Cortex®-M4F 32-bit Microcontroller with FPU, 140 MHz, Up to 1024 KB Dual Bank Code Flash, 32 KB Data Flash, 64 KB SRAM, 3-phase PWM, 12-bit ADC for Motor Control Application

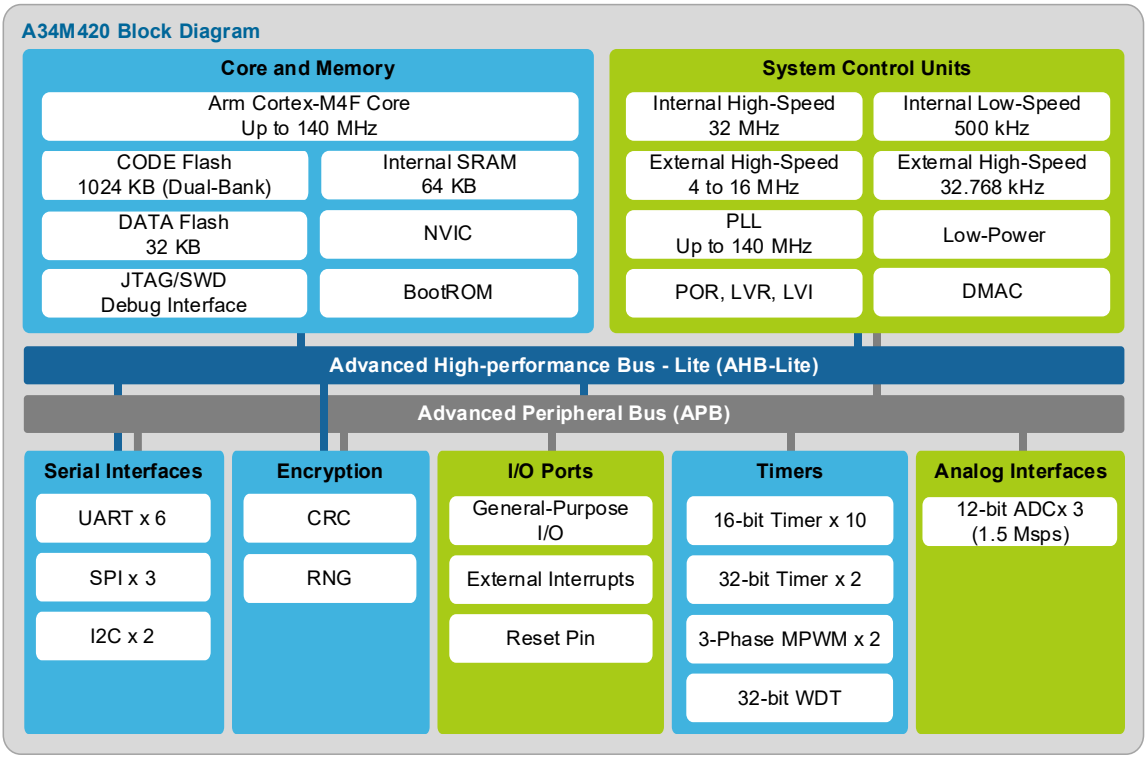
UM Rev. 1.00

The Introduction

The A34M420 series is a 32-bit microcontroller suitable for applications that require high-performance processing such as electric motors. The microcontroller includes the Arm Cortex-M4F, which is a high-performance 32-bit core, and a variety of peripheral devices for motor control. The A34M420 series is equipped with two three-phase PWM generators capable of controlling two inverter motors simultaneously. And an Individual mode optimized for induction heating (IH) has been added.

The microcontroller's three high-speed 12-bit analog-to-digital converters are capable of processing motor driving information coming in through 24 analog channels. With this function, the microcontroller can control either two inverter motors, or one inverter motor and a power factor correction (PFC) process, simultaneously. The microcontroller comes with various communication interfaces, such as UART, SPI, and I2C modules. That enables it to interface with external devices. It also has dual-bank flash memory for versatile uses. The two banks can either function as a unified flash memory or work independently. For instance, one flash memory bank could be used as an upgrade buffer while the other runs a code. This feature allows for swift transition between the applications in each bank, making it ideal for upgrade purposes such as OTA (On-The-Air programming).

Figure 1. A34M420 Block Diagram



Reference Document

- Document 'ID061113' is provided by Arm and contains information on Cortex-M4F.
- A34M420 datasheet is provided by ABOV and available at www.abovsemi.com.

Contents

List of Tables

List of Figures

1. Description

The A34M420 series is a 32-bit microcontroller based on the high-performance Arm Cortex-M4F+ core, with up to 1024 KB of dual bank code flash memory (bank0: 512 KB, bank1: 512 KB), 32 KB of data flash memory, and 64 KB of SRAM. It is a powerful microcontroller with a Floating-Point Unit (FPU) that provides effective solutions for various electrical appliances requiring low power consumption and high performance, such as refrigerators, washing machines, dryers, dishwashers, water purifiers, and blenders. It is also suitable for controlling inverter motor where high-performance is critical.

The A34M420 series also offers bank swap with RWW (Read-While-Write), which allows for simultaneous programming while the system continues to operate. Used in conjunction with dual-bank flash memory, A34M420 series allows code to be executed on one bank while being erased or programmed on the other bank. These functions are effective for implementing features such as OTA (Over-The-Air programming).

1.1 Product Category Definition

Table 1 gives an overview of memory density versus product line.

This document describes the superset of features for each product category.

Refer to Table 2 for the list of features per category.

Table 1. Product Memory Density

Memory Density		Category
Flash	RAM	
1,024 KB	64 KB	A34M420

1.2 Availability of Peripherals

Table 2 summarizes and lists product specific features available in the A34M420 considering the largest package.

Table 2. A34M420 Features and Peripherals

Peripherals		Description
Core	CPU	• Maximum operating frequency: 140 MHz • 32-bit ARM Cortex-M4F core - 32-bit Thumb®-2 instruction set • Register settings in CPU: - General-purpose registers specified - Main stack pointer (MSP) and process stack pointer (PSP): R13 - Link register (LR): R14 - Program counter (PC): R15 • Data ordering format: Little-Endian • Harvard Architecture • AHB / APB
	Interrupt	• NVIC (Nested-Vectored Interrupt Controller) • Up to 86 peripheral interrupts supported • Assignable with 16 different priority levels
	FPU	• Rendered by extending and transforming the ARMv7 floating-point arithmetic functionality • Compliant with the ANSI / IEEE 754 standard • Capable of binary floating-point arithmetic and computation

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
Memory	Code Flash	• Capacity: - A34M420: 1024 KB code flash memory • A high-capacity code flash memory built in • Max 28 MHz Flash access speed • One-word (4-byte) program • Bulk (full-chip), 512-byte, and 2 KB erases • Read protection • Self-programming (Supports updating data in some Code Flash memory region during the execution of user program in the code area.) • CRC code generation and verification for the Flash memory • Dual Bank code flash (bank0: 512 KB, bank1: 512 KB) for OTA (Over-The-Air programming) • Support RWW (Read-While-Write) • Support ECC (Error Correct Code) • Endurance: 10,000 cycles • Lifetime: 10 years
	Data Flash	• Capacity: 32 KB • Max 28 MHz access speed • One-word (4-byte) program • 512 bytes and 2 KB erases • Bulk (full-chip) erase • CRC code generation and verification for the Flash memory • Support ECC (Error Correct Code) • Endurance: 100,000 Cycles • Lifetime: 10 years
	BootROM	• Entering the boot mode by setting the nBOOT pin to low-level duringt a reset procedure. • UART boot mode • In-System Programming • The boot mode can program the internal flash memory via UART.
	SRAM	• Capacity: 64 KB • Usable as a program’s work area • High-speed execution enables the execution of time-critical codes. • Part of the SRAM can be remapped into an interrupt vector area

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
System Control Unit (SCU)	Clock	• High-speed internal oscillator (HSI) • 32 MHz - ±1.2% @ +25°C - ±3.0% @ −40°C to +85°C • Low-speed internal oscillator (LSI) - 500 kHz (±20% at −40°C to +85°C) • External main oscillator (HSE): 4 MHz to 16 MHz • External sub-oscillator (LSE): 32.768 kHz • Phase-locked loop (PLL) frequency generator generates a high-speed clock (up to 140 MHz)
	Clock Monitoring	• System Fail-Safe function by Clock Monitoring • External main oscillator (HSE) • External sub oscillator (LSE) • Main system clock (MCLK)
	Operating Mode	• Run mode • Sleep mode • Deep sleep (STOP1, STOP2) mode
	Reset	• nRESET pin reset • Core reset • Software reset • POR (Power-On Reset) • LVR (Low-Voltage Reset) • WDTR (WatchDog Timer Reset) • Reset due to clock oscillating error
	VDC	• Low-DropOut (LDO) regulator built in for low-voltage operation
	POR	• The POR generator detects an internal 1.4 V voltage and generates a reset signal.
	LVI	• 16 low-voltage detection levels • Supports LVD interrupt • Supports wake-up from SLEEP mode

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
System Control Unit (SCU)	Wake-up	• Wake-up by a general-purpose input/output (GPIO) pin • Wake-up by a free-run timer (FRT) • Wake-up by a watchdog timer (WDT) • Wake-up by a low-voltage indicator (LVI)
General-Purpose I/O (GPIO)		• Input/output (I/O) port for general purposes • 120-LQFP - I/O pins: 107 • 100-LQFP - I/O pins: 89 • 64-LQFP - I/O pins: 51 • Each pin can be set for one of the following modes: - Push-pull output - Open drain output - Input • The use of each pin can be set by setting the mux • Each pin can be configured as an external interrupt source, either the high-level / low-level interrupt or the rising-edge / falling-edge interrupt • Pull-up or pull-down resistor, and debouncing can be set for each pin • Drive strength can be adjusted for each port pin • Each pin bit can be individually set / reset • Wake-up events triggered by external asynchronous inputs
Direct Memory Access Controller (DMA)		• 16-ch direct memory access (DMA) support peripherals • 8-bit / 16-bit / 32-bit data transfers • Compatible with 12 different types of peripherals - SPIn (SPI0, SPI1, SPI2) - UARTn (UART0, UART1, UART2, UART3, UART4, UART5) - CRC - ADC (ADC0, ADC1, ADC2)

