channel n Peripheral Address | 0x0000_0000 |
| **DC*n*.MAR** | 0x000C | R/W | DMA Channel n Memory Address | 0x2000_0000 |

# DC*n*.CR DMA Controller Configuration Register

The DMA operation control register is a 32-bit register.

**DC0.CR=0x4000_0400 , DC1.CR=0x4000_0410**
**DC2.CR=0x4000_0420 , DC3.CR=0x4000_0430**
**DC4.CR=0x4000_0440 , DC5.CR=0x4000_0450**
**DC6.CR=0x4000_0460 , DC7.CR=0x4000_0470**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | TRANSCNT | | | | | | | | | | | | | | | | PERISEL | | | | | | | | SIZE | | DIR | |
| 0 | 0 | 0 | 0 | 0x000 | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | | | | 0 | 0 | 0 | 0 | 00 | | 0 | 0 |
| RW | | | | | | | | | | | | | | | | RW | | | | | | | | RW | | | | | | R/W | |

| | | |
|---|---|---|
| 27<br>16 | TRANSCNT | Number of DMA transfer remained<br>Required transfer number should be written before enable DMA transfer. |
| | | 0 DMA transfer is done. |
| | | N N transfers are remained |
| 11<br>8 | PERISEL | Peripheral selction |
| | | N Associated peripheral selection.<br>Refer to DMA Peripheral connection table |
| 3<br>2 | SIZE | Bus transfer size. |
| | | 00 DMA transfer is byte size transfer |
| | | 01 DMA transfer is half word size transfer |
| | | 10 DMA transfer is word size transfer |
| | | 11 Reserved |
| 1 | DIR | Select transfer direction. |
| | | 0 Transfer direction is from memory to peripheral. (TX) |
| | | 1 Transfer direction is from peripheral to memory (RX) |

A DMA channel is connected with the selected peripheral.

Table 9.3 sTable 9.3. DMAC PERISEL Selectionhows peripheral selection numbers. The PERISEL field should be set with the correct number of the peripheral which will be connected with the DMA interface.

**Table 9.3. DMAC PERISEL Selection**

| PERISEL[3:0] | Associated Peripheral |
|---|---|
| 0 | CHANNEL IDLE |
| 1 | UART0 RX |
| 2 | UART0 TX |
| 3 | UART1 RX |
| 4 | UART1 TX |
| 5 | UART2 RX |
| 6 | UART2 TX |
| 7 | UART3 RX |
| 8 | UART3 TX |
| 9 | SPI0 RX |
| 10 | SPI0 TX |
| 11 | SPI1 RX |
| 12 | SPI1 TX |
| 13 | ADC0 RX |
| 14 | ADC1 RX |

PERISEL cannot have the same value in different channels. If the same PERISEL value is wriiten in more than one channel, proper operation is not guaranteed.

Unused channels should contain a CHANNEL IDLE value in the PERISEL bit postions.

## DC*n*.SR DMA Controller Status Register

The DMA Controller Status register is an 8-bit register. This register represents the current status of DMA Controller and enables DMA function.

**DC0.SR=0x4000_0404 , DC1.SR=0x4000_0414**
**DC2.SR=0x4000_0424 , DC3.SR=0x4000_0434**
**DC4.SR=0x4000_0444 , DC5.SR=0x4000_0454**
**DC6.SR=0x4000_0464 , DC7.SR=0x4000_0474**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EOT | | | | | | | DMAEN |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RO | | | | | | | R/W |

| | | | |
|---|---|---|---|
| 7 | EOT | End of transfer. | |
| | | 0 | Data to be transferred is existing.<br>TRANSCNT shows non zero value |
| | | 1 | All data is transferred.<br>TRANSCNT shows now 0 |
| 0 | DMAEN | DMA Enable | |
| | | 0 | DMA is in stop or hold state |
| | | 1 | DMA is running or enabled |

## DC*n*.PAR DMA Controller Peripheral Address Register

The DMA Controller Peripheral Address register represents the peripheral address.

**DC0.PAR=0x4000_0408 , DC1.PAR=0x4000_0418**
**DC2.PAR=0x4000_0428 , DC3.PAR=0x4000_0438**
**DC4.PAR=0x4000_0448 , DC5.PAR=0x4000_0458**
**DC6.PAR=0x4000_0468 , DC7.PAR=0x4000_0478**

| 31 30 29 28 27 26 25 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 |
|---|---|---|---|
| Peripheral BASE OFFSET | | PAR | |
| 0x4000 | | 0x0000 | |
| RO | | RW | |

| | | |
|---|---|---|
| 31<br>0 | PAR | Target Peripheral address of transmit buffer or receive buffer.<br>User must set exact target peripheral buffer address in this field.<br>If DIR is “0” this address is destination address of data transfer.<br>If DIR is “1”, this address is source address of data transfer. |

## DC*n*.MAR DMA Controller Memory Address Register

The DMA Controller Memory Address register represents the memory address.

**DC0.MAR=0x4000_040C , DC1.MAR=0x4000_041C**
**DC2.MAR=0x4000_042C , DC3.MAR=0x4000_043C**
**DC4.MAR=0x4000_044C , DC5.MAR=0x4000_045C**
**DC6.MAR=0x4000_046C , DC7.MAR=0x4000_047C**

| 31 30 29 28 27 26 25 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 |
|---|---|---|---|
| | | MAR | |
| 0x2000 | | 0x0000 | |
| RO | | RW | |

| | | |
|---|---|---|
| 31<br>0 | MAR | Target memory address of data transfer.<br>Address is automatically incremented according to SIZE bits when each transfer is done.<br>If DIR is “0” this address is source address of data transfer.<br>If DIR is “1”, this address is destination address of data transfer. |

# Functional Description

The DMA controller performs direct memory transfer by sharing the system bus with the CPU core. The system bus is shared by two AHB masters following the round-robin priority strategy. Therefore, the DMA controller can share half of the system bandwidth.

The DMA controller can be triggered only by a peripheral request. When a peripheral requests a transfer to the DMA controller, the related channel is activated and accesses the bus to transfer the requested data from memory to the peripheral data buffer or vice versa.

The transfer process involves the following steps:

1. User sets the peripheral and memory addresses.
2. User configures DMA operation mode and transfer count.
3. User enables the DMA channel.
4. Peripheral generates a DMA request.
5. DMA activates the channel that was requested
6. DMA reads data from the source address and saves it to the internal buffer.
7. DMA writes the buffered data to the destination address.
8. Transfer count number is decreased by 1.
9. When the transfer count is 0, the EOT flag is set and notice is sent to the peripheral to issue the interrupt.
10. DMA does not have an interrupt source; the interrupt-related DMA status can be shown from the assigned peripheral interrupt.

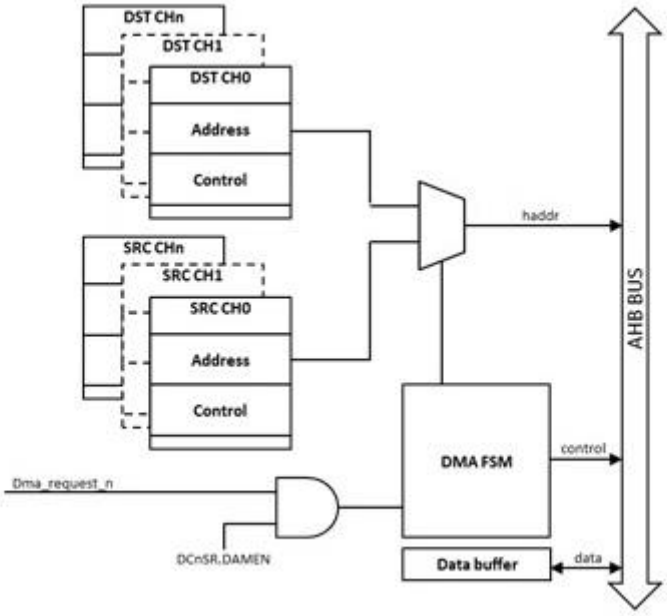



**Figure 9.2. Block Diagram**

Figure 9.3The figure shows the functional timing diagram of the DMA controller. The transfer request from the peripheral is pended internally and it invokes source data read transfer on the AHB bus. The read data from the source address is stored in the internal buffer. This data is transferred to the destination address when the AHB bus is available.

The timing diagram for a DMA transfer from the peripheral to memory is shown in Figure 9.3. A 4-clock cycle latency exists when accessing the peripheral. If the bus is occupied by a different bus master, the number of bus waiting cycles increase until the bus is available.

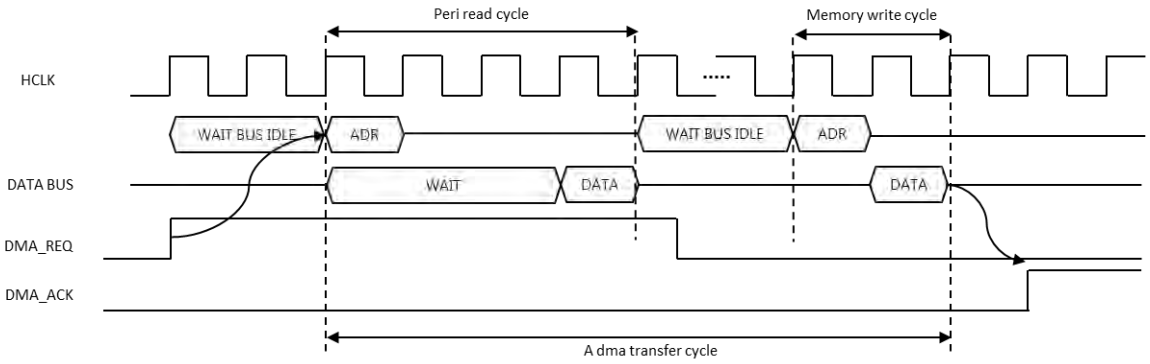



**Figure 9.3. DMA Transfer from Peripheral to Memory**

The timing diagram for a DMA transfer from memory to the peripheral is shown in Figure 9.4. A 4-clock cycle latency exists when accessing the peripheral. If the bus is occupied by a different bus master, the number of bus waiting cycles increase until the bus is available.

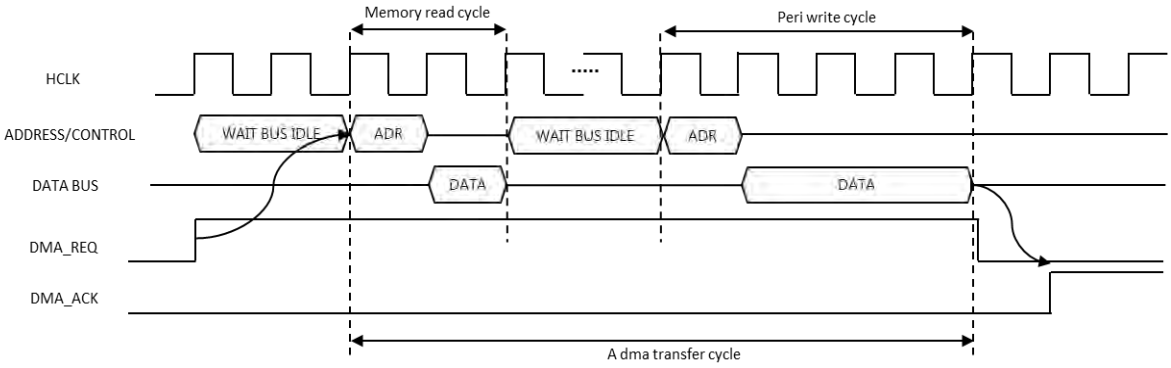



**Figure 9.4. DMA Transfer from Memory to Peripheral**

Figure 9.5 is an example of N data transfers with the DMA. The DMA transfer is started when DCnSR.DMAEN is set and cleared when all the transfers are completed.

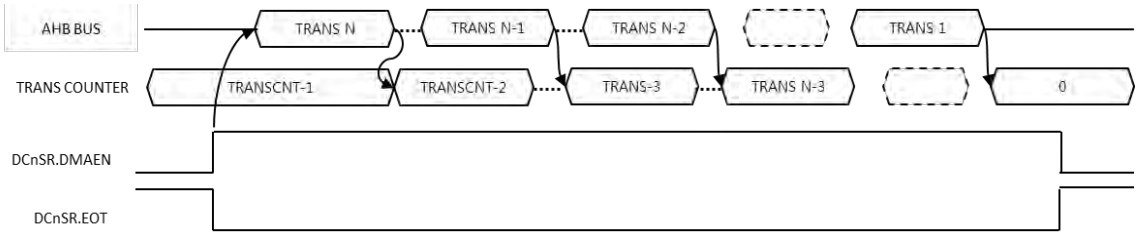



**Figure 9.5. Example of N DMA Transfer**

# 10. Watch-Dog Timer

## Overview

The Watchdog Timer can monitor the system and generate an interrupt or a reset. It has a 32-bit down-counter. Miscellaneous Clock Control Register 3 provides base clock options with clock dividers to drive the WDT clock. This can be selected in the WDTCON register. To prevent the WDT from firing, reload the LR register with the appropriate value before the WDT times out.

- 32-bit down counter (WDTCNT)
- Select reset or periodic interrupt
- Count clock selection
- Dedicated pre-scaler
- Watchdog overflow output signal

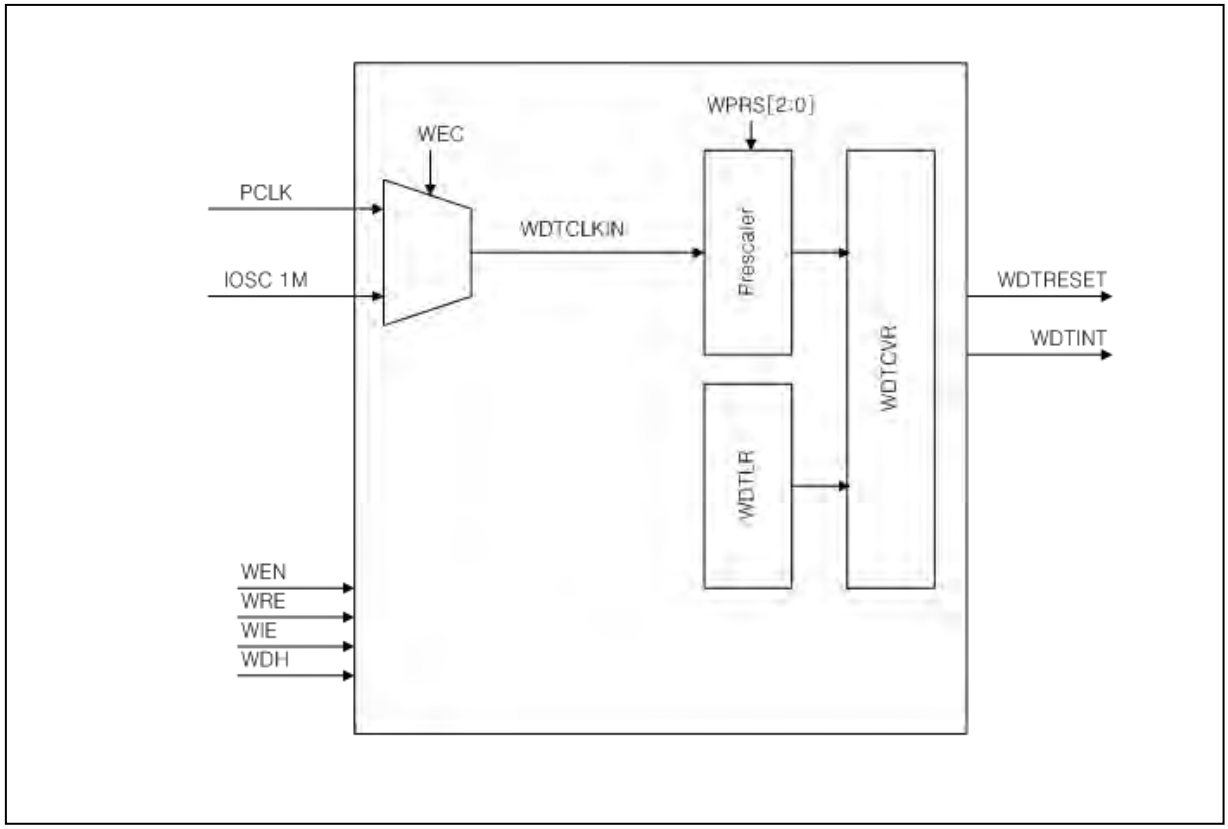



**Figure 10.1. Block Diagram**

# Registers

The base address of the watchdog timer is `0x4000_0200` and the register map is listed in Table 10.1. Initial watchdog time-out period is set to 2000-miliseconds.

**Table 10.1. Watchdog Timer Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **WDT.LR** | 0x0000 | W | WDT Load register | 0x00000000 |
| **WDT.CNT** | 0x0004 | R | WDT Current counter register | 0x0000FFFF |
| **WDT.CON** | 0x0008 | R/W | WDT Control register | 0x0000805C |

## WDT.LR Watchdog Timer Load Register

The Watchdog Timer Load register is used to update the WDTCNT register. To update the WDTCNT register, the WEN bit of WDTCON should be set to 1 and written into the WDTLR register with a target value of WDTCNT.

**WDT.LR=0x4000_0200**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WDTLR** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **0x0000_0000** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **RW** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31<br>0 | WDTLR | Watchdog timer load value register<br>Keeping WEN bit as '1', write WDTLR register will update WDTCNT value with written value |

## WDT.CNT Watchdog Timer Current Counter Register

The Watchdog Timer Current Counter register represents the current count value of the 32-bit down counter .When the counter value reaches 0, the interrupt or reset is awoken.

**WDT.CNT=0x4000_0204**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **WDTCNT** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **0x0000_FFFF** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **RW** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 31<br>0 | WDTCNT | Watchdog timer current counter register<br>32-bit down counter will run from the written value. |

# WDT.CON Watchdog Timer Control Register

The timer module should be correctly configured before running. When the target purpose is defined, the timer can be configured in the TnCON register.

**WDT.CON=0x4000_0208**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WDBG | | | | | | | WUF | WDTIE | WDTRE | | WDTEN | CKSEL | WPRS | | |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 100 | | |
| RW | | | | | | | RW | RW | RW | | RW | RW | RW | | |

| | | | |
|---|---|---|---|
| 15 | WDBG | Watchdog operation control in debug mode | |
| | | 0 | Watchdog counter running when debug mode |
| | | 1 | Watchdog counter stopped when debug mode |
| 8 | WUF | Watchdog timer underflow flag | |
| | | 0 | No underflow |
| | | 1 | Underflow is pending |
| 7 | WDTIE | Watchdog timer counter underflow interrupt enable | |
| | | 0 | Disable interrupt |
| | | 1 | Enable interrupt |
| 6 | WDTRE | Watchdog timer counter underflow interrupt enable | |
| | | 0 | Disable reset |
| | | 1 | Enable reset |
| 4 | WDTEN | Watchdog Counter enable | |
| | | 0 | Watch dog counter disabled |
| | | 1 | Watch dog counter enabled |
| 3 | CKSEL | WDTCLKIN clock source select | |
| | | 0 | PCLK |
| | | 1 | External clock (Configured in MCCR3) |
| 2<br>0 | WPRS[2:0] | Counter clock prescaler<br>WDTCLK = WDTCLKIN/WPRS | |
| | | 000 | WDTCLKIN / 1 |
| | | 001 | WDTCLKIN / 4 |
| | | 010 | WDTCLKIN / 8 |
| | | 011 | WDTCLKIN / 16 |
| | | 100 | WDTCLKIN / 32 |
| | | 101 | WDTCLKIN / 64 |
| | | 110 | WDTCLKIN / 128 |
| | | 111 | WDTCLKIN / 256 |

# Functional Description

The MCCR3 register must be configured to enable the clock source and divider for the Watch Dog Timer (WDT) to run. To prevent the WDT from resetting or interrupting, load a new value into the WDTLR register before the WDTCNT reaches 0.

The watchdog timer count is enabled by setting WDTEN (WDT.CON[4]) to 1. When the watchdog timer is enabled, the down counter starts counting from the load value. If WDTRE (WDT.CON[6]) is set to 1, WDT reset is asserted when the WDT counter value reaches 0 (underflow event) from the WDTLR value. Before the WDT counter goes down to 0, the software can write a certain value to the WDTLR register to reload the WDT counter.

## Timing Diagram

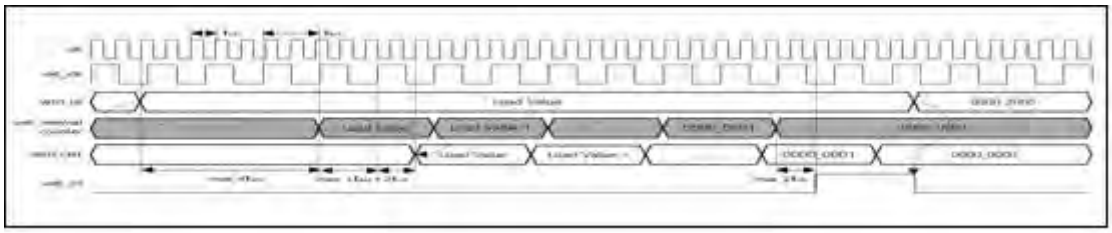


**Figure 10.2. Timing Diagram in Interrupt Mode Operation when WDT Clock is External Clock**

In WDT interrupt mode, after the WDT underflow occurs, a certain count value is reloaded to prevent the next WDT interrupt in a short time period. This reloading action can only be activated when the watchdog timer counter is set to Interrupt mode (set WDTIE of WDT.CON). It takes up to 5 cycles from the load value to the CNT value. The WDT interrupt signal and CNT value data might be delayed by a maximum of 2 system bus clocks in synchronous logic.

## Prescale Table

The WDT includes a 32-bit down counter with programmable prescaler to define different time-out intervals.

The clock sources of the watchdog timer can include the peripheral clock (PCLK) or one of 3 external clock sources. An external clock source can be enabled by CKSEL (WDT.CON[3]) set to ‘1’ and external clock source chosen in MCCR3 register of the System Control Unit block.

To make the WDT counter base clock, users can control the 3-bit prescaler WPRS [2:0] in the WDT.CON register and the maximum prescaled value is “clock source frequency/256”. The prescaled WDT counter clock frequency values are listed in Table 10.2.

**Selectable clock source (40 kHz ~ 16 MHz) and the time out interval when 1 count**

**Time out period = {(Load Value) * (1/pre-scaled WDT counter clock frequency) + max $5T_{ext}$} + max $4T_{clk}$**

*Time out period (time out period from load Value to interrupt set ‘1’)

Table 10.2. Prescaled WDT Counter Clock Frequency

| Clock Source | WDTCLKIN | WDTCLKIN/4 | WDTCLKIN/8 | WDTCLKIN/16 | WDTCLKIN/32 | WDTCLKIN/64 | WDTCLKIN/128 | WDTCLKIN/256 |
|---|---|---|---|---|---|---|---|---|
| **Ring OSC** | 1MHz | 250kHz | 125kHz | 62.5kHz | 31.25kHz | 15.625kHz | 7.8125kHz | 3.90625kHz |
| **MCLK** | MCLK (BUS CLK) | MCLK/4 | MCLK/8 | MCLK/16 | MCLK/32 | MCLK/64 | MCLK/128 | MCLK/256 |
| **IOSC20** | 20MHz | 5MHz | 2.5MHz | 1.25MHz | 625kHz | 312.5kHz | 156.25kHz | 78.125kHz |
| **EOSC** | XTAL | XTAL/4 | XTAL/8 | XTAL/16 | XTAL/32 | XTAL/64 | XTAL/128 | XTAL/256 |
| **SubOSC** | 32.768kHz | 8.192kHz | 4.096kHz | 2.048kHz | 1.024kHz | 512Hz | 256Hz | 128Hz |

# 11. 16-Bit Timer

## Overview

The Timer block consists of 10 channels of 16-bit general-purpose timers. They can support periodic timer, PWM pulse, one-shot timer, and capture mode.

- 16-bit up-counter
- Periodic timer mode
- One-shot timer mode
- PWM pulse mode
- Capture mode
- 10-bit prescaler
- Multi-channel synchronization function

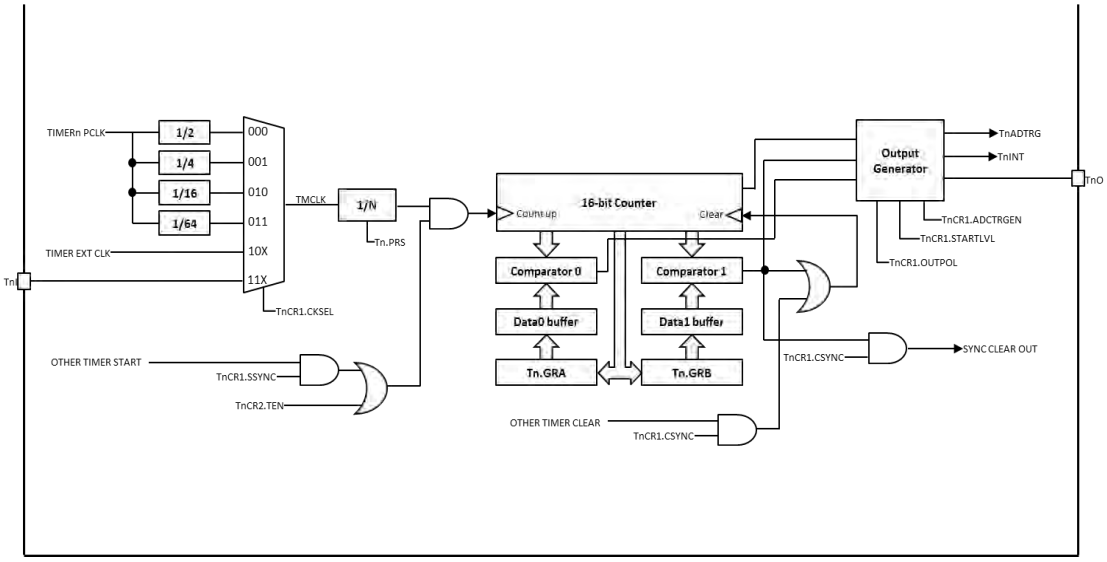



**Figure 11.1. Block Diagram**

# Pin Description

**Table 11.1. External Pin**

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **TnC** | I | External clock / capture input |
| **TnO** | O | Timer output |

# Registers

The base address of the TIMER is `0x4000_3000` and the register map is described in Table.11.2 and 11.3.

**Table 11.2. Base Address of Each Channel**

| CHANNEL | Address |
|---|---|
| **T0** | 0x4000_3000 |
| **T1** | 0x4000_3020 |
| **T2** | 0x4000_3040 |
| **T3** | 0x4000_3060 |
| **T4** | 0x4000_3080 |
| **T5** | 0x4000_30A0 |
| **T6** | 0x4000_30C0 |
| **T7** | 0x4000_30E0 |
| **T8** | 0x4000_3100 |
| **T9** | 0x4000_3120 |

**Table 11.3. Timer Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **T*n*.CR1** | 0x-000 | R/W | Timer control register 1 | 0x00000000 |
| **T*n*.CR2** | 0x-004 | R/W | Timer control register 2 | 0x00000000 |
| **T*n*.PRS** | 0x-008 | R/W | Timer prescaler register | 0x00000000 |
| **T*n*.GRA** | 0x-00C | R/W | Timer general data register A | 0x00000000 |
| **T*n*.GRB** | 0x-010 | R/W | Timer general data register B | 0x00000000 |
| **T*n*.CNT** | 0x-014 | R/W | Timer counter register | 0x00000000 |
| **T*n*.SR** | 0x-018 | R/W | Timer status register | 0x00000000 |
| **T*n*I.ER** | 0x-01C | R/W | Timer interrupt enable register | 0x00000000 |

# Tn.CR1 Timer n Control Register 1

The Timer Control Register 1 is a 16-bit register.

The timer module should be correctly configured before running. When the target purpose is defined, the timer can be configured in the TnCR1 register.

**T0.CR1=0x4000_3000, T1.CR1=0x4000_3020**
**T2.CR1=0x4000_3040, T3.CR1=0x4000_3060**
**T4.CR1=0x4000_3080, T5.CR1=0x4000_30A0**
**T6.CR1=0x4000_30C0, T7.CR1=0x4000_30E0**
**T8.CR1=0x4000_3100, T9.CR1=0x4000_3120**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSYNC | CSYNC | UAO | OUTPOL | | | | ADCTRGEN | STARTLVL | CKSEL | | | CLRMOD | | MODE | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 000 | | | 00 | | 00 | |
| R/W | R/W | R/W | R/W | | | | R/W | | R/W | | | R/W | | R/W | |

| | | | |
|---|---|---|---|
| 15 | SSYNC | Synchronize start counter with other synchronized timers | |
| | | 0 | Single counter mode |
| | | 1 | Synchronized counter start mode |
| 14 | CSYNC | Synchronize clear counter with other synchronized timers | |
| | | 0 | Single counter mode |
| | | 1 | Synchronized counter clear mode |
| 13 | UAO | Select GRA, GRB update mode | |
| | | 0 | Writing GRA or GRB takes effect after current period |
| | | 1 | Writing GRA or GRB takes effect in current period |
| 12 | OUTPOL | Timer output polarity | |
| | | 0 | Normal output |
| | | 1 | Negated output |
| 8 | ADCTRGEN | ADC Trigger enable control | |
| | | 0 | Disable adc trigger |
| | | 1 | Enable adc trigger |
| 7 | STARTLVL | Timer output polarity control | |
| | | 0 | Default output level is HIGH |
| | | 1 | Defulat output level is LOW |
| 6<br>4 | CKSEL[2:0] | Counter clock source select | |
| | | 000 | PCLK/2 |
| | | 001 | PCLK/4 |
| | | 010 | PCLK/16 |
| | | 011 | PCLK/64 |
| | | 10X | EXT0 (MCCR3) |
| | | 11X | TnC pin input |
| 3<br>2 | CLRMD | Clear select when capture mode | |
| | | 00 | Rising edge clear mode |
| | | 01 | Falling edge clear mode |
| | | 10 | Both edge clear mode |
| | | 11 | None clear mode |
| 1<br>0 | MODE[1:0] | Timer operation mode control | |
| | | 00 | Normal periodic operation mode |
| | | 01 | PWM mode |
| | | 10 | One shot mode |
| | | 11 | Capture mode |

## Tn.CR2Timer n Control Register 2

The Timer Control Register 2 is an 8-bit register.

**T0.CR2=0x4000_3004, T1.CR2=0x4000_3024**
**T2.CR2=0x4000_3044, T3.CR2=0x4000_3064**
**T4.CR2=0x4000_3084, T5.CR2=0x4000_30A4**
**T6.CR2=0x4000_30C4, T7.CR2=0x4000_30E4**
**T8.CR2=0x4000_3104, T9.CR2=0x4000_3124**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | TCLR | TEN |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R | R | R | R | R | R | R/W | R/W |

| | | |
|---|---|---|
| 1 | TCLR | Timer register clear |
| | 0 | No |
| | 1 | Clear count register (This bit will be cleared after next timer clock) |
| 0 | TEN | Timer enable bit |
| | 0 | Stop Timer Counting |
| | 1 | Start Timer Counting |

It is recommended to start the timer with the TCLR bit set to '1'.

## Tn.PRS Timer n Prescaler Register

The Timer Prescaler Register is a 16-bit register to prescale the counter input clock.

**T0.PRS=0x4000_3008, T1.PRS=0x4000_3028**
**T2.PRS =0x4000_3048, T3.PRS=0x4000_3068**
**T4.PRS=0x4000_3088, T5.PRS=0x4000_30A8**
**T6.PRS =0x4000_30C8, T7.PRS=0x4000_30E8**
**T8.PRS=0x4000_3108, T9.PRS=0x4000_3128**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | PRS | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 000 | | | | | | | | | |
| | | | | | | R/W | | | | | | | | | |

| | | |
|---|---|---|
| 9 | PRS | Pre-scale value of count clock |
| 0 | | TCLK = PCLK/(PRS+1) |

## Tn.GRA Timer n General Register A

The Timer General Register A is a 16-bit register. The GRA register is the duty register. This register controls the TxIO pin.

**T0.GRA=0x4000_300C, T1.GRA=0x4000_302C**
**T2.GRA =0x4000_304C, T3.GRA=0x4000_306C**
**T4.GRA=0x4000_308C, T5.GRA=0x4000_30AC**
**T6.GRA =0x4000_30CC, T7.GRA=0x4000_30EC**
**T8.GRA=0x4000_310C, T9.GRA=0x4000_312C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GRA | | | | | | | | | | | | | | | |
| 0x0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15<br>0 | GRA | Timer n General Register A (Duty/Interrupt Register)<br>Periodic mode / PWM / One-shot mode<br>- Duty count value<br>- When the counter value is matched with this value, GRA Match interrupt is requested<br>Capture mode<br>- Falling edge of TnC port will capture the count value when rising edge clear mode<br>- Rising edge of TnC port will capture the count value when falling edge clear mode |

## Tn.GRB Timer n General Register B

Timer General Register B is a 16-bit register. The GRB register is the period match counter. It does not toggle the TxIO pins and is required in most modes as the period.

**T0.GRB=0x4000_3010, T1.GRB=0x4000_3030**
**T2.GRB=0x4000_3050, T3.GRB=0x4000_3070**
**T4.GRB=0x4000_3090, T5.GRB=0x4000_30B0**
**T6.GRB=0x4000_30D0, T7.GRB=0x4000_30F0**
**T8.GRB=0x4000_3110, T9.GRB=0x4000_3130**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GRB | | | | | | | | | | | | | | | |
| 0x0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15<br>0 | GRB | Timer n General Register A (Period/Interrupt register)<br>Periodic mode / PWM / One-shot mode<br>- In periodic mode or PWM mode, this register is used as Period value. The counter will count up to (GRB-1) value.<br>- When the counter value is matched with this value, GRB Match interrupt is requested only in PWM and one-shot modes.<br>Capture mode<br>- Rising edge of TnC port will capture the count value when rising edge clear mode<br>- Falling edge of TnC port will capture the count value when falling edge clear mode |

## Tn.CNT Timer n Counter Register

The Timer Counter Register is a 16-bit register.

**T0.CNT=0x4000_3014, T1.CNT=0x4000_3034**
**T2.CNT=0x4000_3054, T3.CNT=0x4000_3074**
**T4.CNT=0x4000_3094, T5.CNT=0x4000_30B4**
**T6.CNT=0x4000_30D4, T7.CNT=0x4000_30F4**
**T8.CNT=0x4000_3114, T9.CNT=0x4000_3134**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT | | | | | | | | | | | | | | | |
| 0x0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 0 | CNT | Timer count value |

## Tn.SR Timer n Status Register

The Timer Status Register is an 8-bit register. This register indicates the current status of the timer module.

**T0.SR=0x4000_3018, T1.SR=0x4000_3038**
**T2.SR=0x4000_3058, T3.SR=0x4000_3078**
**T4.SR=0x4000_3098, T5.SR=0x4000_30D8**
**T6.SR=0x4000_30F8, T7.SR=0x4000_30F8**
**T8.SR=0x4000_3118, T9.SR=0x4000_3138**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | MFA | MFB | OVF |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | R/W | R/W | | R/W | R/W | R/W |

| | | |
|---|---|---|
| 2 | MFA | GRA Match flag (Duty Match) |
| | | 0 No direction change |
| | | 1 Match flag with GRA |
| 1 | MFB | GRB Match flag (Period Match) |
| | | 0 No direction change |
| | | 1 Match flag with GRB |
| 0 | OVF | Counter overflow flag (Only occurs when counter rolls over at 0xFFFF) |
| | | 0 No direction change |
| | | 1 Counter overflow flag |

# Tn.IER Timer n Interrupt Enable Register

The Timer Interrupt Enable Register is an 8-bit register. Each status flag of the timer block can issue the interrupt. To enable the interrupt, write “1” in the corresponding bit in the TnIER register.

**T0.IER=0x4000_301C, T1.IER=0x4000_303C**
**T2 .IER=0x4000_305C, T3.IER=0x4000_307C**
**T4.IER=0x4000_309C, T5.IER=0x4000_30BC**
**T6.IER=0x4000_30DC, T7.IER=0x4000_30FC**
**T8.IER=0x4000_311C, T9.IER=0x4000_313C**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | MAIE | MBIE | OVIE |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | R/W | R/W | R/W | | W | R/W | W |

| | | |
|---|---|---|
| 2 | MAIE | GRA Match interrupt enable |
| | | 0 Not effect |
| | | 1 Enable match register A interrupt |
| 1 | MBIE | GRB Match interrupt enable |
| | | 0 Not effect |
| | | 1 Enable match register B interrupt |
| 0 | OVIE | Counter overflow interrupt enable |
| | | 0 Not effect |
| | | 1 Enable counter overflow interrupt |

# Functional Description

## Basic Operation of Timer

In Figure 11.2, TMCLK is a reference clock for operation of the timer. This clock is divided by the prescaler setting and the counting clock will work. The following images show the starting point of the counter and the ending of the period point of the counter in normal periodic mode.