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
TIMER	16-bit timer	• General-purpose 16-bit up-count timer • 10 channels - 10 timer n capture port (TnIO) input channels - 10 timer n output port (TnIO) output channels • Timer operating modes - Periodic timer mode - One-shot mode - PWM mode - Capture mode • Interrupt events - Timer/counter match interrupt - Timer overflow interrupt • Input clock selection - Clocks are freely selectable through the miscellaneous clock control registers (SCU_MCCRx) - External clocks are selectable • Timer signals can be generated through TnIO pins • 10-bit prescaler
	WDT	• 32-bit down-count timer • Reset and periodic interrupts • Clocks are freely selectable through the miscellaneous clock control registers (SCU_MCCRx) • Eight different prescalers are selectable
	FRT	• Two 32-bit Free-Run Timer - Capable of calculating the internal system time - 32-bit up-count timer • Interrupt events - Period interrupt - Overflow interrupt

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
Serial Interface	UART	• A total of six 16450 asynchronous serial communication ports • Configurable standard asynchronous communication bits (start, stop, and parity) • Flexible communication available through programming - 5- to 8-bit data transfers - Even / Odd / Non-parity generation and checking - 1-bit, 1.5-bit, or 2-bit stop bit generation and checking • 8-bit fraction controller and 16-bit baud rate generator
	SPI	• Three synchronous serial communication port channels • Master/slave operation • Loop-back mode • Programmable and flexible communication • 8-bit / 9-bit / 16-bit / 17-bit data transmit / receive - SPI clock speed - Both least significant bit (LSB)-first and most significant bit (MSB)-first modes available
	I2C	• Standard I2C communication protocol • Two channels supported • Master and slave modes supported for each channel • 7-bit addressing supported for slave mode • SCL signal's high/low periods and SDA signal's hold time settable
Motor Pulse-Width Modulation	MPWM	• Two Motor PWM generators • Six channels (high and low signals of phases U, V, and W) generate different waveforms • 16-bit up / down counters • Six ADC trigger sources • Interrupt events - Bottom interrupts - Top (period) interrupts - Interval interrupt mode

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
Motor Pulse-Width Modulation	MPWM	• Falling / rising dead-time applicable • A special operating mode: • Phases U, V, and W are independently controlled - Different carrier counters running for phases U, V, and W - Different duty and periods configurable for phases U, V, and W - Different interrupts used for phases U, V, and W • Capture functionality • Protection and over-voltage detection supported
12-bit A/D Converter	ADC	• Three independent ADC blocks • 24 analog input channels • A number of operating modes: - Single conversion - Sequence conversion - Burst conversion - Multiple conversion • Up to eight sequential conversions supported • Software triggers supported • Three internal trigger sources (MPWM and timers) supported • Sample time and hold time are adjustable
Random Number Generator	RNG	• Random-number generator • Interrupt events • Generator ready interrupt • Error interrupt

Table 3. A34M420 Features and Peripherals (continued)

Peripherals		Description
Cyclic Redundancy Check	CRC	• CRC operating modes: - CRC-32 (0x04C1_1DB7) - CRC-16-IBM (0x8005) - CRC-8 (0x07) - CRC-7 (0x09) • Input/output data reversion supported • Compatible with DMA
Operating Voltage		• 2.5 V to 5.5 V
Operating Temperature		• Commercial grade (−40°C to +85°C)
Package		• Three types of package options - 120-pin LQFP - 100-pin LQFP - 64-pin LQFP

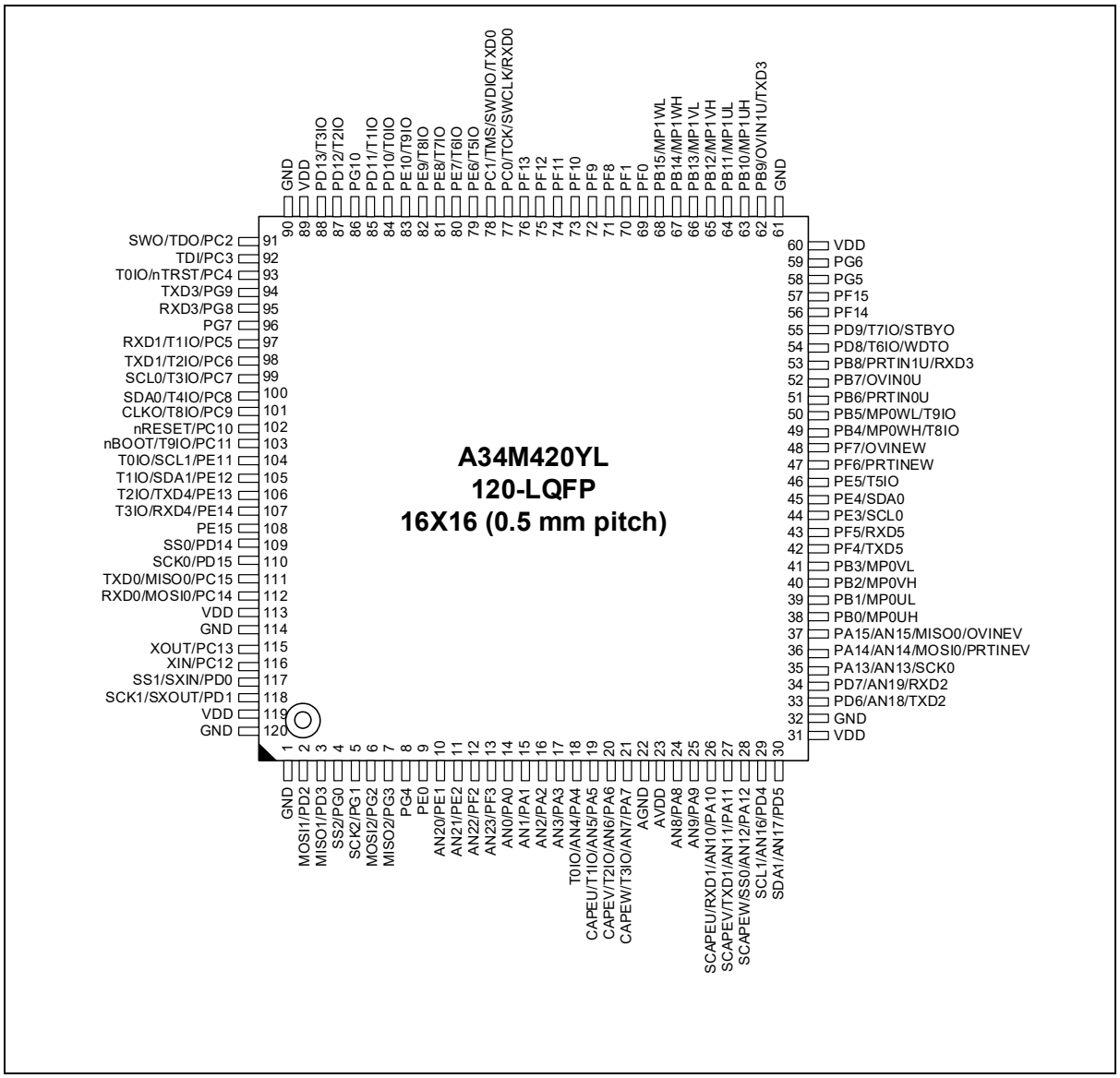
2. Pinouts and Pin Descriptions

In this chapter, pinouts and pin descriptions of the product are described.

2.1 Pinouts

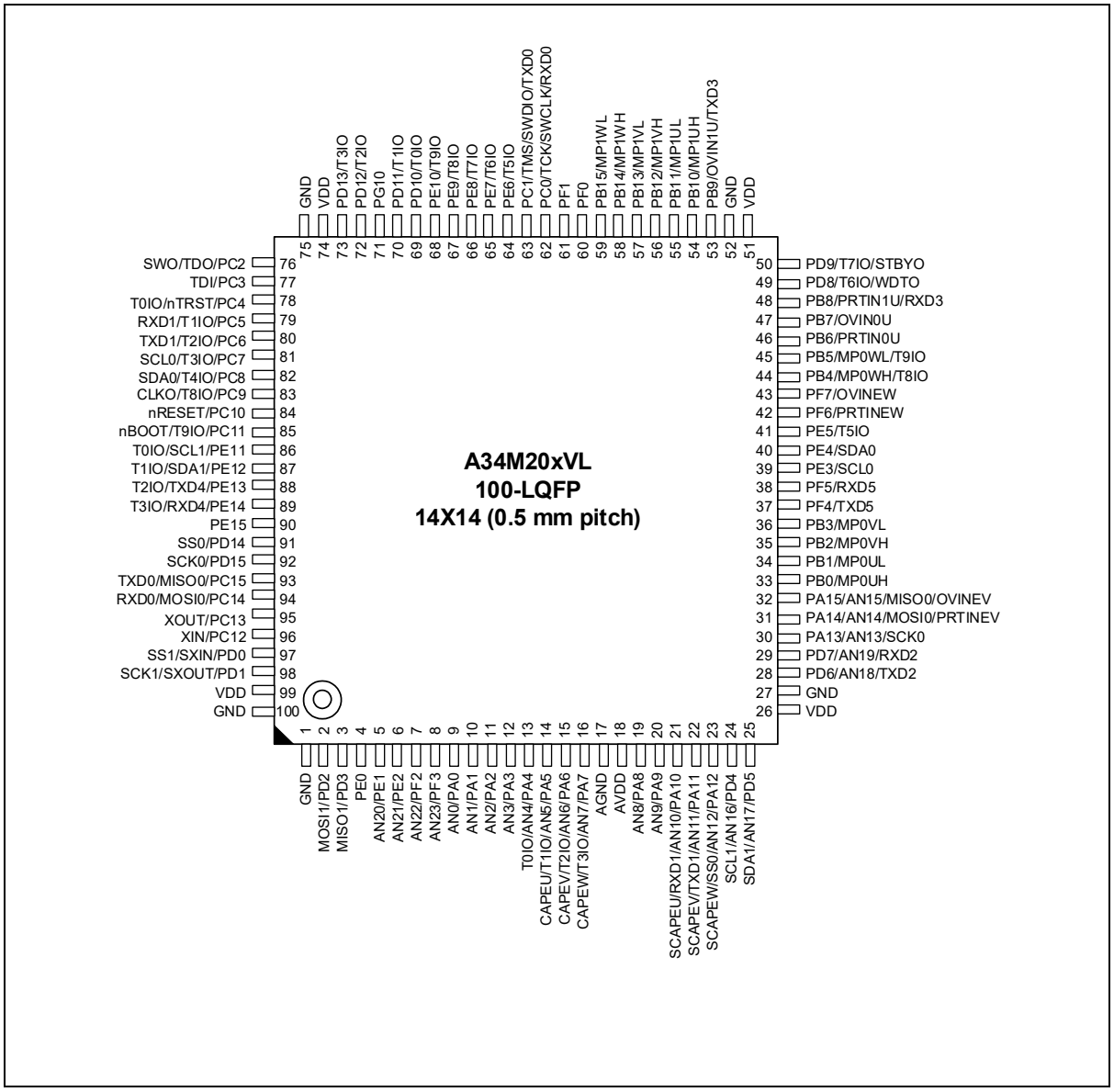
2.1.1 A34M420YL (120-LQFP)

Figure 2. 120-LQFP Pinouts



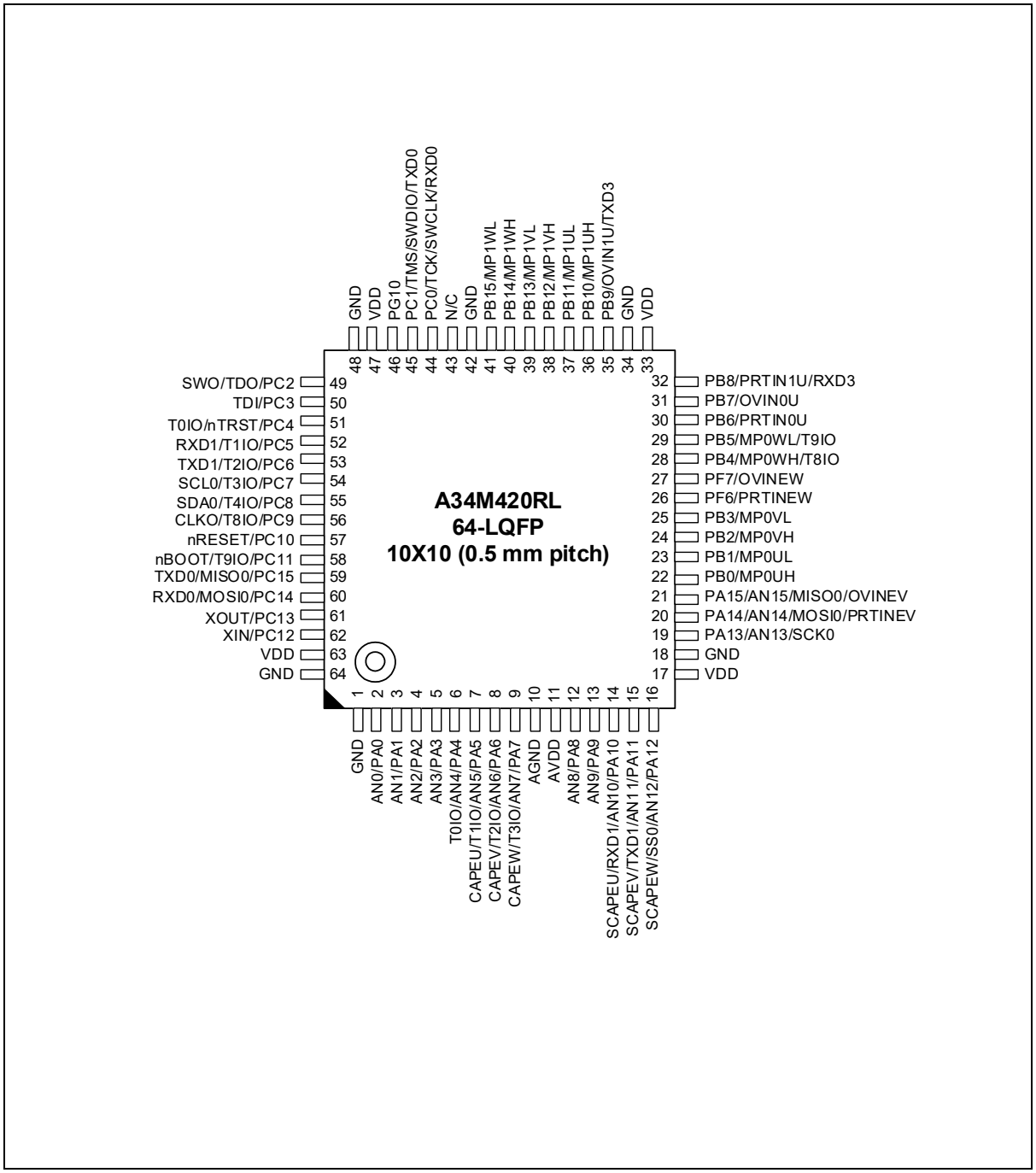
2.1.2 A34M420VL (100-LQFP)

Figure 3. 100-LQFP Pinouts



2.1.3 A34M420RL (64-LQFP)

Figure 4. 64-LQFP Pinouts



2.2 Pin Description

Pin configuration information in Table 3 contains two pairs of power, ground, and other dedicated pins. These multi-function pins have up to six selections of functions including GPIO. Configuration including pin ordering can be changed without notice.