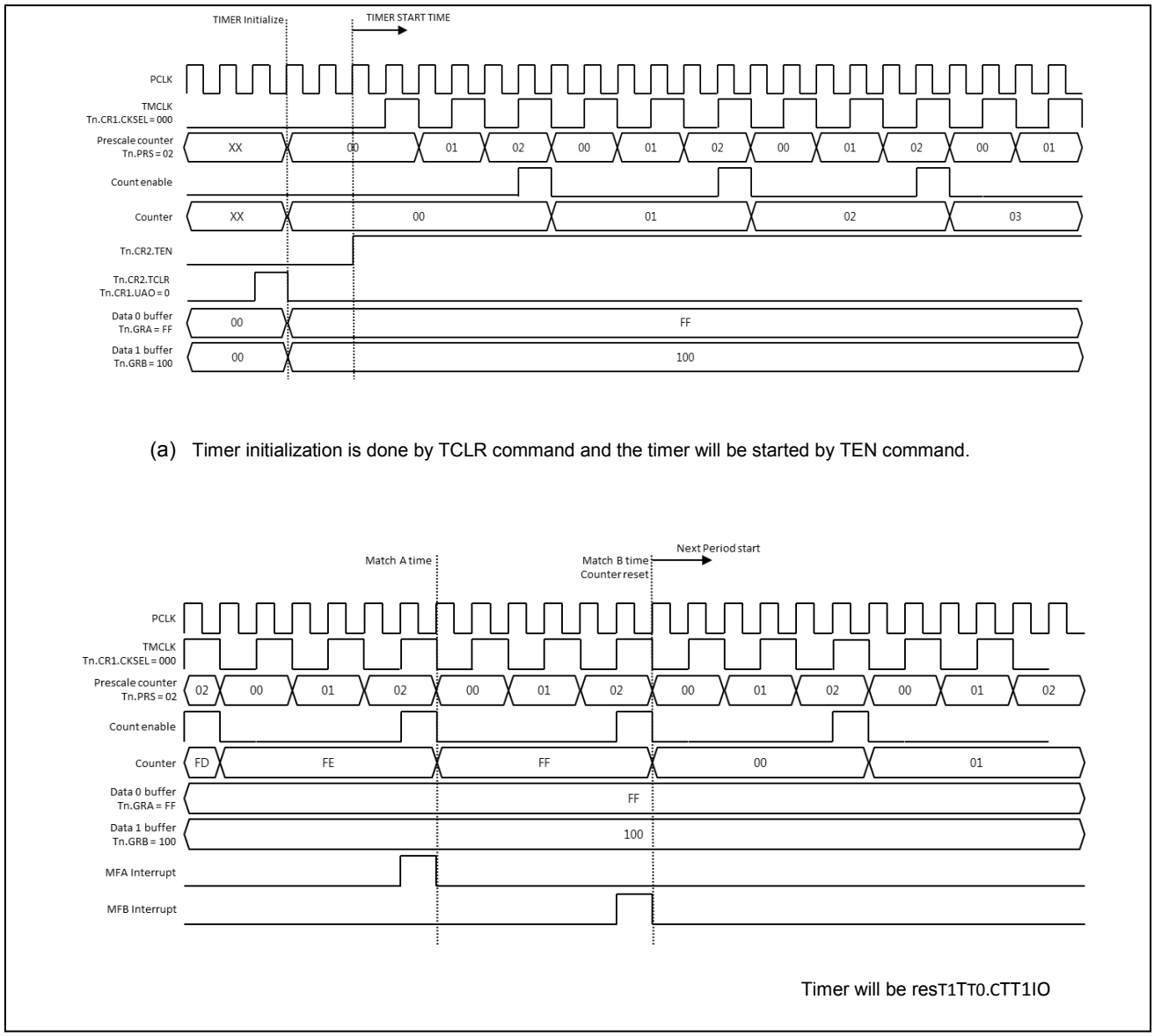



**Figure 11.2. Basic Start and Match Operation**

The period of timer count can be calculated as shown in the following equation:

**The period = TMCLK Period * Tn.GRB value**
**Match A interrupt time = TMCLK Period * Tn.GRA value**

If the Tn.CR1.UAO bit is “0”, the Tn.CR2.TCLR command will initialize all the registers in the timer block and load the GRA and GRB value into the Data0 and Data1 buffers. When you change the timer setting and restart the timer with the new setting, write the CR2.TCLR command before the CR2.TEN command.

The update timing of the Data0 and Data1 buffers in dynamic operation is different in each operating mode and depends on the Tn.CR1.UAO bit.

## Normal Periodic Mode

Figure 11.3 shows the timing diagram in normal periodic mode. The Tn.GRB value decides the timer period. An additional comparison point is provided with the Tn.GRA register value.

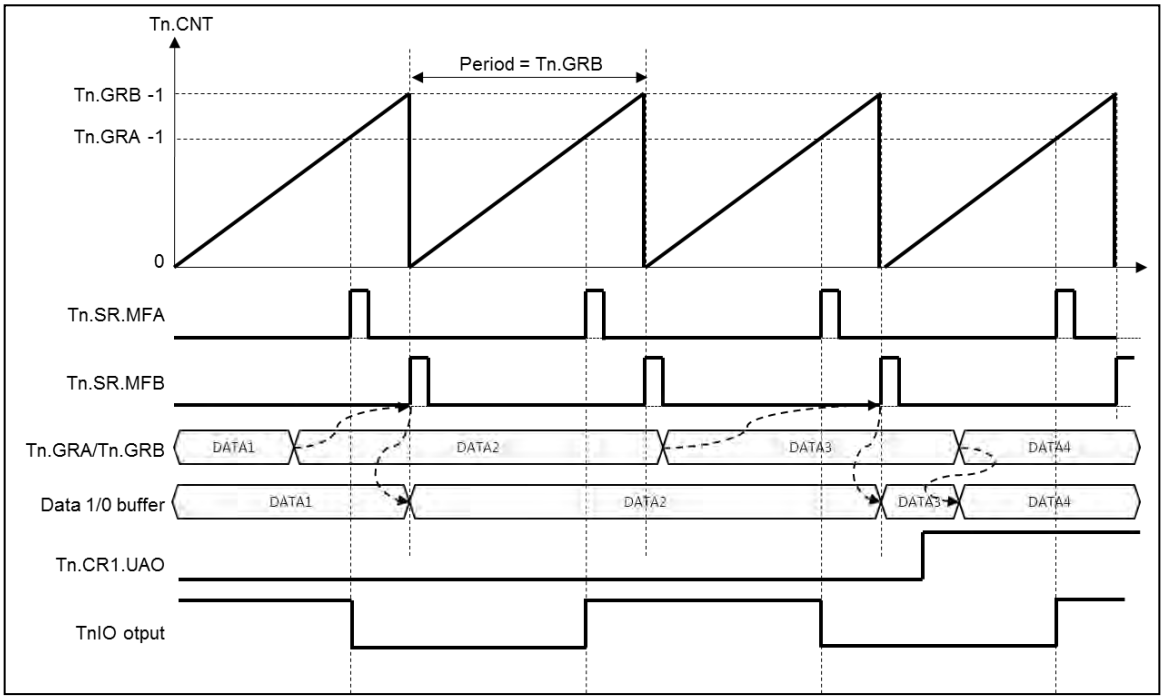



**Figure 11.3. Normal Periodic Mode Operation**

The period of the timer count can be calculated as shown in the following equation:

**The period = TMCLK Period * Tn.GRB value**
**Match A interrupt time = TMCLK Period * Tn.GRA value**

If Tn.GRB = 0, the timer cannot be started even if TnCR2.TEN is “1”, because the period is “0”.

The value in Tn.GRA and Tn.GRB is loaded into the internal compare data buffers 0 and 1 when the loading condition occurs. In this periodic mode with TnCR1.UAO =0, the Tn.CR2.TCLR write operation and the GRB match event will load the compare data buffers.

When TnCR1.UAO is 1, the internal compare data buffer is updated when the Tn.GRA or Tn.GRB data is updated.

The TnIO output signal is toggled at the time of every Match A condition. If the value of TnGRA is 0, the TnIO output does not change its previous level. If TnGRA is the same as TnGRB, the TnIO ouput toggles at the same time as the counter start time. The initial level of the TnIO signal is decided by the TnCR1.STARTLVL value.

## One Shot Mode

**Figure 11.4** shows the timing diagram in One Shot mode. The Tn.GRB value decides the one shot period. An additional comparison point is provided with the Tn.GRA register value.

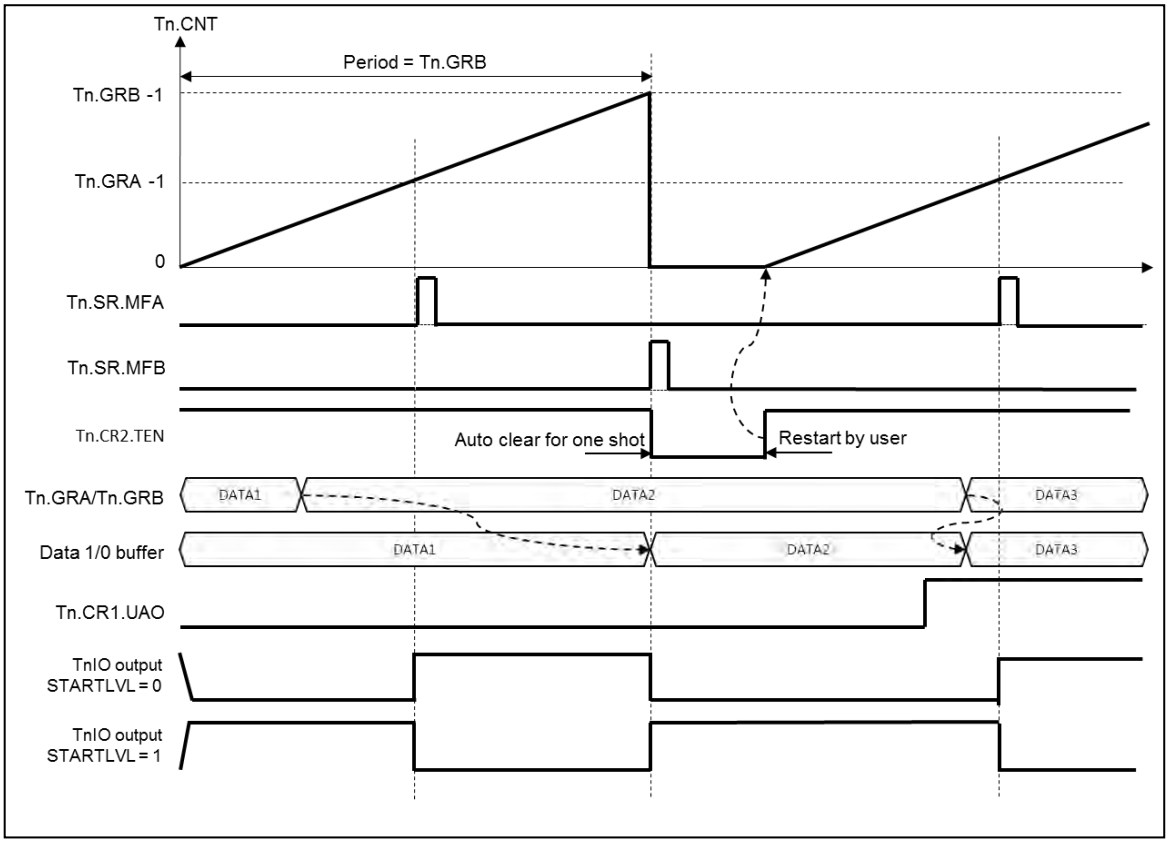



**Figure 11.4. One Shot Mode Operation**

The period of one shot count can be calculated as shown in the following equation:

**The period = TMCLK Period * Tn.GRB value**
**Match A interrupt time = TMCLK Period * Tn.GRA value**

If Tn.GRB = 0, the timer cannot be started even if TnCR2.TEN is “1” because the period is “0”.

The value in Tn.GRA and Tn.GRB is loaded into the internal compare data buffers 0 and 1 when the loading condition occurs. In this periodic mode with TnCR1.UAO =0, the Tn.CR2.TCLR write operation and the GRB match event will load the compare data buffers.

When TnCR1.UAO is 1, the internal compare data buffer is updated when the Tn.GRA or Tn.GRB data is updated.

The TnIO output signal format is the same as in PWM mode. The Tn.GRB value defines the output pulse period and the Tn.GRA value defines the pulse width of one shot pulse.

# PWM Timer Output

Figure 11.5 shows the timing diagram in PWM output mode. The Tn.GRB value decides the PWM pulse period. An additional comparison point is provided with the Tn.GRA register value which defines the pulse width of PWM output.

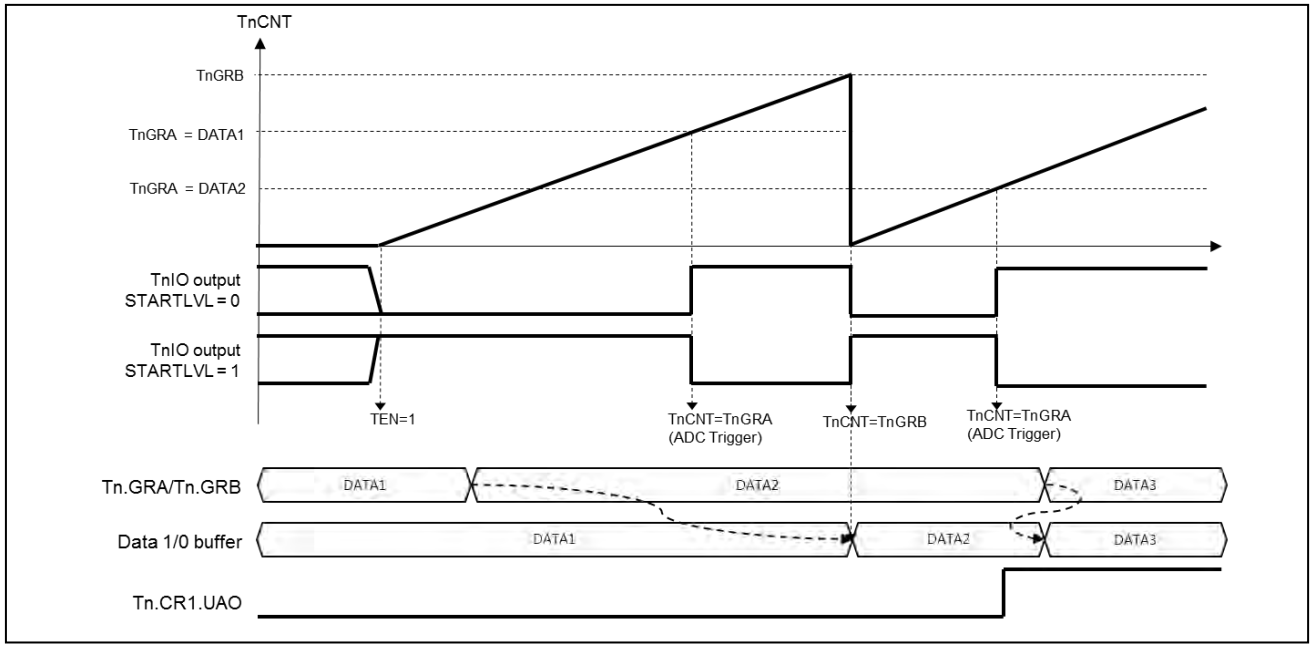



**Figure 11.5. PWM Output Operation**

The period of PWM pulse can be calculated as shown in the following equation:

**The period = TMCLK Period * Tn.GRB value**
**Match A interrupt time = TMCLK Period * Tn.GRA value**

If Tn.GRB = 0, the timer cannot be started even if TnCR2.TEN is “1” because the period is “0”.

The value in Tn.GRA and Tn.GRB is loaded into the internal compare data buffer 0 and 1 when the loading condition occurs. In this periodic mode with TnCR1.UAO =0, the Tn.CR2.TCLR write operation and the GRB match event will load the compare data buffers.

When TnCR1.UAO is 1, the internal compare data buffer is updated when the Tn.GRA or Tn.GRB data is updated.

The TnIO output signal generates a PWM pulse. The Tn.GRB value defines the output pulse period and the Tn.GRA value defines the pulse width of one shot pulse.The active level of the PWM pulse can be controlled by the Tn.CR1.STARTLVL bit value.

ADC Trigger generation is available at Match A interrupt time.

# PWM Synchronization Function

**2 PWM outputs are usually used as synchronous PWM signal control. This function is provided with a synchronous start function.**

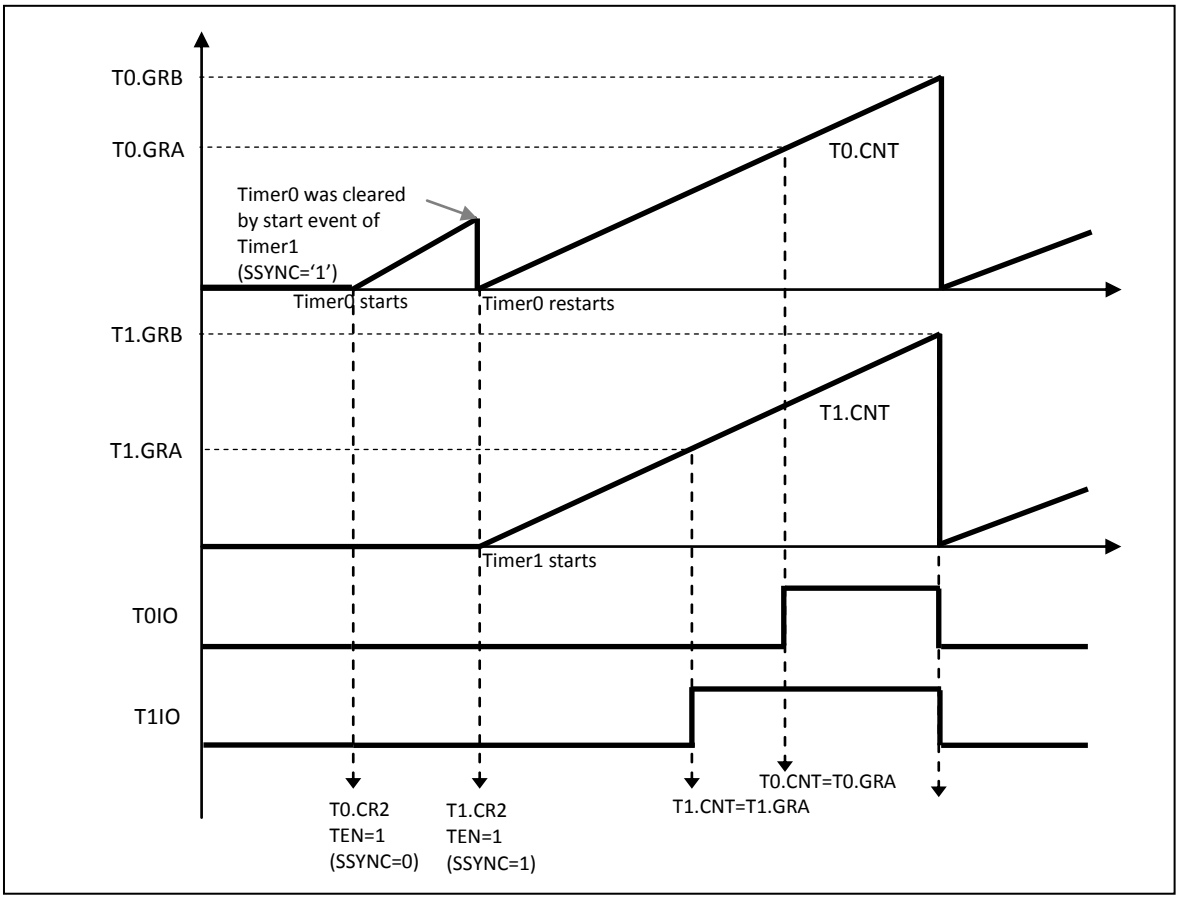



Figure 11.6 shows the synchronous PWM generation function.

**Figure 11.6. An Example of the Timer Synchronization Function (SSYNC='0')**

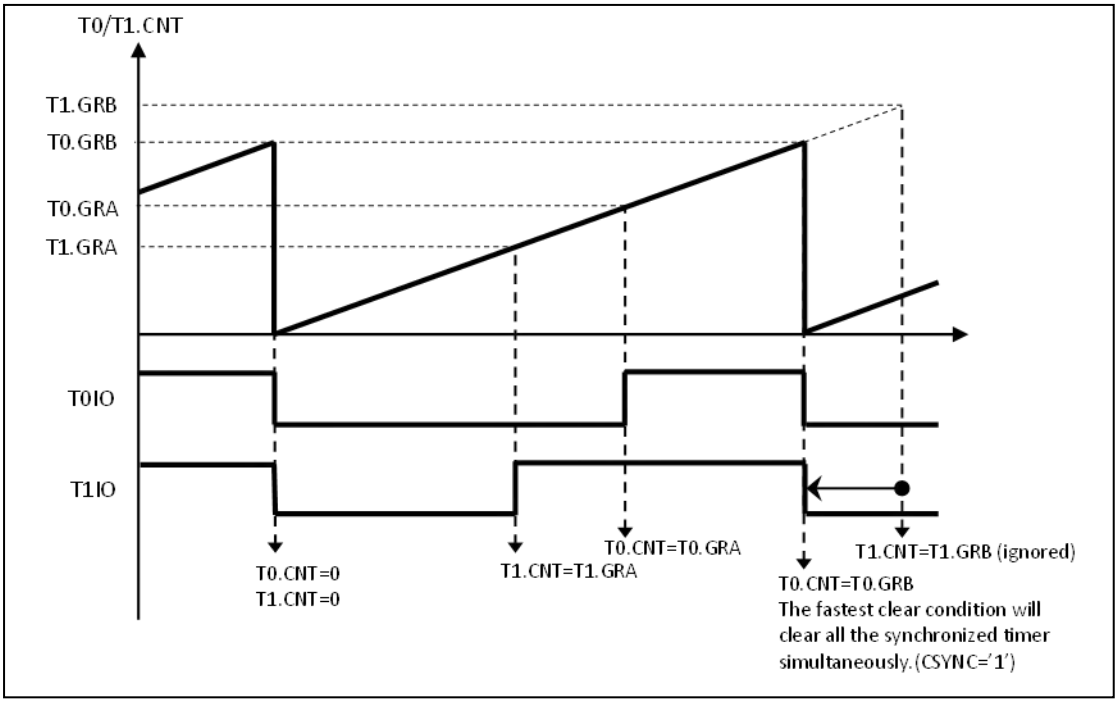



**Figure 11.7. An Example of the Timer Synchronization Function (CSYNC='1')**

The TnCR1.SSYNC bit controls start synchronization with other timer blocks. The TnCR1.CSYCN bit controls clear sync with other timer blocks. The SSYNC and CSYNC bits are only effective when used with tow or more timers.

For example, timer0 and timer1 set the SSYNC and CCSYNC bits in each CR1 register; both timers are started when one of them is enabled. Both timers are cleared with a short period match value. However, others are not affected by these 2 timers, and they can be operated independently because their SYNC control bit is 0.

## Capture Mode

Figure 11.8 shows the timing diagram in Capture mode operation. The TnIO input signal is used for the capture pulse. Both rising and falling edges can capture the counter value in each capture condition.

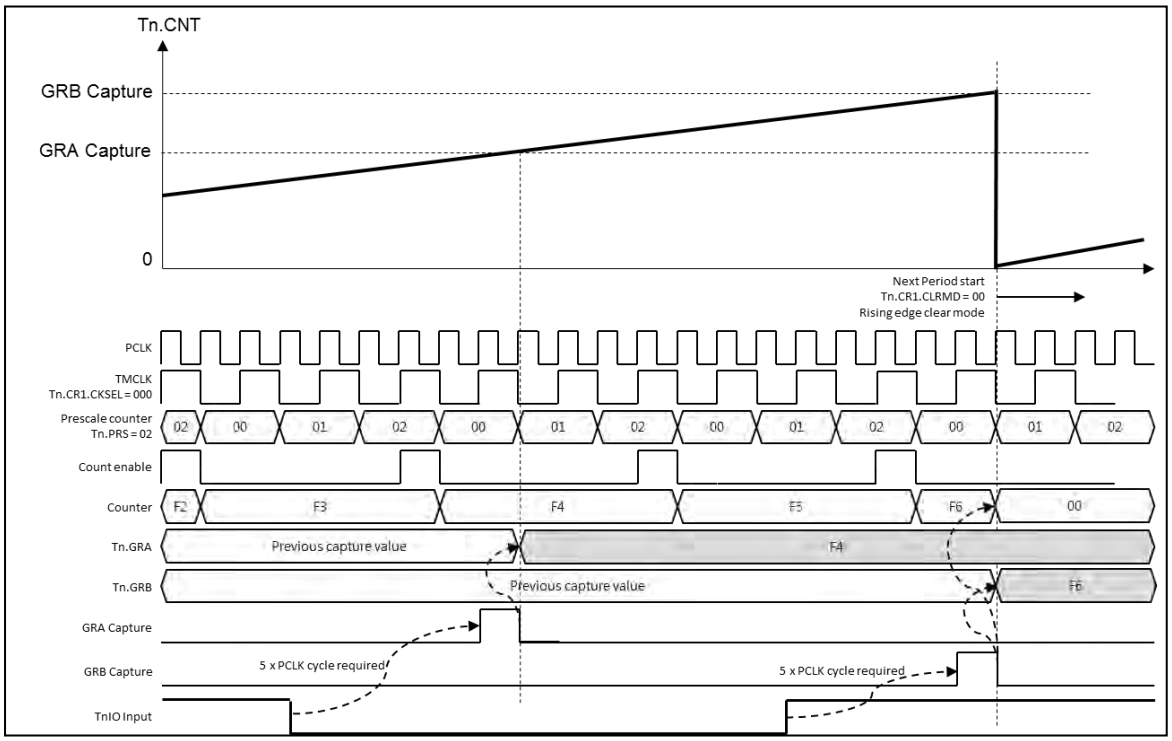



**Figure 11.8. Capture Mode Operation**

A 5 PCLK clock cycle is required internally. Therefore, the actual capture point is after 5 PCLK clock cycles from the rising or falling edge of the TnIO input signal.

**The internal counter can be cleared in various modes. The TnCR1.CLRMD field controls the counter clear mode. Rising Edge clear mode, Falling Edge clear mode, Both Edges clear mode, and None clear mode are supported.**

Figure 11.8 displays the rising edge clear mode.

## ADC Trigger Function

The timer module can generate ADC start trigger signals. One timer can be one trigger source of the ADC block. Trigger source control is performed by the ADC control register.

Figure 11.9 shows the ADC trigger function.

The conversion rate must be shorter than the timer period. If this is not true, an overrun situation can occur. ADC acknowledge is not required because the trigger signal will be cleared automatically after 3 PCLK clock pulses.

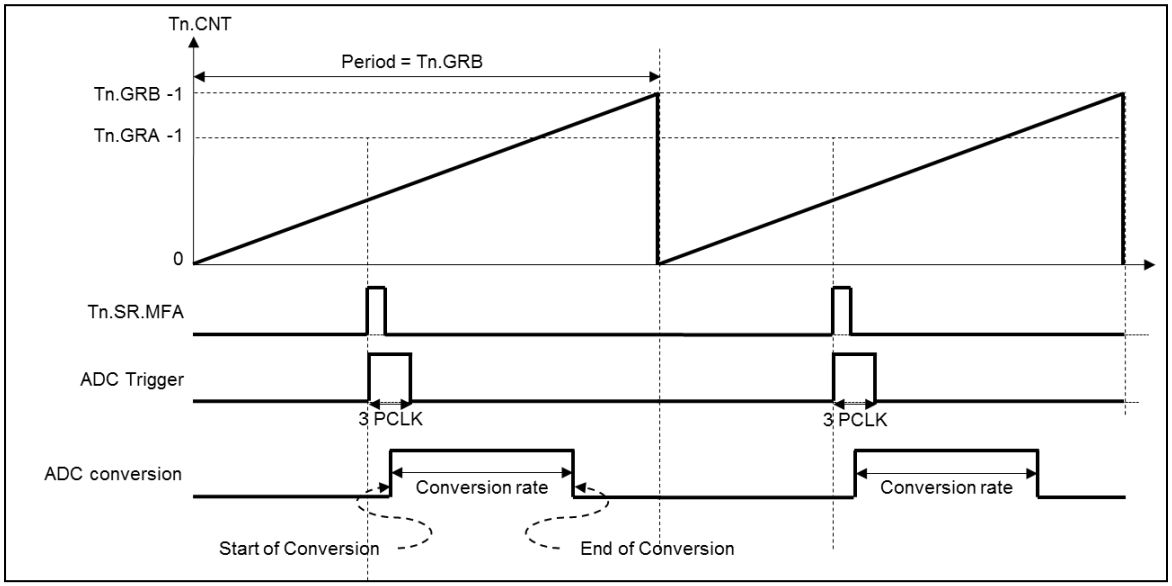



**Figure 11.9. ADC Trigger Function Timing Diagram**

# 12. 32-Bit Free Run Timer

## Overview

The FRT block is a 32-bit Free Run Timer. It can be used in power down mode.

- 32-bit up-counter with SUB OSC
- Matched interrupt

## Registers

The base address of FRT is 0x4000_30E0 and the register map is described in Tables 12.1 and 12.2.

**Table 12.1. Base Address of Each Channel**

| Channel | Address |
|---|---|
| **FRT** | 0x4000_0600 |

**Table 12.2. Timer Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **FRT.MR** | 0x0000 | R/W | FRT mode register | 0x00000000 |
| **FRT.CR** | 0x0004 | R/W | FRT control register | 0x00000000 |
| **FRT.PER** | 0x0008 | R/W | FRT period register | 0x00000000 |
| **FRT.CNT** | 0x000C | RO | FRT counter register | 0x00000000 |
| **FRT.SR** | 0x0010 | R/W | FRT status register | 0x00000000 |

## FRT.MR FRT Mode Register

FRT is a 32-bit up counter. It can be used in power down mode. The SUB OSC clock is directly connected to FRT. The clock is uncontrollable in the SCU block. The FRT Mode Register is an 8-bit register.

**FRTMR=0x4000_0600**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | CLKSEL | | MCD | OVIE | MIE |
| 0 | 0 | 0 | 00 | | 0 | 0 | 0 |
| | | | RW | | RW | RW | RW |

| | | |
|---|---|---|
| 3 | CLKSEL | FRT counter clock source control |
| 2 | | 0 Internal Oscillator clock divided by 16 |
| | | 1 External Oscillator clock divided by 16 |
| | | 2 Sub Oscillator clock |
| | | 3 Reserved |
| 1 | MCD | Counter Match Clear Disable bit |
| | | 0 Counter Match Clear function is enabled. Whenever the counter matches FRT.PER, the counter will be set zero and waiting for MF to be cleared. |
| | | 1 Counter Match Clear function is disabled. The counter will keep countering without set zero |
| 1 | OVIE | Overflow Interrupt Enable bit |
| | | 0 Not effect |
| | | 1 Interrupt enabled |
| 0 | MIE | Match Interrupt Enable bit |
| | | 0 Not effect |
| | | 1 Interrupt enabled |

## FRT.CR FRT Control Register

The FRT Control register is an 8-bit register.

**FRTCR=0x4000_3E04**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | CNTREQ | FCLR | FHOLD | FEN |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | RW | WO | R/W | R/W |

| | | |
|---|---|---|
| 3 | CNTREQ | FRT Counter read request bit |
| | | 0 No |
| | | 1 Request to read FCNT (cleared when CNTACK(FSR[1]) is high) |
| 2 | FCLR | FRT Counter register clear bit |
| | | 0 No |
| | | 1 Clear the counter |
| 1 | FHOLD | FRT Counter register hold bit |
| | | 0 No |
| | | 1 Hold the counter |
| 0 | FEN | FRT enable bit |
| | | 0 FRT Disabled |
| | | 1 FRT Enabled |

## FRT.PER FRT Period Match Register

The FRT Period Match register is a 32-bit register.

FRTPER=0x4000_3E08

| 31 30 29 28 27 26 25 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 |
|---|---|---|---|
| DATA | | | |
| 0x0000_0000 | | | |
| R/W | | | |

| | | |
|---|---|---|
| 32<br>0 | DATA | FRT match data |

## FRT.CNT FRT Counter Register

The FRT Counter Register is a 32-bit register.

FRTCNT=0x4000_3E0C

| 31 30 29 28 27 26 25 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 |
|---|---|---|---|
| CNT | | | |
| 0x0000_0000 | | | |
| RO | | | |

| | | |
|---|---|---|
| 32<br>0 | CNT | FRT Counter |

## FRT.SRFRT Status Register

FRT Status Register is an 8-bit register.FRTSR=0x4000_0610

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | RACK | OVIF | MIF |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | WC1 | WC1 | WC1 |

| | | |
|---|---|---|
| 2 | RACK | Read Counter Acknowledge bit |
| | 0 | Not ready to read CNT value |
| | 1 | Ready to read CNT value |
| 1 | OVIF | Overflow interrupt flag bit |
| | 0 | Overflow interrupt did not occur |
| | 1 | Overflow interrupt occurred |
| 0 | MIF | Match interrupt flag bit |
| | 0 | Match interrupt did not occur. |
| | 1 | Match Interrupt occurred<br>In Counter Match Clear mode, this bit should be cleared for restarting the counter. |

# Functional Description

FRT has two types of interrupt:

- Match interrupt
- Overflow interrupt

## Match Interrupt Operation

**The match interrupt timing diagram is shown in**

Figure 12.1. FRT.MR.MIE should be set as '1' for using match-interrupt.

The FRT clock starts the FRT counter after FRT.CR.EN is '1'. Interrupt and wakeup signals occur when the counter is matched with the value of FRT.PER. The 'interrupt' signal might be delayed in a maximum of 2 system clocks and 'wakeup' signal might be delayed in a maximum of (1 clk + 2 frt_clk).

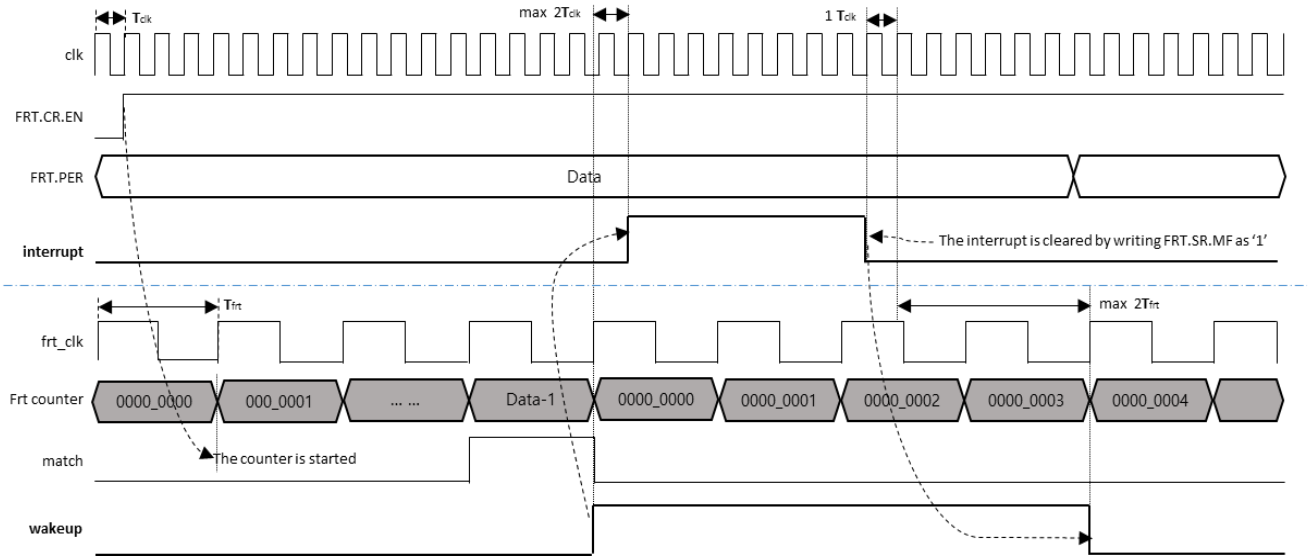



**Figure 12.1. Match Interrupt Operation Timing Diagram**

## Overflow Interrupt Operation

**The overflow interrupt timing diagram is shown in**

Figure 12.2. The overflow-interrupt operation is similar to the match-interrupt operation. The overflow-interrupt is started to set when the FRT counter matches `0xFFFFFFFF`.

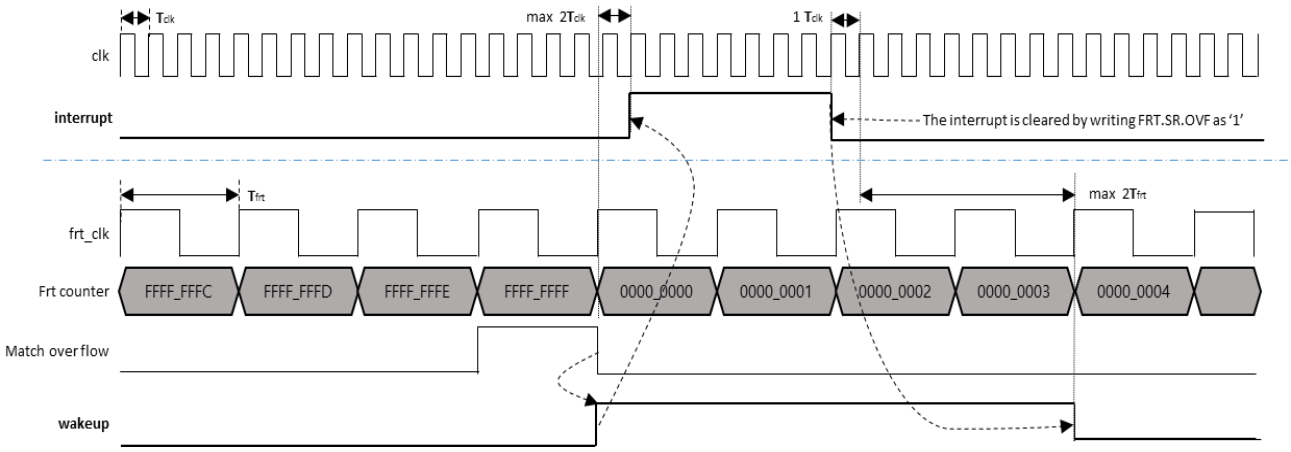



**Figure 12.2. Overflow Interrupt Operation Timing Diagram**

# 13. UART

## Overview

4-Channel Universal Asynchronous Receiver/Transmitter (UART) modules are provided. There is dedicated DMA support to data transfer between the memory buffer and the transmit or receive buffer of the UART block.

The UART operation status, including error status, can be read from the status register. The prescaler, which generates the correct baud rate, exists for each UART channel. The prescaler can divide the UART clock source, PCLK/2, from 1 to 65535. The baud rate is generated by the clock which is internally divided by 16 of the prescaled clock and 8-bit precision clock tuning function.

The programmable interrupt generation function helps control communication via the UART channel.

**Features**

- Compatible with 16450
- Supports DMA transfer
- Standard asynchronous control bit (start, stop, and parity) configurable
- Programmable 16-bit fractional baud generator
- Programmable serial communication
- 5-, 6-, 7- or 8- bit data transfer
- Even, odd, or no-parity bit insertion and detection
- 1-, 1.5- or 2-stop bit-insertion and detection
- 16-bit baud rate generation with 8-bit fraction control
- Hardware inter-frame delay function
- Stop bit error detection
- Detail status register

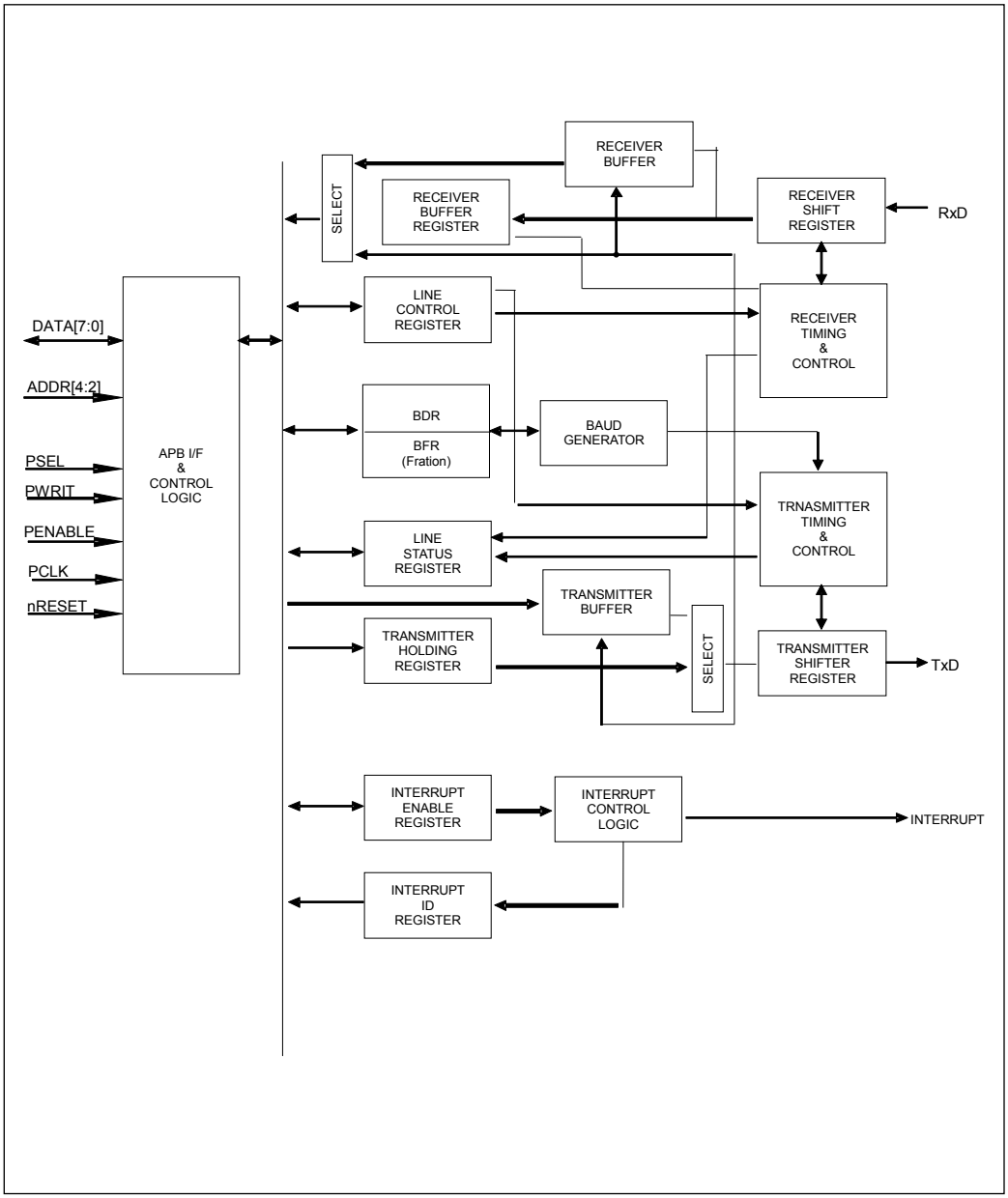



**Figure 13.1. Block Diagram**

# Pin Description

Table 13.1. External Signal

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **TXD0** | O | UART Channel 0 transmit output |
| **RXD0** | I | UART Channel 0 receive input |
| **TXD1** | O | UART Channel 1 transmit output |
| **RXD1** | I | UART Channel 1 receive input |
| **TXD2** | O | UART Channel 2 transmit output |
| **RXD2** | I | UART Channel 2 receive input |
| **TXD3** | O | UART Channel 3 transmit output |
| **RXD3** | I | UART Channel 3 receive input |

# Registers

The base address of UART is `0x4000_8000` and the register map is described in Tables 13.2 and 13.3.

Table 13.2. Base Address of Each Port

| UART Channel | Address |
|---|---|
| **UART0** | 0x4000_8000 |
| **UART1** | 0x4000_8100 |
| **UART2** | 0x4000_8200 |
| **UART3** | 0x4000_8300 |

Table 13.3. UART Register Map

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **Un.RBR** | 0x00 | R | Receive data buffer register | 0x00 |
| **Un.THR** | 0x00 | W | Transmit data hold register | 0x00 |
| **Un.IER** | 0x04 | R/W | Interrupt enable register | 0x00 |
| **Un.IIR** | 0x08 | R | Interrupt ID register | 0x01 |
| **Un.LCR** | 0x0C | R/W | Line control register | 0x00 |
| **Un.DCR** | 0x10 | R/W | Data Control Register | |
| **Un.LSR** | 0x14 | R | Line status register | 0x00 |
| **Un.SCR** | 0x1C | R/W | Scratch pad register | 0x00 |
| **Un.BDR** | 0x20 | R/W | Baud rate Divisor Latch Register | |
| **Un.BFR** | 0x24 | R/W | Baud rate Fractional Counter Value | 0x00 |
| **Un.IDTR** | 0x30 | R/W | Inter-frame Delay Time Register | 0x00 |

## Un.RBR Receive Buffer Register

The UART Receive Buffer Register is an 8-bit read-only register.

**U0.RBR=0x4000_8000, U1.RBR=0x4000_8100**
**U2.RBR=0x4000_8200, U3.RBR=0x4000_8300**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RBR[7:0] | | | | | | | |
| - | | | | | | | |
| RO | | | | | | | |

| | | |
|---|---|---|
| 7<br>0 | RBR | Receive Buffer Register |

## Un.THR Transmit Data Hold Register

The UART Transmit Data Hold Register is an 8-bit write-only register.

**U0.THR=0x4000_8000, U1.THR=0x4000_8100**
**U2.THR=0x4000_8200, U3.THR=0x4000_8300**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| THR | | | | | | | |
| - | | | | | | | |
| WO | | | | | | | |

| | | |
|---|---|---|
| 7<br>0 | THR | Transmit Data Hold Register |

# Un.IER UART Interrupt Enable Register

The UART Interrupt Enable Register is an 8-bit register.

**U0.IER=0x4000_8004, U1.IER=0x4000_8104**
**U2.IER=0x4000_8204, U3.IER=0x4000_8304**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | DTXIEN | DRXIEN | TXEIE | RLSIE | THREIE | DRIE |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 5 | DTXIEN | DMA transmit done interrupt enable |
| | | 0 Receive line status interrupt is disabled |
| | | 1 Receive line status interrupt is enabled |
| 4 | DRXIEN | DMA receive done interrupt enable |
| | | 0 DMA receive done interrupt is disabled |
| | | 1 DMA receive done interrupt is enabled |
| 3 | TXEIE | End-of-transmit interrupt enable |
| | | 0 End-of-transmit interrupt is disabled |
| | | 1 End-of-transmit interrupt is enabled |
| 2 | RLSIE | Receiver line status interrupt enable |
| | | 0 Receive line status interrupt is disabled |
| | | 1 Receive line status interrupt is enabled |
| 1 | THREIE | Transmit holding register empty interrupt enable |
| | | 0 Transmit holding register empty interrupt is disabled |
| | | 1 Transmit holding register empty interrupt is enabled |
| 0 | DRIE | Data receive interrupt enable |
| | | 0 Data receive interrupt is disabled |
| | | 1 Data receive interrupt is enabled |

## Un.IIR UART Interrupt ID Register

The UART Interrupt ID Register is an 8-bit register. Reading this register will clear the interrupts.

**U0.IIR=0x4000_8008, U1.IIR=0x4000_8108**
**U2.IIR=0x4000_8208, U3.IIR=0x4000_8308**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | TXE | IID | | | IPEN |
| 0 | 0 | 0 | 0 | 000 | | | 0 |
| | | | R | R | | | R |

| | | |
|---|---|---|
| 4 | TXE | Interrupt source ID<br>See interrupt source ID table |
| 3<br>1 | IID | Interrupt source ID<br>See interrupt source ID table |
| 0 | IPEN | Interrupt pending bit<br>0 Interrupt is pending<br>1 No interrupt is pending. |

The UART supports 3-priority interrupt generation and the Interrupt Source ID register shows one interrupt source which has the highest priority amongst pending interrupts. The priority is defined as follows:

- Receive line status interrupt
- Receive data ready interrupt/ character timeout interrupt
- Transmit hold register empty interrupt
- Tx/Rx DMA complete interrupt

| Priority | TXE | DMA | IID | | IPEN | Interrupt sources | | |
|---|---|---|---|---|---|---|---|---|
| | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Interrupt | Interrupt condition | Interrupt clear |
| - | 0 | 0 | 0 | 0 | 1 | None | - | - |
| 1 | 0 | 0 | 1 | 1 | 0 | Receiver Line Status | Overrun, Parity, Framing or Break Error | Read LSR register |
| 2 | 0 | 0 | 1 | 0 | 0 | Receiver Data Available | Receive data is available. | Read receive register or read IIR register |
| 3 | 0 | 0 | 0 | 1 | 0 | Transmitter Holding Register Empty | Transmit buffer empty | Write transmit hold register or read IIR register |
| 4 | 1 | X | X | X | X | Transmitter Register Empty | Transmit register empty | Write transmit hold register or read IIR register |
| 5 | 0 | 1 | 1 | 0 | 0 | Rx DMA done | Rx DMA completed. | Read IIR register |
| 6 | 0 | 1 | 0 | 1 | 0 | Tx DMA done | Tx DMA completed. | Read IIR register |
| 7 | 1 | X | X | X | X | Transmitter register Empty and DMA done | Transmitter regiser Empty and Tx DMA completed. | Read IIR register |

# Un.LCR UART Line Control Register

The UART Line Control Register is an 8-bit register.

**U0.LCR=0x4000_800C, U1.LCR=0x4000_810C**
**U2.LCR=0x4000_820C, U3.LCR=0x4000_830C**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | BREAK | STICKP | PARITY | PEN | STOPBIT | DLEN | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | |
|---|---|---|
| 6 | BREAK | When this bit is set, TxD pin will be driven at low state in order to notice the alert to the receiver. |
| | 0 | Normal transfer mode |
| | 1 | Break transmit mode |
| 5 | STICKP | Force parity and it will be effective when PEN bit is set. |
| | 0 | Parity stuck is disabled |
| | 1 | Parity stuck is enabled and parity always the bit of PARITY. |
| 4 | PARITY | Parity mode selection bit and stuck parity select bit |
| | 0 | Odd parity mode |
| | 1 | Even parity mode |
| 3 | PEN | Parity bit transfer enable |
| | 0 | The parity bit disabled |
| | 1 | The parity bit enabled |
| 2 | STOPBIT | The number of stop bit followed by data bits. |
| | 0 | 1 stop bit |
| | 1 | 1.5 / 2 stop bit<br>In case of 5 bit data case, 1.5 stop bit is added. In case of 6,7 or 8 bit data, 2 stop bit is added |
| 1<br>0 | DLEN | The data length in one transfer word. |
| | 00 | 5 bit data |
| | 01 | 6 bit data |
| | 10 | 7 bit data |
| | 11 | 8 bit data |

The parity bit is generated according to bits 3,4,5 of the UnLCR register. Table 13.4 shows the variation of parity bit generation.

**Table 13.4. Parity Bit Generation**

| STICKP | PARITY | PEN | Parity |
|---|---|---|---|
| X | X | 0 | No Parity |
| 0 | 0 | 1 | Odd Parity |
| 0 | 1 | 1 | Even Parity |
| 1 | 0 | 1 | Force parity as “1” |
| 1 | 1 | 1 | Force parity as “0” |

# Un.DCR UART Data Control Register

The UART Data Control Register is an 8-bit register.

**U0DCR=0x4000_8010, U1DCR=0x4000_8110**
**U2DCR=0x4000_8210, U3DCR=0x4000_8310**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | RXINV | TXINV | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | R/W | R/W | | |

| | | |
|---|---|---|
| 3 | RXINV | Rx Data Inversion Selection |
| | | 0 Normal RxData Input |
| | | 1 Inverted RxData Input |
| 2 | TXINV | Tx Data Inversion Selection |
| | | 0 Normal TxData Output |
| | | 1 Inverted TxData Output |

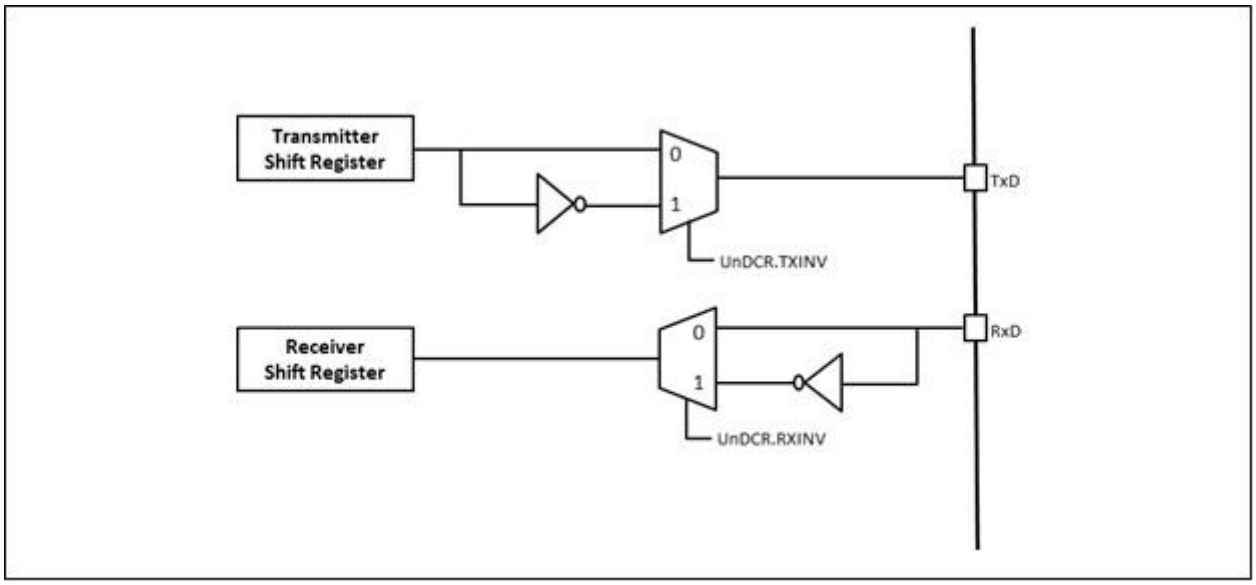



**Figure 13.2. Data Inversion Diagram**

## Un.LSR UART Line Status Register

The UART Line Status Register is an 8-bit register.

U0LSR=0x4000_8014, U1LSR=0x4000_8114
U2LSR=0x4000_8214, U3LSR=0x4000_8314

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | TEMT | THRE | BI | FE | PE | OE | DR |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | R | R | R | R | R | R | R |

| | | |
|---|---|---|
| 6 | TEMT | Transmit empty. |
| | 0 | Transmit register has the data is now transferring |
| | 1 | Transmit register is empty. |
| 5 | THRE | Transmit holding empty. |
| | 0 | Transmit holding register is not empty. |
| | 1 | Transmit holding register empty |
| 4 | BI | Break condition indication bit |
| | 0 | Normal status |
| | 1 | Break condition is detected |
| 3 | FE | Frame Error. |
| | 0 | No framing error. |
| | 1 | Framing error. The receive character did not have a valid stop bit |
| 2 | PE | Parity Error |
| | 0 | No parity error |
| | 1 | Parity error. The receive character does not have correct parity information. |
| 1 | OE | Overrun error |
| | 0 | No overrun error |
| | 1 | Overrun error. Additional data arrives while the RHR is full |
| 0 | DR | Data received |
| | 0 | No data in receive holding register. |
| | 1 | Data has been received and is saved in the receive holding register |

This register provides the status of data transfers between the transmitter and receiver. Users can get line status information from this register and can handle the next process. Bits 1,2,3,4 will arise the line status interrupt when the RLSIE bit in the UnIEN register is set. Other bits can generate its interrupt when the interrupt enable bit in the UnIEN register is set.

## Un.BDR Baud rate Divisor Latch Register

The UART Baud rate Divisor Latch Register is a 16-bit register.

U0.BDR=0x4000_8020, U1.BDR=0x4000_8120
U2.BDR=0x4000_8220, U3.BDR=0x4000_8320

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BDR | | | | | | | | | | | | | | | |
| 0x0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 0 | BDR | Baud rate Divider latch value |

To establish communication with the UART channel, the baud rate should be set correctly. The programmable baud rate generation allows division from 1 to 65535. The 16 bit divider register (UnBDR) should be written for the expected baud rate.

The baud rate calculation formula is shown in the following equation:

$$\mathrm{BDR} = \frac{UART_{PCLK}}{32 \times BaudRate}$$

For an 72 MHz UART_PCLK speed, the divider value and error rate is described in Table 13.5.

**Table 13.5. Example of Baud Rate Calculation**

| UART_PCLK=72 MHz | | |
|---|---|---|
| **Baud rate** | **Divider** | **Error (%)** |
| 1200 | 1875 | 0.00% |
| 2400 | 937 | 0.05% |
| 4800 | 468 | 0.16% |
| 9600 | 234 | 0.16% |
| 19200 | 117 | 0.16% |
| 38400 | 58 | 1.02% |
| 57600 | 39 | 0.16% |
| 115200 | 19 | 2.79% |

# Un.BFR Baud rate Fraction Counter Register

The Baud rate Fraction Counter Register is an 8-bit register.

**U0.BFR=0x4000_8024, U1.BFR=0x4000_8124**
**U2.BFR=0x4000_8224, U3.BFR=0x4000_8324**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BFR | | | | | | | |
| 0x00 | | | | | | | |
| R/W | | | | | | | |

| | | | |
|---|---|---|---|
| 7 | BFR | Fractions counter value. | |
| 0 | | 0 | Fraction counter is disabled |
| | | N | Fraction counter enabled. Fraction compensation mode is operating. Fraction counter is incremented by FCNT. |

**Table 13.6 Example of Baud Rate Calculation**

| UART_PCLK=72 MHz | | | |
|---|---|---|---|
| **Baud rate** | **Divider** | **FCNT** | **Error (%)** |
| 1200 | 1875 | 0 | 0.0% |
| 2400 | 937 | 128 | 0.0% |
| 4800 | 468 | 192 | 0.0% |
| 9600 | 234 | 96 | 0.0% |
| 19200 | 117 | 48 | 0.0% |
| 38400 | 58 | 152 | 0.0% |
| 57600 | 39 | 16 | 0.0% |
| 115200 | 19 | 136 | 0.0% |

$$FCNT = Float * 256$$

The 8-bit fractional counter will count up by the FCNT value every (baud rate)/16 period and when the fractional counter overflow occurs, the divisor value increments by 1. Therefore. this period is compensated. In the next period, the divisor value returns to the original set value.

For example, if 9600 bps,

$$\frac{PCLK / 2}{16 \times BaudRate} = \frac{72000000 / 2}{16 \times 9600} = 234.375 \quad Divider = 234 \quad Float = .375$$

$$FCNT = Float \times 256 = .375 \times 256 = 96$$

BDR = 234, BFR = 96

## Un.IDTR Inter-frame Delay Time Register

The UART Inter-frame Time Register is an 8-bit register. A dummy delay can be inserted between two continuous transmits.

**U0.IDTR=0x4000_8030, U1.IDTR=0x4000_8130**
**U2.IDTR=0x4000_8230, U3.IDTR=0x4000_8330**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | | | | | WAITVAL | | |
| 0 | 0 | 0 | 0 | 0 | 000 | | |
| | | | | | RW | | |

| | | |
|---|---|---|
| 2 0 | WAITVAL | Wait time is decided by this value<br>$\text{Wait Time} = \frac{WAITVAL}{BAUDRATE}$ |

# Functional Description

The UART module is compatible with the 16450 UART. Additionally, dedicated DMA channels and fractional baud rate compensation logic are provided. The UART does not have an internal FIFO block. Therefore, data transfers are established interactively or with DMA support.

Two DMA channels are provided for each UART module – one channel for TX transfer and the other channel for RX transfer. Each channel has a 32-bit memory address register and a 16-bit transfer counter register. Prior to the DMA operation, the DMA memory address register and transfer count register should be configured. For the RX operation, the memory address is the destination memory address and for the TX operation, the memory address is the source memory address.

The transfer counter register stores the number of data transfers. When a single transfer is done, the counter is decremented by 1. When the counter reaches zero, the DMA done flag is delivered to the UART control block. If the interrupt is enabled, this flag generates the interrupt.

## Receiver Sampling Timing

The UART operates per the following timing:

If the falling edge is on the receive line, the UART judges it as the start bit. From the start timing, the UART oversamples 16 times of 1-bit and detects the bit value at the 7th sample of 16 samples.

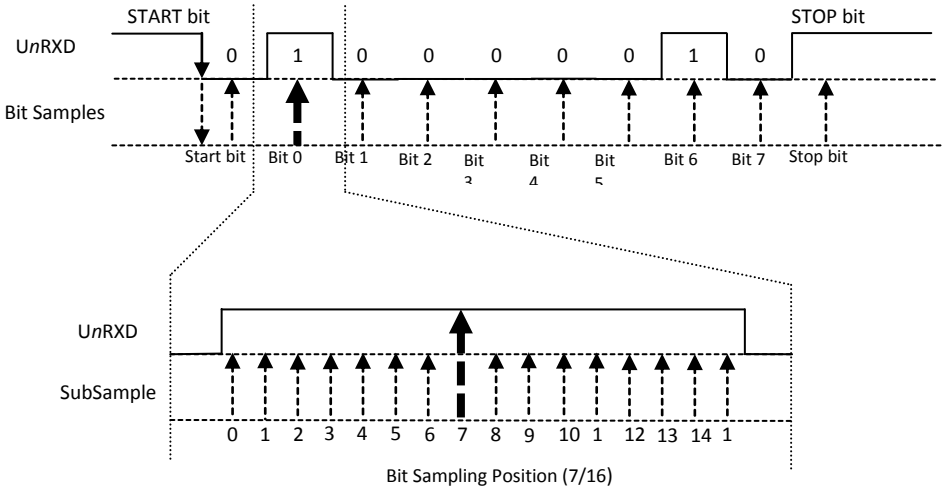



**Figure 13.3 The Sampling Timing of UART Receiver**

**Note:** Zilog recommends enabling debounce settings in the PCU block to reinforce the immunity of external glitch noise.

## Transmitter

The transmitter's function is to transmit data. The start bit, data bits, optional parity bit, and stop bit are serially shifted with the least significant bit first. The number of data bits is selected in the DLAN[1:0] field in the Un.LCR register.

The parity bit is set according to the PARITY and PEN bit field in the Un.LCR register. If the parity type is even, then the parity bit depends on the one bit sum of all data bits. For odd parity, the parity bit is the inverted sum of all data bits.

The number of stop bits is selected in the STOPBIT field in the Un.LCR register.

An example of transmit data format is shown in Figure 13.4.

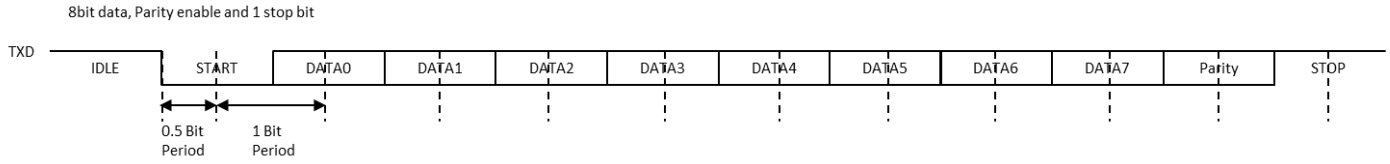



**Figure 13.4. Transmit Data Format Example**

## Inter-frame Delay Transmission

The Inter-frame Delay function allows the transmitter to insert an idle state on the TXD line between two characters. The width of the idle state is defined in the WAITVAL field in the Un.IDTR register. When this field is set to 0, no time-delay is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted character during the number of bit periods defined in the WATIVAL field.

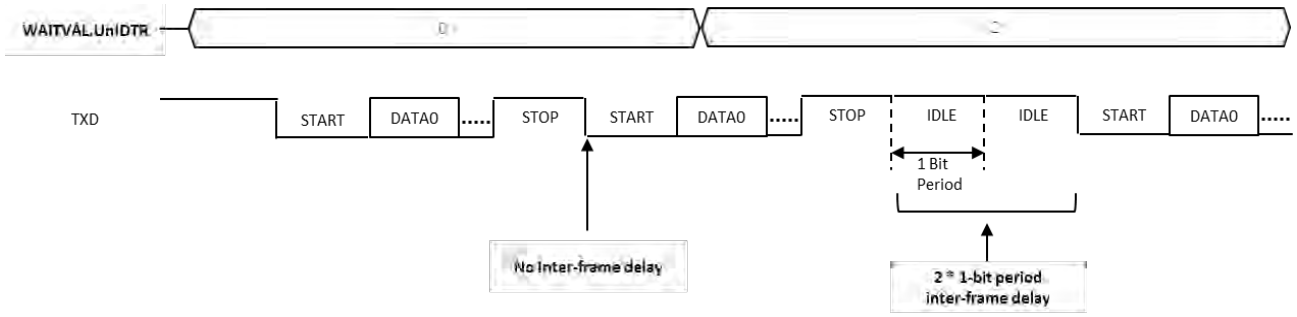



**Figure 13.5. Inter-frame Delay Timing Diagram**

## Transmit Interrupt

The transmit operation generates interrupt flags. When the transmitter holding register is empty, the THRE interrupt flag is set. Whe the transmitter shifter register is empty, the TXE interrupt flag is set. Users can select which interrupt timing is best for the application.

**Figure 13.6. Transmit Interrupt Timing Diagram**

## DMA Transfers

The UART supports the DMA interface function (optionally provided, depending on the device). The start memory address for transfer data and the length of transfer data are programmd in the registers in the DMA block.

The end of transfer is notified via the related transfer done flag.

Transmit with DMA operation invokes the DTX.UnIIR DMA TX done flag and sets the DMA TX done interrupt ID when all the transmit data are written to the transmit holding register. Two transmit data values remain in the registers of the UART block after the DMA transfer done interrupt.

Receive with DMA operation invokes the RXT.UnIIR DMA RX done flag and sets the DMA RX done interrupt ID when all the receive data are written to the destination memory. Therefore, the UART RXD signal is already in IDLE state when the DMA RX done interrupt is issued.

# 14. Serial Peripheral Interface

## Overview

2-channel serial interfaces are provided for synchronous serial communications with external peripherals. The SPI block supports the master and slave modes. Four signals are used for SPI communication – SS, SCK, MOSI, and MISO.

- Master or Slave operation
- Programmable clock polarity and phase
- 8,9,16,17-bit wide transmit/receive register
- 8,9,16,17-bit wide data frame
- Loop-back mode
- Programmable start, burst, and stop delay time
- DMA handshake operation

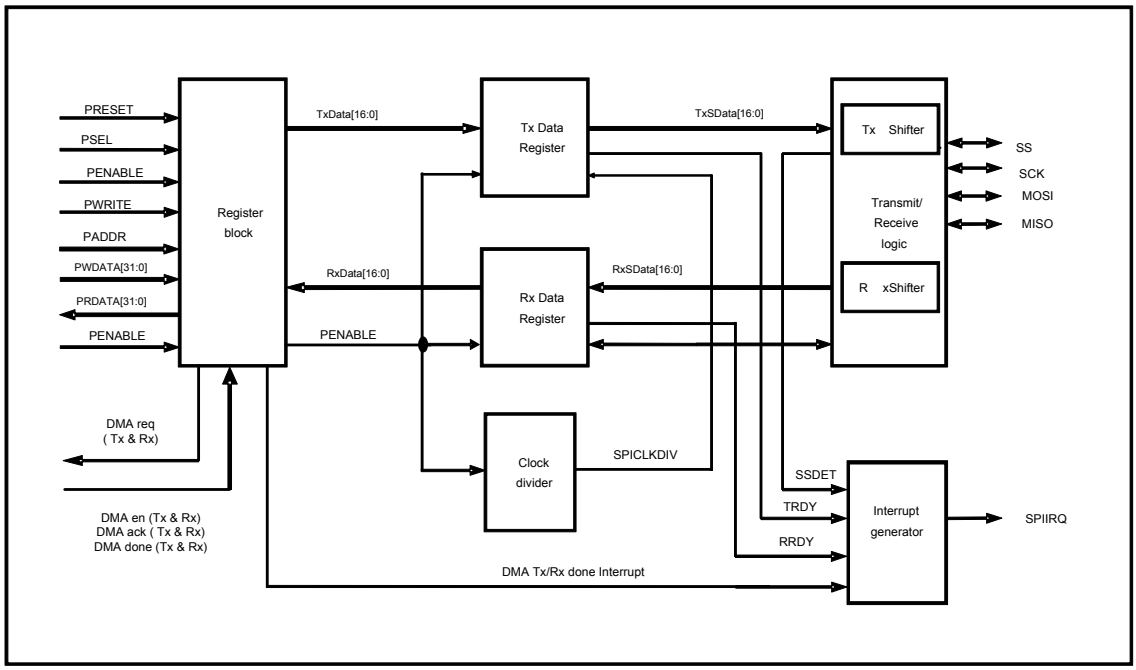



**Figure 14.1. SPI Block Diagram**

# Pin Description

**Table 14.1. External Pins**

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **SS0** | I/O | SPI0 Slave select input / output |
| **SCK0** | I/O | SPI0 Serial clock input / output |
| **MOSI0** | I/O | SPI0 Serial data ( Master output, Slave input ) |
| **MISO0** | I/O | SPI0 Serial data ( Master input, Slave output ) |
| **SS1** | I/O | SPI1 Slave select input / output |
| **SCK1** | I/O | SPI1 Serial clock input / output |
| **MOSI1** | I/O | SPI1 Serial data ( Master output, Slave input ) |
| **MISO1** | I/O | SPI1 Serial data ( Master input, Slave output ) |

# Registers

The base address of SPI is `0x4000_9000` and the register map is described in Tables 14.2 and 14.3.

**Table 14.2. SPI Base Address**

| Channel | Base address |
|---|---|
| **SPI0** | 0x4000_9000 |
| **SPI1** | 0x4000_9100 |

**Table 14.3 SPI Register Map**

| Name | Offset | TYPE | Description | Reset |
|---|---|---|---|---|
| **SP*n*.TDR** | 0x--00 | W | SPI n Transmit Data Register | - |
| **SP*n*.RDR** | 0x--00 | R | SPI n Receive Data Register | 0x000000 |
| **SP*n*.CR** | 0x--04 | R/W | SPI n Control Register | 0x001020 |
| **SP*n*.SR** | 0x--08 | R/W | SPI n Status Register | 0x000006 |
| **SP*n*.BR** | 0x--0C | R/W | SPI n Baud rate Register | 0x0000FF |
| **SP*n*.EN** | 0x--10 | R/W | SPI n Enable register | 0x000000 |
| **SPn.LR** | 0x--14 | R/W | SPI n delay Length Register | 0x010101 |

# SP*n*.CR SPI n Control Register

SPnCR is a 20-bit read/write register and can be set to configure SPI operation mode.