Table 3. Pin Description

Pin No.			Pin Name	Type	Description	Remark
120-pin	100-pin	64-pin				
1	1	1	GND	P	Ground	
2	2	-	PD2	IOUS	PORT D bit 2 input/output	
			MOSI1	IO	SPI channel 1 MOSI (Master out / Slave in) signal	
3	3	-	PD3	IOUS	PORT D bit 3 input/output	
			MISO1	IO	SPI channel 1 MISO (Master in / Slave out) signal	
4	-	-	PG0	IOUS	PORT G bit 0 input/output	
			SS2	IO	SPI channel 2 select signal input/output	
5	-	-	PG1	IOUS	PORT G bit 1 input/output	
			SCK2	IO	SPI channel 2 clock signal input/output	
6	-	-	PG2	IOUS	PORT G bit 2 input/output	
			MOSI2	IO	SPI channel 2 MOSI (Master out / Slave in) signal	
7	-	-	PG3	IOUS	PORT G bit 3 input/output	
			MISO2	IO	SPI channel 2 MISO (Master in / Slave out) signal	
8	-	-	PG4	IOUS	PORT G bit 4 input/output	
9	4	-	PE0	IOUS	PORT E bit 0 input/output	
10	5	-	PE1	IOUS	PORT E bit 1 input/output	
			AN20(2)	IA	Analog input 20	
11	6	-	PE2	IOUS	PORT E bit 2 input/output	
			AN21(2)	IA	Analog input 21	
12	7	-	PF2	IOUS	PORT F bit 2 input/output	
			AN22(2)	IA	Analog input 22	
13	8	-	PF3	IOUS	PORT F bit 3 input/output	
			AN23(2)	IA	Analog input 23	
14	9	2	PA0	IOUS	PORT A bit 0 input/output	
			AN0(2)	IA	Analog input 0	
15	10	3	PA1	IOUS	PORT A bit 1 input/output	
			AN1(2)	IA	Analog input 1	
16	11	4	PA2	IOUS	PORT A bit 1 input/output	
			AN2(2)	IA	Analog input 1	
17	12	5	PA3	IOUS	PORT A bit 3 input/output	
			AN3(2)	IA	Analog input 3	
18	13	6	PA4	IOUS	PORT A bit 4 input/output	
			AN4(2)	IA	Analog input 4	
			T0IO	IO	Timer 0 input/output	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-pin	100-pin	64-pin				
19	14	7	PA5	IOUS	PORT A bit 5 input/output	
			AN5[2]	IA	Analog input 5	
			T1IO	IO	Timer 1 input/output	
			CAPEU	Input	Individual PWM capture U phase	
20	15	8	PA6	IOUS	PORT A bit 6 input/output	
			AN6[2]	IA	Analog input 6	
			T2IO	IO	Timer 2 input/output	
			CAPEV	Input	Individual PWM capture V phase	
21	16	9	PA7	IOUS	PORT A bit 7 input/output	
			AN7[2]	IA	Analog input 7	
			T3IO	IO	Timer 3 input/output	
			CAPEW	Input	Individual PWM capture W phase	
22	17	10	AGND	P	Analog Ground	
23	18	11	AVDD	P	Analog VDD	
24	19	12	PA8	IOUS	PORT A bit 8 input/output	
			AN8[2]	IA	Analog input 8	
25	20	13	PA9	IOUS	PORT A bit 9 input/output	
			AN9[2]	IA	Analog input 9	
26	21	14	PA10	IOUS	PORT A bit 10 input/output	
			AN10[2]	IA	Analog input 10	
			RXD1	Input	UART channel 1 RXD input	
			SCAPEU	Input	Individual PWM sub capture U phase	
27	22	15	PA11	IOUS	PORT A bit 11 input/output	
			AN11[2]	IA	Analog input 11	
			TXD1	Output	UART channel 1 TXD output	
			SCAPEV	Input	Individual PWM sub capture V phase	
28	23	16	PA12	IOUS	PORT A bit 12 input/output	
			AN12[2]	IA	Analog input 12	
			SS0	IO	SPI channel 0 select signal input/output	
			SCAPEW	Input	Individual PWM sub capture W phase	
29	24	-	PD4	IOUS	PORT D bit 4 input/output	
			AN16[2]	IA	Analog input 16	
			SCL1[9]	IO	I2C channel 1 output	Open-drain
30	25	-	PD5	IOUS	PORT D bit 5 input/output	
			AN17[2]	IA	Analog input 17	
			SDA1[9]	IO	I2C channel 1 SDA input/output	Open-drain
31	26	17	VDD	P	VDD	
32	27	18	GND	P	Ground	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-pin	100-pin	64-pin				
33	28	-	PD6	IOUS	PORT D bit 6 input/output	
			AN18[2]	IA	Analog input 18	
			TXD2	Output	UART channel 2 TXD output	
34	29	-	PD7	IOUS	PORT D bit 7 input/output	
			AN19[2]	IA	Analog input 19	
			RXD2	Input	UART channel 2 RXD input	
35	30	19	PA13	IOUS	PORT A bit 13 input/output	
			AN13[2]	IA	Analog input 13	
			SCK0	IO	SPI channel 0 clock input/output	
36	31	20	PA14	IOUS	PORT A bit 14 input/output	
			AN14[2]	IA	Analog input 14	
			MOSI0	IO	SPI channel 0 MOSI (Master put / Slave in signal)	
			PRTINEV	Input	Individual PWM phase V protection input	
37	32	21	PA15	IOUS	PORT A bit 15 input/output	
			AN15[2]	IA	Analog input 15	
			MISO0	IO	SPI channel 0 MISO (Master in / Slave out signal)	
			OVINEV	Input	Individual PWM phase V over-voltage input	
38	33	22	PB0	IOUS	PORT B bit 0 input/output	
			MP0UH	Output	PWM0 UH output	
39	34	23	PB1	IOUS	PORT B bit 1 input/output	
			MP0UL	Output	PWM channel 0 UL output	
40	35	24	PB2	IOUS	PORT B bit 2 input/output	
			MP0VH	Output	PWM channel 0 VH output	
41	36	25	PB3	IOUS	PORT B bit 3 input/output	
			MP0VL	Output	PWM channel 0 VL output	
42	37	-	PF4	IOUS	PORT F bit 4 input/output	
			TXD5	Output	UART channel 5 TXD output	
43	38	-	PF5	IOUS	PORT F bit 5 input/output	
			RXD5	Input	UART channel 5 RXD input	
44	39	-	PE3	IOUS	PORT E bit 3 input/output	
			SCL0[9]	IO	I2C channel 0 output	Open-drain
45	40	-	PE4	IOUS	PORT E bit 4 input/output	
			SDA0[9]	IO	I2C channel 0 SDA input/output	Open-drain
46	41	-	PE5	IOUS	PORT E bit 5 input/output	
			T5IO	IO	Timer 5 input/output	
47	42	26	PF6	IOUS	PORT F bit 6 input/output	
			PRTINEW	Input	Individual PWM phase W protection input	
48	43	27	PF7	IOUS	PORT F bit 7 input/output	
			OVINEW	Input	Individual PWM phase W over-voltage input	
49	44	28	PB4	IOUS	PORT B bit 4 input/output	
			MP0WH	Output	PWM channel 0 WH output	
			T8IO	IO	Timer 8 input/output	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type(1)	Description	Remark
120-pin	100-pin	64-pin				
50	45	29	PB5	IOUS	PORT B bit 5 input/output	
			MP0WL	Output	PWM channel 0 WL output	
			T9IO	IO	Timer 9 input/output	
51	46	30	PB6	IOUS	PORT B bit 6 input/output	
			PRTIN0U	Input	PWM0 Protection input Signal Individual PWM 0 Phase U Protection input	
52	47	31	PB7	IOUS	PORT B bit 7 input/output	
			OVIN0U	Input	PWM0 Over-voltage input Signal Individual PWM 0 Phase U Over voltage input	
53	48	32	PB8	IOUS	PORT B bit 8 input/output	
			PRTIN1U	Input	PWM1 Protection input Signal Individual PWM 1 Phase U Protection input	
			RXD3	Input	UART channel 3 RXD input	
54	49	-	PD8	IOUS	PORT D bit 8 input/output	
			T6IO	IO	Timer 6 input/output	
			WDTO	Output	WDT output	
55	50	-	PD9	IOUS	PORT D bit 9 input/output	
			T7IO	IO	Timer 7 input/output	
			STBYO	Output	Power-down Mode Indication Signal	
56	-	-	PF14	IOUS	PORT F bit 14 input/output	
57	-	-	PF15	IOUS	PORT F bit 15 input/output	
58	-	-	PG5	IOUS	PORT G bit 5 input/output	
59	-	-	PG6	IOUS	PORT G bit 6 input/output	
60	51	33	VDD	P	VDD	
61	52	34	GND	P	Ground	
62	53	35	PB9	IOUS	PORT B bit 9 input/output	
			OVIN1U	Input	PWM1 Overvoltage input Signal Individual PWM 1 Phase U Over voltage input	
			TXD3	Output	UART channel 3 TXD output	
63	54	36	PB10	IOUS	PORT B bit 10 input/output	
			MP1UH	Output	PWM channel 1 UH output	
64	55	37	PB11	IOUS	PORT B bit 11 input/output	
			MP1UL	Output	PWM channel 1 UL output	
65	56	38	PB12	IOUS	PORT B bit 12 input/output	
			MP1VH	Output	PWM channel 1 VH output	
66	57	39	PB13	IOUS	PORT B bit 13 input/output	
			MP1VL	Output	PWM channel 1 VL output	
67	58	40	PB14	IOUS	PORT B bit 14 input/output	
			MP1WH	Output	PWM channel 1 WH output	
68	59	41	PB15	IOUS	PORT B bit 15 input/output	
			MP1WL	Output	PWM channel 1 WL output	
69	60	-	PF0	IOUS	PORT F bit 0 input/output	
70	61	-	PF1	IOUS	PORT F bit 1 input/output	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-pin	100-pin	64-pin				
71	-	-	PF8	IOUS	PORT F bit 8 input/output	
72	-	-	PF9	IOUS	PORT F bit 9 input/output	
73	-	-	PF10	IOUS	PORT F bit 10 input/output	
74	-	-	PF11	IOUS	PORT F bit 11 input/output	
75	-	-	PF12	IOUS	PORT F bit 12 input/output	
76	-	-	PF13	IOUS	PORT F bit 13 input/output	
-	-	42	GND	P	Ground	
-	-	43	N/C	-	N/C	
77	62	44	PC0	IOUS	PORT C bit 0 input/output	
			TCK / SWCLK[2][3][6][7]	Input	JTAG TCK, SWD clock input	Pull-up
			RXD0	Input	UART channel 0 RXD input	
78	63	45	PC1	IOUS	PORT C bit 1 input/output	
			TMS / SWDIO[2][3][6][7]	IO	JTAG TMS, SWD data input/output	Pull-up
			TXD0	Output	UART channel 0 TXD output	
79	64	-	PE6	IOUS	PORT E bit 6 input/output	
			T5IO	IO	Timer 5 input/output	
80	65	-	PE7	IOUS	PORT E bit 7 input/output	
			T6IO	IO	Timer 6 input/output	
81	66	-	PE8	IOUS	PORT E bit 8 input/output	
			T7IO	IO	Timer 7 input/output	
82	67	-	PE9	IOUS	PORT E bit 9 input/output	
			T8IO	IO	Timer 8 input/output	
83	68	-	PE10	IOUS	PORT E bit 10 input/output	
			T9IO	IO	Timer 9 input/output	
84	69	-	PD10	IOUS	PORT D bit 10 input/output	
			T0IO	IO	Timer 0 input/output	
85	70	-	PD11	IOUS	PORT D bit 11 input/output	
			T1IO	IO	Timer 1 input/output	
86	71	46	PG10	IOUS	PORT G bit 10 input/output	
87	72	-	PD12	IOUS	PORT D bit 12 input/output	
			T2IO	IO	Timer 2 input/output	
88	73	-	PD13	IOUS	PORT D bit 13 input/output	
			T3IO	IO	Timer 3 input/output	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-pin	100-pin	64-pin				
89	74	47	VDD	P	VDD	
90	75	48	GND	P	Ground	
91	76	49	PC2	IOUS	PORT C bit 2 input/output	
			TDO / SWO[2]	Output	JTAG TDO output, Serial Wire Output	Output Low
92	77	50	PC3	IOUS	PORT C bit 3 input/output	
			TDI[2]	Input	JTAG TDI input	Pull-up
93	78	51	PC4	IOUS	PORT C bit 4 input/output	
			nTRST[2]	Input	JTAG nTRST input	Pull-up
			T0IO	IO	Timer 0 input/output	
94	-	-	PG9	IOUS	PORT G bit 9 input/output	
			TXD3	Output	UART channel 3 TXD output	
95	-	-	PG8	IOUS	PORT G bit 8 input/output	
			RXD3	Input	UART channel 3 RXD input	
96	-	-	PG7	IOUS	PORT G bit 7 input/output	
97	79	52	PC5	IOUS	PORT C bit 5 input/output	
			T1IO	IO	Timer 1 input/output	
			RXD1	Input	UART channel 1 RXD input	
98	80	53	PC6	IOUS	PORT C bit 6 input/output	
			T2IO	IO	Timer 2 input/output	
			TXD1	Output	UART channel 1 TXD output	
99	81	54	PC7	IOUS	PORT C bit 7 input/output	
			T3IO	IO	Timer 3 input/output	
			SCL0[9]	IO	I2C channel 0 output	Open-drain
100	82	55	PC8	IOUS	PORT C bit 8 input/output	
			T4IO	IO	Timer 4 input/output	
			SDA0[9]	IO	I2C channel 0 SDA input/output	Open-drain
101	83	56	PC9	IOUS	PORT C bit 9 input/output	
			T8IO	IO	Timer 8 input/output	
			CLKO	Output	System Clock output	
102	84	57	PC10	IOUS	PORT C bit 10 input/output	
			nRESET[2][3]	Input	External Reset input	Pull-up
103	85	58	PC11	IOUS	PORT C bit 11 input/output	
			T9IO	IO	Timer 9 input/output	
			nBOOT[2][3][5]	Input	Boot Mode Selection input	Pull-up
104	86	-	PE11	IOUS	PORT E bit 11 input/output	
			SCL1[9]	IO	I2C channel 1 output	Open-drain
			T0IO	IO	Timer 0 input/output	
105	87	-	PE12	IOUS	PORT E bit 12 input/output	
			SDA1[9]	IO	I2C channel 1 SDA input/output	Open-drain
			T1IO	IO	Timer 1 input/output	
106	88	-	PE13	IOUS	PORT E bit 13 input/output	
			TXD4	Output	UART channel 4 TXD output	
			T2IO	IO	Timer 2 input/output	

Table 4. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-pin	100-pin	64-pin				
107	89	-	PE14	IOUS	PORT E bit 14 input/output	
			RXD4	Input	UART channel 4 RXD input	
			T3IO	IO	Timer 3 input/output	
108	90	-	PE15	IOUS	PORT E bit 15 input/output	
109	91	-	PD14	IOUS	PORT D bit 14 input/output	
			SS0	IO	SPI channel 0 select signal input/output	
110	92	-	PD15	IOUS	PORT D bit 15 input/output	
			SCK0	IO	SPI channel 0 clock input/output	
111	93	59	PC15	IOUS	PORT C bit 15 input/output	
			MISO0	IO	SPI channel 0 MISO (Master in / Slave out) signal	
			TXD0	Output	UART channel 0 TXD output	
112	94	60	PC14	IOUS	PORT C bit 14 input/output	
			MOSI0	IO	SPI channel 0 MOSI (Master out / Slave in) signal	
			RXD0	Input	UART channel 0 RXD input	
113	-	-	VDD	P	VDD	
114	-	-	GND	P	Ground	
115	95	61	PC13	IOUS	PORT C bit 13 input/output	
			XOUT[2][9]	OA	External crystal oscillator output	
116	96	62	PC12	IOUS	PORT C bit 12 input/output	
			XIN[2][9]	IA	External crystal oscillator input	
117	97	-	PD0	IOUS	PORT D bit 0 input/output	
			SXIN[2][9]	IA	Sub crystal oscillator input	
			SS1	IO	SPI channel 1 select signal input/output	
118	98	-	PD1	IOUS	PORT D bit 1 input/output	
			SXOUT[2][9]	OA	Sub crystal oscillator output	
			SCK1	IO	SPI channel 1 clock input/output	
119	99	63	VDD	P	VDD	
120	100	64	GND	P	Ground	