**SP0.CR=0x4000_9004, SP1.CR=0x4000_9104**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | TXBC | RXBC | DTXIE | DRXIE | SSCIE | TXIE | RXIE | SSMOD | SSOUT | LBE | SSMASK | SSMO | SSPOL | | | MS | MSBF | CPHA | CPOL | BITSZ | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 00 | |
| | | | | | | | | | | | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | | | RW | RW | RW | RW | RW | | |

| | | |
|---|---|---|
| **20** | **TXBC** | Tx buffer clear bit. |
| | | 0 No action |
| | | 1 Clear Tx buffer |
| **19** | **RXBC** | Rx buffer clear bit |
| | | 0 No action |
| | | 1 Clear Rx buffer |
| **18** | **DTXIE** | DMA Tx Done Interrupt Enable bit. |
| | | 0 DMA Tx Done Interrupt is disabled. |
| | | 1 DMA Tx Done Interrupt is enabled. |
| **17** | **DRXIE** | DMA Rx Done Interrupt Enable bit. |
| | | 0 DMA Rx Done Interrupt is disabled. |
| | | 1 DMA Rx Done Interrupt is enabled. |
| **16** | **SSCIE** | nSS Edge Change Interrupt Enable bit. |
| | | 0 nSS interrupt is disabled. |
| | | 1 nSS interrupt is enabled for both edges (L→H, H→L) |
| **15** | **TXIE** | Transmit Interrupt Enable bit. |
| | | 0 Transmit Interrupt is disabled. |
| | | 1 Transmit Interrupt is enabled. |
| **14** | **RXIE** | Receive Interrupt Enable bit.. |
| | | 0 Receive Interrupt is disabled. |
| | | 1 Receive Interrupt is enabled. |
| **13** | **SSMOD** | SS Auto/Manual output select bit. |
| | | 0 SS output is not set by SSOUT (SPnCR[12]).<br>- SS signal is in normal operation mode. |
| | | 1 SS output signal is set by SSOUT. |
| **12** | **SSOUT** | SS output signal select bit. |
| | | 0 SS output is 'L.' |
| | | 1 SS output is 'H'. |
| **11** | **LBE** | Loop-back mode select bit in master mode. |
| | | 0 Loop-back mode is disabled. |
| | | 1 Loop-back mode is enabled. |
| **10** | **SSMASK** | SS signal masking bit in slave mode. |
| | | 0 SS signal masking is disabled.<br>- Receive data when SS signal is active. |
| | | 1 SS signal masking is enabled.<br>- Receive data at SCLK edges. SS signal is ignored. |
| **9** | **SSMO** | SS output signal select bit. |
| | | 0 SS output signal is disabled. |
| | | 1 SS output signal is enabled. |
| **8** | **SSPOL** | SS signal Polarity select bit. |
| | | 0 SS signal is Active-Low. |
| | | 1 SS signal is Active-High. |
| **7**<br>**6** | | Reserved |

| | | |
|---|---|---|
| 5 | MS | Master/Slave select bit. |
| | | 0 SPI is in Slave mode. |
| | | 1 SPI is in Master mode. |
| 4 | MSBF | MSB/LSB Transmit select bit. |
| | | 0 LSB is transferred first. |
| | | 1 MSB is transferred first. |
| 3 | CPHA | SPI Clock Phase bit. |
| | | 0 Sampling of data occurs at odd edges (1,3,5,...,15). |
| | | 1 Sampling of data occurs at even edges (2,4,6,...,16). |
| 2 | CPOL | SPI Clock Polarity bit. |
| | | 0 Active-high clocks selected. |
| | | 1 Active-low clocks selected. |
| 1 | BITSZ | Transmit/Receive Data Bits select bit. |
| | | 00 8 bits |
| | | 01 9 bits |
| | | 10 16 bits |
| 0 | | 11 17 bits |

CPOL=0, CPHA=0 : data sampling at rising edge, data changing at falling edge
CPOL=0, CPHA=1 : data sampling at falling edge, data changing at rising edge
CPOL=1, CPHA=0 : data sampling at falling edge, data changing at rising edge
CPOL=1, CPHA=1 : data sampling at rising edge, data changing at falling edge

# SP*n*.SR SPI n Status Register

SPnSR is a 10-bits read/write register. It contains the status of SPI interface.

SP0.SR=0x4000_9008, SP1.SR=0x4000_9108

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TXDMAF | RXDMAF | SBUSY | SSDET | SSON | OVRF | UDRF | SRDY | TRDY | RRDY |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| | | | | | | RC1 | RC1 | R | RC1 | RC1 | RC1 | RC1 | R | R | R |

| | | |
|---|---|---|
| 9 | **TXDMAF** | DMA Transmit Operation Complete flag. (DMA to SPI) |
| | 0 | DMA Transmit Op is working or is disabled. |
| | 1 | DMA Transmit Op is done. |
| 8 | **RXDMAF** | DMA Receive Operation Complete flag. (SPI to DMA ) |
| | 0 | DMA Receive Operation is working or is disabled. |
| | 1 | DMA Transmit Op is done. |
| 8 | **SBUSY** | Transmit/Receive Operation flag |
| | 0 | SPI is in IDLE state |
| | 1 | SPI is operating |
| 6 | **SSDET** | The rising or falling edge of SS signal Detect flag. |
| | 0 | SS edge is not detected. |
| | 1 | SS edge is detected.<br>- The bit is cleared when it is written as “0”. |
| 5 | **SSON** | SS signal Status flag. |
| | 0 | SS signal is inactive. |
| | 1 | SS signal is active. |
| 4 | **OVRF** | Receive Overrun Error flag. |
| | 0 | Receive Overrun error is not detected. |
| | 1 | Receive Overrun error is detected.<br>- This bit is cleared by writing or reading SPnRDR. |
| 3 | **UDRF** | Transmit Underrun Error flag. |
| | 0 | Transmit Underrun is not occurred. |
| | 1 | Transmit Underrun is occurred.<br>- This bit is cleared by writing or reading SPnTDR. |
| 2 | **SRDY** | Shift register Empty flag. |
| | 0 | Shift register is busy. |
| | 1 | Shift register is ready.<br>- This bit is cleared by writing SPnTDR to Shift rgister and is TRDY’s complement when SSON is active. |
| 1 | **TRDY** | Transmit buffer Empty flag. |
| | 0 | Transmit buffer is busy. |
| | 1 | Transmit buffer is ready.<br>- This bit is cleared by writing data to SPnTDR. |
| 0 | **RRDY** | Receive buffer Ready flag. |
| | 0 | Receive buffer has no data. |
| | 1 | Receive buffer has data.<br>- This bit is cleared by writing data to SPnRDR. |

## SP*n*.TDR SPI n Transmit Data Register

SPnTDR is a 17-bit read/write register. It contains serial transmit data.

**SP0.TDR=0x4000_9000, SP1.TDR=0x4000_9100**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | TDR | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0x00000 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 16<br>0 | TDR | Transmit Data Register |

## SP*n*.RDR SPI n Receive Data Register

SPnRDR is a 17-bit read/write register. It contains serial receive data.

**SP0.RDR=0x4000_9000, SP1.RDR=0x4000_9100**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | RDR | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0x00000 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 16<br>0 | RDR | Receive Data Register |

## SP*n*.BR SPI n Baud Rate Register

SPnBR is an16-bit read/write register. Baud rate can be set by writing to this register.

**SP0.BR=0x4000_900C, SP1.BR=0x4000_910C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BR | | | | | | | | | | | | | | | |
| 0x00FF | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15<br><br>0 | BR | Baud rate setting bits<br>- Baud Rate = PCLK / (BR + 1).<br>(BR must be bigger than “0”, BR >= 2 ) |

# SP*n*.EN SPI n Enable Register

SPnEN is a single-bit read/write register. It contains SPI enable bit.

**SP0.EN=0x4000_9010, SP1.EN=0x4000_9110**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | ENABLE |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | | RW |

| | | |
|---|---|---|
| **0** | **ENABLE** | SPI Enable bit |
| | | 0 SPI is disabled.<br>- SPnSR is initialized by writing “0” to this bit but other registers aren’t initialized. |
| | | 1 SPI is enabled.<br>- When this bit is written as “1”, the dummy data of transmit buffer will be shifted. To prevent this, write data to SPTDR before this bit is active. |

**Note:** When in SPI Slave mode, be sure to disable the SPI prior to loading the TDR register, then enable it to prevent an extra byte from being sent.

# SP*n*.LR SPI n delay Length Register

SPnLR is a 24-bit read/write register. It contains start, burst, and stop length value.

**SP0.CR=0x4000_9014, SP1.CR=0x4000_9114**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SPL | | | | | | | | BTL | | | | | | | | STL | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x01 | | | | | | | | 0x01 | | | | | | | | 0x01 | | | | | | | |
| | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |

| | | |
|---|---|---|
| 23 | SPL | StoPLength value |
| 16 | | 0x01 ~ 0xFF : 1 ~ 255 SCLKs. (SPL >= 1) |
| 15 | BTL | BursTLength value |
| 8 | | 0x01 ~ 0xFF : 1 ~ 255 SCLKs. (BTL >= 1) |
| 7 | STL | STart Length value |
| 0 | | 0x01 ~ 0xFF : 1 ~ 255 SCLKs. (STL >= 1) |

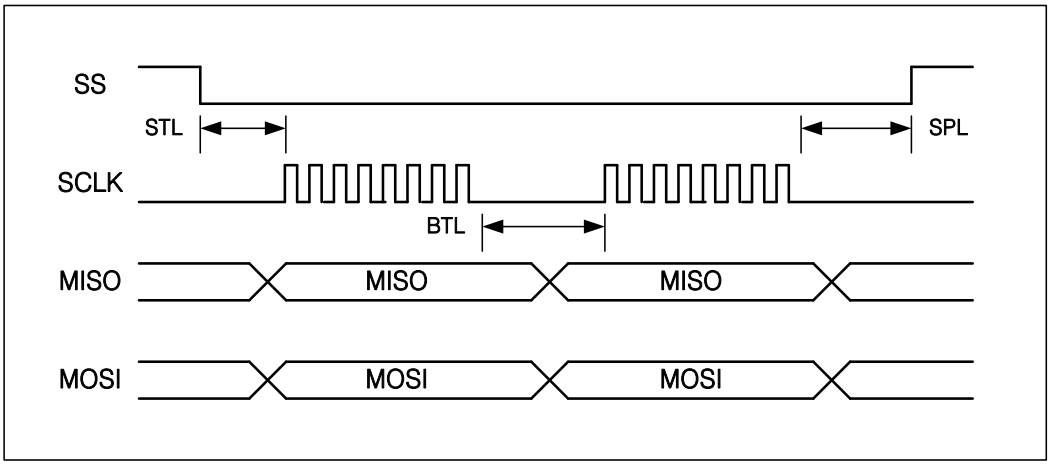



**Figure14.1. SPI Waveform (STL, BTL, and SPL)**

## Functional Description

The SPI Transmit and Receive blocks share Clock Gen Block but they are independent of each other. The Transmit and Receive blocks have double buffers and SPI is available for back-to-back transfer operation.

## SPI Timing

The SPI has four modes of operation. These modes essentially control the way data is clocked in or out of an SPI device. The configuration is done by two bits in the SPI control register (SPnCR). The clock polarity is specified by the CPOL control bit, which selects an active high or active low clock. The clock phase (CPHA) control bit selects one of the two fundamentally different transfer formats. To ensure effective communication between master and slave, both devices have to run in the same mode. This can require a reconfiguration of the master to match the requirements of different peripheral slaves.

The clock polarity has no significant effect on the transfer format. Switching this bit causes the clock signal to be inverted (active high becomes active low and idle low becomes idle high). The settings of the clock phase, however, select one of the two different transfer timings, which are described in the next two chapters. Since the MOSI and MISO lines of the master and the slave are directly connected to each other, the diagrams show the timing of both devices, master and slave. The nSS line is the slave select input of the slave. The nSS pin of the master is not shown in the diagrams. It has to be inactive by a high level on this pin (if configured as input pin) or by configuring it as an output pin.

The timing of a SPI transfer where CPHA is zero is shown in Figures 10.3 and 10.4. Two wave forms are shown for the SCK signal – one where CPOL equals zero and another where CPOL equals one.

When the SPI is configured as a slave, the transmission starts with the falling edge of the /SS line. This activates the SPI of the slave and the MSB of the byte stored in its data register (SPnTDR) is output on the MISO line. The actual transfer is started by a software write to the SPnTDR of the master. This causes the clock signal to be generated. If the CPHA equals zero, the SCLK signal remains zero for the first half of the first SCLK cycle. This ensures that the data is stable on the input lines of both the master and the slave. The data on the input lines is read with the edge of the SCLK line from its inactive to its active. The edge of the SCLK line from its active to its inactive state (falling edge if CPOL equals zero and rising edge if CPOL equals one) causes the data to be shifted one bit further so that the next bit is output on the MOSI and MISO lines.

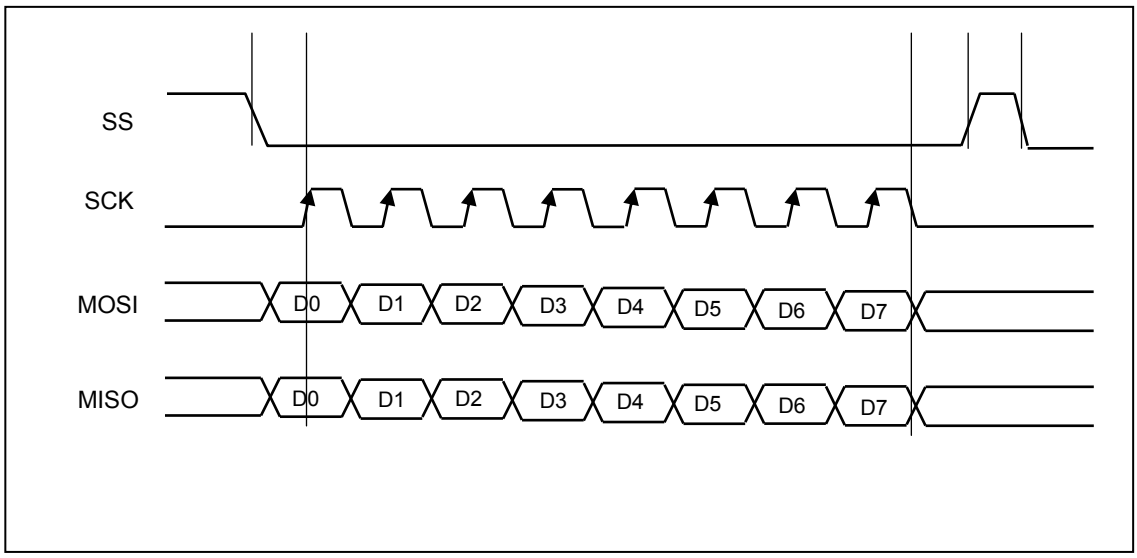



**Figure 14.2.SPI Transfer Timing 1/4 (CPHA=0, CPOL=0, MSBF=0)**

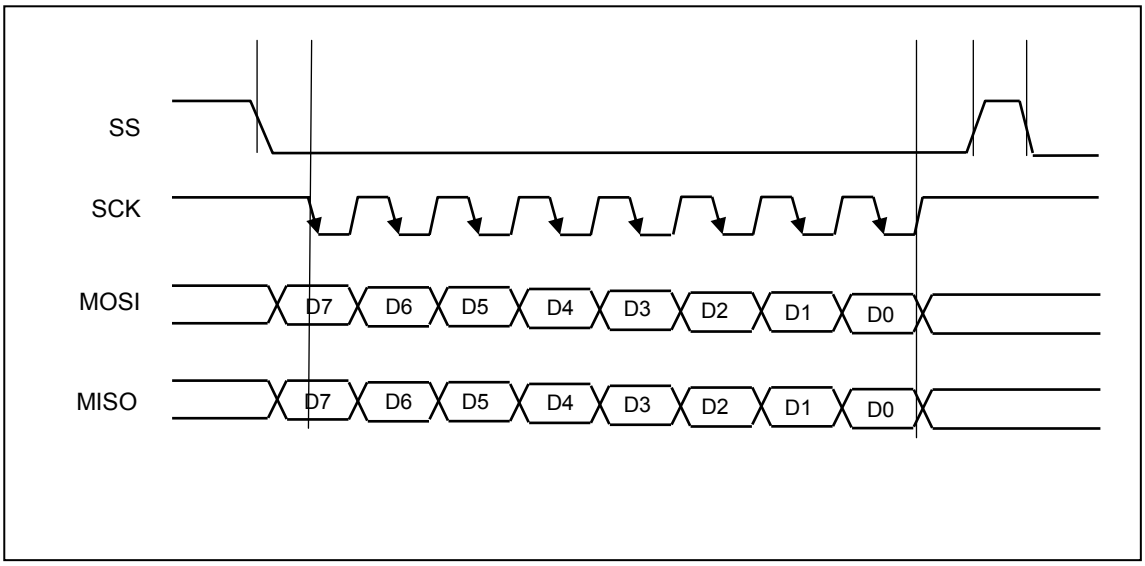



**Figure 14.3.SPI Transfer Timing 2/4 (CPHA=0, CPOL=1, MSBF=1)**

The timing of a SPI transfer where CPHA is one is shown in Figure 10.5 and 10.6. Two wave forms are shown for the SCLK signal – one when CPOL equals zero and another when CPOL equals one.

Similar to the pevious case, the falling edge of the nSS lines selects and activates the slave. Compared to the previous case, where CPHA equals zero, the transmission is not started and the MSB is not output by the slave at this stage. The actual transfer is started by a software write to the SPnTDR of the master that causes the clock signal to be generated. The first edge of the SCLK signal from its inactive to its active state (rising edge if CPOL equals zero and falling edge if CPOL equals one) causes both the master and the slave to output the MSB of the byte in the SPnTDR.

As shown in Figures 14.3 and 14.4, there is no delay of half a SCLK-cycle. The SCLK line changes its level immediately at the beginning of the first SCLK-cycle. The data on the input lines is read with the edge of the SCLK line from its active to its inactive state (falling edge if CPOL equals zero and rising edge if CPOL equals one). After eight clock pulses the transmission is completed.

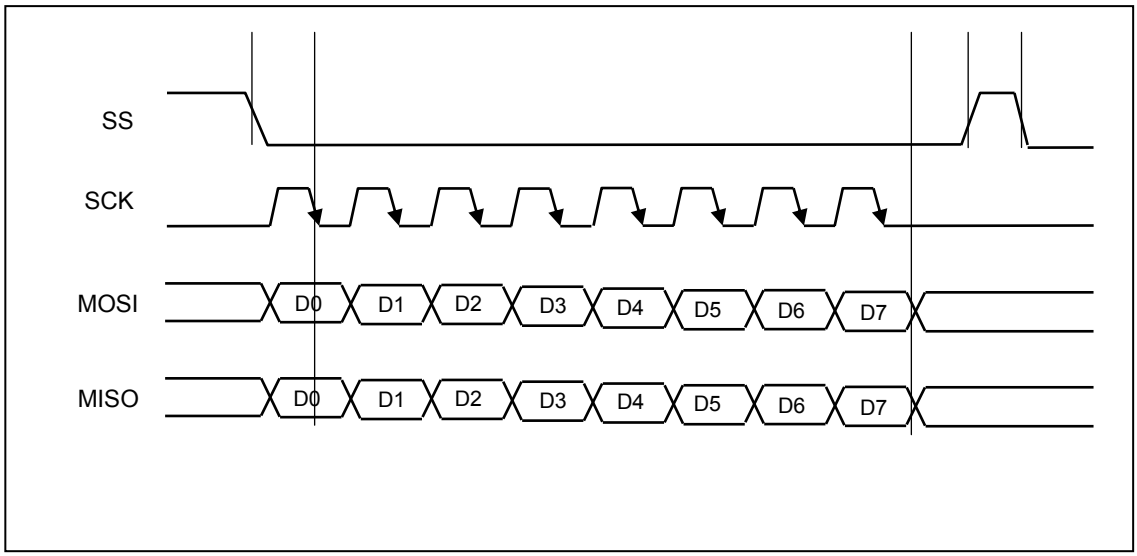



**Figure 14.4.SPI Transfer Timing 3/4 (CPHA=1, CPOL=0, MSBF=0)**

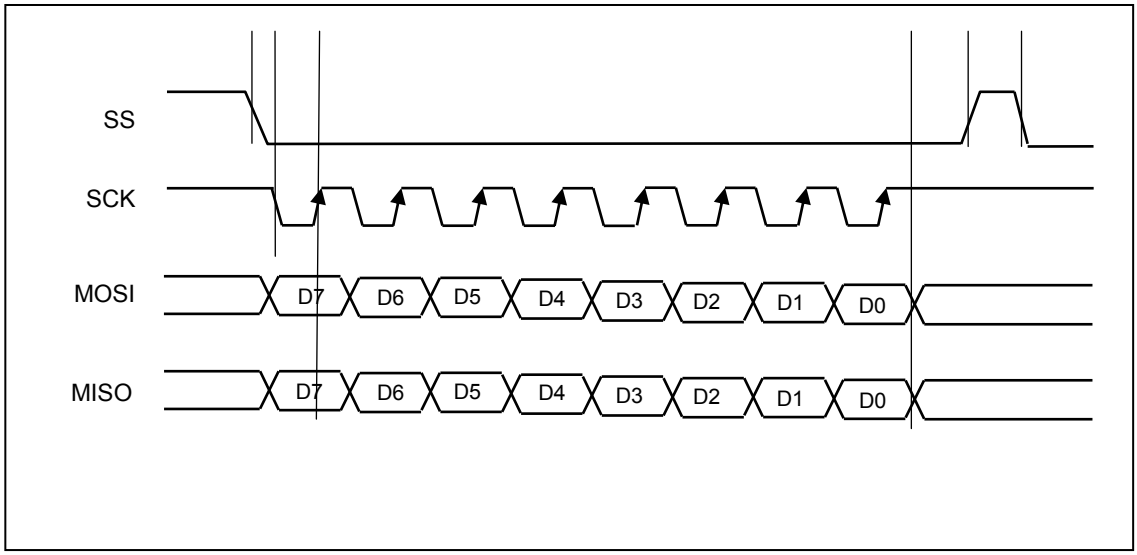



**Figure 14.5.SPI Transfer Timing 4/4 (CPHA=1, CPOL=1, MSBF=1)**

## DMA Handshake

SPI supports DMA handshaking operation. In order to operate DMA handshake, DMA registers should be set first. (See Chapter 9. Direct Memory Access Contoller). SPI0 has 2 channels of DMA, channel 8 for receiver and channel 9 for transmitter. SPI1 has channel 10 for receiver and channel 11 for transmitter. Because the transmitter and receiver are independent of each other, SPI can operate the two channels at the same time.

After the DMA channel for receiver is enabled and the receive buffer is filled, SPI sends Rx request to DMA to empty the buffer and waits for an ACK signal from DMA. If the Receive buffer is filled again after ACK signal, SPI sends an Rx request. If DMA Rx DONE becomes high, RXDMAF (SPnSR[8]) goes “1” and an interrupt is serviced when RXDIE (SPnCR[17]) is set.

Similarly, if the transmit buffer is empty after the DMA channel for transmitter is enabled, SPI sends a Tx request to DMA to fill the buffer and waits for an ACK signal from DMA. If the transmit buffer is empty again after the ACK signal, SPI sends a Tx request. If DMA Tx DONE becomes high, TXDMAF(SPnSR[9]) goes “1” and an interrupt is serviced when TXDIE(SPnCR[18]) is set.

The slave transmitter sends dummy data at the first transfer (8~17 SCLKs) in DMA handshake mode.

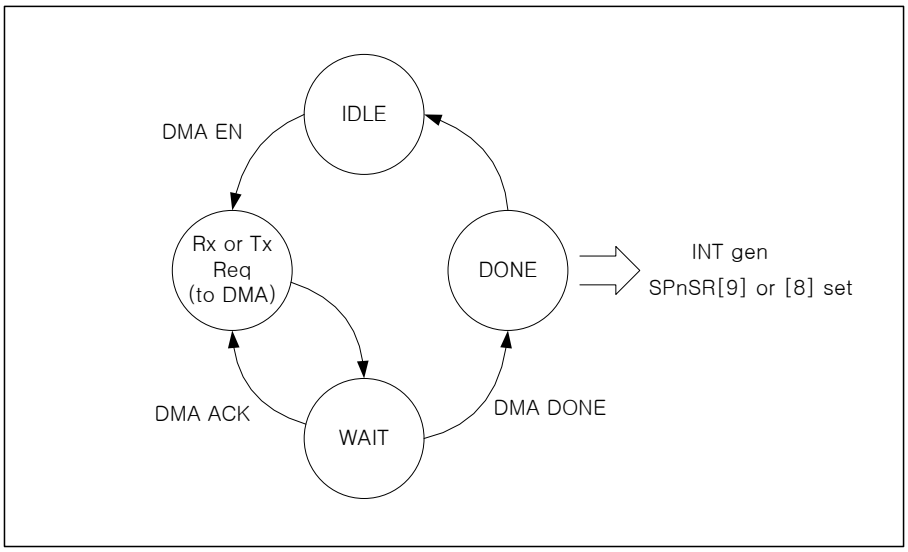



**Figure 14.6.DMA Handshake Flowchart**

# 15. I²C Interface

## Overview

The Inter-Integrated Circuit (I²C) bus serves as an interface between the microcontroller and the serial I²C bus. It provides two wires and a serial bus interface to a large number of popular devices and allows parallel-bus systems to communicate bidirectionally with the I²C-bus.

- Master and slave operation
- Programmable communication speed
- Multi-master bus configuration
- 7-bit addressing mode
- Standard data rate of 100/400 kbps
- STOP signal generation and detection
- START signal generation
- ACK bit generation and detection

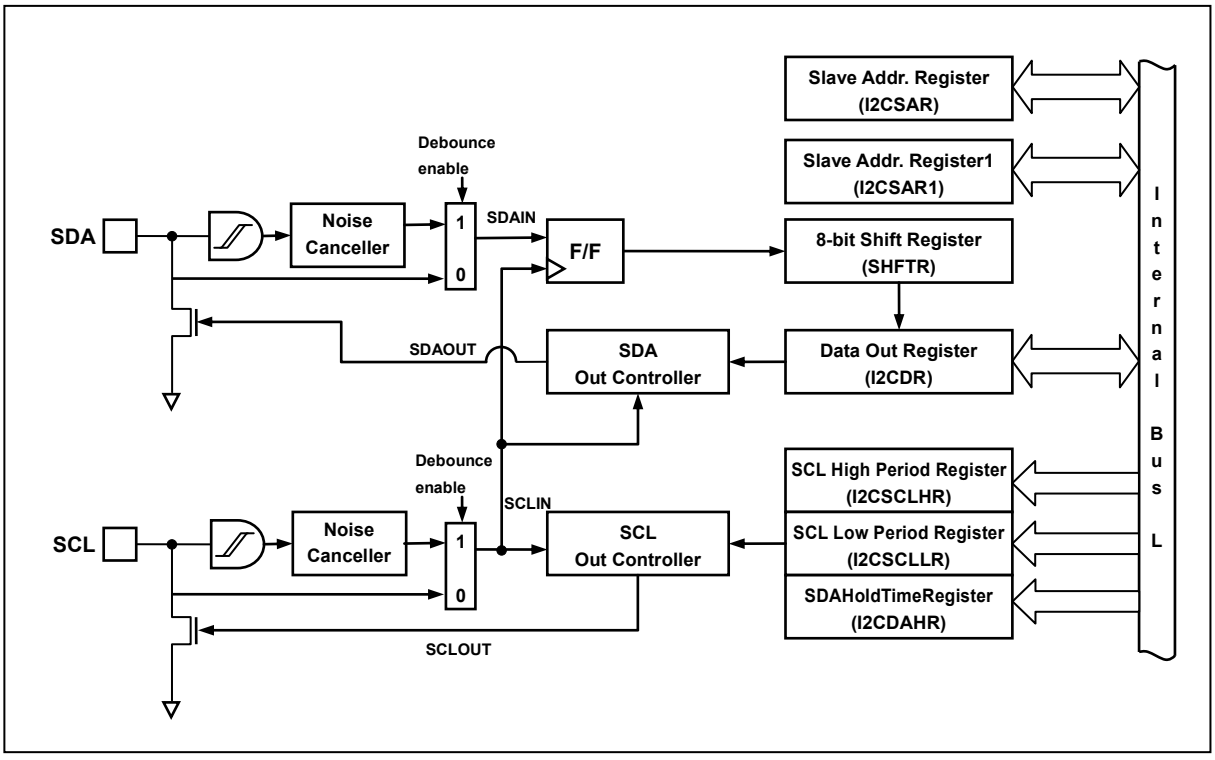



**Figure 15.1. I²C Block Diagram**

# Pin Description

Table 15.1. I²C Interface External Pins

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **SCL0** | I/O | I²C channel 0 Serial clock bus line (open-drain) |
| **SDA0** | I/O | I²C channel 0 Serial data bus line (open-drain) |
| **SCL1** | I/O | I²C channel 1 Serial clock bus line (open-drain) |
| **SDA1** | I/O | I²C channel 1 Serial data bus line (open-drain) |

# Registers

The base address of I²C0 is `0x4000_A000` and the base address of I²C1 is `0x4000_A100`. The register map is described in Tables 15.2 and 15.3.

Table 15.2. I²C Interface Base Address

| Channel | Base address |
|---|---|
| **I²C0** | 0x4000_A000 |
| **I²C1** | 0x4000_A100 |

Table 15.3. I²C Register Map

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **IC0.DR** | 0xA000 | R/W | I²C0 Data Register | 0xFF |
| **IC0.SR** | 0xA008 | R, R/W | I²C0 Status Register | 0x00 |
| **IC0.SAR** | 0xA00C | R/W | I²C0 Slave Address Register | 0x00 |
| **IC0.CR** | 0xA014 | R/W | I²C0 Control Register | 0x00 |
| **IC0.SCLL** | 0xA018 | R/W | I²C0 SCL LOW duration Register | 0xFFFF |
| **IC0.SCLH** | 0xA01C | R/W | I²C0 SCL HIGH duration Register | 0xFFFF |
| **IC0.SDH** | 0xA020 | R/W | I²C0 SDA Hold Register | 0x7F |
| **IC1.DR** | 0xA100 | R/W | I²C1 Data Register | 0xFF |
| **IC1.SR** | 0xA108 | R, R/W | I²C1 Status Register | 0x00 |
| **IC1.SAR** | 0xA10C | R/W | I²C1 Slave Address Register | 0x00 |
| **IC1.CR** | 0xA114 | R/W | I²C1 Control Register | 0x00 |
| **IC1.SCLL** | 0xA118 | R/W | I²C1 SCL LOW duration Register | 0xFFFF |
| **IC1.SCLH** | 0xA11C | R/W | I²C1 SCL HIGH duration Register | 0xFFFF |
| **IC1.SDH** | 0xA120 | R/W | I²C1 SDA Hold Register | 0x007F |

## ICn.DR I²C Data Register

ICnDR is an 8-bit read/write register. It contains a byte of serial data to be transmitted or a byte which has just been received.

**IC0.DR=0x4000_A000, IC1.DR=0x4000_A100,**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ICDR | | | | | | | |
| 0xFF | | | | | | | |
| RW | | | | | | | |

| | | |
|---|---|---|
| 7<br>0 | ICDR | The most recently received data or data to be transmitted. |

## ICn.SR I²C Status Register

ICnSR is an 8-bit read/write register. It contains the status of I²C bus interface. Writing to the register clears the status bits except for IMASTER.

**IC0.SR=0x4000_A008, IC1.SR=0x4000_A008**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GCALL | TEND | STOP | SSEL | MLOST | BUSY | TMODE | RXACK |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | RW | RW | RW | RW | RW | RW | RW |

| | | | |
|---|---|---|---|
| 7 | GCALL | General call flag | |
| | | 0 | General call is not detected. |
| | | 1 | General call detected. |
| 6 | TEND | 1 Byte transmission complete flag | |
| | | 0 | The transmission is working or not completed. |
| | | 1 | The transmission is completed. |
| 5 | STOP | STOP flag | |
| | | 0 | STOP is not detected. |
| | | 1 | STOP is detected. |
| 4 | SSEL | Slave flag (Start condition received) | |
| | | 0 | Slave is not selected. |
| | | 1 | Slave is selected. |
| 3 | MLOST | Mastership lost flag | |
| | | 0 | Mastership is not lost. |
| | | 1 | Mastership is lost. |
| 2 | BUSY | BUSY flag | |
| | | 0 | I²C bus is in IDLE state. |
| | | 1 | I²C bus is busy. |
| 1 | TMODE | Transmitter/Receiver mode flag | |
| | | 0 | Receiver mode. |
| | | 1 | Transmitter mode. |
| 0 | RXACK | Rx ACK flag | |
| | | 0 | Rx ACK is not received. |
| | | 1 | Rx ACK is received. |

# ICn.SAR I²C Slave Address Register

ICnSAR is an 8-bit read/write register. It shows the address in slave mode.

IC0.SAR=0x4000_A00C, IC1.SAR=0x4000_A10C

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SVAD | | | | | | | GCEN |
| 0x00 | | | | | | | 0 |
| RW | | | | | | | RW |

| | | |
|---|---|---|
| 7 1 | SVAD | 7-bit Slave Address |
| 0 | GCEN | General call enable bit |
| | | 0 General call is disabled. |
| | | 1 General call is enabled. |

# ICn.CR I²C Control Register

ICnCR is an 8-bit read/write register. The register can be set to configure I²C operation mode and simultaneously allows for I²C transactions to be kicked off.

**IC0CR=0x4000_A014, IC1CR=0x4000_A114**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | INTDEL | | IIF | I2CEN | SOFTRST | INTEN | ACKEN | | STOP | START |
| 0 | 0 | 0 | 0 | 0 | 0 | 00 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | RW | | R | RW | RW | RW | RW | | RW | RW |

| | | |
|---|---|---|
| 9 | INTDEL | Interval delay value between address and data transfer (or DATA and DATA) |
| 8 | | 0 1 * ICnSCLL |
| | | 1 2 * ICnSCLL |
| | | 2 4 * ICnSCLL |
| | | 3 8 * ICnSCLL |
| 7 | IIF | Interrupt status bit |
| | | 0 Interrupt is inactive |
| | | 1 Interrupt is active |
| 6 | I2CEN | I2C enable bit |
| | | 0 I2C disabled |
| | | 1 I2C enabled |
| 5 | SOFTRST | Soft Reset enable bit. |
| | | 0 Soft Reset is disabled. |
| | | 1 Soft Reset is enabled.. |
| 4 | INTEN | Interrupt enabled bit. |
| | | 0 Interrupt is disabled. |
| | | 1 Interrupt is enabled. |
| 3 | ACKEN | ACK enable bit in Receiver mode. |
| | | 0 ACK is not sent after receiving data. |
| | | 1 ACK is sent after receiving data. |
| 1 | STOP | Stop enable bit. When this bit is set as "1" in transmitter mode, next transmission will be stopped even though ACK signal has been received. |
| | | 0 Stop is disabled. |
| | | 1 Stop is enabled. When this bit is set, transmission will be stopped. |
| 0 | START | Transmission start bit in master mode. |
| | | 0 Waits in slave mode. |
| | | 1 Starts transmission in master mode. |

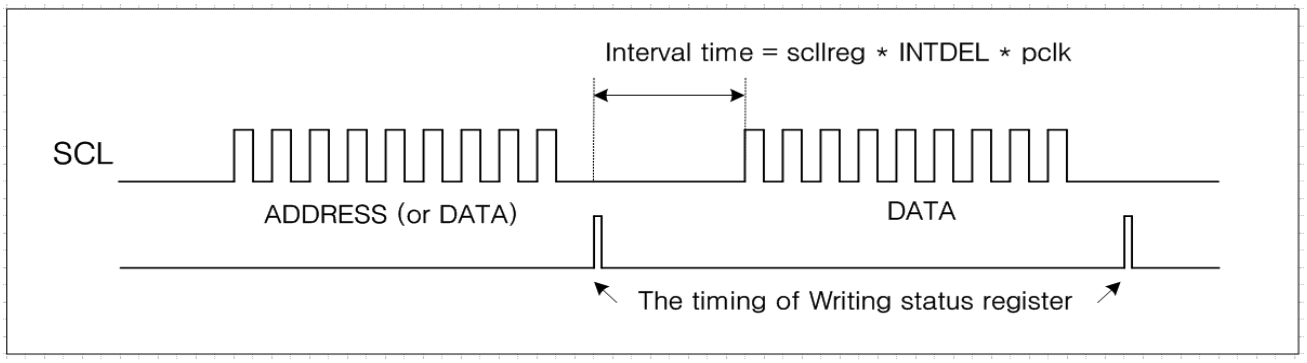



**Figure15.1. INTDEL in Master Mode**

# ICn.SCLL I²C SCL LOW Duration Register

ICnSCLL is a 16-bit read/write register. The SCL LOW time is set by writing this register in master mode.

IC0.SDLL=0x4000_A018, IC1.SDLL=0x4000_A118

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCLL | | | | | | | | | | | | | | | |
| 0xFFFF | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 | SCLL | SCL LOW duration value.<br>SCLL = ( PCLK * SCLL[15:0] ) + 2*PCLKs |
| 0 | | Default value is 0xFFFF. |

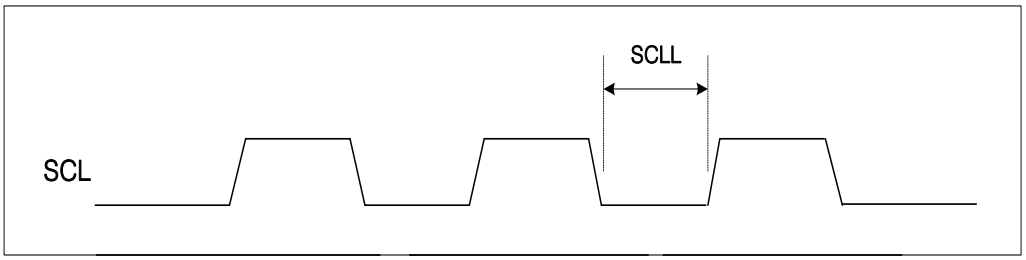



**Figure 15.2. SCL LOW Timing**

# ICn.SCLH I²C SCL HIGH Duration Register

ICnSCLH is a 16-bit read/write register. The SCL HIGH time is set by writing this register in master mode.

IC0.SDLH=0x4000_A01C, IC1.SDLH=0x4000_A11C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCLH | | | | | | | | | | | | | | | |
| 0xFFFF | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 | SCLH | SCL HIGH duration value.<br>SCLH = ( PCLK * SCLH[15:0] ) + 3 PCLKs |
| 0 | | Default value is 0xFFFF. |

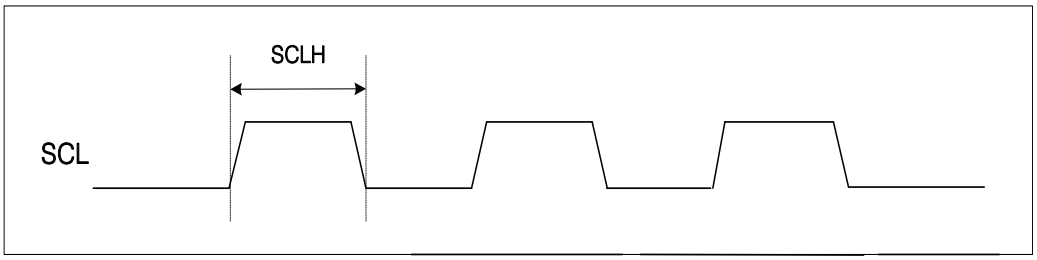



**Figure 15.3.SCL HIGH Timing**

## ICn.SDH SDA Hold Register

ICnSDH is a 15-bit read/write register. The SDA HOLD time is set by writing this register in master mode.

IC0.SDH=0x4000_A020, IC1.SDH=0x4000_A120

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SDH | | | | | | | | | | | | | | |
| 0x007F | | | | | | | | | | | | | | | |
| RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 14 | SDH | SDA HOLD time setting value.<br>SDH = ( PCLK * SDH[14:0] ) + 4 PCLKs |
| 0 | | Default value is 0x3FFF. |

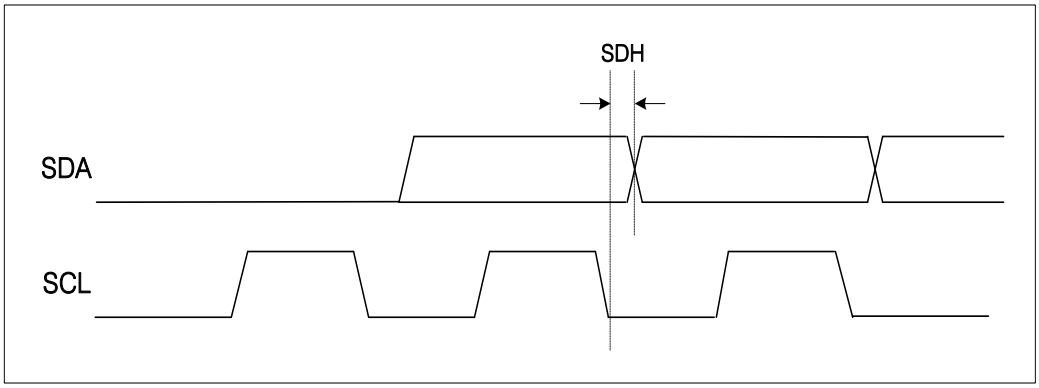



**Figure 15.4.SDA HOLD Timing**

# Functional Description

## I²C Bit Transfer

The data on the SDA line must be stable during the “H” period of the clock. The “H” or “L” state of the data line can only change when the clock signal on the SCL line is “L” as shown in Figure 15.5.

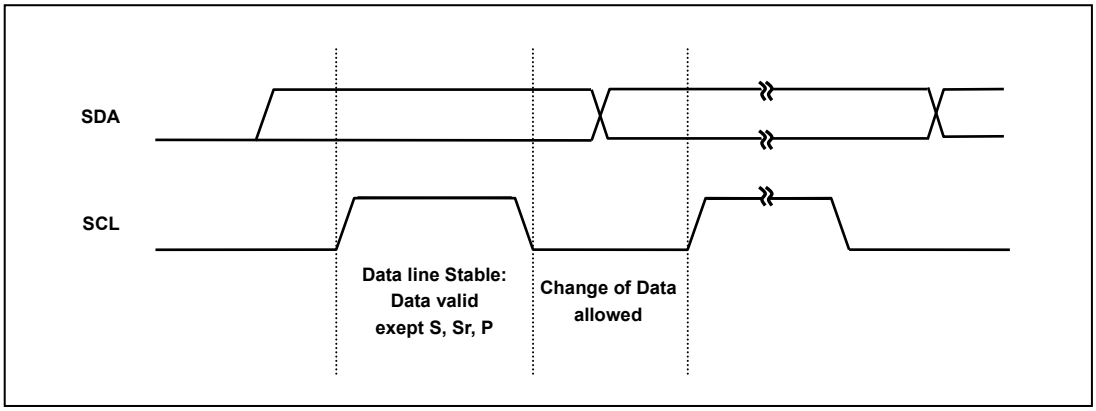



**Figure 15.5. I²C Bus Bit Transfer**

## START/Repeated START/STOP

Within the procedure of the $I^2C$-bus, unique situations arise which are defined as START(S) and STOP(P) conditions (see Figure 15.6.).

An “H” to “L” transition on the SDA line while SCL is “H” is one such unique case. This situation indicates a START condition.

An “L” to “H” transition on the SDA line while SCL is “H” defines a STOP condition.

START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition.

The bus is busy if a repeated START(Sr) is generated instead of a STOP condition. In this respect, the START(S) and repeated START(Sr) conditions are functionally identical. Therefore, for the remainder of this document, the S symbol will be used as a generic term to represent both the START and repeated START conditions, unless Sr is particularly relevant.

Detection of START and STOP conditions by devices connected to the bus is easy if they incorporate the necessary interfacing hardware. However, microcontrollers with no such interface have to sample the SDA line at least twice per clock period to sense the transition.

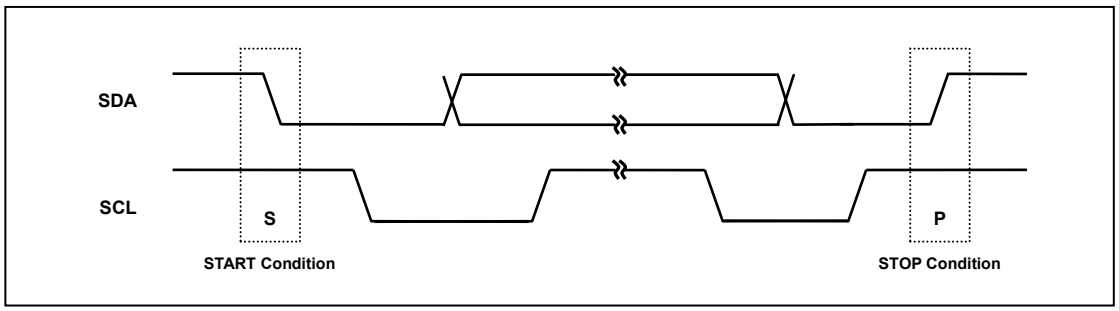



**Figure 15.6. START and STOP Condition**

## Data Transfer

Every byte put on the SDA line must be 8-bit long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first (see Figure 15.7). If a slave cannot receive or transmit another complete byte of data until it has performed some other function, for example servicing an internal interrupt, it can hold the clock line SCL “L” to force the master into a wait state. Data transfer then continues when the slave is ready for another byte of data and releases clock line SCL.

A message which starts with such an address can be terminated by generation of a STOP conditions, even during the transmission of a byte. In this case, no acknowledge is generated.

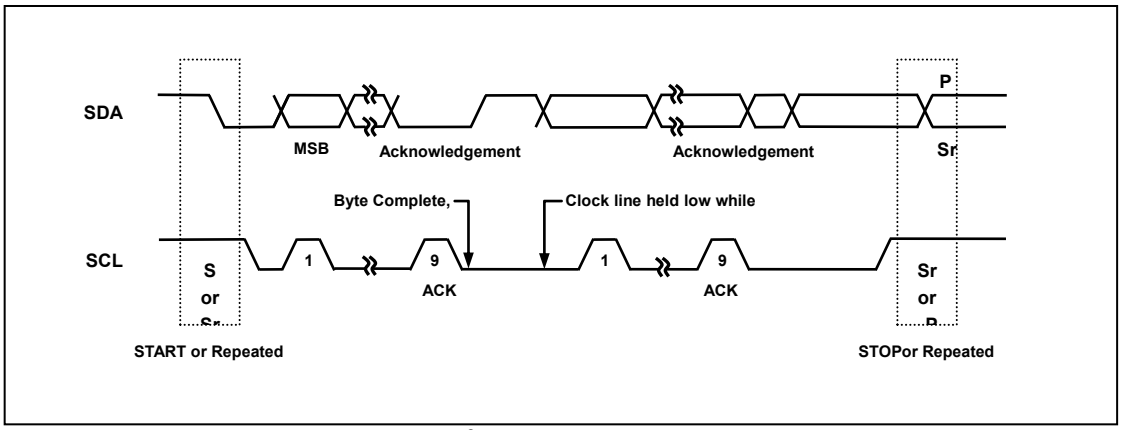



**Figure 15.7. I²C Bus Data Transfer**

## Acknowledge

Data transfer with acknowledge is obligatory. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse.

The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable “L” during the “H” period of this clock pulse (see Figure 15.8). Additionally, set-up and hold times must also be taken into account.

When a slave doesn’t acknowledge the slave address (for example, it’s unable to receive or transmit because it’s performing some real-time function), the data line must be left “H” by the slave. The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer.

If a slave-receiver acknowledges the slave address, but later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave generating the not-acknowledge on the first byte to follow. The slave leaves the data line “H” and the master generates a STOP or a repeated START condition.

If a master-receiver is involved in a transfer, it must signal the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave-transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

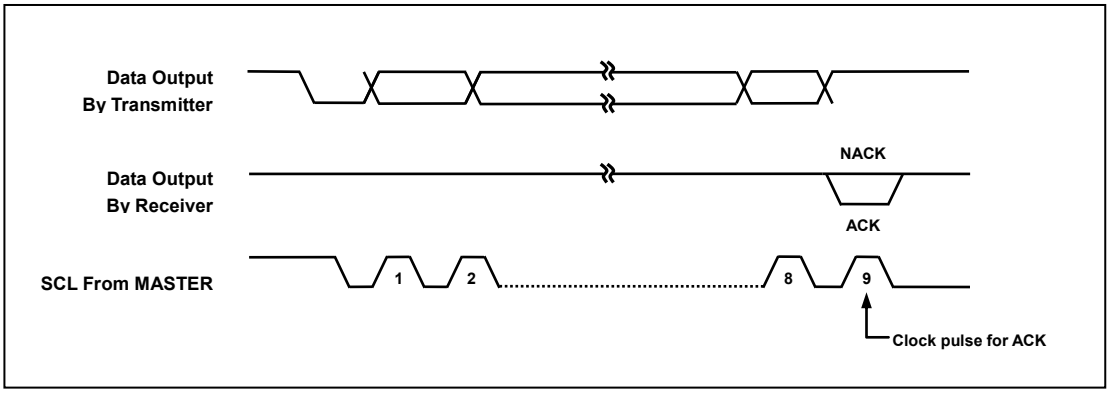



**Figure 15.8. I²C Bus Acknowledge**

## Synchronization

All masters generate their own clock on the SCL line to transfer messages on the $I^2C$-bus. Data is only valid during the “H” period of the clock. A defined clock is therefore needed for the bit-by-bit arbitration procedure to take place.

Clock synchronization is performed using the wired-AND connection of $I^2C$ interfaces to the SCL line. This means that an “H” to “L” transition on the SCL line will cause the devices to start counting off their “L” period and, after a device clock has gone “L”, it will hold the SCL line in that state until the clock “H” state is reached (see Figure 15.9). However, the “L” to “H” transition of this clock may not change the state of the SCL line if another clock is still within its “L” by the device with the longest “L” period. Devices with shorter “L” periods enter an “H” wait-state during this time.

When all devices have counted off their “L” period, the clock line will be released and go “H”. There will then be no difference between the device clocks and the state of the SCL line, and the devices will start counting their “H” periods. The first device to complete its “H” period will again pull the SCL line “L”.

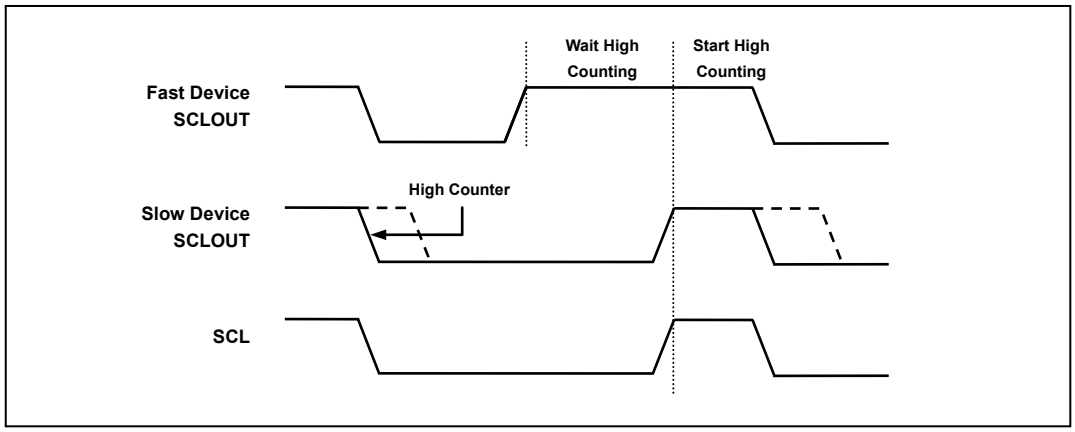



**Figure 15.9.Clock Synchronization During the Arbitration Procedure**

## Arbitration

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition within the minimum hold time of the START condition which results in a defined START condition to the bus.

Arbitration takes place on the SDA line, while the SCL line is at the “H” level, in such a way that the master which transmits “H” level, while another master is transmitting a “L” level will switch off its DATA output stage because the level on the bus doesn’t correspond to its own level.

Arbitration can continue for many bits. Its first stage is comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with comparison of the data-bits if they are master-transmitter, or acknowledge-bits if they are master-receiver. Because address and data information on the $I^2C$-bus is determined by the winning master, no information is lost during the arbitration process.

A master that loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration.

If a master also incorporates a slave function and it loses arbitration during the addressing stage, it is possible that the winning master is trying to address it. The losing master must therefore switch over immediately to its slave mode.

Figure 15.10 shows the arbitration procedure for two masters. There may be additional masters involved, depending on how many masters are connected to the bus. As soon as there is a difference between the internal data level of the master generating Device1 Dataout and the actual level on the SDA line, its data output is switched off, which means that an “H” output level is then connected to the bus. This will not affect the data transfer initiated by the winning master.

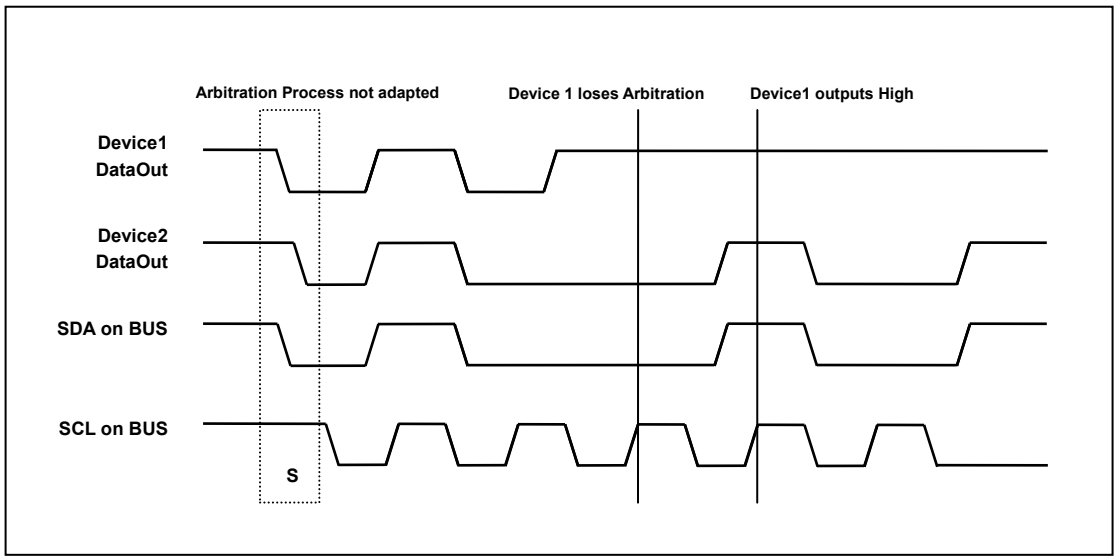



**Figure 15.10. Arbitration Procedure of Two Masters**

# I²C OPERATION

I²C supports interrupt operation. After the interrupt is serviced, the IIF(ICnSR[10]) flag is set. ICnSR shows I²C-bus status information and the SCL line stays “L” before the register is written as a certain value. The status register can be cleared by writing any value to the status register.

## Master Transmitter

The master transmitter shows the flow of the transmitter in Master mode as shown in Figure 15.11.

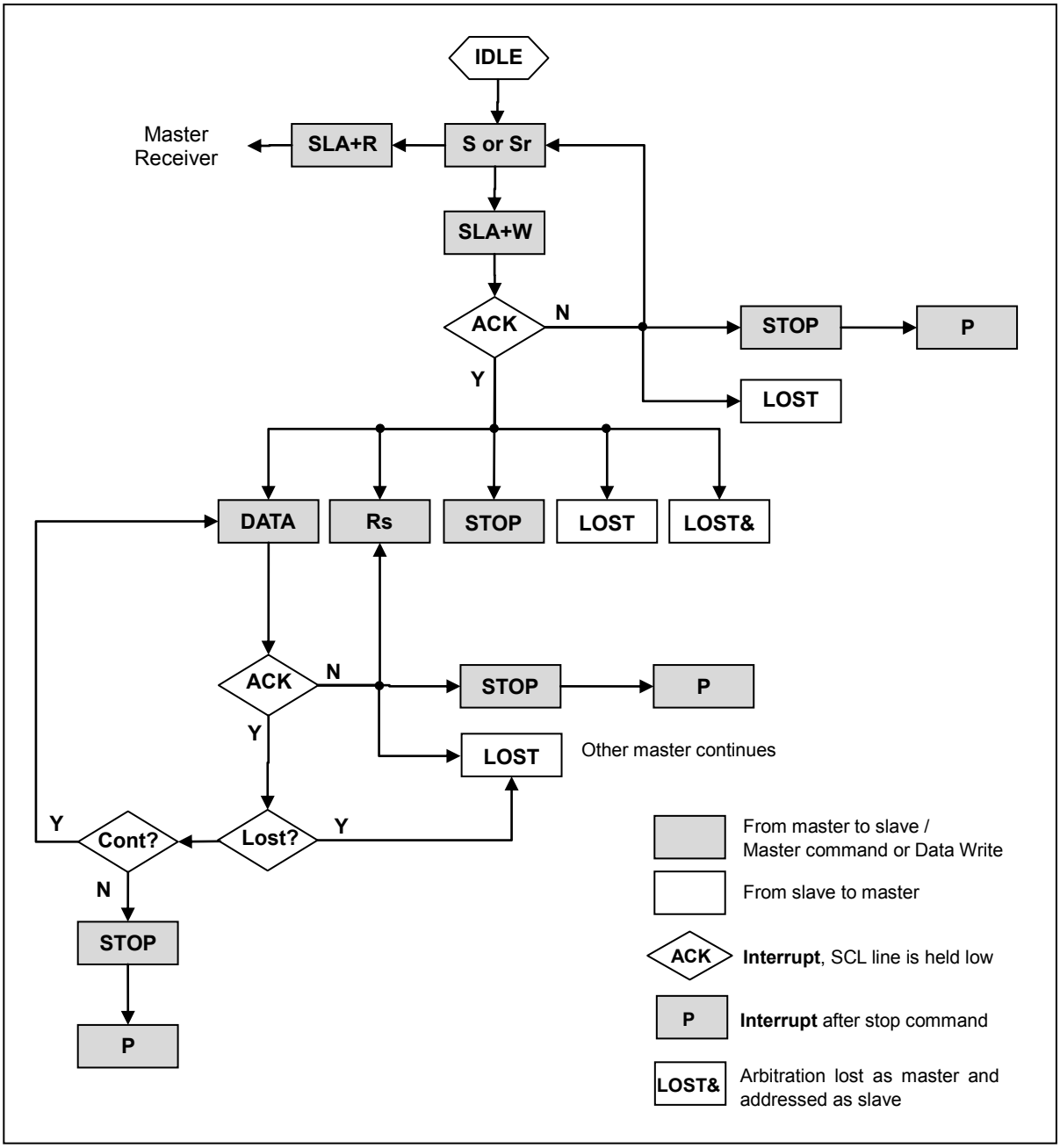



**Figure 15.11. Transmitter Flowchart in Master Mode**

## Master Receiver

The master receiver shows the flow of the receiver in Master mode as shown in Figure 15.12.

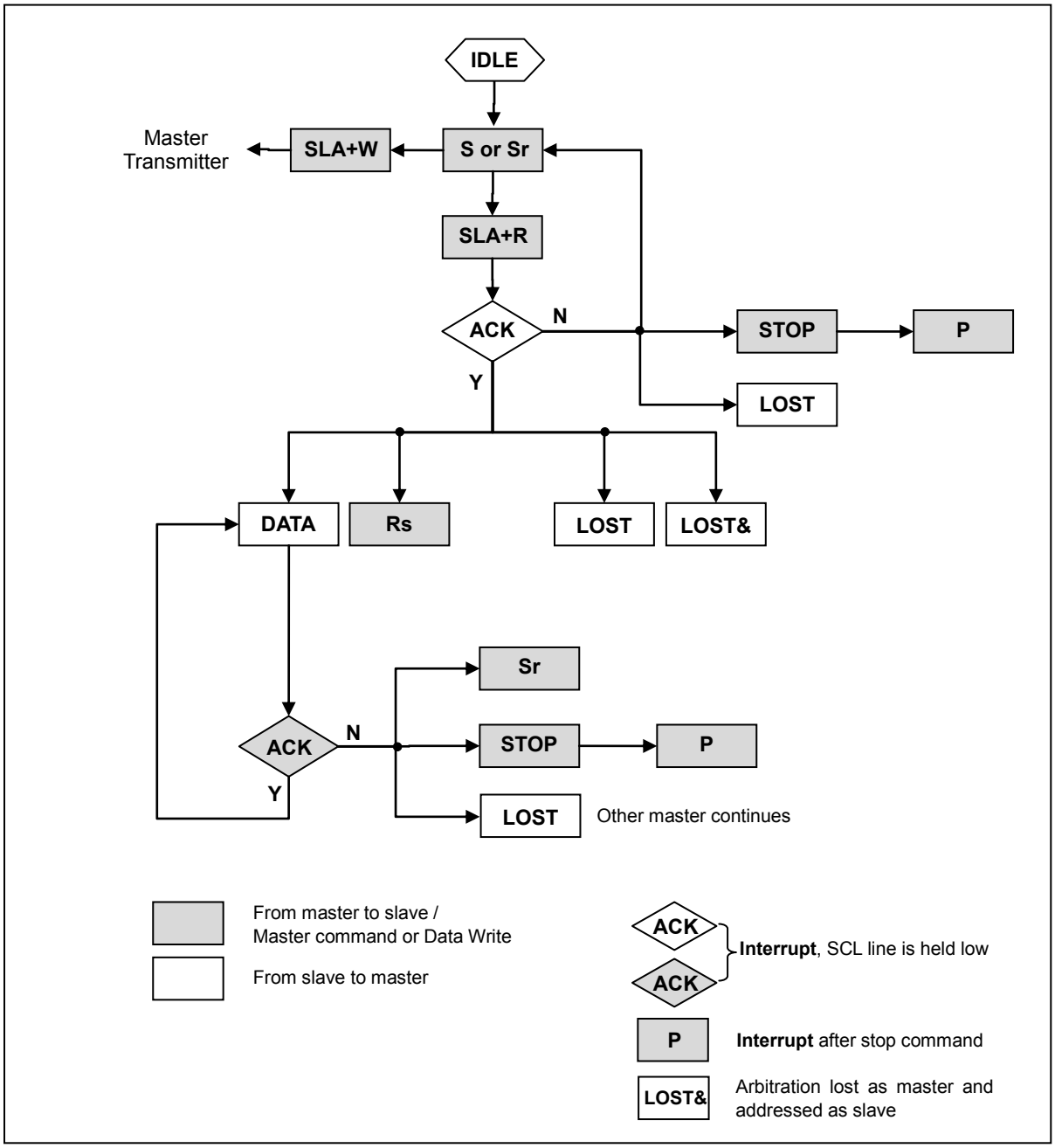



**Figure 15.12. Receiver Flowchart in Master Mode**

## Slave Transmitter

The slave transmitter shows the flow of the transmitter in Slave mode, as shown in Figure 15.13.

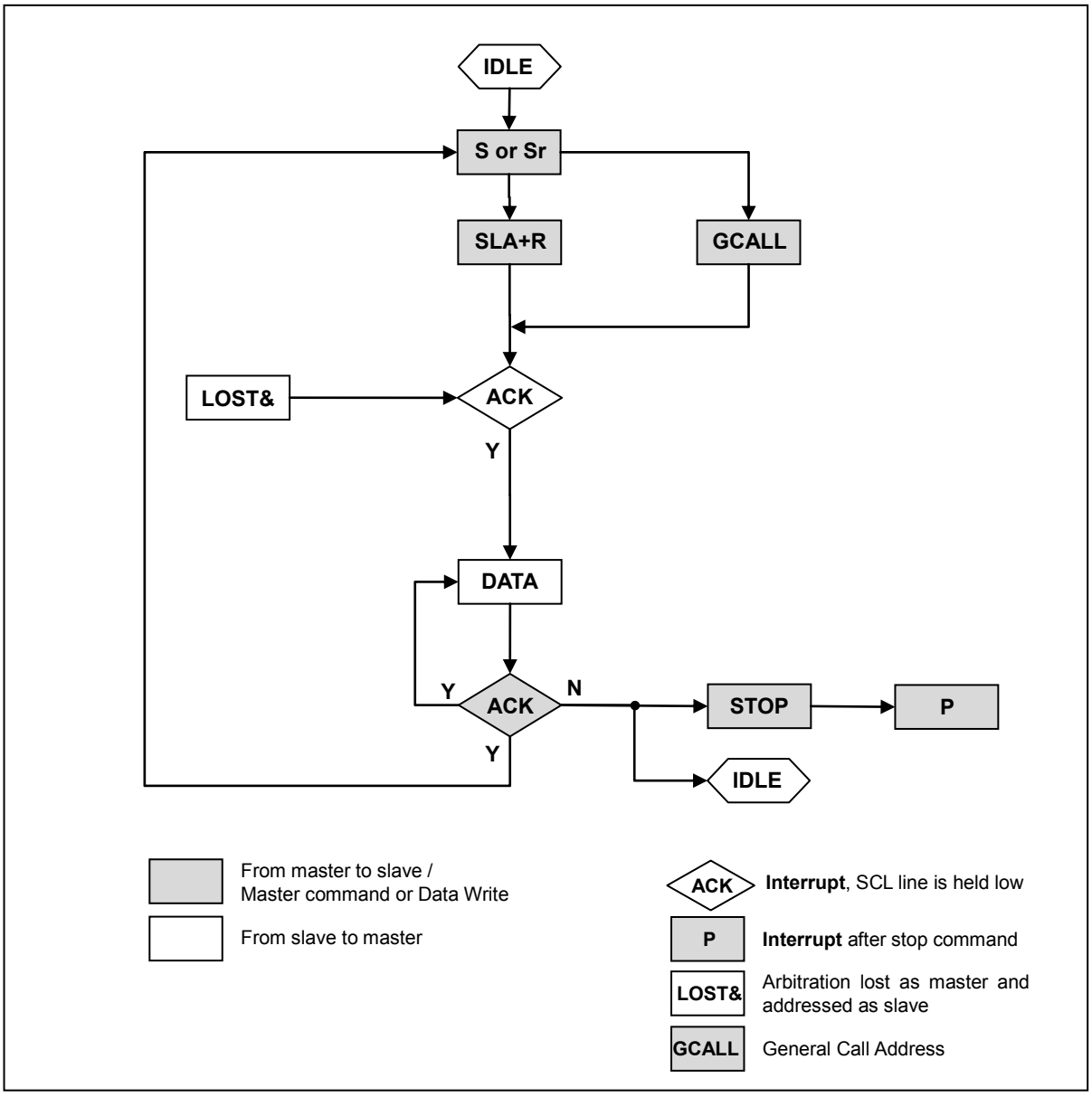



**Figure 15.13. Transmitter Flowchart in Slave Mode**

## Slave Receiver

The slave receiver shows the flow of the receiver in Slave mode, as shown in Figure 15.14.

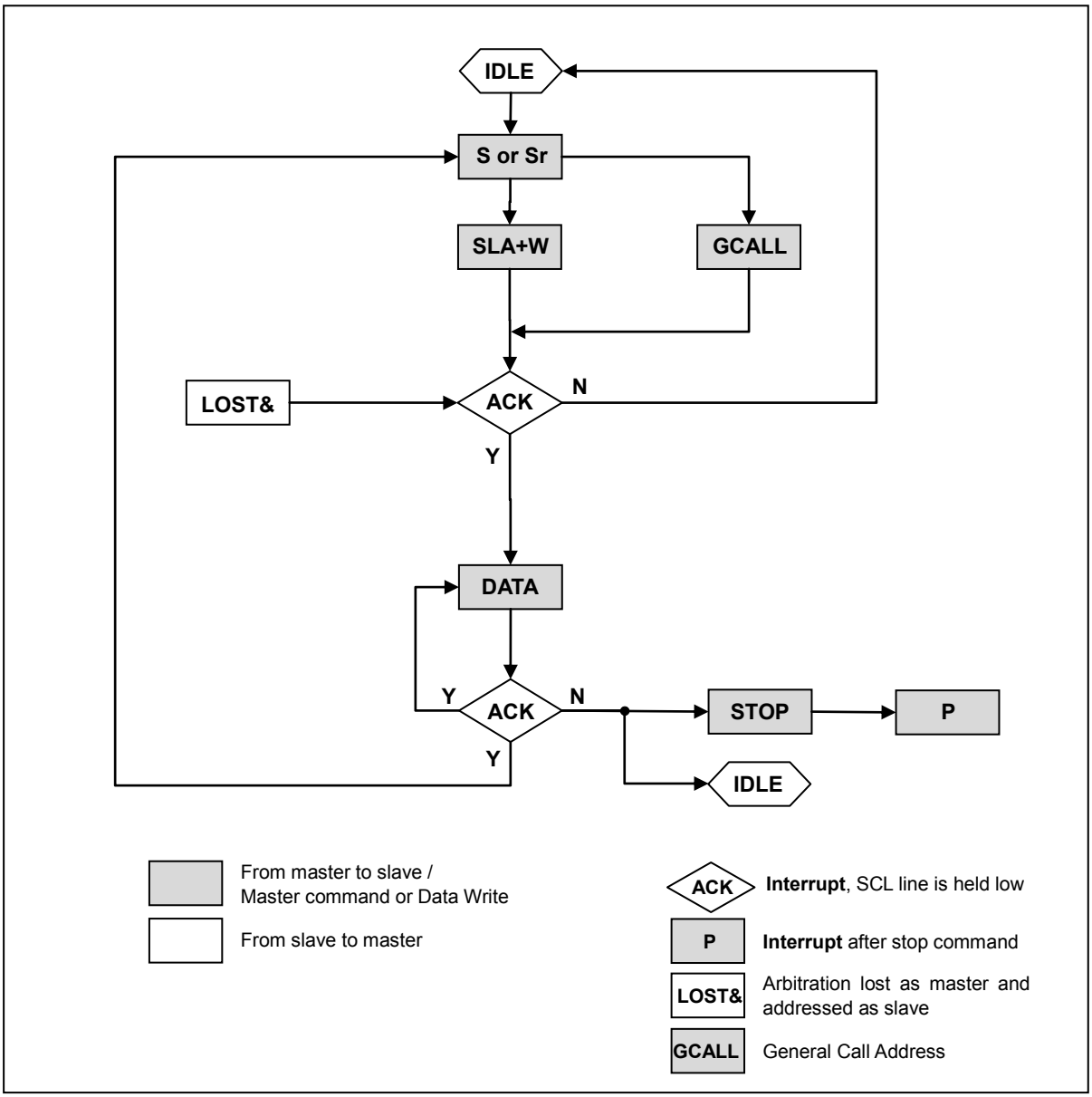



**Figure 15.14. Receiver Flowchart in Slave Mode**

# 16. Motor Pulse-Width-Modulator

## Introduction

The Motor Pulse Width Modulator (MPWM) is a programmable motor controller which is optimized for 3-phase inverter control applications. It can be used in several other applications that require timing, counting, and comparison.

MPWM includes three channels, each of which controls a pair of outputs that in turn can control an AC or DC motor through an inverter bridge. Features include:

- 6 channel outputs for motor control
- Dead- time zone support
- Protection event and over voltage event handling
- 6 trigger outputs for ADC
- Interval interrupt mode
- Up-down count mode

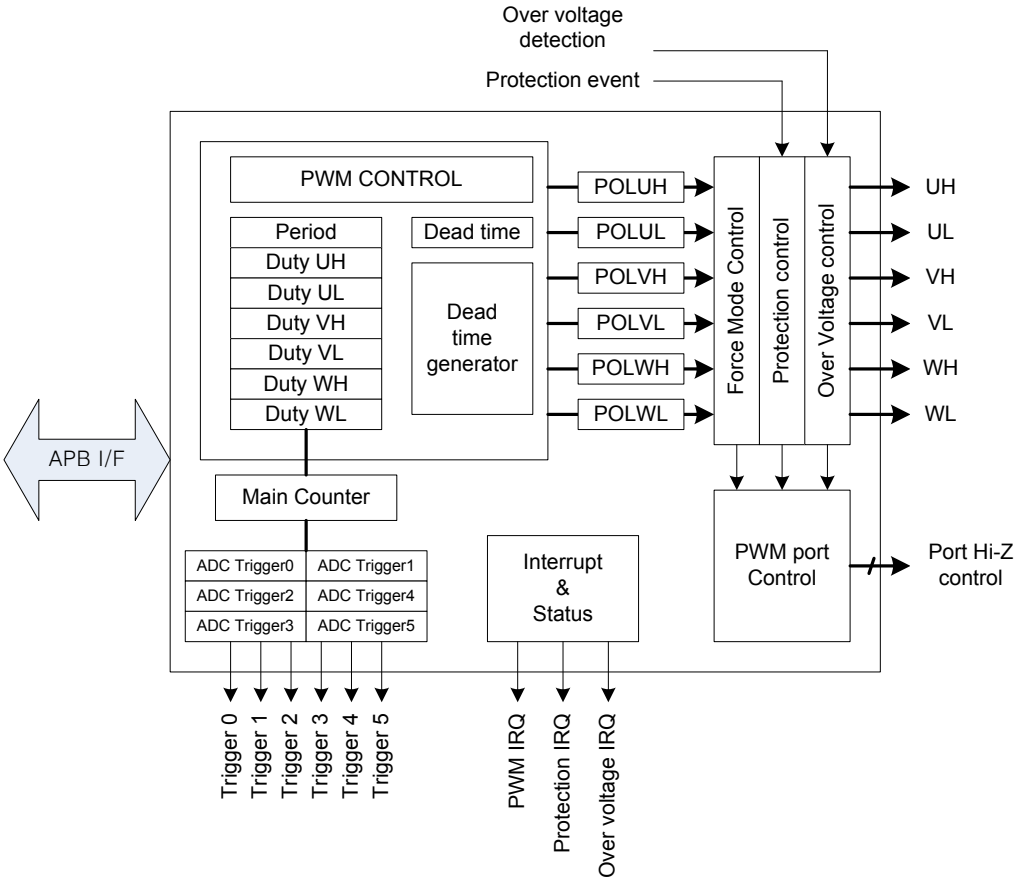



**Figure 16.1. Block Diagram**

# Pin Description

Table16.1. External Signals

| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| **MP0UH** | O | MPWM 0 Phase-U H-side output |
| **MP0UL** | O | MPWM 0 Phase-ULH-side output |
| **MP0VH** | O | MPWM 0 Phase-V H-side output |
| **MP0VL** | O | MPWM 0 Phase-V L-side output |
| **MP0WH** | O | MPWM 0 Phase-W L-side output |
| **MP0WL** | O | MPWM 0 Phase-W L-side output |
| **MP1UH** | O | MPWM 1 Phase-U H-side output |
| **MP1UL** | O | MPWM 1 Phase-U L-side output |
| **MP1VH** | O | MPWM 1 Phase-V H-side output |
| **MP1VL** | O | MPWM 1 Phase-V L-side output |
| **MP1WH** | O | MPWM 1 Phase-W L-side output |
| **MP1WL** | O | MPWM 1 Phase-W L-side output |
| **PRTIN0** | I | MPWM 0 Protection Input |
| **OVIN0** | I | MPWM 0 Over-voltage Input |
| **PRTIN1** | I | MPWM 1 Protection Input |
| **OVIN1** | I | MPWM 1 Over-voltage Input |

# Registers

The base address of MPWM is Table 16.2.