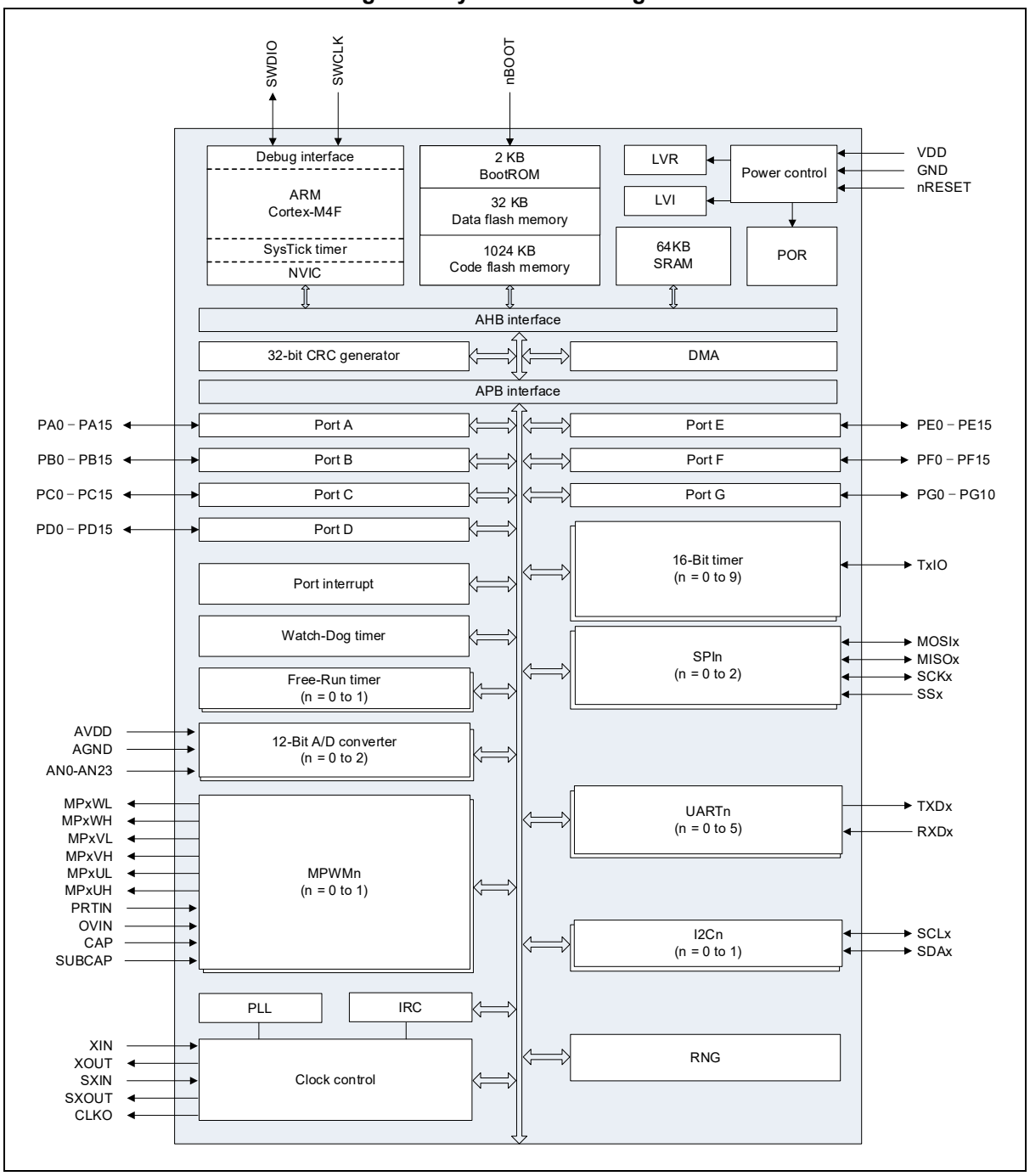
NOTES:

1. I = Input, O = Output, U = Pull-up, D = Pull-down, S = Schmitt-Trigger Input Type, C = CMOS Input Type, A = Analog, P = Power
2. After a reset, all the pins are configured to function defined by their initial values. The initial value of the pin depends on the package type. This configuration is in compliance with the 120-pin standard.
3. nBOOT, nRESET, PC1 (SWDIO), PC0 (SWCLK), PC3 and PC4 are the default pull-up pins.
4. Do not configure unused pins as floating inputs.
5. After a reset, the internal pull-up for the boot pin is enabled.
6. After a reset, the internal pull-up for the serial wire clock (SWCLK) and the serial wire data I/O (SWDIO) is enabled.
7. The SWCLK and SWDIO pins should not be switched to other functions while they are being used.
8. PC7, PC8, PD4, PD5, PE3, PE4, PE11, and PE12 pins are configured as open-drain ports when used as I2C ports.
9. When the PC12 (XIN), PC13 (XOUT), PD1 (SXOUT), and PD0 (SXIN) pins are configured for a function other than a clock, and if the clock is enabled by software, the other functions may not operate normally.

3. System and Memory Overview

3.1 System Architecture

Figure 5. System Block Diagram



3.1.1 Cortex-M4F Core

The Arm® Cortex-M4F has the same functions as the Cortex-M4 and includes optional floating point arithmetic functionality. The two processors are intended for deeply embedded applications that require fast interrupt response features.

3.1.2 Floating Point Unit (FPU)

Cortex-M4 FPU is an implementation of the single precision variant of the ARMv7-M Floating-Point Extension (FPv4-SP). It provides floating-point computation functionality that is compliant with the ANSI / IEEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic, referred to as the IEEE 754 standard. The FPU supports all single-precision data-processing instructions and data types described in the ARMv7-M Architecture Reference Manual.

3.1.3 Interrupt Controller

Table 4. Interrupt Vector Map

Priority	Vector Address	Interrupt Source
-16	0x0000_0000	Stack Pointer
-15	0x0000_0004	Reset Handler
-14	0x0000_0008	NMI Handler
-13	0x0000_000C	Hard Fault Handler
-12	0x0000_0010	MemManage Handler
-11	0x0000_0014	BusFault Handler
-10	0x0000_0018	UsageFault Handler
-9	0x0000_001C	Reserved
-8	0x0000_0020	
-7	0x0000_0024	
-6	0x0000_0028	
-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	LVI
1	0x0000_0044	Reserved
2	0x0000_0048	HSEFAIL
3	0x0000_004C	LSEFAIL
4	0x0000_0050	Reserved
5	0x0000_0054	
6	0x0000_0058	WDT

Table 5. Interrupt vector map (continued)

Priority	Vector Address	Interrupt Source
7	0x0000_005C	Reserved
8	0x0000_0060	FRT0
9	0x0000_0064	FRT1
10	0x0000_0068	Reserved
11	0x0000_006C	CFMC
12	0x0000_0070	DFMC
13	0x0000_0074	Reserved
14	0x0000_0078	
15	0x0000_007C	TIMER0
16	0x0000_0080	TIMER1
17	0x0000_0084	TIMER2
18	0x0000_0088	TIMER3
19	0x0000_008C	TIMER4
20	0x0000_0090	TIMER5
21	0x0000_0094	TIMER6
22	0x0000_0098	TIMER7
23	0x0000_009C	TIMER8
24	0x0000_00A0	TIMER9
25	0x0000_00A4	Reserved
26	0x0000_00A8	
27	0x0000_00AC	RNG
28	0x0000_00B0	Reserved
29	0x0000_00B4	
30	0x0000_00B8	
31	0x0000_00BC	
32	0x0000_00C0	
33	0x0000_00C4	
34	0x0000_00C8	
35	0x0000_00CC	
36	0x0000_00D0	GPIOA
37	0x0000_00D4	GPIOB
38	0x0000_00D8	GPIOC
39	0x0000_00DC	GPIOD
40	0x0000_00E0	GPIOE
41	0x0000_00E4	GPIOF

Table 5. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
42	0x0000_00E8	GPIOG
43	0x0000_00EC	Reserved
44	0x0000_00F0	
45	0x0000_00F4	MPWM0PROT
46	0x0000_00F8	MPWM0OVV
47	0x0000_00FC	MPWM0 (U)
48	0x0000_0100	MPWM0 (V)
49	0x0000_0104	MPWM0 (W)
50	0x0000_0108	MPWM1PROT
51	0x0000_010C	MPWM1OVV
52	0x0000_0110	MPWM1 (U)
53	0x0000_0114	MPWM1 (V)
54	0x0000_0118	MPWM1 (W)
55	0x0000_011C	SPI0
56	0x0000_0120	SPI1
57	0x0000_0124	SPI2
58	0x0000_0128	Reserved
59	0x0000_012C	
60	0x0000_0130	I2C0
61	0x0000_0134	I2C1
62	0x0000_0138	Reserved
63	0x0000_013C	UART0
64	0x0000_0140	UART1
65	0x0000_0144	UART2
66	0x0000_0148	UART3
67	0x0000_014C	UART4
68	0x0000_0150	UART5
69	0x0000_0154	Reserved
70	0x0000_0158	
71	0x0000_015C	
72	0x0000_0160	
73	0x0000_0164	
74	0x0000_0168	ADC0
75	0x0000_016C	ADC1
76	0x0000_0170	ADC2

Table 5. Interrupt Vector Map (continued)

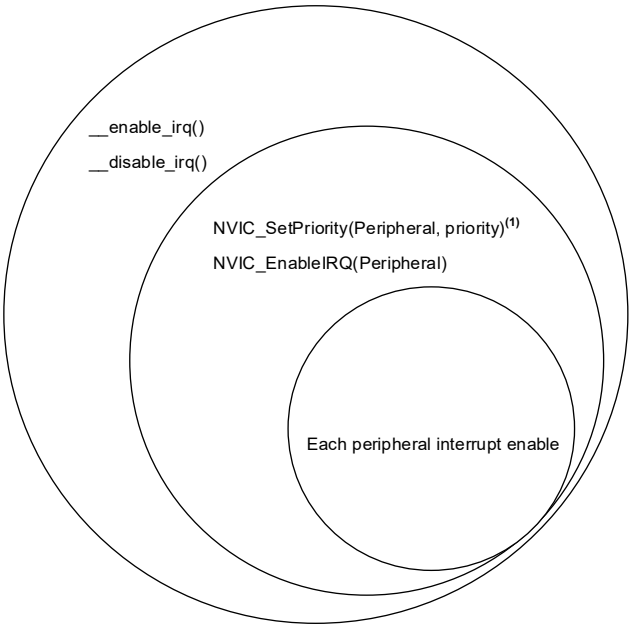
Priority	Vector Address	Interrupt Source
77	0x0000_0174	Reserved
78	0x0000_0178	
79	0x0000_017C	
80	0x0000_0180	
81	0x0000_0184	
82	0x0000_0188	
83	0x0000_018C	
84	0x0000_0190	
85	0x0000_0194	CRC

NOTE:

1. Each interrupt is linked to the priority level registers. Each interrupt priority register is 1-byte length and only the upper 4 bits of it are used for the priority level. NVIC registers of Cortex-M4F processor use four interrupt priority bytes at a time because it supports the transmission in word unit.