Table16.2. MPWM Base Address

| | BASE ADDRESS |
|---|---|
| **MPWM0** | 0x4000_4000 |
| **MPWM0** | 0x4000_5000 |

Table 16.3 shows the register memory map.

**Table16.3. MPWM Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **MPn.MR** | 0x0000 | R/W | PWM Mode register | 0x0000_0000 |
| **MPn.OLR** | 0x0004 | R/W | PWM Output Level register | 0x0000_0000 |
| **MPn.FOR** | 0x0008 | R/W | PWM Force Output register | 0x0000_0000 |
| **MPn.PRD** | 0x000C | R/W | PWM Period register | 0x0000_0002 |
| **MPn.DUH** | 0x0010 | R/W | PWM Duty UH register | 0x0000_0001 |
| **MPn.DVH** | 0x0014 | R/W | PWM Duty VH register | 0x0000_0001 |
| **MPn.DWH** | 0x0018 | R/W | PWM Duty WH register | 0x0000_0001 |
| **MPn.DUL** | 0x001C | R/W | PWM Duty UL register | 0x0000_0001 |
| **MPn.DVL** | 0x0020 | R/W | PWM Duty VL register | 0x0000_0001 |
| **MPn.DWL** | 0x0024 | R/W | PWM Duty WL register | 0x0000_0001 |
| **MPn.CR1** | 0x0028 | R/W | PWM Control register 1 | 0x0000_0000 |
| **MPn.CR2** | 0x002C | R/W | PWM Control register 2 | 0x0000_0000 |
| **MPn.SR** | 0x0030 | R | PWM Status register | 0x0000_0000 |
| **MPn.IER** | 0x0034 | R/W | PWM Interrupt Enable | 0x0000_0000 |
| **MPn.CNT** | 0x0038 | R | PWM counter register | 0x0000_0001 |
| **MPn.DTR** | 0x003C | R/W | PWM dead time control | 0x0000_0000 |
| **MPn.PCR0** | 0x0040 | R/W | PWM protection 0 control register | 0x0000_0000 |
| **MPn.PSR0** | 0x0044 | R/W | PWM protection 0 status register | 0x0000_0080 |
| **MPn.PCR1** | 0x0048 | R/W | PWM protection 1 control register | 0x0000_0000 |
| **MPn.PSR1** | 0x004C | R/W | PWM protection 1 status register | 0x0000_0000 |
| **-** | 0x0054 | - | Reserved | - |
| **MPn.ATR1** | 0x0058 | R/W | PWM ADC Trigger reg1 | 0x0000_0000 |
| **MPn.ATR2** | 0x005C | R/W | PWM ADC Trigger reg2 | 0x0000_0000 |
| **MPn.ATR3** | 0x0060 | R/W | PWM ADC Trigger reg3 | 0x0000_0000 |
| **MPn.ATR4** | 0x0064 | R/W | PWM ADC Trigger reg4 | 0x0000_0000 |
| **MPn.ATR5** | 0x0068 | R/W | PWM ADC Trigger reg5 | 0x0000_0000 |
| **MPn.ATR6** | 0x006C | R/W | PWM ADC Trigger reg6 | 0x0000_0000 |

# MPn.MR MPWM Mode Register

The PWM operation Mode register is a 16-bit register.

MP0.MR=0x4000_4000, MP1.MR=0x4000_5000

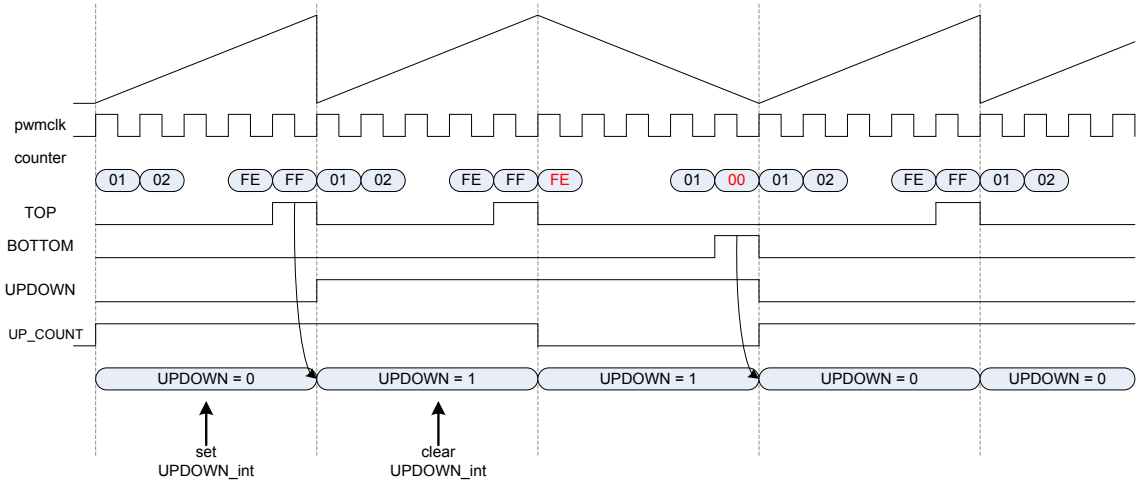
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOTORB | | | | | | | | UAO | | TUP | BUP | | MCHMOD | | UPDOWN |
| 0 | | | | | | | | | | 0 | 0 | | 00 | | 0 |
| R/W | | | | | | | | R/W | | R/W | R/W | | RW | | R/W |

| | | | |
|---|---|---|---|
| 15 | MOTORB | | Set PWM Mode |
| | | 0 | Motor Mode |
| | | 1 | Normal (PWM) mode |
| 7 | UAO | 0 | Update will be executed at designated timing. |
| | | 1 | Update all duty, period register at once.<br>When UPDATE set, Duty and Period registers are updated after two PWM clocks |
| 5 | TUP | 0 | Period, duty values are not updated at every period match. |
| | | 1 | Period, duty values are updated at every period match. |
| 4 | BUP | 0 | Period, duty values are not updated at every bottom match |
| | | 1 | Period, duty values are updated at every bottom match |
| 2<br>1 | MCHMOD | 00 | 2 channels symmetric mode<br>Duty H decides toggle high/low time of H-ch<br>Duty L decides toggle high/low time of L-ch |
| | | 01 | 1 channel asymmetric mode<br>Duty H decides toggle high time of H-ch<br>Duty L decides toggle low time of H-ch<br>L channel become the inversion of H channel |
| | | 10 | 1 channel symmetric mode<br>Duty H decides toggle high/low time of H-ch<br>L channel become the inversion of H channel |
| | | 11 | Not valid (same with 00) |
| 0 | UPDOWN | 0 | PWM Up count mode (only in Normal mode) |
| | | 1 | PWM Up/Down count mode |

After initial PWM period and duty setting is completed, the UAO bit should be set once for updating the setting value into the internal operating registers. This action will help to transfer the setting data from the user interface register to the internal operating register. The UAO bit should stay at the set state for at least 2-PWM clock periods. Otherwise, the update command can be missed and internal registers will retain the previous data.

MCHMOD in the MPn.MR field is only effective when MOTORB in MPn.MR is clear “0”. Otherwise, the MCHMOD field value will be ignored internally and will keep the “00” value.

UPDOWN in the MPn.MR field is only effective when MOTORB in MPn.MR is set to “1”. Otherwise, the UPDOWN field value will be ignored internally and will keep the “1” value. In the motor mode, the counter is always updown count operation.




# MPn.OLR MPWM Output Level Register

PWM Port Mode register is a 16-bit register.

**MP0.OLR=0x4000_4004, MP1.OLR=0x4000_5004**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | WHL | VHL | UHL | WLL | VLL | ULL |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| WHL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |
| VHL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |
| UHL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |
| WLL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |
| VLL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |
| ULL | 0 | Normal Output = L / Active Output = H |
| | 1 | Normal Output = H / Active Output = L |

**Table 16.1. MPWM Register Map**

| PWM Output | Level | NORMAL mode | | MOTOR mode |
|---|---|---|---|---|
| | | UP mode | UPDOWN mode | |
| WH | Default level | LOW | HIGH | LOW |
| | Active level | HIGH | LOW | HIGH |
| WL | Default level | LOW | LOW | HIGH |
| | Active level | HIGH | HIGH | LOW |
| VH | Default level | LOW | HIGH | LOW |
| | Active level | HIGH | LOW | HIGH |
| VL | Default level | LOW | LOW | HIGH |
| | Active level | HIGH | HIGH | LOW |
| UH | Default level | LOW | HIGH | LOW |
| | Active level | HIGH | LOW | HIGH |
| UL | Default level | LOW | LOW | HIGH |
| | Active level | HIGH | HIGH | LOW |

Figure 16.2 shows the polarity control block. This is an example of WH signal polarity control.

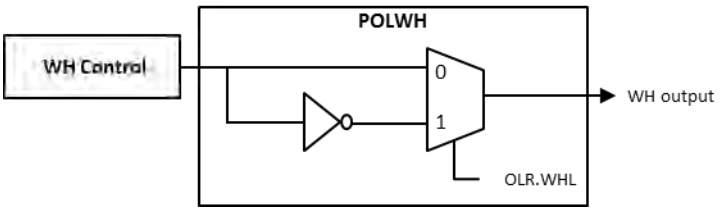



**Figure 16.2 Polarity Control Block**

# MPn.FOR MPWM Force Output Level Register

The PWM force output register is an 8-bit register. The PWM output level can be forced by an abnormal event from an external or user-intended condition. When the forced condition occurs, each PWM output level which is programmed in the FOLR register will be forced.

**MP0.FOR=0x4000_4008, MP1.FOR=0x4000_5008,**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | WHFL | VHFL | UHFL | WLFL | VLFL | ULFL |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 5 | **WHFL** | Select WH Output Force Level |
| | | 0 Output Force Level is 'L' |
| | | 1 Output Force Level is 'H' |
| 4 | **VHFL** | Select VH Output Force Level |
| | | 0 Output Force Level is 'L' |
| | | 1 Output Force Level is 'H' |
| 3 | **UHFL** | Select UH Output Force Level |
| | | 0 Output Force Level is 'L' |
| | | 1 Output Force Level is 'H' |
| 2 | **WLFL** | Select WL Output Force Level |

| | | |
|---|---|---|
| | | 0 Output Force Level is 'L'<br>1 Output Force Level is 'H' |
| 1 | VLFL | Select VL Output Force Level |
| | | 0 Output Force Level is 'L'<br>1 Output Force Level is 'H' |
| 0 | ULFL | Select UL Output Force Level |
| | | 0 Output Force Level is 'L'<br>1 Output Force Level is 'H' |

## MPn.CR1 MPWM Control Register 1

PWM Control Register 1 is a 16-bit register.

**MP0.CR1=0x4000_4028, MP1.CR1=0x4000_5028**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | IRQN | | | | | | | | | | PWMEN |
| | | | | | 000 | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | | RW | | | | | | | | | | RW |

| | | |
|---|---|---|
| 10<br>8 | IRQN | IRQ interval number<br>(Every 1~8th PRDIRQ,BOTIRQ,ATRn) |
| 0 | PWMEN | PWM enable<br>When this bit is set to 0, the PWM block stays in the reset state but the user interface can be accessed. To operate the PWM block, this bit should be set to 1. |

## MPn.CR2 MPWM Control Register 2

PWM Control Register 2 is an 8-bit register.

**MP0.CR2=0x4000_402C, MP1.CR2=0x4000_502C,**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HALT | | | | | | | PSTART |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | | | | | | | RW |

| | | |
|---|---|---|
| 7 | HALT | PWM HALT (PWM counter stop but not reset)<br>PWM outputs keep previous state |
| 0 | PSTART | 0 PWM counter stop and clear<br>1 PWM counter start (will be resynced @PWM clock twice)<br><br>PWMEN should be "1" to start PWM counter |

## MPn.PRD MPWM Period Register

PWM Period Register is a 16-bit register.

MP0.PRD=0x4000400C, MP1.PRD=0x40000500C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PERIOD | | | | | | | | | | | | | | | |
| 0x0002 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | PERIOD | 16-bit PWM period. It should be larger than 0x0010 (if Duty is 0x0000, PWM will not work) |

## MPn.DUH MPWM Duty UH Register

PWM U channel duty register is an 16-bit register.

MP0DUH=0x4000_4010, MP1DUH=0x4000_5010

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY UH | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | DUTY UH | 16-bit PWM Duty for UH output. It should be larger than 0x0001 (if Duty is 0x0000, PWM will not work) |

## MPn.DVH MPWM Duty VH Register

PWM V channel duty register is a 16-bit register.

MP0.DVH=0x4000_4014, MP1DVH=0x4000_5014

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY VH | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | DUTY VH | 16-bit PWM Duty for VH output. It should be larger than 0x0001 (if Duty is 0x0000, PWM will not work) |

## MPn.DWH MPWM Duty WH Register

PWM W channel duty register is a 16-bit register.

MP0.DWH=0x4000_4018, MP1.DWH=0x4000_5018

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY WH | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | DUTY WH | 16-bit PWM Duty for WH output.<br>It should be larger than 0x0001<br>(if Duty is 0x0000, PWM will not work) |

## MPn.DUL MPWM Duty UL Register

PWM U channel duty register is a 16-bit register.

MP0.DUL=0x4000_401C, MP1.DUL=0x4000_501C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY UL | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | DUTY UL | 16-bit PWM Duty for UL output.<br>It should be larger than 0x0001<br>(if Duty is 0x0000, PWM will not work) |

## MPn.DVL MPWM Duty VL Register

PWM V channel duty register is a 16-bit register.

MP0.DVL=0x4000_4020, MP1.DVL=0x4000_5020

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY VL | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | DUTY VL | 16-bit PWM Duty for VL output.<br>It should be larger than 0x0001<br>(if Duty is 0x0000, PWM will not work) |

## MPn.DWL MPWM Duty WL Register

PWM W channel duty register is a 16-bit register.

**MP0.DWL=0x4000_4024, MP1.DWL=0x4000_5024**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DUTY WL | | | | | | | | | | | | | | | |
| 0x0001 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15:0 | **DUTY WL** | 16-bit PWM Duty for WL output.<br>It should be larger than 0x0001<br>(if Duty is 0x0000, PWM will not work) |

## MPn.IER MPWM Interrupt Enable Register

PWM Interrupt Enable Register is an 8-bit register.

**MP0.IER=0x4000_4034, MP1.IER=0x4000_5034,**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PRDIEN | BOTIEN | WHIE | VHIE | UHIE | WLIE | VLIE | ULIE |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | RW | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 7 | **PRDIEN** | PWM Counter Period Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 6 | **BOTIEN** | PWM Counter Bottom Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 5 | **WHIE**<br>**ATR6IE** | WH Duty or ATR6 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 4 | **VHIE**<br>**ATR5IE** | VH Duty or ATR5 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 3 | **UHIE**<br>**ATR4IE** | UH Duty or ATR4 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 2 | **WLIE**<br>**ATR3IE** | WL Duty or ATR3 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 1 | **VLIE**<br>**ATR2IE** | VL Duty or ATR2 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |
| 0 | **ULIE**<br>**ATR1IE** | UL Duty or ATR1 Match Interrupt enable<br>0 interrupt disable<br>1 interrupt enable |

MPn.IER[5:0] control bits are shared by the duty match interrupt event and ADC trigger match interrupt event. When ADC trigger mode is disabled, the interrupt is generated by the duty match condition; else the interrupt is generated by the ADC trigger counter match condition. The ADC trigger mode is selected by the ATMOD bit field in the ATRm register.

# MPn.SR MPWM Status Register

PWM Status Register is a 16-bit register.

MP0.SR=0x4000_4030, MP1.SR=0x4000_5030

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DOWN | IRQCNT | | | | | | | PRDIF | BOTIF | DWHIF ATR6F | DVHIF ATR5F | DUHIF ATR4F | DWLIF ATR3F | DVLIF ATR2F | DULIF ATR1F |
| 0 | 000 | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | | | | | | | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | |
|---|---|---|---|
| 15 | **DOWN** | | PWM Count Up |
| | | 0 | PWM Count Down |
| 14<br>12 | **IRQCNT[2:0]** | | Interrupt count number of period match<br>(Interval PRDIRQ mode) |
| 7 | **PRDIF** | | PWM Period Interrupt flag(write “1” to clear flag) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 6 | **BOTIF** | | PWM Bottom Interrupt flag(write “1” to clear flag) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 5 | **DWHIF**<br>**ATR6F** | | PWM duty WH interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR6 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 4 | **DVHIF**<br>**ATR5F** | | PWM duty VH interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR5 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 3 | **DUHIF**<br>**ATR4F** | | PWM duty UH interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR4 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 2 | **DWLIF**<br>**ATR3F** | | PWM duty WL interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR3 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 1 | **DVLIF**<br>**ATR2F** | | PWM duty VL interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR2 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |
| 0 | **DULIF**<br>**ATR1F** | | PWM duty UL interrupt flag(write “1” to clear flag)<br>(Duty interrupt is enabled if ATR1 was disabled) |
| | | 0 | No interrupt occurred |
| | | 1 | Interrupt occurred |

## MPn.CNT MPWM Counter Register

PWM Counter Register is a 16-bit Read-Only register.

MP0.CNT=0x4000_4038, MP1.CNT=0x4000_5038

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MPnCNT | | | | | | | | | | | | | | | |
| 0x0000 | | | | | | | | | | | | | | | |
| R/W | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| | MPnCNT | PWM Counter Value |

## MPn.DTR MPWM Dead Time Register

PWM Dead Time Register is a 16-bit register.

MP0.DTR=0x4000_403C, MP1.DTR=0x4000_503C

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DTEN | PSHRT | | | | | | DTCLK | DT | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0x00 | | | | | | | |
| RW | | | | | | | RW | RW | | | | | | | |

| | | |
|---|---|---|
| | DTEN | Dead-time function enable |
| | | 0 Disable Dead-time function |
| | | 1 Enable Dead-time function |
| | PSHRT | Protect short condition |
| | | 0 Protection disable |
| | | 1 When H-side and L-side are active, disable both side |
| | DTCLK | Dead-time prescaler |
| | | 0 Dead time counter uses PWM CLK/4 |
| | | 1 Dead time counter uses PWM CLK/16 |
| | DT[7:0] | Dead Time value (Dead time setting makes output delay of 'low to high transition' in normal polarity)<br>0x01 ~0xFF : Dead time |

**Note:** Protect short condtion is for only internal PWM level and not for external PWM level. When the internal signal of H-side and L-side are the same high level, the protection short function works to force both H-side and L-side to low level.

# MPn.PCR0/1 MPWM Protection 0/1 Control Register

PWM Protection Control Register is a 16-bit register.

**MP0.PCR=0x4000_4040, MP1.PCR=0x4000_5040**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PROTEN | PROTPOL | | | | PROTD | | | PROTIE | | WHPROTM | VHPROTM | UHPROTM | WLPROTM | VPROTM | UPROTM |
| 0 | 0 | | | | 000 | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| RW | RW | | | | RW | | | RW | | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 15 | **PROT0EN** | Enable Protection Input 0 |
| 14 | **PROT0POL** | Select Protection Input Polarity<br>0: Low-Active<br>1: High-Active |
| 10<br>8 | **PROTD** | Protection Input debounce<br>0 – no debounce<br>1~7 – debounce by (MPWMCLK * PROTD[2:0]) |
| 7 | **PROTIE** | Protection Interrupt enable<br>0 Disable protection interrupt<br>1 Enable protection interrupt |
| 5 | **WHPROTM** | Activate W-phase H-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |
| 4 | **VHPROTM** | Activate V-phase H-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |
| 3 | **UHPROTM** | Activate U-phase H-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |
| 2 | **WLPROTM** | Activate W-phase L-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |
| 1 | **VLPROTM** | Activate V-phase L-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |
| 0 | **ULPROTM** | Activate U-phase L-side protection output<br>0 Disable Protection Output<br>1 Enable Protection Output with FOR value |

# MPn.PSR0/1 MPWM Protection 0/1 Status Register

PWM Protection Status Register is a 16-bit register. This register indicates which outputs are disabled and users can set the output masks manually. Without writing PROTKEY when writing any value, the written values are ignored.

**MP0.PSR=0x4000_4044, MP1.PSR=0x4000_5044**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PROTKEY | | | | | | | | PROTIF | | WHPROTF | VHPROTF | UHPROTF | WLPROTF | VPROTF | UPROTF |
| - | | | | | | | | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |
| WO | | | | | | | | RC | | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 15<br>8 | **PROTKEY** | Protection Clear Access Key<br>To clear flags, write 0xCA with protection flag<br>(PSR0 key is 0xCA and PSR1 key is 0xAC)<br>Writing without PROTKEY prohibited. |
| 7 | **PROTIF** | Protection Interrupt status<br>0 No Protection Interrupt<br>1 Protection Interrupt occurred |
| 5 | **WHPROT** | Activate W-phase H-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |
| 4 | **VHPROT** | Activate V-phase H-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |
| 3 | **UHPROT** | Activate U-phase H-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |
| 2 | **WLPROT** | Activate W-phase L-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |
| 1 | **VLPROT** | Activate V-phase L-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |
| 0 | **ULPROT** | Activate U-phase L-side protection flag<br>0 Protection not occurred.<br>1 Protection occurred or protection output enabled |

**Note:** MPn.PCR0 is related to the PRTINn pin and MPn.PCR1 is related to OVINn.

# MPn.ATRm MPWMn ADC Trigger Counter m Register

MPn.ATR1 MPWM ADC Trigger Counter 1 Register
MPn.ATR2 MPWM ADC Trigger Counter 2 Register
MPn.ATR3 MPWM ADC Trigger Counter 3 Register
MPn.ATR4 MPWM ADC Trigger Counter 4 Register
MPn.ATR5 MPWM ADC Trigger Counter 5 Register
MPn.ATR6 MPWM ADC Trigger Counter 6 Register

The PWM ADC Trigger Counter Register is a 32-bit register.

**MP0.ATR1=0x4000_4058, MP1.ATR1=0x4000_5058**
**MP0.ATR2=0x4000_405C, MP1.ATR2=0x4000_505C**
**MP0.ATR3=0x4000_4060, MP1.ATR3=0x4000_5060**
**MP0.ATR4=0x4000_4064, MP1.ATR4=0x4000_5064**
**MP0.ATR5=0x4000_4068, MP1.ATR5=0x4000_5068**
**MP0.ATR6=0x4000_406C, MP1.ATR6=0x4000_506C**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | ATUDT | | ATMOD | | ATCNT | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0x0000 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | RW | | RW | | RW | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 19 | **ATUDT** | Trigger register update mode |
| | | 0 ADC trigger value applied at period match event (at the same time with period and duty registers update) |
| | | 1 Trigger register update mode When this bit set, written Trigger register values are sent to trigger compare block after two PWM clocks (through synchronization logic) |
| 17 | **ATMOD** | ADC trigger Mode register |
| 16 | | 00 ADC trigger Disable |
| | | 01 Trigger out when up count match |
| | | 10 Trigger out when down count match |
| | | 00 Trigger out when up-down count match |
| 15 | **ATCNT** | ADC Trigger counter |
| 0 | | (it should be less than PWM period) |

# Functional Description

The PWMx module allows users to configure the PWM for different types of modulation schemes described in the previous section. The PER2 and PCER2 registers must be configured to enable the PWMx peripheral and the PWMx peripheral clock.

Setting or resetting the MOTOR bit in the MPnMR register allows users to operate the motor in Independent or Complementary PWM modes. For more information about operating modes, refer to the diagrams in the following section.

Figure 16.3 shows the diagram for generating a PWM output signal.

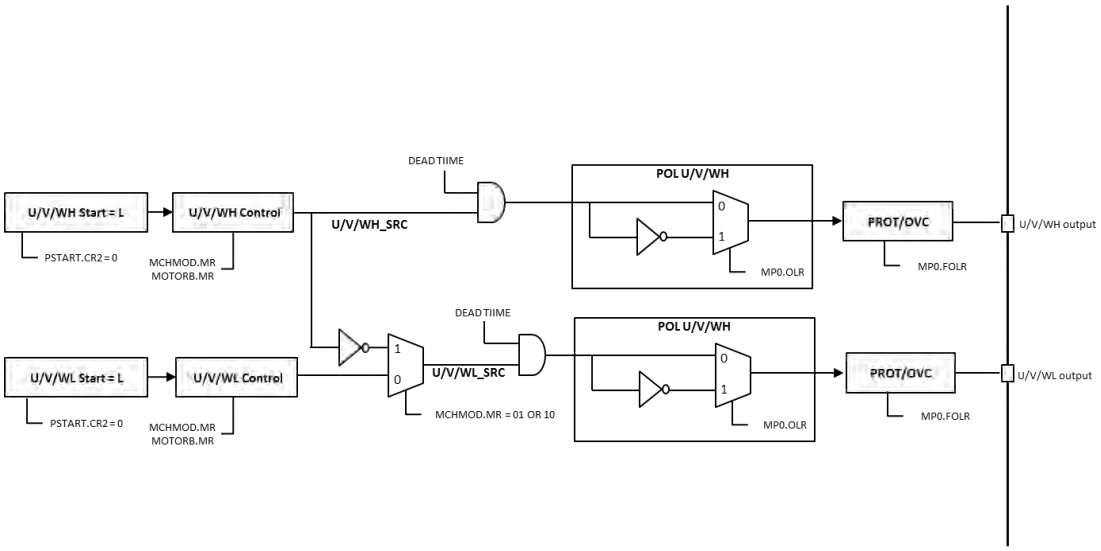



**Figure 16.3 PWM Output Generation Chain**

## Normal PWM Up Count Mode Timing

In normal PWM mode, each channel runs independently. 6 PWM outputs can be generated. An example waveform is shown in Figure 16.4. Before PSTART is activated, the PWM output stay at the default value L. When PSTART is enabled, the period counter starts up count until the MP0.PRD count value. In the first period, the MPWM does not generate a PWM pulse.

The PWM pulse is generated from the second period. The active level is derived at the start of the counter value during duty value time.

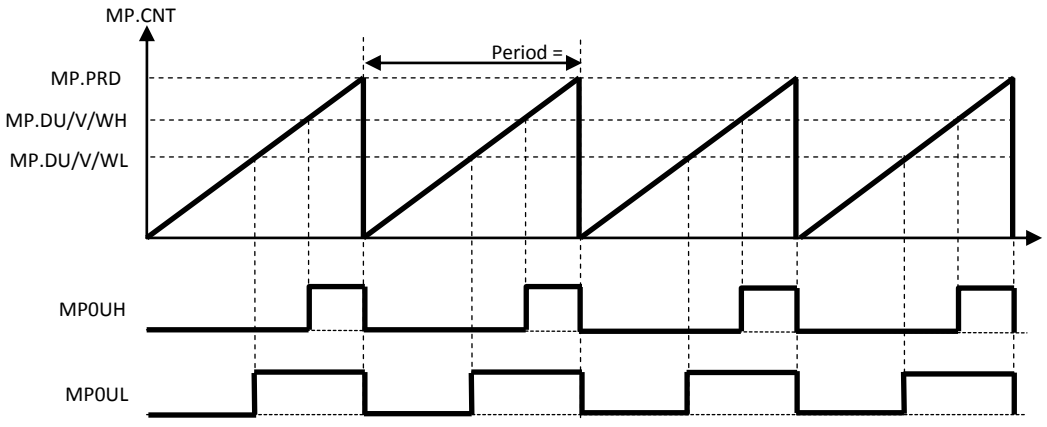



**Figure 16.4. Up Count Mode Waveform (MOTORB=1, UPDOWN=0)**

## Normal PWM Up/Down Count Mode Timing

The basic operation of the Up/Down count mode is the same as the Up count mode except that one up/down period is twice as long as an UP count mode period. The default active level is opposite in a pair PWM output. This output polarity can be controlled by the MP0.OLR register.

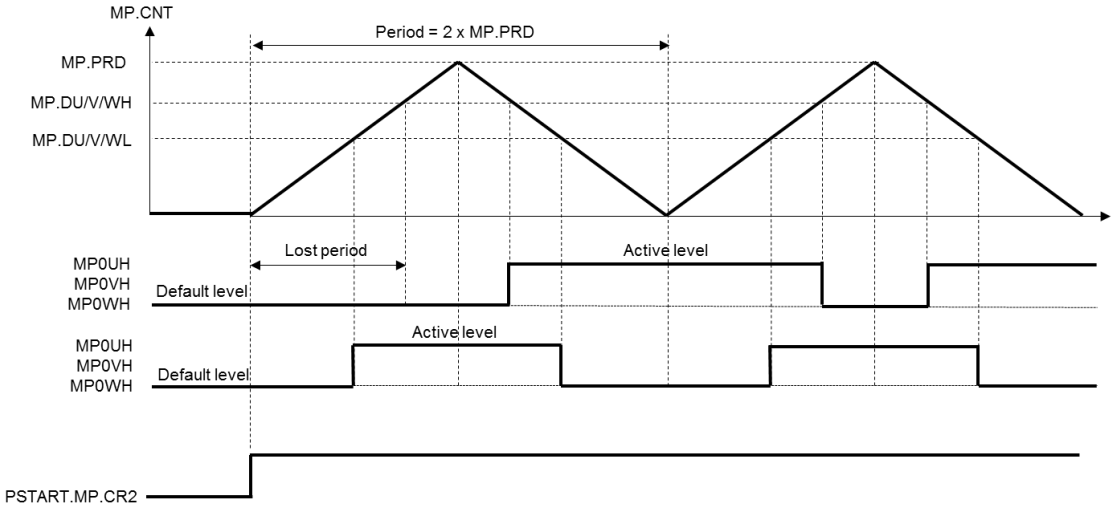



**Figure 16.5. Up/Down Count Mode Waveform (MOTORB=0, MCHMOD=0, UPDOWN=1)**

## Motor PWM 2-Channel Symmetric Mode Timing

The motor PWM operation has three types of operating mode:

- 2-Channel symmetric mode
- 1-Channel symmetric mode
- 1-Channel asymmetric mode

Figure 16.6 shows a 2-channel symmetric mode waveform.

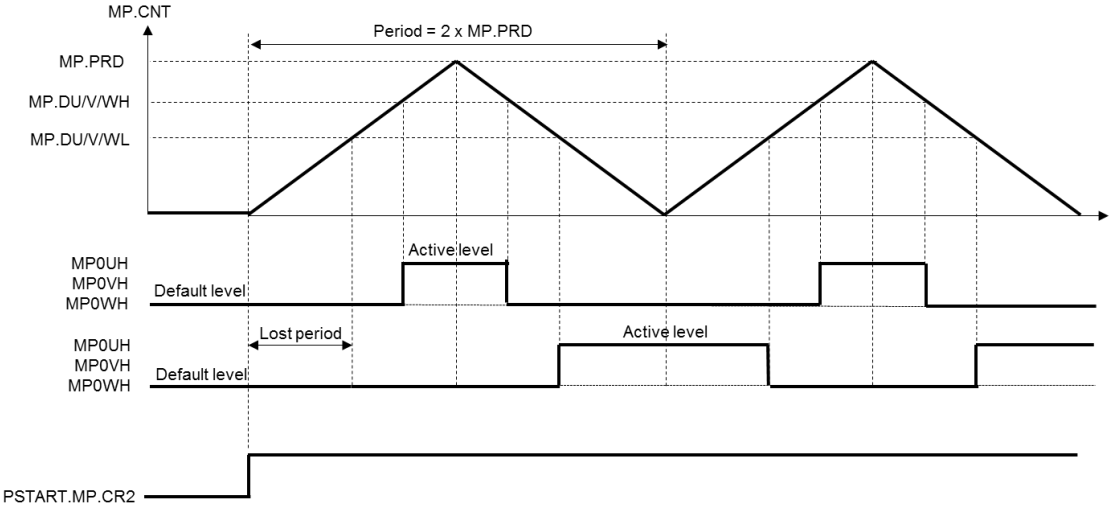



**Figure 16.6. 2-Channel Symmetric Mode Wave Form (MOTORB=0,MCHMOD=00)**

The default start level of both H-side and L-side is low. For the H-side, the PWM ouput level is changed to active level when the duty level is matched in the up count period and is returned to the default level when the duty level is matched in the down count period.

The symmetrical feature appears in each channel which is controlled by the corresponding DUTY register value.

## Motor PWM 1-Channel Asymmetric Mode Timing

The 1 channel asymmetric mode generates asymmetric duration pulses which are defined by the H-side and L-side DUTY register. Therefore, the L-side signal is always the negative signal of H-side. During the up count period, the H-side DUTY register matching condition generates the active level pulse and during the down count period, the L-side DUTY register matiching condition generates the default level pulse.

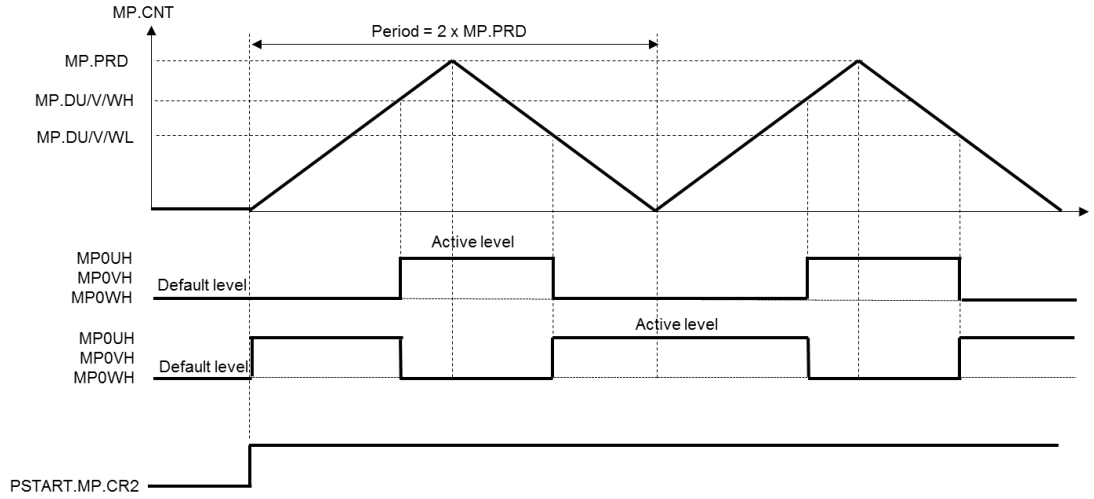



**Figure 16.7. 1-Channel Asymmetric Mode Waveform (MOTORB=0,MCHMOD=01)**

The default start level of both H-side and L-side is low. For the H-side, PWM ouput level is changed to active level when the H-side duty level is matched in up count period and is returned to default level when the L-side duty level is matched in down count period.

When the PSTART is set, the L-side PWM output is changed to the active level then the L-side PWM output is inverse output of H-side output.

## Motor PWM 1-Channel Symmetric Mode Timing

The 1-channel symmetric mode generates a symmetric duration pulse which is defined by the H-side DUTY register. Therefore, the L-side signal is always the negative of the H-side signal. During up count period, the H-side DUTY register matching condition creates the active level pulse and during down count period, the H-side DUTY register matiching condition also generates the default level pulse.

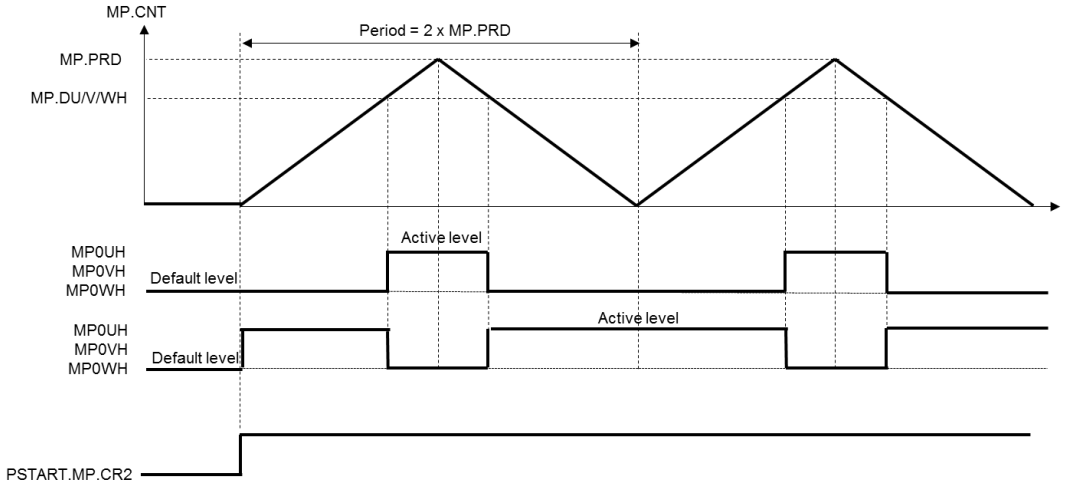


**Figure 16.8. 1-Channel Symmetric Mode Waveform (MOTORB=0,MCHMOD=10)**

The default start level of both H-side and L-side is low. For the H-side, PWM ouput level is changed to active level when the H-side duty level is matched in up count period and is returned to the default level when the H-side duty level is matched again in down count period.

When PSTART is set, the L-side PWM output is changed to the active level, then the L-side PWM output is inverse output of H-side output.

## PWM Dead-time Operation

To prevent external short conditions, the MPWM provides dead time functionality. This function is only available for motor PWM mode. When one of H-sdie or L-side output changes to active level, the amount of dead time is inserted if the DTEN.MP.DTR bit is enabled.

The duration of dead time is decided by the value in the DT.MP.DTR[7:0] field.

When DTCLK = 0, the dead time duration = DT[7:0] * (PWM clock period * 4)
When DTCLK = 1, the dead time duration = DT[7:0] * (PWM clock period * 16)

When the PWM counter reaches the duty value, the PWM output is masked and the dead time counter starts to run. When the dead time counter reaches the value in the DT[7:0] register, the output mask is disabled.

Figure 16.9 is an example of dead time operation in 1-channel symmetric mode.

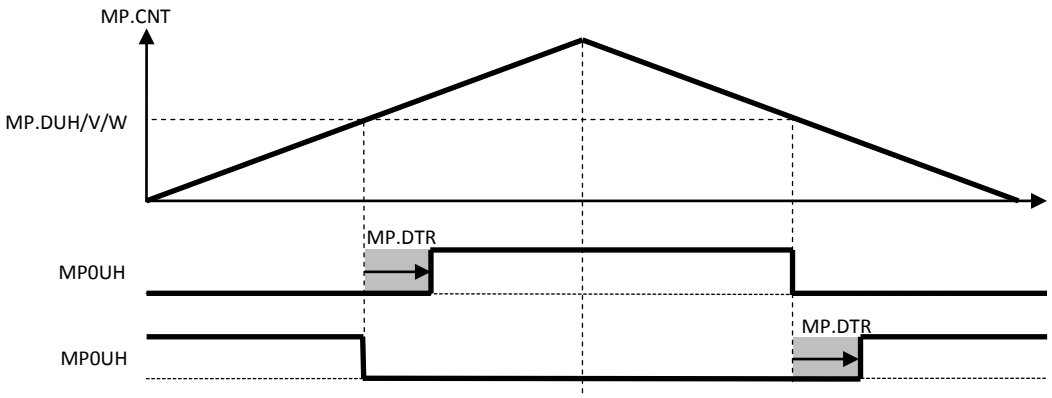



**Figure 16.9. PWM Dead-time Operation Timing Diagram (Symmetric Mode)**

Figure 16.10 is an example of dead time operation in 1-channel asymmetric mode

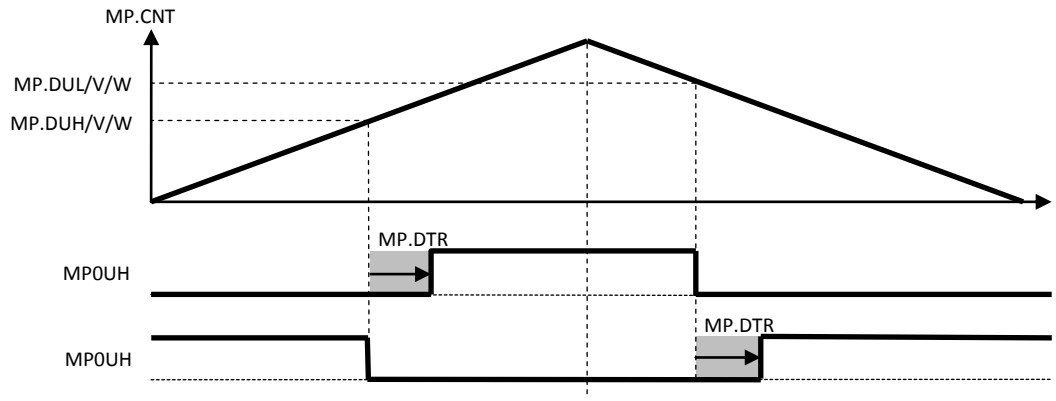



**Figure 16.10. PWM Dead-time Operation Timing Diagram (Asymmetric Mode)**

The dead time function is not available for 2-channel symmetric mode. Therefore, the dead condition is generated by each channel's duty control.

## MPWM Dead-time Timing Examples

The following images show how the dead-time operates. In normal situations, the dead time masking is activated at duty match time and the dead time counter runs. When the dead time counter reaches the dead time value, the mask is disabled.

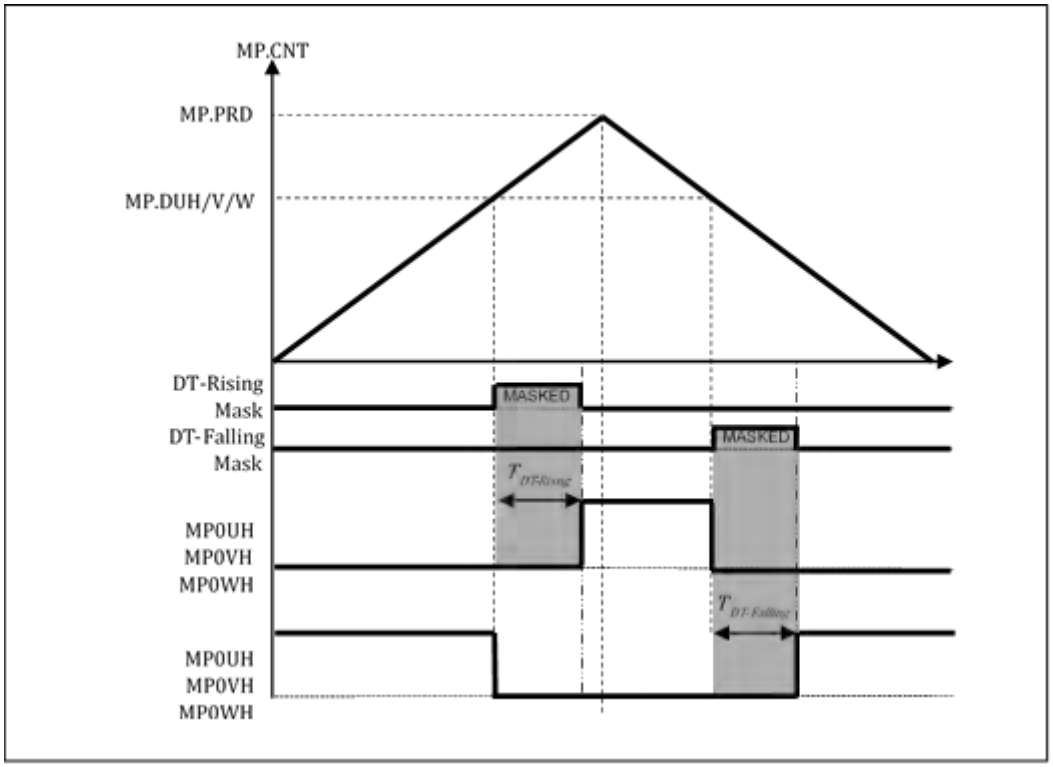



**Figure 16.11. Normal Dead-time Operation ($T_{DUTY}$>$T_{DT}$)**

The following figures show the dead time configuration in special situations.

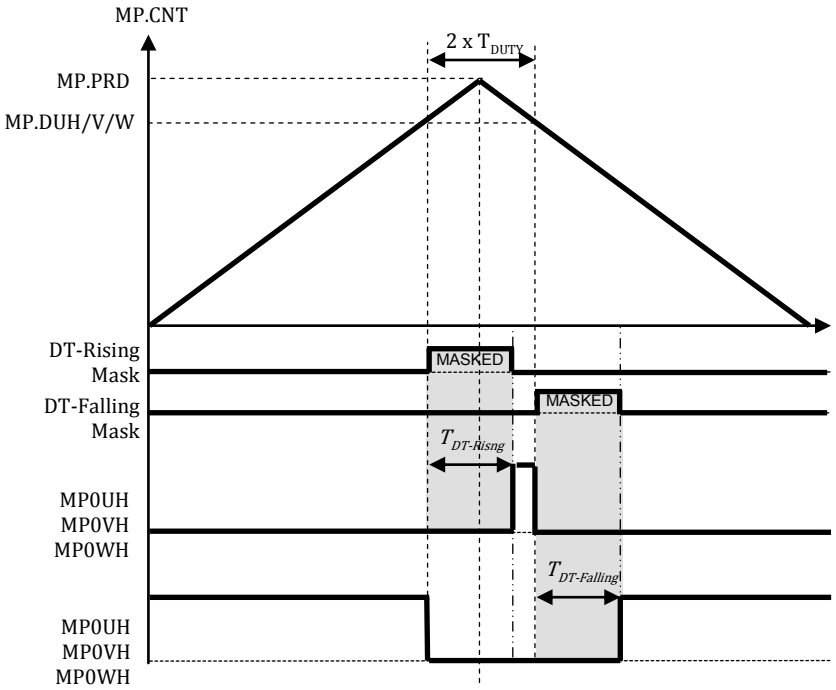



**Figure 16.12. Minimum H-side Pulse Timing ($T_{DUTY}$<$T_{DT}$<2x$T_{DUTY}$)**

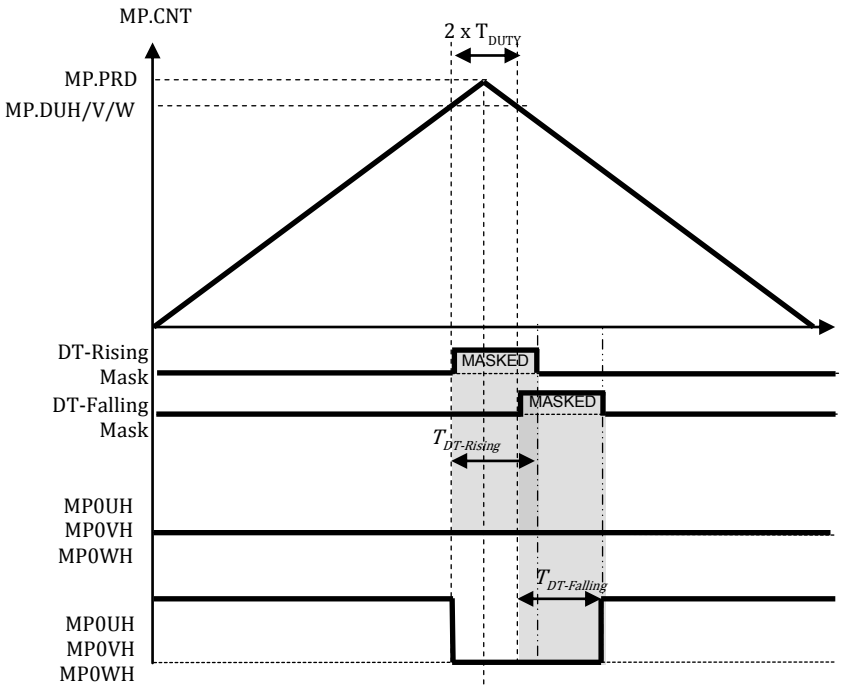



**Figure 16.13. Zero H-side Pulse Timing ($T_{DT}$>2x$T_{DUTY}$)**

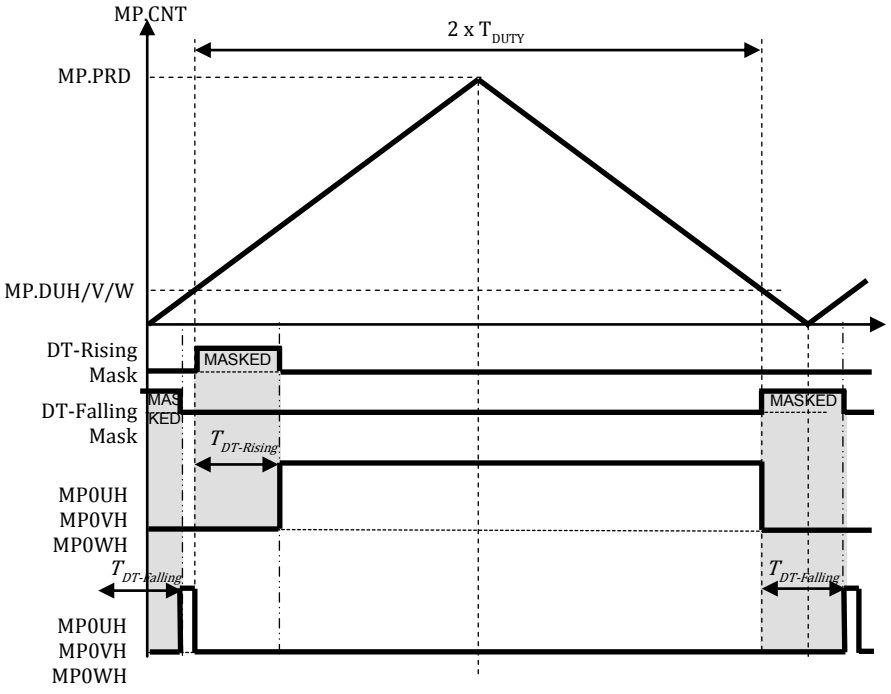



**Figure 16.14. Minimum L-side Pulse Timing ($T_{DT}$<Period-$T_{DUTY}$)**

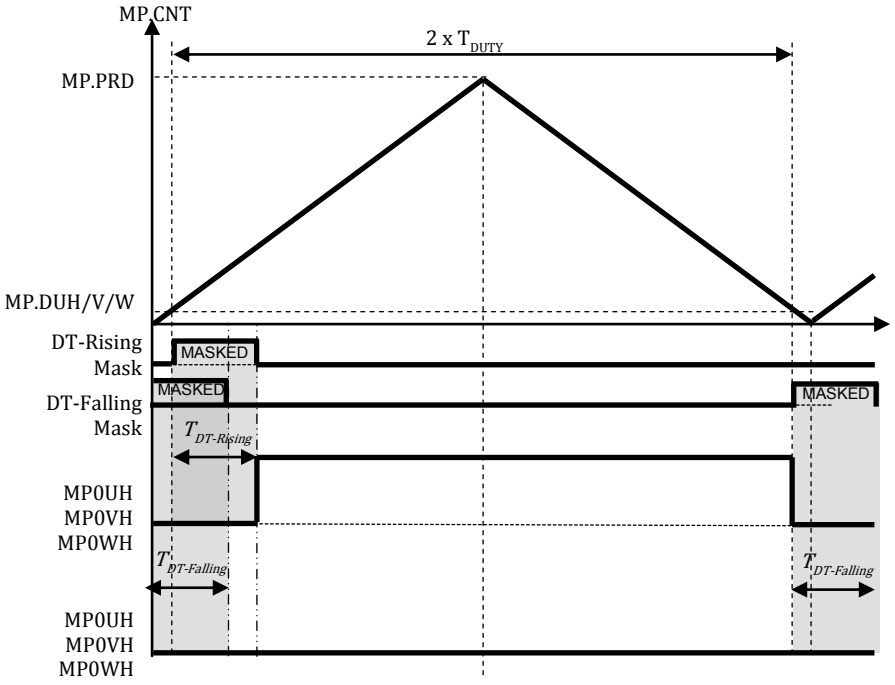



**Figure 16.15. Zero L-side Pulse Timing ($T_{DT}$>Period-$T_{DUTY}$)**

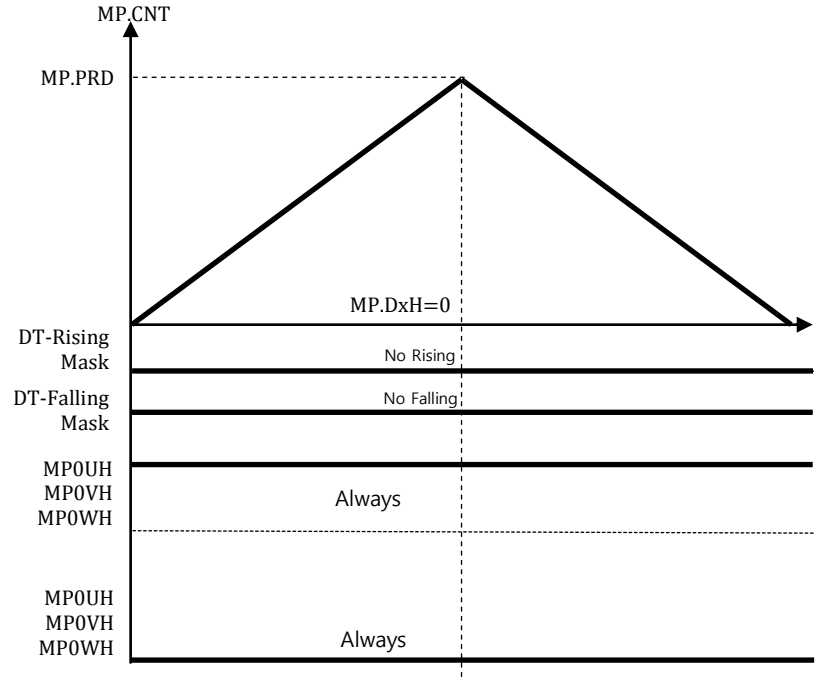



**Figure 16.16. H-side Always On ($T_{DUTY}$=Period: Dead-time Disabled)**

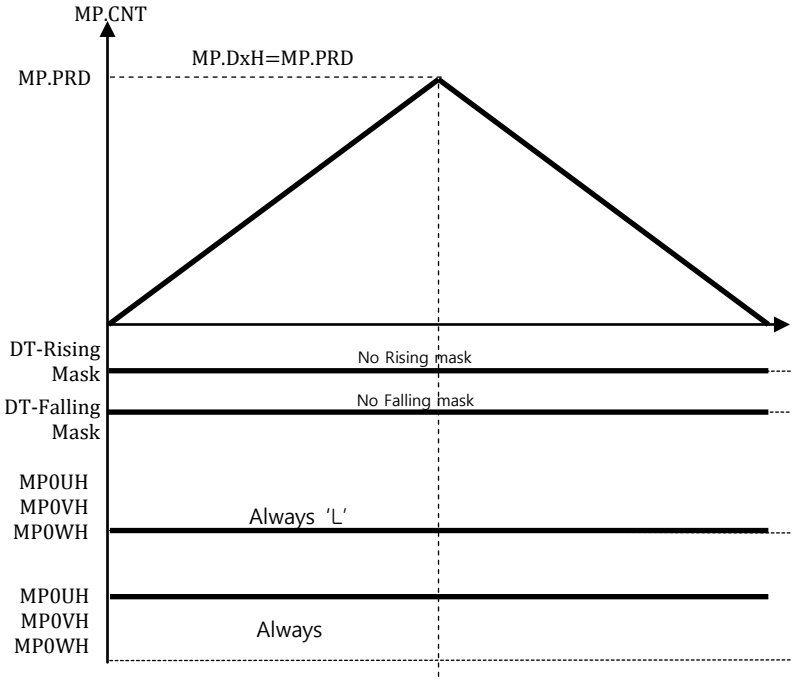



**Figure 16.17. L-side Always On ($T_{DUTY}$='0': Dead-time Disabled)**

## Symmetrical Mode vs Asymmetrical Mode

In symmetrical mode, the wave form is symmetrical on both sides of the mid-point of the period. The duty comparison is performed twice in both up and down count periods.

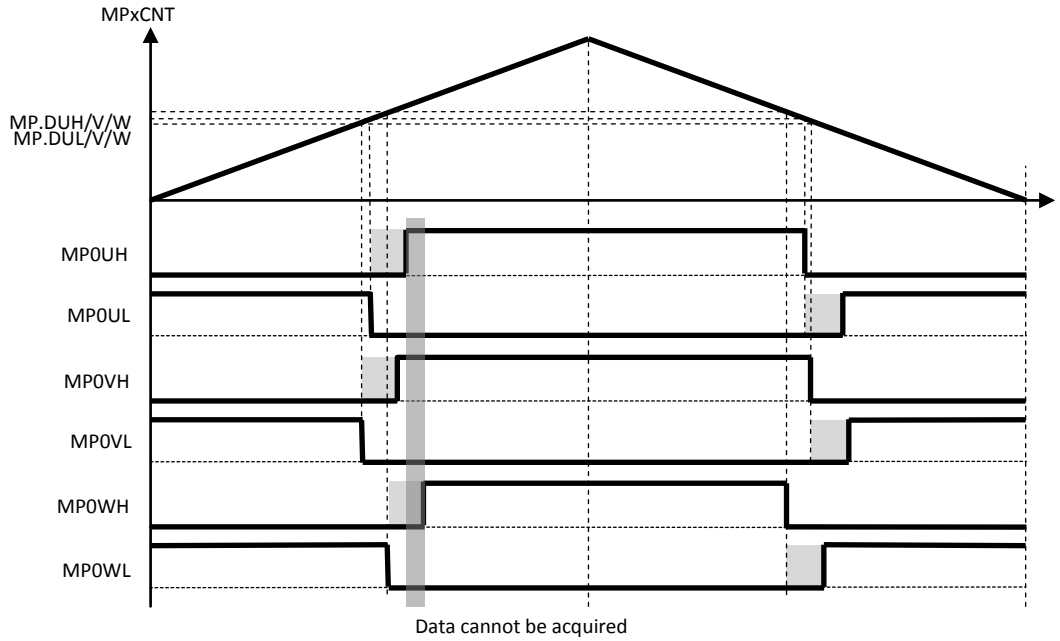



**Figure 16.18. Symmetrical PWM Timing**

In asymmetrical mode, the wave from is not symmetrical from the mid-point of the period. The duty comparison of H-side is performed in up count period. The duty comparison of L-side is performed in down count period.

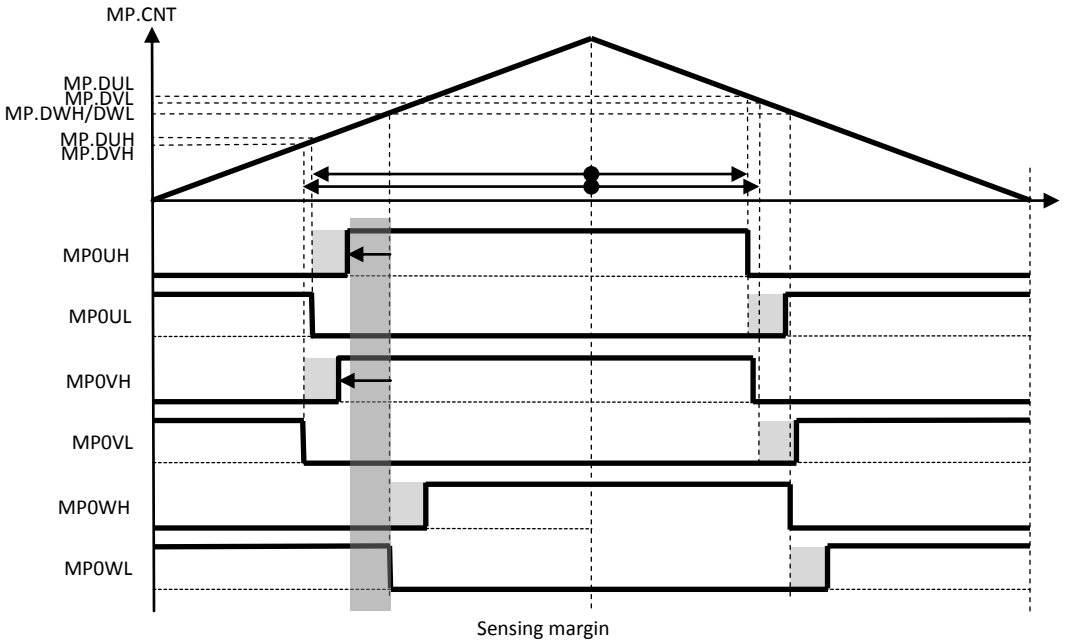



**Figure 16.19. Asymmetrical PWM Timing and Sensing Margin**

## Description of ADC Triggering Function

A total of 6 ADC trigger timing registers are provided. This dedicated register generates a trigger signal to start ADC conversion. The conversion channel of ADC is defined in the ADC control register.

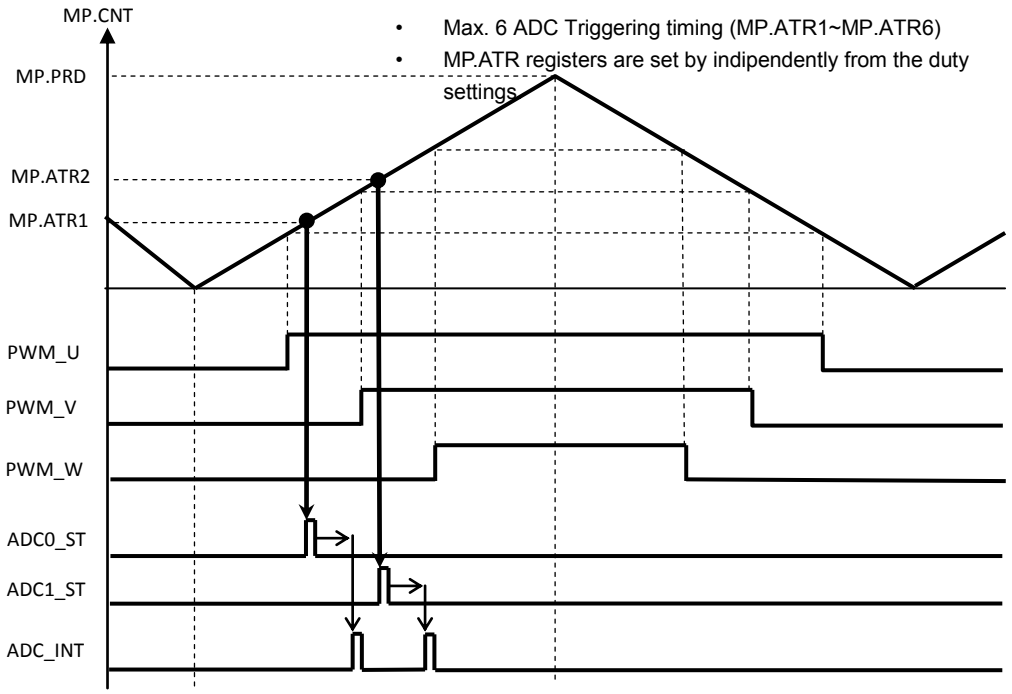



**Figure 16.20. ADC Triggering Function Timing Diagram**

An example of ADC data acquisition is shown in Figure 16.21.

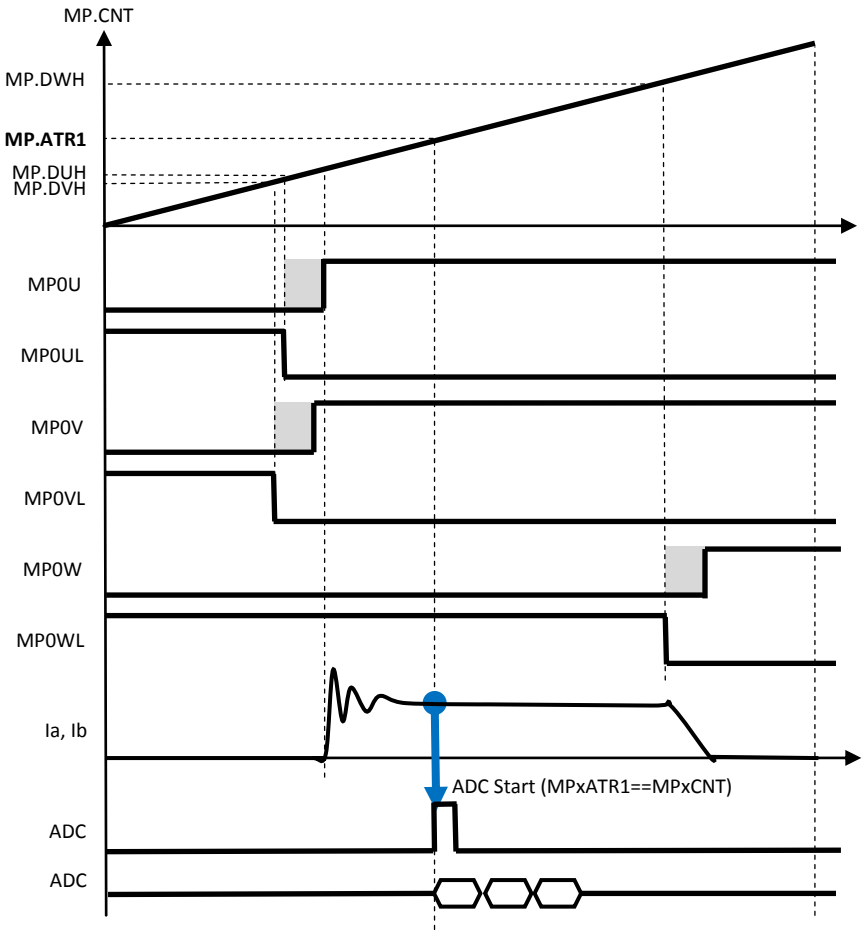



**Figure 16.21. An Example of ADC Aquisition Timing by Event from MPWM**

## Interrupt Generation Timing

Each timing event can make interrupt requests to the CPU.

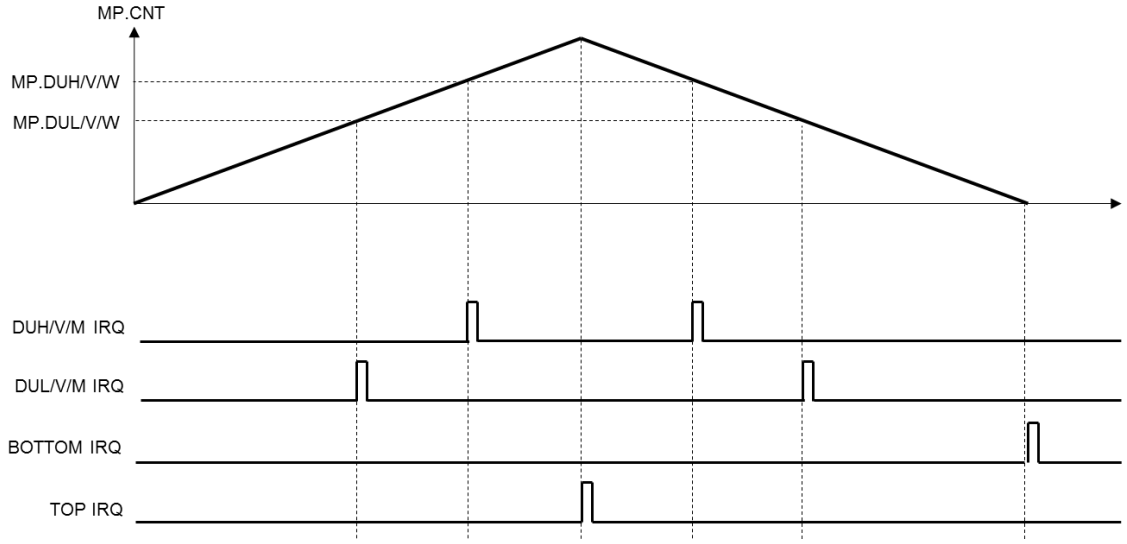



**Figure 16.22. Interrupt Generation Timing**

# 17. 12-Bit A/D Converter

## Introduction

ADC block consists of 2 independent ADC units.

- 16 Channels of analog inputs
- Single and Sequential conversion mode
- Up to 8 times sequential conversion support
- External pin trigger support
- 8 internal trigger sources support (PWMs, timers)
- Adjustable sample & hold time

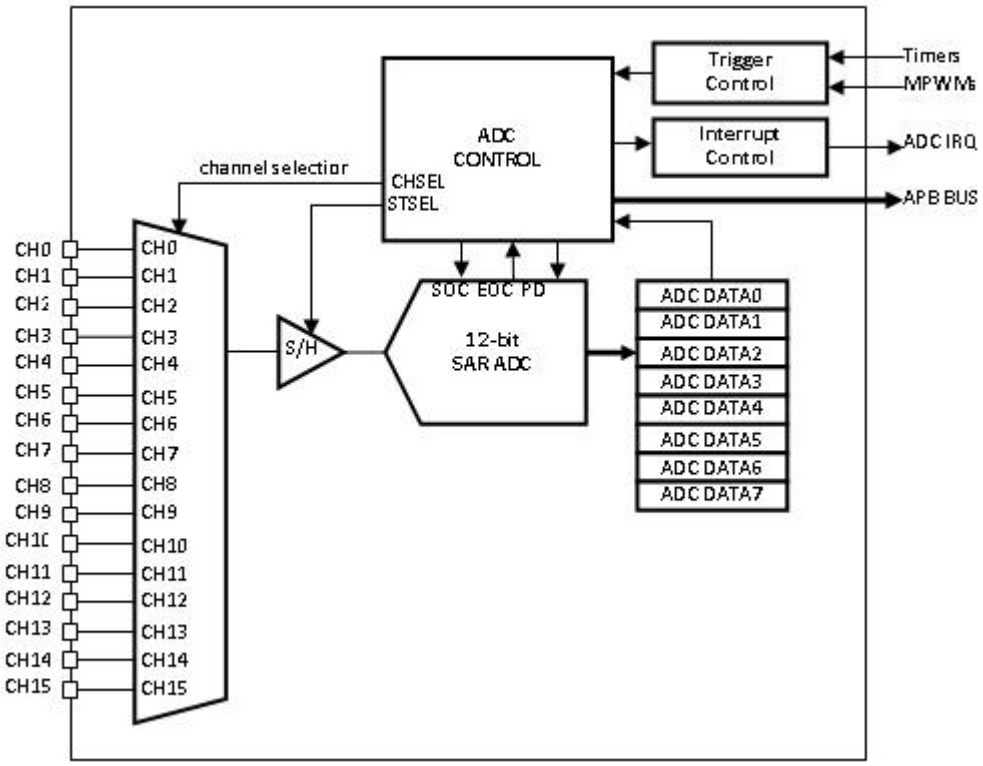



**Figure 17.1. Block Diagram**

# Pin Description

Table 17.1. External Signal

| PIN NAME | TYPE | DESCRIPTION | AINx PIN to ADC Channel Mapping | |
|---|---|---|---|---|
| **AVDD** | P | Analog Power(3.0V~VDD) | ADC0 | ADC1 |
| **AVSS** | P | Analog GND | | |
| **AN0** | A | ADC Input 0 | Channel 0 | Channel 0 |
| **AN1** | A | ADC Input 1 | Channel 1 | Channel 1 |
| **AN2** | A | ADC Input 2 | Channel 2 | Channel 2 |
| **AN3** | A | ADC Input 3 | Channel 3 | Channel 3 |
| **AN4** | A | ADC Input 4 | Channel 4 | Channel 4 |
| **AN5** | A | ADC Input 5 | Channel 5 | Channel 5 |
| **AN6** | A | ADC Input 6 | Channel 6 | -- |
| **AN7** | A | ADC Input 7 | Channel 7 | -- |
| **AN8** | A | ADC Input 8 | Channel 8 | -- |
| **AN9** | A | ADC Input 9 | Channel 9 | -- |
| **AN10** | A | ADC Input 10 | Channel 10 | -- |
| **AN11** | A | ADC Input 11 | Channel 11 | -- |
| **AN12** | A | ADC Input 12 | Channel 12 | -- |
| **AN13** | A | ADC Input 13 | Channel 13 | -- |
| **AN14** | A | ADC Input 14 | Channel 14 | Channel 6 |
| **AN15** | A | ADC Input 15 | -- | Channel 7 |
| **AN16** | A | ADC Input 16 | -- | Channel 8 |
| **AN17** | A | ADC Input 17 | -- | Channel 9 |
| **AN18** | A | ADC Input 18 | -- | Channel 10 |
| **AN19** | A | ADC Input 19 | -- | Channel 11 |
| **AN20** | A | ADC Input 20 | -- | Channel 12 |
| **AN21** | A | ADC Input 21 | -- | Channel 13 |

# Registers

The base addresses of ADC units are shown in Table 17.2.

**Table 17.2. ADC Base Address**

| | BASE ADDRESS |
|---|---|
| **ADC0** | 0x4000_B000 |
| **ADC1** | 0x4000_B100 |

Table 17.3 shows the register memory map.