** __NVIC_PRIO_BITS = 4

Figure 6. Interrupt Block Diagram



NOTE:

1. Option for interrupt priority

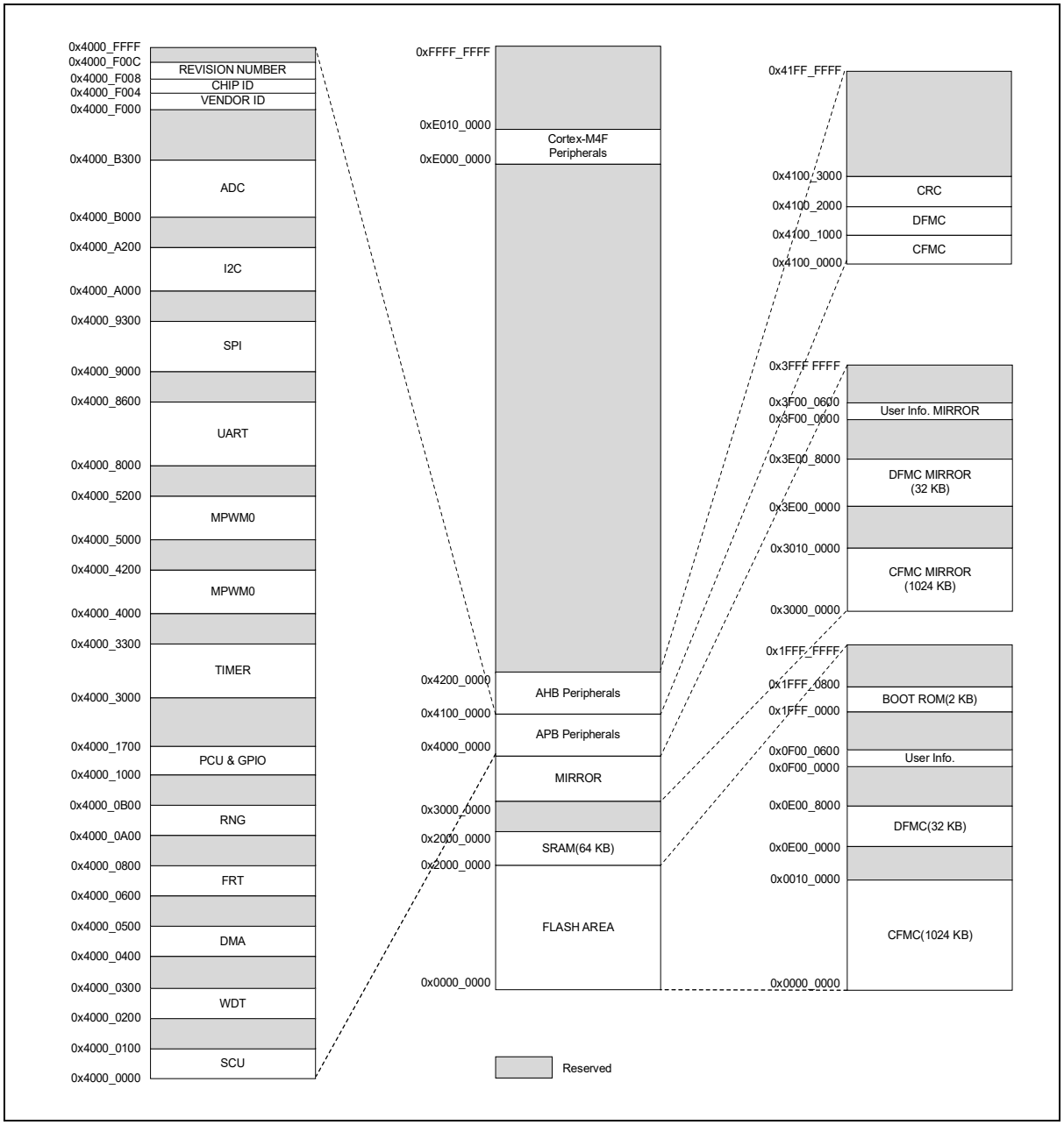
3.2 Memory Organization

3.2.1 Introduction

Program memory, data memory, registers and I/O ports are organized in the same address space. The bytes are coded in memory in Little-Endian format. The lowest numbered byte in a word is considered the word's least significant byte and the highest numbered byte the most significant.

3.2.2 Memory Map and Register Boundary Address

Figure 7. Memory Map



All the memory map areas that are not allocated to on-chip memories and peripherals are considered “Reserved”. For the detailed mapping of available memory and register areas, refer to the following table.

The following Table 5 gives the boundary addresses of the peripherals available in the devices.

Table 5. Peripheral Address

Base Address	Peripheral Name
0x4000_0000	SCU
0x4000_0200	WDT
0x4000_0400	DMA 0/1/2/3/4/5/6/7/8/9/10/11/12/13/14/15
0x4000_0600	FRT 0/1
0x4000_0A00	RNG
0x4000_1000	PCU A/B/C/D/E/F/G
0x4000_3000	Timer 0/1/2/3/4/5/6/7/8/9
0x4000_4000	MPWM0
0x4000_5000	MPWM1
0x4000_8000	UART 0/1/2/3/4/5
0x4000_9000	SPI 0/1/2
0x4000_A000	I2C 0/1
0x4000_B000	ADC 0/1/2
0x4100_0000	CFMC
0x4100_1000	DFMC
0x4100_2000	CRC
0x2000_0000	Internal SRAM

3.3 Memory map

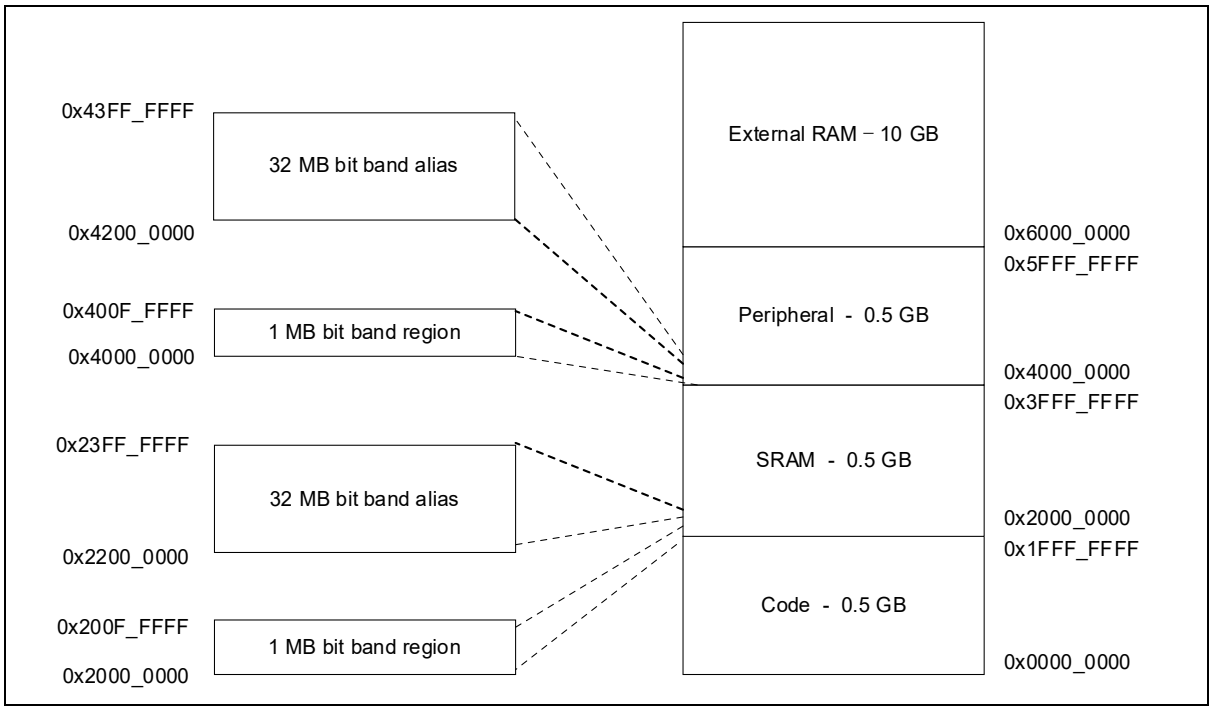
3.3.1 Bit-Banding

The Cortex-M4 with FPU memory map includes two bit-band regions. These regions map each word in an alias region of memory to a bit in a bit-band region of memory. Writing to a word in the alias region has the same effect as a read-modify-write operation on the targeted bit in the bit-band region.

In the A34M420 series devices both the peripheral registers and the SRAM are mapped to a bit-band region, so that single bit-band write and read operations are allowed.

The following is the bit-banding memory map and calculation formula of peripheral and SRAM.

Figure 8. Bit-Banding Memory Map



- Bit_Word_Addr = bit_band_base + (byte_offset × 32) + (bit_number × 4)
  - bit_band_base:
    - Peripheral start address = 0x4200_0000
    - SRAM start address = 0x2200_0000
  - byte_offset:
    - Peripheral = (Peripheral_base_addr - 0x4000_0000 + offset) × 32
    - SRAM = (SRAM_addr - 0x2000_0000 + offset) × 32

3.3.2 Embedded SRAM

The A34M420 series has a block of 0-wait on-chip SRAM. The size of the SRAM is 64 KB and its base address is 0x2000_0000.

This SRAM memory area is usually used for data memory and stack memory. Sometimes the code is dumped into the SRAM memory for fast operation or flash memory erase and programming operations.

3.3.3 Flash Memory Overview

The A34M420 series provides internal 1024 KB code Flash memory and a controller. This is enough to control the general system. Self-programming is available and ISP and SWD programming is also supported in boot mode or in debugging mode.

Instruction and data cache buffer are present and overcome the low bandwidth Flash memory. CPU can access Flash memory with one wait state up to 28 MHz bus frequency.

3.4 Boot Configuration

The A34M420 series has a boot mode option to program internal flash memory. Boot mode can be entered by setting nBOOT pin to ‘L’ at reset timing (Normal state is ‘H’).

Boot mode supports UART boot. The pins for the boot mode are listed in Table 6.

Table 6. Boot Mode Pin List

Block	Pin Name	I/O	Pin Direction	Description
SYSTEM	nRESET	PC10	Input	Reset input signal
	nBOOT	PC11	Input	Boot mode setting pin
UART0	RXD0	PC14	Input	UART boot receive data
	TXD0	PC15	Output	UART boot transmit data

4. System Control Unit (SCU)

4.1 SCU Introduction

The A34M420 series has a built-in intelligent power control block which manages system analog blocks and operating modes. System Control Unit (SCU) module controls an internal reset and clock signals to maintain optimized system performance and power dissipation.

4.2 SCU Main Features

The clock features of A34M420 series are as follows:

- Operating frequency: Up to 140 MHz
- High-Speed Internal Oscillator (HSI): 32 MHz
- Low-Speed Internal Oscillator (LSI): 500 kHz
- High-Speed External Oscillator (HSE): 4 MHz to 16 MHz
- Low-Speed External Oscillator (LSE): 32.768 kHz
- Phase-locked loop (PLL) frequency generator generates a high-speed clock (Up to 140 MHz)

A34M420 series have clock monitoring as a system fail-safe function.

- The reset features of A34M420 series are as follows:
- nRESET pin reset
- CPU reset
- Software reset
- Power-On Reset (POR)
- Low-Voltage Detect Reset (LVR)
- WatchDog Timer Reset (WDT)
- Reset due to clock oscillatiion error

Operating mode features of A34M420 series are as follows:

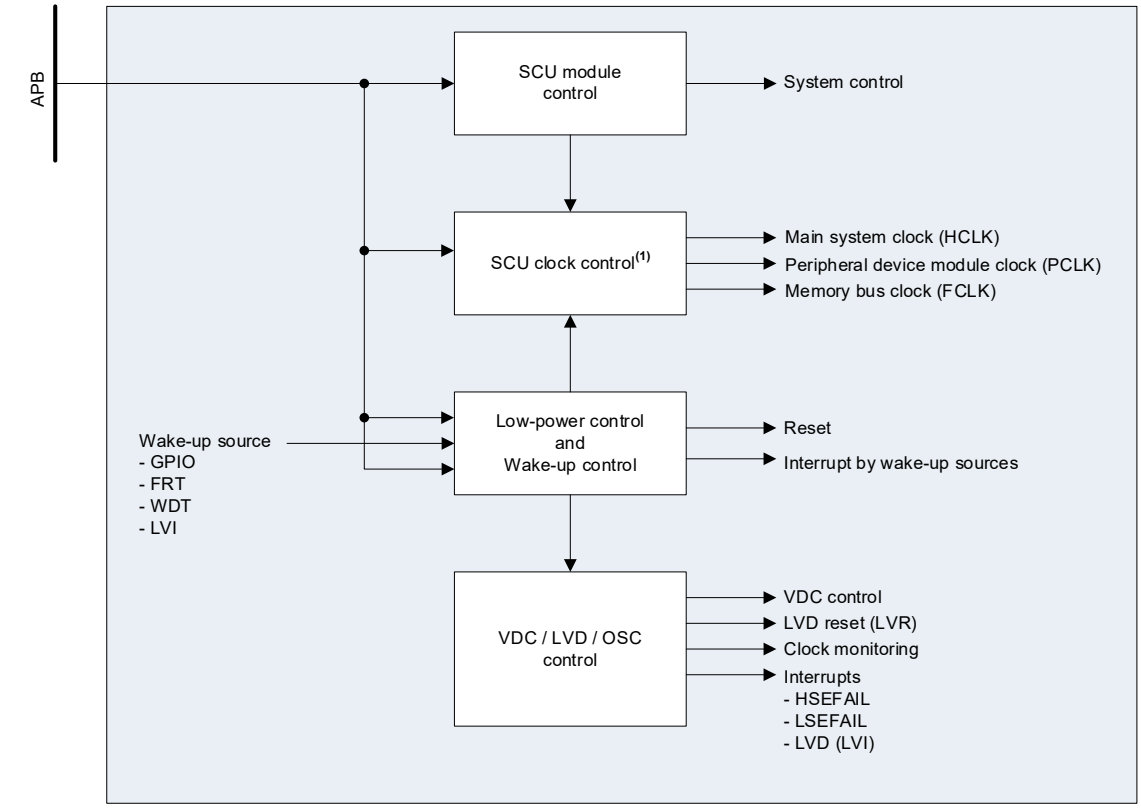
- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP1, STOP2) mode

4.3 SCU Functional Description

4.3.1 SCU Block Diagram

Figure 9 describes the SCU in a block diagram.

Figure 9. SCU Block Diagram



NOTE:

1. For details on the SCU clock control, see section 4.4.2.

4.3.2 SCU Pin and Type

Six pins in Table 7 are assigned for the SCU module.

Table 7. SCU Pins

Pin Name	Type	Description
nRESET	Input	External reset input
XIN / XOUT	OSC	External crystal oscillator (4 MHz to 16 MHz)
SXIN / SXOUT	OSC	External sub-crystal oscillator (32.768 kHz)
CLKO	Output	Clock output monitoring signal

4.4 Clocks

4.4.1 Overview

The A34M420 series has a clock generation circuit. The clock generation circuit consists of external oscillation circuit, PLL circuit, internal oscillation circuit. The overview of each circuit is as follows:

- External oscillation circuit
  - High-Speed External Oscillator (HSE): 4 MHz to 16 MHz
    - Generates main clocks by connecting oscillator to XIN pin and XOUT pin.
    - It is recommended to use 8 MHz.
  - Low-Speed External Oscillator (LSE): 32.768 kHz
    - Generates sub clocks by connecting oscillator to SXIN pin and SXOUT pin.
- PLL circuit
  - Generates clocks by multiplying clocks which are generated in the external oscillation, internal oscillation circuit.
  - Phase-locked loop (PLL) frequency generator generates a high-speed clock (Up to 140 MHz)
- Internal oscillation circuit
  - High-speed internal oscillator (HSI): 32 MHz
  - Low-speed internal oscillator (LSI): 500 kHz

NOTE:

1. After the reset is released, the 4 MHz HSI signal, derived from the 32 MHz HSI signal divided by 8, and a 500 kHz LSI signal are automatically activated as the internal oscillator.

Table 8 describes the clock sources of the A34M420.

Table 8. Clock Sources

Clock Name	Frequency	Description
HSE	4 to 16 MHz	High-speed external oscillator
PLL	8 to 140 MHz	Phase-locked loop
LSE	32.768 kHz	Low-speed external oscillator
HSI	32 MHz	High-speed internal oscillator
LSI	500 kHz	Low-speed internal oscillator

Five different clock sources can be used to drive the system clock:

- Low-Speed Internal oscillator (LSI)
- Low-Speed External oscillator (LSE)
- High-Speed Internal oscillator (HSI)
- High-Speed external oscillator (HSE)
- Phase-Locked Loop (PLL)

Peripheral clock features are as follows:

Table 9 describes the peripheral's clock and the register selection:

Table 9. Peripheral Clock Selection

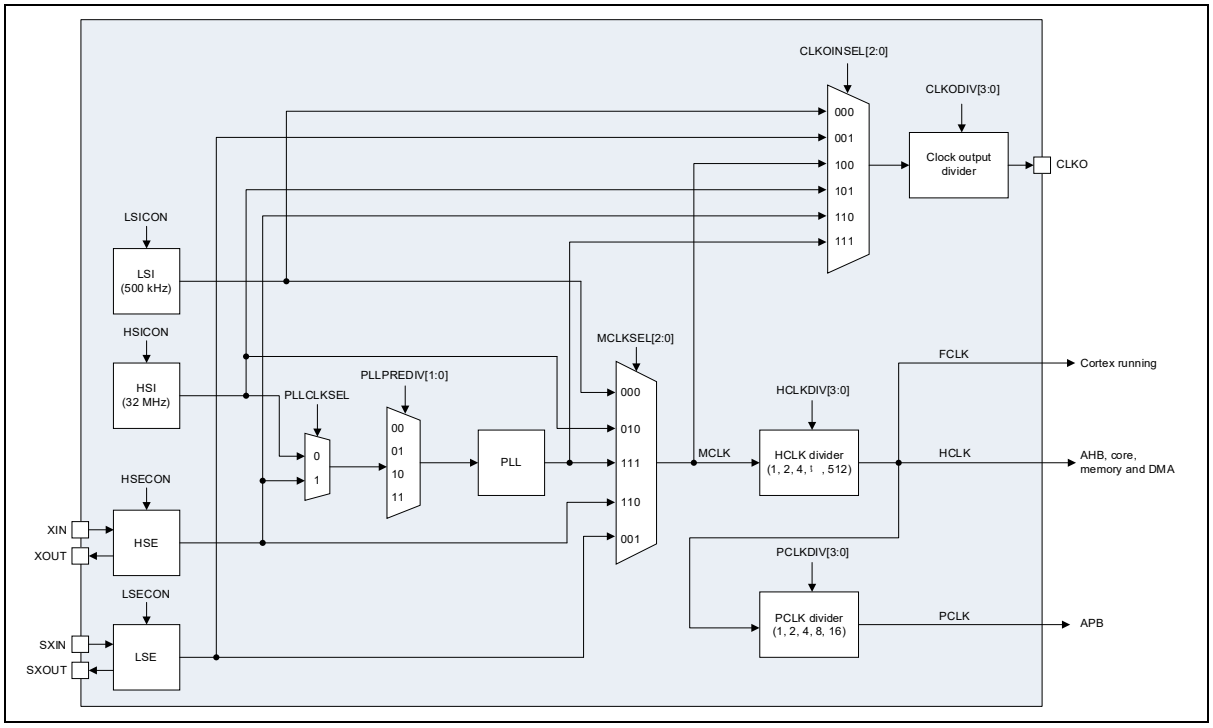
Peripheral	PCLK	Miscellaneous Clocks						
		SCU_MCCRx	MCLK	LSI	LSE	HSI	HSE	PLL
Systick	N/A	SCU_MCCR1	✓	✓	✓	✓	✓	✓
WDT	✓		✓	✓	✓	✓	✓	✓
MPWM0	N/A	SCU_MCCR2	✓	✓	✓	✓	✓	✓
MPWM1	N/A		✓	✓	✓	✓	✓	✓
TIMERn (n = 0 to 4)	✓	SCU_MCCR3	✓	✓	✓	✓	✓	✓
TIMERn (n = 5 to 9)	✓		✓	✓	✓	✓	✓	✓
ADC	✓	SCU_MCCR4	✓	✓	✓	✓	✓	✓
PGAD	N/A		✓	✓	✓	✓	✓	✓
PGBD	N/A	SCU_MCCR5	✓	✓	✓	✓	✓	✓
PGCD	N/A		✓	✓	✓	✓	✓	✓
FRT0	N/A	SCU_MCCR6	✓	✓	✓	✓	✓	✓
FRT1	N/A		✓	✓	✓	✓	✓	✓
UART	N/A	SCU_MCCR7	✓	✓	✓	✓	✓	✓

4.4.2 Clock Tree Configuration

Users can maintain the clock system variation under software control. Figure 10 allows the users to learn about the clock system on the A34M420.