**Table 17.3. ADCIF Register Map**

| Name | Offset | R/W | Description | Reset |
|---|---|---|---|---|
| **ADn.MR** | 0x0000 | R/W | ADC Mode register | 0x00 |
| **ADn.CSCR** | 0x0004 | R/W | ADC Current Sequence/Channel register | 0x00 |
| **ADn.CCR** | 0x0008 | R/W | ADC Clock Control register | 0x80 |
| **ADn.TRG** | 0x000C | R/W | ADC Trigger Selection register | 0x00 |
| **-** | 0x0010 | - | Reserved | |
| **-** | 0x0014 | - | Reserved | |
| **ADn.SCSR** | 0x0018 | R/W | ADC Burst mode channel select | 0x00 |
| **ADn.CR** | 0x0020 | R/W | ADC Control register | 0x00 |
| **ADn.SR** | 0x0024 | R/W | ADC Status register | 0x00 |
| **ADn.IER** | 0x0028 | R/W | ADC Interrupt Enable register | 0x00 |
| **ADn.DDR** | 0x002C | R | ADCn DMA Data Register | 0x00 |
| **ADn.DR0** | 0x0030 | R | ADCn Sequence 0 Data register | 0x00 |
| **ADn.DR1** | 0x0034 | R | ADCn Sequence 1 Data register | 0x00 |
| **ADn.DR2** | 0x0038 | R | ADCn Sequence 2 Data register | 0x00 |
| **ADn.DR3** | 0x003C | R | ADCn Sequence 3 Data register | 0x00 |
| **ADn.DR4** | 0x0040 | R | ADCn Sequence 4 Data register | 0x00 |
| **ADn.DR5** | 0x0044 | R | ADCn Sequence 5 Data register | 0x00 |
| **ADn.DR6** | 0x0048 | R | ADCn Sequence 6 Data register | 0x00 |
| **ADn.DR7** | 0x004C | R | ADCn Sequence 7 Data register | 0x00 |

# ADn.MR ADC*n* Mode Register

ADC Mode Registers are 32-bit registers. This register configures ADC operation mode. This register should be written first before the other registers.

**AD0.MR=0x4000_B000, AD1.MR=0x4000_B100**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | DMACH | DMAEN | | STSEL | | | | | SEQCNT | | | ADEN | ARST | ADMOD | | | | | TRGSEL | |
| | | | | | | | | | | | | | 0x0 | 0x0 | | 0x0 | | | | | 0x0 | | | 0x0 | 0x0 | 0x0 | | | | | 0x0 | |
| | | | | | | | | | | | | | RW | RW | | RW | | | | | RW | | | RW | RW | RW | | | | | RW | |

| | | | | | |
|---|---|---|---|---|---|
| 18 | **DMACH** | DMA channel option<br>When DMACH is set, Channel information of DMA data will be located at ADDMAR[3:0] for half word size transfer.<br>Channel information is at ADDMAR[19:16] in default.(DMACH is low) | | | |
| 17 | **DMAEN** | DMA enable bit – should be set to '1' when ADCEN='1'.<br>When DMA function is enabled, DMA request at every end of conversion (also in burst mode) and interrupt request only be generated when ADC receives DMA done from DMAC. | | | |
| 16<br>12 | **STSEL** | Sampling Time Selection<br>ADC Sample & Hold circuit sampling time become (2 + STSEL[4:0]) MCLK cycles<br>Minimum sampling time is 2 MCLK cycle<br>When STSEL[4:0]=11111, the sampling channel is always on. | | | |
| 10<br>8 | **SEQCNT** | Number of coversions in a sequence<br>If ADMOD is 2'h0 and SEQCNT is not 3'h0, CSEQN will be increased up to SEQCNT by trigger event. | | | |
| | | 000 | Single mode | 100 | 5 sequence AD conversion |
| | | 001 | 2 sequence AD conversion | 101 | 6 sequence AD conversion |
| | | 010 | 3 sequence AD conversion | 110 | 7 sequence AD conversion |
| | | 011 | 4 sequence AD conversion | 111 | 8 sequence AD conversion |
| 7 | **ADEN** | 0 | ADC disable | | |
| | | 1 | ADC enable | | |
| 6 | **ARST** | 0 | Stop at the end of sequence.<br>Should set ASTART as 1 to restart again | | |
| | | 1 | Restart at the end of sequence. | | |
| 5<br>4 | **ADMOD** | 00 | Single/Continuous conversion mode | | |
| | | 01 | Sequenced conversion mode | | |
| | | 10 | Reserved | | |
| | | 11 | Reserved | | |
| 1<br>0 | **TRGSEL** | 00 | Event Trigger Disabled/Soft-Trigger Only | | |
| | | 01 | Timer Event Trigger | | |
| | | 10 | MPWM0 Event Trigger | | |
| | | 11 | MPWM1 Event Trigger | | |

If ADn.MR.ADCMOD was set as Single Sequential Conversion mode (Single sequential mode), the AD channels are controlled by ADn.SCSR.SEQ0CH ~ SEQ7CH. Single sequential mode always starts from SEQ0CH when not writing CSEQN. If in single sequential mode and SEQCNT is set as 3, AD converts the channels which are assigned at ADn.SCSR.SEQ0CH, ADn.SCSR.SEQ1CH, and ADn.SCSR.SEQ2CH to ADn.TRG.SEQTRG0, ADn.TRG.SEQTRG1, and ADn.TRG.SEQTRG2 when ADn.MR.TRGSEL is 0x1(Timer Event Trigger) or 0x2(MPWM Event Trigger).

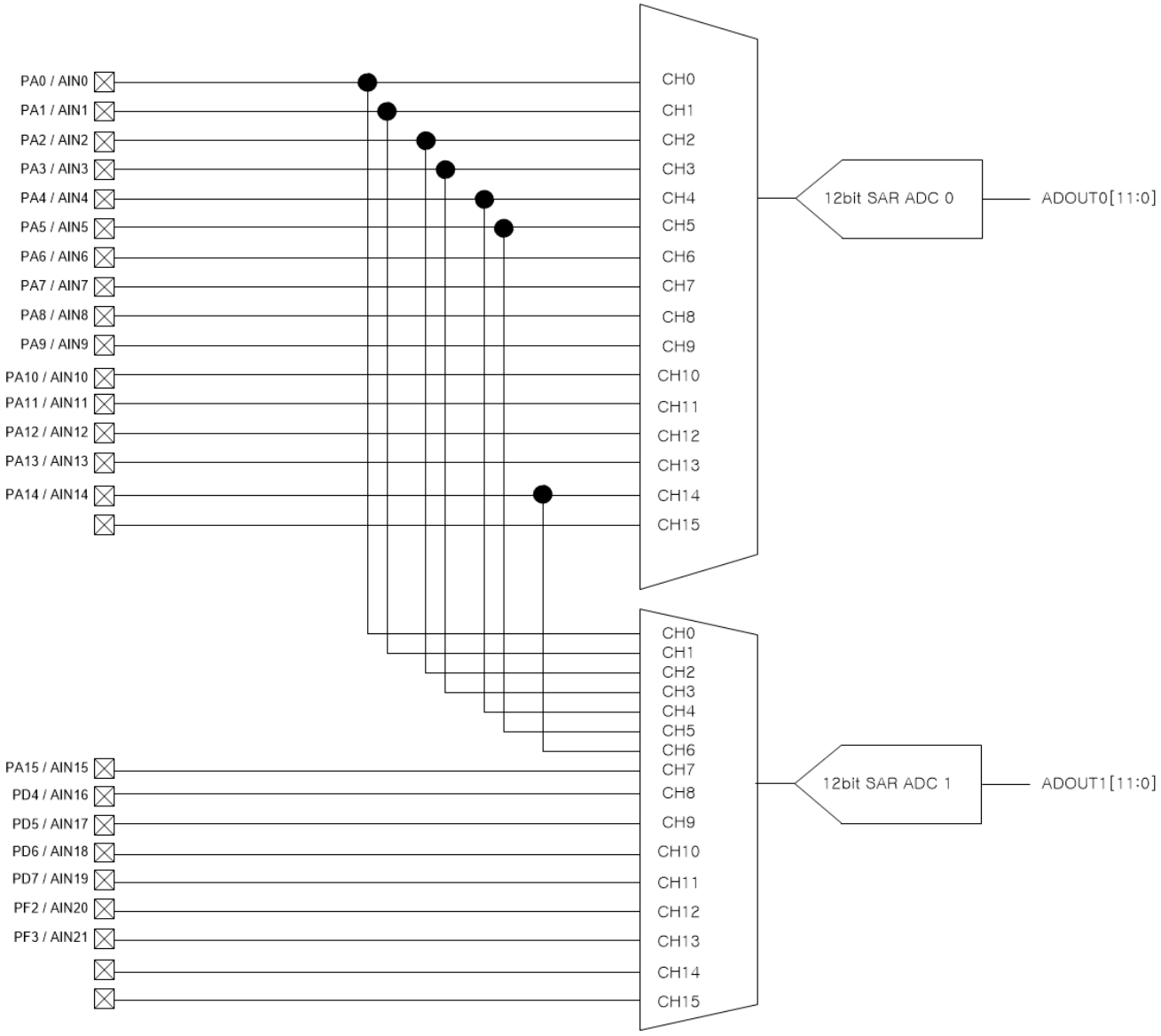



**Figure 17.2. Analog Channel Block Diagram**

# ADn.CSCR ADC*n* Current Sequence/Channel Register

ADC Current Sequence/Channel Registers are 7-bit registers. These registers consist of Current Sequence Number and Current Active Channel value. CSEQN (Current Sequence Number) can be written to change the next sequence number. Upon writing CSEQN as 0x7 when CSEQN is 0x3 and ADn.MR.SEQCNT is 0x7, the next sequence number is 0x7 and AD converts this channel of ADn.SCSR.SEQ7CH. The 4,5,6 sequence is skipped. This register should be written first before ADn.SCSR.

**D0.CSCR=0x4000_B004, AD1.CSCR=0x4000_B104**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | CSEQN | | | CACH | | | |
| - | 0x0 | | | 0x0 | | | |
| - | RW | | | RO | | | |

| | | | |
|---|---|---|---|
| 7<br>4 | **CSEQN** | Current Sequence Number , can write when not busy | |
| | | 0000 | Current Sequence is 0 |
| | | 0001 | Current Sequence is 1 |
| | | 0010 | Current Sequence is 2 |
| | | 0011 | Current Sequence is 3 |
| | | 0100 | Current Sequence is 4 |
| | | 0101 | Current Sequence is 5 |
| | | 0110 | Current Sequence is 6 |
| | | 0111 | Current Sequence is 7 |
| 3<br>R0 | **CACH** | Current Active Channel | |
| | | 0000 | ADC channel 0 is active |
| | | 0001 | ADC channel 1 is active |
| | | 0010 | ADC channel 2 is active |
| | | 0011 | ADC channel 3 is active |
| | | 0100 | ADC channel 4 is active |
| | | 0101 | ADC channel 5 is active |
| | | 0110 | ADC channel 6 is active |
| | | 0111 | ADC channel 7 is active |
| | | 1000 | ADC channel 8 is active |
| | | 1001 | ADC channel 9 is active |
| | | 1010 | ADC channel 10 is active |
| | | 1011 | ADC channel 11 is active |
| | | 1100 | ADC channel 12 is active |
| | | 1101 | ADC channel 13 is active |
| | | 1110 | ADC channel 14 is active |
| | | 1111 | ADC channel 15 is active |

# ADn.CR1 ADC*n* Clock Control Register

ADC Control Registers are 16-bit registers. ADC period register

**AD0.CR1=0x4000_B008, AD1.CR1=0x4000_B108**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADCPDA | CKDIV | | | | | | | ADCPD | EXTCLK | CLKINVT | - | - | - | - | - |
| 0 | 0x00 | | | | | | | 1 | 0 | 0 | | | | | |
| RW | RW | | | | | | | RW | RW | RW | | | | | |

| | | |
|---|---|---|
| 15 | ADCPDA | ADC R-DAC disable to save power<br>Don't set "1" here(it's optional bit) |
| 14<br>8 | CLKDIV[6:0] | ADC clock divider when EXTCLK is '0'.<br>ADC clock = system clock/CLKDIV<br>CKDIV=0 : ADC clock=system clock<br>CKDIV=1 : ADC clock=stop |
| 7 | ADCPD | ADC Power Down<br>0 – ADC normal mode<br>1 – ADC Power Down mode |
| 6 | EXTCLK | Select if ADC uses external clock.<br>0 – internal clock(CKDIV enabled)<br>1 – external clock(Clock configured in SCU_MCCR4) |
| 5 | CLKINVT | Divided clock inversion(optional bit)<br>0 – duty ratio of divided clock is larger than 50%<br>1 – duty ratio of divided clock is less than 50% |

# AD*n*.TRG ADC Trigger Selection Register

ADC Trigger registers are 32-bt registers. ADC Trigger channel register. In Single/Continuous mode, all the bit fields are used. In Burst conversion mode, only BSTTRG bit field(bit3~bit0) is used.

**AD0.TRG=0x4000_B00C, AD1.TRG=0x4000_B10C**

| 31 | 30 29 28 | 27 | 26 25 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SEQTRG7 | | SEQTRG6 | | SEQTRG5 | | SEQTRG4 | | SEQTRG3 | | SEQTRG2 | | SEQTRG1 | | SEQTRG0<br>BSTTRG |
| | 0x0 | | 0x0 | | 0x0 | | 0x0 | | 0x0 | | 0x0 | | 0x0 | | 0x0 |
| | RW | | RW | | RW | | RW | | RW | | RW | | RW | | RW |

| | | |
|---|---|---|
| 30<br>28 | SEQTRG7 | 8th Sequence Trigger Source |
| 26<br>24 | SEQTRG6 | 7th Sequence Trigger Source |
| 22<br>20 | SEQTRG5 | 6th Sequence Trigger Source |
| 18<br>16 | SEQTRG4 | 5th Sequence Trigger Source |
| 14<br>12 | SEQTRG3 | 4th Sequence Trigger Source |
| 10<br>8 | SEQTRG2 | 3rd Sequence Trigger Source |
| 6<br>4 | SEQTRG1 | 2nd Sequence Trigger Source |
| 2<br>0 | SEQTRG0 | 1st Sequence Trigger Source |

| Value | Timer<br>(TRGSEL '2'h1) | MPWM0<br>(TRGSEL '2'h2) | MPWM1<br>(TRGSEL '2'h3) |
|---|---|---|---|
| 0 | Timer 0 | MP0ATR1 | MP1ATR1 |
| 1 | Timer 1 | MP0ATR2 | MP1ATR2 |
| 2 | Timer 2 | MP0ATR3 | MP1ATR3 |
| 3 | Timer 3 | MP0ATR4 | MP1ATR4 |
| 4 | Timer 8 | MP0ATR5 | MP1ATR5 |
| 5 | Timer 9 | MP0ATR6 | MP1ATR6 |
| 6 | - | BOTTOM | BOTTOM |
| 7 | - | PERIOD | PERIOD |

## AD*n*.SCSR ADC Sequence Channel Selection Register

The ADC Sequence Channel Select Register is a 32-bit register.

**AD0.BCSR=0x4000_B018, AD1.BCSR=0x4000_B118**

| 31 30 29 28 | 27 26 25 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 |
|---|---|---|---|---|---|---|---|
| SEQ7CH | SEQ6CH | SEQ5CH | SEQ4CH | SEQ3CH | SEQ2CH | SEQ1CH | SEQ0CH |
| 0x0 | 0x0 | 0x0 | 0x0 | 0x0 | 0x0 | 0x0 | 0x0 |
| RW | RW | RW | RW | RW | RW | RW | RW |

| | | |
|---|---|---|
| 31 28 | SEQ7CH | 8th conversion sequence channel selection |
| 27 24 | SEQ6CH | 7th conversion sequence channel selection |
| 23 20 | SEQ5CH | 6th conversion sequence channel selection |
| 19 16 | SEQ4CH | 5th conversion sequence channel selection |
| 15 12 | SEQ3CH | 4th conversion sequence channel selection |
| 11 8 | SEQ2CH | 3rd conversion sequence channel selection |
| 7 4 | SEQ1CH | 2nd conversion sequence channel selection |
| 3 0 | SEQ0CH | 1st conversion sequence channel selection |

## AD*n*.CR ADCn Control Register

This is the ADC start register. This register is an 8-bit register.

**AD0.CR=0x4000_B020, AD1.CR=0x4000_B120**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ASTOP | | | | | | | ASTART |
| 0 | | | | | | | 0 |
| W | | | | | | | RW |

| | | | |
|---|---|---|---|
| 7 | ASTOP | 0 | No |
| | | 1 | ADC conversion stop (will be clear next @ADC clock)<br>If ASTOP set after conversion cycle start, present conversion would be completed. |
| 0 | ASTART | 0 | No ADC conversion |
| | | 1 | ADC conversion start (will be clear next @ADC clock)<br>ADCEN should be "1" to start ADC.<br>If ASTART is set as 0 when ARST is 0 in Timer/MPWM trigger event mode, AD converts to ADn.MR.SEQCNT once and AD stops. ASTART should be written to start the conversion sequence again |

## ADn.SR ADC*n* Status Register

The ADC Status Register is a 32-bit register.

**AD0.SR=0x4000_B024, AD1.SR=0x4000_B124**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EOC | ABUSY | DOVRUN | DMAIRQ | TRGIRQ | EOSIRQ | - | EOCIRQ |
| 0 | 0 | 0 | 0 | 0 | 0 | - | 0 |
| RO | RO | RO | RO | RC | RC | - | RC |

| | | |
|---|---|---|
| 7 | EOC | ADC End-of-Conversion flag<br>(Start-of-Conversion made by ADC_CLK clears this bit , not ASTART) |
| 6 | ABUSY | ADC conversion busy flag |
| 5 | DOVRUN | DMA overrun flag (not interrupt)<br>(DMA ACK didn't come until end of next conversion) |
| 4 | DMAIRQ | DMA done received (DMA transfer is completed) |
| 3 | TRGIRQ | ADC Trigger interrupt flag(Write "1" to clear flag)<br>(0: no int / 1: int occurred) |
| 2 | EOSIRQ | This flag will be set upon final end of a sequence (Write "1" to clear flag) |
| | | 0 None. |
| | | 1 End-of-Sequence(burst) Interrupt occurred |
| 0 | EOCIRQ | This flag will be set upon each conversion in a sequence is occurred(Write "1" to clear flag) |
| | | 0 None. |
| | | 1 End-of-Conversion Interrupt occurred |

## ADn.IER Interrupt Enable Register

**AD0.IER=0x4000_B028, AD1.IER=0x4000_B128**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | DMAIRQE | TRGIRQE | EOSIRQE | - | EOCIRQE |
| | | | 0 | 0 | 0 | - | 0 |
| | | | RW | RW | RW | - | RW |

| | | |
|---|---|---|
| 4 | DMAIRQE | DMA done interrupt enable<br>0: interrupt disable<br>1: interrupt enable |
| 3 | TRGIRQE | ADC trigger conversion interrupt enable |
| 2 | EOSIRQE | ADC sequence conversion interrupt enable |
| 0 | EOCIRQE | ADC single conversion interrupt enable |

## AD*n*.DDR ADC*n* DMA Data Register

ADC DMA Data Registers are 16-bit registers. This is the ADC conversion result register for DMA (AD data of just completed conversion).

**AD0.DDR=0x4000B02C, AD1.DDR=0x4000B12C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADC DMA Temporary Data | | | | | | | | | | | | ADMACH | | | |
| 0x000 | | | | | | | | | | | | 0x0 | | | |
| R | | | | | | | | | | | | R | | | |

| | | |
|---|---|---|
| 15 4 | ADDMAR | ADC conversion result data (12-bit) |
| 3 0 | ADMACH | ADC data channel indicator |

## AD*n*.DRx ADC*n* Sequence Data Register 0~7

The DRx Data Registers are 16-bit registers. This is the ADC conversion result register for the related sequence number.

**AD0.DR0=0x4000_B030, AD0.DR1=0x4000_B034, AD0.DR2=0x4000_B038, AD0.DR3=0x4000_B03C**
**AD0.DR4=0x4000_B040, AD0.DR5=0x4000_B044, AD0.DR6=0x4000_B048, AD0.DR7=0x4000_B04C**
**AD1.DR0=0x4000_B130, AD1.DR1=0x4000_B134, AD1.DR2=0x4000_B138, AD1.DR3=0x4000_B13C**
**AD1.DR4=0x4000_B140, AD1.DR5=0x4000_B144, AD1.DR6=0x4000_B148, AD1.DR7=0x4000_B14C**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADCDATA | | | | | | | | | | | | | | | |
| 0x000 | | | | | | | | | | | | | | | |
| R | | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| 15 4 | ADC DATA | ADC Sequence Data (12-bit) |

# Functional Description

The ADC block provides the ability to convert an analog signal to a digital value. The ADC compares the input channel with the AVDD voltage and provides a 12-bit value.

```
Voltage value = (ADC Reading / 4096) * AVDD voltage
```

The ADC clock can be configured up to 22.5Mhz and be driven from any of the available clocks – System clock, Ring OSC, Bus Clock, Int OSC, External OSC or the PLL clock. There is a 6 bit divider available for the system clock (divider must be greater than 1) or the ADC clock can be configured in the MCCR6 register, which provides access to all clocks and 8 bit divider. The clock is selected in CR1 register (and optionally configured in the SCU MCCR4 register).

The ADC takes 15 ADC clocks to complete one sample. There is a single clock to start then sample and hold time (minimum of 2 ADC clocks) then 1 clock per bit (12 bits). To increase sample time, you can configure up to 511 clock sampling time (which would then take 511 + 15 = 526 ADC clocks per sample).

To calculate the maximum ADC clock that can be used is:

```
ADC Clock = 1.5Msps * (15 clocks per sample + Sample time)

Example (Sampling time = 0):

ADC clock = 1.5Msps * (15 clocks + 0) = 22.5Mhz
```

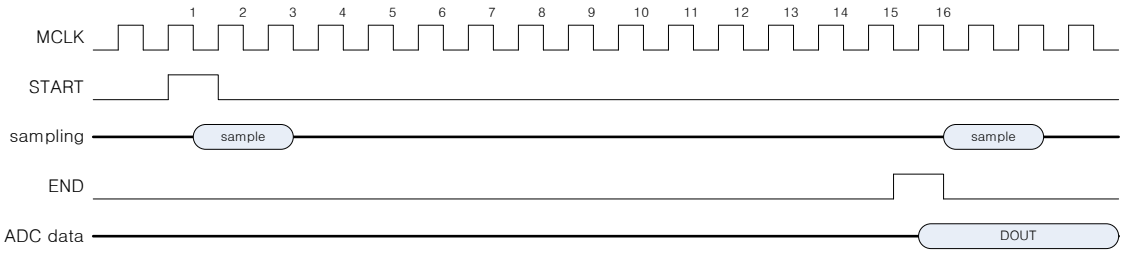
The above example shows that if the system clock was running at 72 MHz the divider cannot be less than 4.

The sequence feature allows the programmer to retrieve multiple readings (up to 8) with only one start request. You can either continue to run the sequence or run it just once. The ADC block will automatically go through all 8 taking readings without intervention. Each sequence can be triggered on different events in order and sequence result has its own data register.

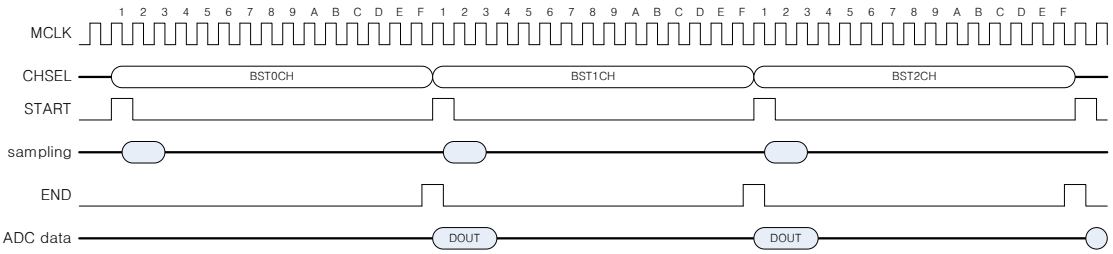
**General ADC Setup Procedure**

1. Allow the modification of the I/O pins to use the ADC inputs needed by writing the unlock sequence as described in PORT CONTROL UNIT (PCU), no pullups enabled.
2. Enable the ADC peripherals needed in PER2 register.
3. Enable the ADC peripheral clock in the PCER2 register.
4. Select the alternating function for the ADC inputs (Port n MUX registers).
5. Configure the ADC Pins to Analog
6. Configure the ADC mode in the ADCnMODE register and enabled the channel ADCn.
7. Configure the ADCnCR1 register and write an appropriate clock divider value.
8. Configure any special features such as triggers, sequencing, etc.
9. Start ADC conversion

## ADC Single Mode Timing Diagram




## ADC Sequencing Mode Timing Diagram



# 18. Electrical Characteristics

## DC Characteristics

### Absolute Maximum Ratings

Absolute maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions..

**Table 18.1. Absolute Maximum Rating**

| Parameter | Symbol | min | max | unit |
|---|---|---|---|---|
| Power Supply (VDD) | VDD | -0.5 | +6 | V |
| Analog Power Supply (AVDD) | AVDD | -0.5 | +6 | V |
| VDC Output Voltage | VDD18 | | | V |
| Input High Voltage | | - | VDD+0.5 | V |
| Input Low Voltage | | VSS – 0.5 | - | V |
| Output Low Current per pin | $I_{OL}$ | | 20 | mA |
| Output Low Current Total | $\sum I_{OL}$ | | | mA |
| Output Low Current per pin | $I_{OH}$ | | 10 | mA |
| Output Low Current Total | $\sum I_{OH}$ | | | mA |
| Power consumption | | | | mW |
| Input Main Clock Range | | 0.4 | 8 | MHz |
| Operating Frequency | | - | 72 | MHz |
| Storage Temperature | Tst | -55 | +125 | ℃ |
| Operating Temperature | Top | -40 | +85 | ℃ |

# DC Characteristics

**Table 18.2 Recommended Operating Conditions**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Supply Voltage | VDD | | 3.0 | | 5.5 | V |
| Supply Voltage | AVDD | | 3.0 | 5.0 | 5.5 | V |
| Operating Frequency | FREQ | MOSC | 4 | | 16 | MHz |
| | | INTOSC | | 20 | | MHz |
| | | PLL | 4 | | 75 | MHz |
| Operating Temperature | Top | Top | -40 | | +85 | ℃ |

**Table 18.3 DC Electrical Characteristics (VDD = +5V, Ta = 25 ℃)**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Input Low Voltage | $V_{IL}$ | Schmitt input | - | - | 0.2VDD | V |
| Input High Voltage | $V_{IH}$ | Schmitt input | 0.8VDD | - | - | V |
| Output Low Voltage | $V_{OL}$ | $I_{OL}$ = 10mA | - | - | VSS+1.0 | V |
| Output High Voltage | $V_{OH}$ | $I_{OH}$ = -3mA | VDD–1.0 | - | - | V |
| Output Low Current | $I_{OL}$ | | - | - | 3 | mA |
| Output High Current | $I_{OH}$ | | – 1.2 | - | | mA |
| Input High Leakage | $I_{IH}$ | | | | 4 | uA |
| Input Low Leakage | $I_{IL}$ | | -4 | | | |
| Pull-up Resister | $R_{PU}$ | Rmax:VDD =3.0V Rmin:VDD =5V | 30 | - | 70 | kΩ |

# POR Electrical Characteristics

**Table18.4 POR Electrical Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD18 | | 1.6 | 1.8 | 2.0 | V |
| Operating Current | $IDD_{PoR}$ | Typ. <6uA If always on | - | 60 | - | nA |
| POR Set Level | $VR_{PoR}$ | VDD rising (slow) | 1.3 | 1.4 | 1.55 | V |
| POR Reset Level | $VF_{PoR}$ | VDD falling (slow) | 1.1 | 1.2 | 1.4 | V |

# BOD Electrical Characteristics

**Table18.5 BOD Electrical Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD | | 1.7 | | 5 | V |
| Operating Current | $IDD_{BOD}$ | Typ. <6uA when always on | - | 1 | - | mA |
| BOD Set Level 0 | VBOD0 | VDD falling (slow) | 1.7 | 1.8 | 1.9 | V |
| BOD Set Level 1 | VBOD1 | VDD falling (slow) | 2.1 | 2.2 | 2.3 | V |
| BOD Set Level 2 | VBOD2 | VDD falling (slow) | 3.2 | 3.3 | 3.4 | V |
| BOD Set Level 3 | VBOD3 | VDD falling (slow) | 4.2 | 4.3 | 4.4 | V |

# VDC Electrical Characteristics

**Table18.6VDC Electrical Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | $VDD_{VDC}$ | | 3.0 | - | 5.5 | V |
| VDC Output Voltage | $VOUT_{VDC}$ | @RUN | 1.62 | 1.8 | 1.98 | V |
| | | @STOP | 1.4 | 1.8 | 2.0 | V |
| Regulation Current | $I_{OUT}$ | | | | 100 | mA |
| Drop-out Voltage | $VDROP_{VDC}$ | VDDVDC=3.0V IOUT=100mA | - | - | 200 | mV |
| Current Consumption | $IDD_{NORM}$ | @RUN | - | 100 | 150 | uA |
| | $IDD_{STOP}$ | @STOP | - | 1 | 2 | uA |

# External OSC Characteristics

**Table18.7External OSC Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD | | 3.0 | - | 5.5 | V |
| IDD | | @4MHz/5V | - | 240 | | uA |
| Frequency | $OSCF_{req}$ | | 4 | 8 | 10 | MHz |
| Output Voltage | $OSC_{VOUT}$ | | 1.2 | 2.4 | - | V |
| Load Capacitance | $LOAD_{CAP}$ | | 5 | 22 | 35 | pF |

# Sub External OSC Characteristics

**Table18.8External OSC Characteristics (Temperature: -40 ~ +85℃)**

| Parameter | Symbol | Condition | Min | Typ | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD | | | 1.8 | | V |
| IDD | | @4MHz/5V | - | 2.93 | | uA |
| Frequency | $OSCF_{req}$ | | | 32.768 | | KHz |
| Output Voltage | $OSC_{VOUT}$ | | | 1.8 | - | V |
| Load Capacitance | $LOAD_{CAP}$ | | | | | pF |

# Internal RC OSC Characteristics

**Table 18.9 Internal RC OSC Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD | | 1.65 | 1.8 | 1.95 | V |
| IDD | $I_{OSC}$ | @20MHz | - | 240 | | uA |
| Frequency | $IOSCF_{req}$ | | | 20 | | MHz |

# PLL Electrical Characteristics

**Table18.10PLL Electrical Characteristics (Temperature: -40 ~ +105℃)**

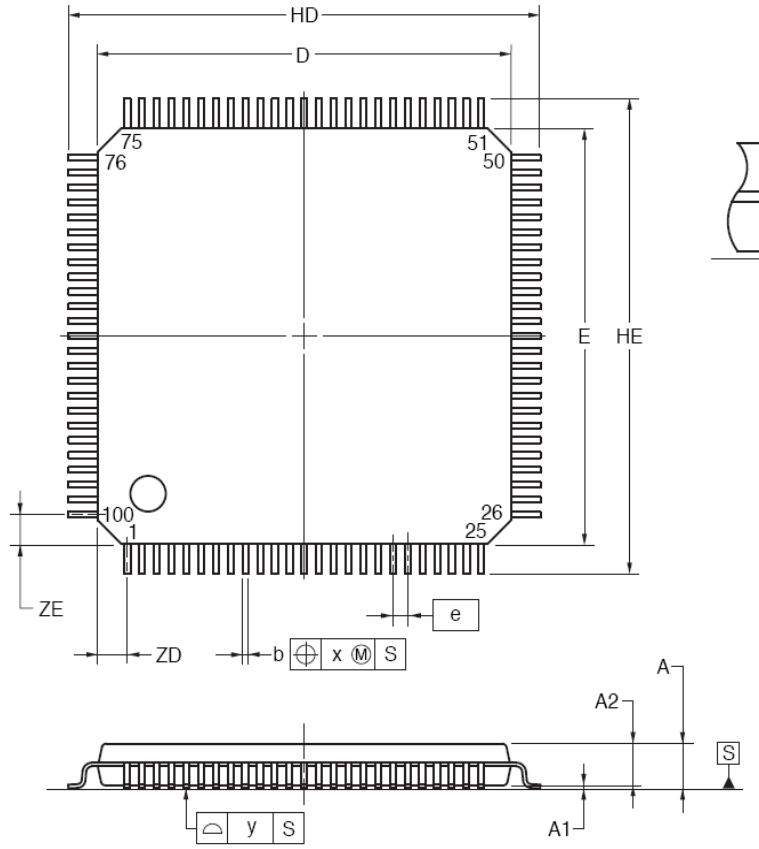
| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | $VDD_{PLL}$ | | 1.65 | 1.8 | 1.95 | V |
| Output Frequency | FOUT | | 4 | | 80 | MHz |
| Operating Current | $IDD_{PLL}$ | @80MHz | | 1.3 | | mA |
| Duty | $FOUT_{DUTY}$ | | 40 | - | 60 | % |
| P-P Jitter | JITTER | @Lock | | | 500 | Ps |
| VCO | VCO | | 30 | | 80 | MHz |
| Input Frequency | FIN | | 4 | | 8 | MHz |
| Locking time | LOCK | | | | 1 | ms |

# ADC Electrical Characteristics

**Table 18.11ADC Electrical Characteristics (Temperature: -40 ~ +105℃)**

| Parameter | Symbol | Condition | Min | Typ. | Max | unit |
|---|---|---|---|---|---|---|
| Operating Voltage | AVDD | | 3.0 | 5 | 5.5 | V |
| Reference Voltage | AVREF | | 3.0 | 5 | 5.5 | V |
| Resolution | | | | 12 | | Bit |
| Operating Current | IDDA | | | | 2.8 | mA |
| Analog Input Range | | | 0 | | AVDD | V |
| Conversion Rate | | | | - | 1.5 | MHz |
| Operating Frequency | ACLK | | | | 15 | MHz |
| DC Accuracy | INL | | | ±2.5 | | LSB |
| | DNL | | | ±1.0 | | LSB |
| Offset Error | | | | ±1.5 | | LSB |
| Full Scale Error | | | | ±1.5 | | LSB |
| SNDR | SNDR | | | 68 | | dB |
| THD | | | | -70 | | dB |

# LQFP-100 Package Dimension




(UNIT:mm)

| ITEM | DIMENSIONS |
|---|---|
| D | 14.00±0.20 |
| E | 14.00±0.20 |
| HD | 16.00±0.20 |
| HE | 16.00±0.20 |
| A | 1.60 MAX. |
| A1 | 0.10±0.05 |
| A2 | 1.40±0.05 |
| A3 | 0.25 |
| b | $0.20^{+0.07}_{-0.03}$ |
| c | $0.125^{+0.075}_{-0.025}$ |
| L | 0.50 |
| Lp | 0.60±0.15 |
| L1 | 1.00±0.20 |
| θ | $3°^{+5°}_{-3°}$ |
| e | 0.50 |
| x | 0.08 |
| y | 0.08 |
| ZD | 1.00 |
| ZE | 1.00 |

P100GC-50-UEU-2

NOTE:

ALL DIMENSIONS REFER TO JEDEC STANDARD MS-02MMP0UL

**Figure 18.1. Package Dimension (LQFP-100 14X14)**

LifeElectronics
Живое партнерство

**ООО «ЛайфЭлектроникс»** **“LifeElectronics” LLC**

*ИНН 7805602321 КПП 780501001 Р/С 40702810122510004610 ФАКБ "АБСОЛЮТ БАНК" (ЗАО) в г.Санкт-Петербурге К/С 30101810900000000703 БИК 044030703*

*Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.*

*С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические), за счёт заключения дистрибьюторских договоров*

*Мы предлагаем:*

- *Конкурентоспособные цены и скидки постоянным клиентам.*
- *Специальные условия для постоянных клиентов.*
- *Подбор аналогов.*
- *Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.*

- *Приемлемые сроки поставки, возможна ускоренная поставка.*
- *Доставку товара в любую точку России и стран СНГ.*
- *Комплексную поставку.*
- *Работу по проектам и поставку образцов.*
- *Формирование склада под заказчика.*

- *Сертификаты соответствия на поставляемую продукцию (по желанию клиента).*
- *Тестирование поставляемой продукции.*
- *Поставку компонентов, требующих военную и космическую приемку.*
- *Входной контроль качества.*
- *Наличие сертификата ISO.*

*В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.*

*Конструкторский отдел помогает осуществить:*

- *Регистрацию проекта у производителя компонентов.*
- *Техническую поддержку проекта.*
- *Защиту от снятия компонента с производства.*
- *Оценку стоимости проекта по компонентам.*
- *Изготовление тестовой платы монтаж и пусконаладочные работы.*

*Тел: +7 (812) 336 43 04 (многоканальный)*
*Email: org@lifeelectronics.ru*

***www.lifeelectronics.ru***