The A34M420 has two operation clocks. One is HCLK, which supplies the clock to the CPU and AHB bus system. The other one is PCLK, which supplies the clock to the peripheral systems. Users can control this clock system variation by configuring internal registers.

Figure 10. Clock Configuration



Each clock multiplexer (selecting a clock source) has a glitch free circuit. So, a clock can be changed without glitch risks. When users change the clock mux control, be sure that both clocks (current clock and new clock) must be operating. If either is not alive, the clock change process will stop, and the system will shut down and will not recover.

Five pins in Table 10 are assigned for the clock pins.

Table 10. Clock Generation Circuit Input/Output Pins

Pin Name	I/O	Description
XIN	Input	External crystal oscillator (4 MHz to 16 MHz)
XOUT	Output	
SXIN	Input	External sub-crystal oscillator (32.768 kHz)
SXOUT	Output	
CLKO	Output	Clock output monitoring signal

4.4.3 HSE Clock

The HSE clock signal can be generated from two clock sources below:

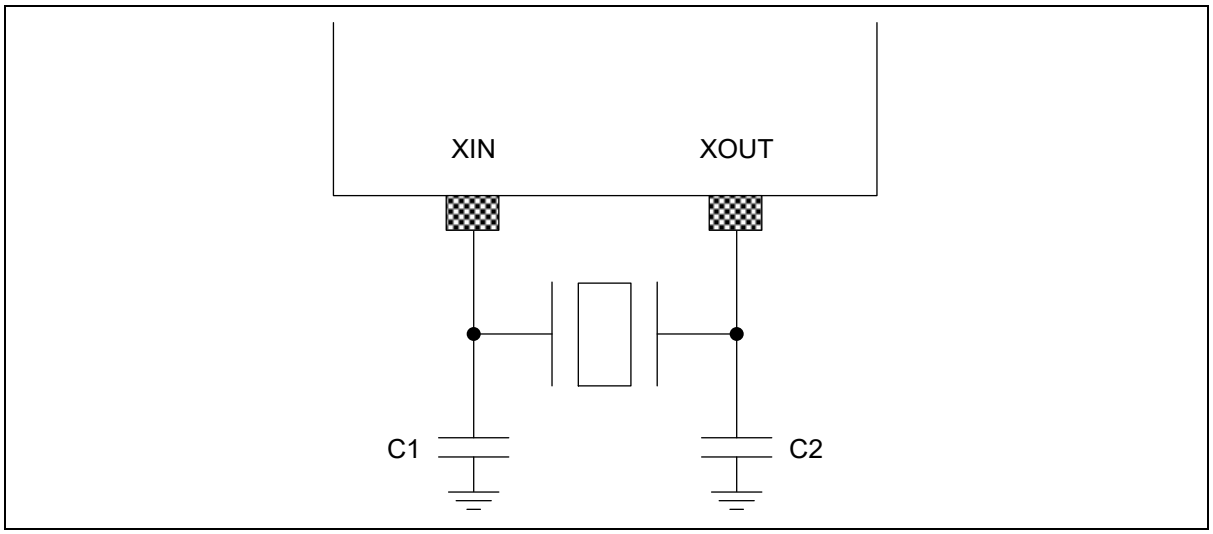
Two pins in Table 11 are assigned for the HSE clock pins block.

- HSE external crystal / ceramic oscillator

Table 11. HSE Clock Pins

Pin Name	I/O	Description
XIN	Input	4 MHz to 16 MHz These pins are used to connect a crystal or ceramic oscillator. For details, refer to Figure 11.
XOUT	Output	

Figure 11. HSE Crystal / Ceramic Oscillator



NOTE:

1. When not using HSE (8 MHz), If the HSECON bit is enabled (SCU_CSCR), it may collide with XOUT (output) and receive internal damage to the microcontroller when used as a GPIO output. When not using an external HSE (8 MHz), enable only the clock source to be used in the SCU_CSCR register.

4.4.4 HSI Clock

The HSI is generated from the internal 32 MHz oscillator.

- HSI: 32 MHz
  - ±1.2% @ 25ºC
  - ±3.0% @ −40ºC to +85ºC

4.4.5 PLL Clock

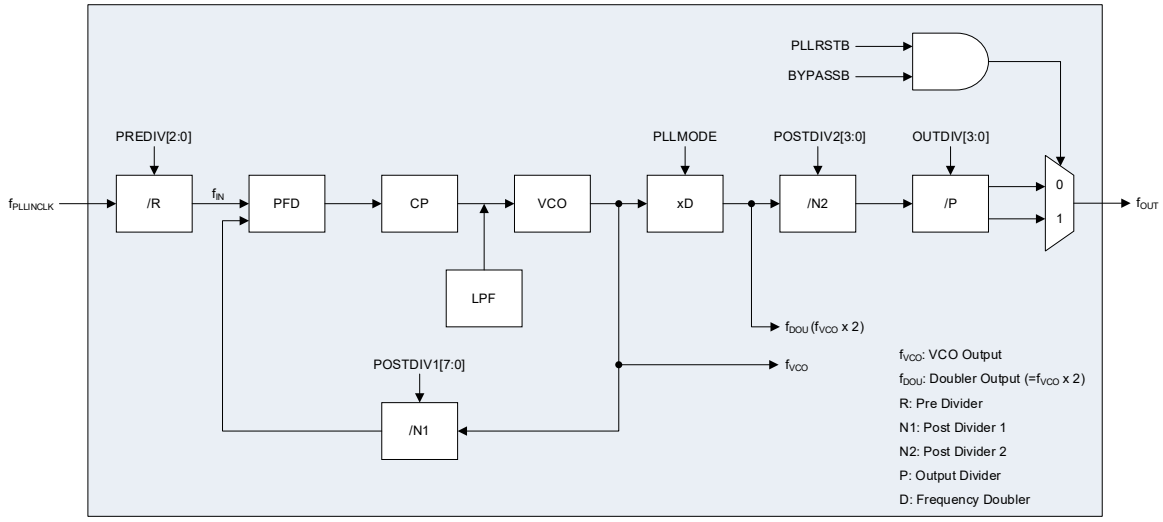
Phase-locked Loop (PLL) can be used to multiply the HSI or HSE output clock frequencies.

The PLL frequency generator generates a high-speed clock (up to 140 MHz). For more information on the PLL frequency generator, refer to Figure 10 and section 4.8.22.

Users must complete the PLL configuration (Selection of the input clock and multiplication factor) before enabling the PLL.

Once the PLL is enabled, these parameters cannot be changed.

Figure 12. PLL Block Diagram



NOTE:

1. PLLEN ='1' → min. 1 us delay → PLLRSTB = '1' to enable PLL block & set PLL frequency.
2. Wait 500 μs for the stabilization of PLL output.
3. Wait until PLLLOCK (= LOCKSTS) flag is occurred.
4. PLLMODE (D) must be '0'.
5. POSTDIV2 (N_2) must be '1'.
6. The output calculation formula is as follows:

$$f_{OUT} = \frac{f_{PLLINCLK} \times (N_1 + 1)}{(R + 1) \times (N_2 + 1) \times (P + 1)} \times (D + 1)$$

Register	Bit	Symbol	Description
SCU_PLLCON	PREDIV[2:0]	R	PLLINCLK pre-divider value
	POSTDIV1[7:0]	N_1	Post multiplier value
	POSTDIV2[3:0]	N_2	Post divider value
	OUTDIV[3:0]	P	Output divider value
	PLLMODE	D	Frequency doubler value

4.4.5.1 PLL Output Frequency Settings

For the A34M420, the PLL's output frequency f_{OUT} can be precisely set in 1 MHz steps. The formula below calculates the input frequency (f_{IN}) fed to the PLL's VCO. The frequency range from 1 MHz to 3 MHz is supported, but it is recommended to set the f_{IN} to 2 MHz if possible.

$$f_{IN} = \frac{PLLINCLK}{(R+1)}, \qquad 1\ MHz \le f_{IN} \le 3\ MHz\ (Recommended\ f_{IN} = 2\ MHz)$$

At this time, the range of VCO frequency, f_{VCO} should be set to Max. VCO frequency (Max. f_{VCO}) or less, and the calculation formula is shown below. (Refer to PLL electrical characteristics in Datasheet.):

$$f_{\mathrm{VCO}}\ (\mathrm{MHz}) = f_{\mathrm{IN}} \times (\mathrm{N}_1 + 1), f_{\mathrm{VCO}} \le \mathrm{Max.}\, f_{\mathrm{VCO}}$$

As a result, the final frequency of the PLL block, f_{PLLOUT} can be obtained from the formula below:

$$f_{\mathrm{PLLOUT}}\ [\mathrm{MHz}] = \frac{f_{\mathrm{PLLINCLK}} \times (\mathrm{N}_1 + 1)}{(\mathrm{R}+1) \times (\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1)$$

$$= \frac{f_{\mathrm{IN}} \times (\mathrm{N}_1 + 1)}{(\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1)$$

$$= \frac{f_{\mathrm{VCO}}}{(\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \ \mathrm{when}, D = 0\ (D\ \ must\ be\ \ '0'.)$$

NOTE:

1. If the main clock is changed from HSI (4 MHz) to PLL, it is recommended to change the clock gradually in the order below section 4.4.5.2:

4.4.5.2 Example PLL Clock Change

1. Once the microcontroller is powered on, the system clock source is HSI (4 MHz).
2. Initialize data cache and instruction cache, and then set flash wait to '7'.
3. Set XIN and XOUT on pin MUX to use the HSE.
4. Enable the HSE.
5. Set an additional time of about 10 ms for clock stabilization.
6. Change the system clock source to the HSE.
7. Enable the PLL. Set the PLL options.
8. Set additional time of about 500 µs for clock stabilization.
9. Set the HCLK to the MCLK divided by 2, and the PCLK to the HCLK.
10. Change the system clock source to the PLL.
11. Set the HCLK to the MCLK divided by '1', and the PCLK to the HCLK.
12. Set Flash wait to '4'.

4.4.6 LSE Clock

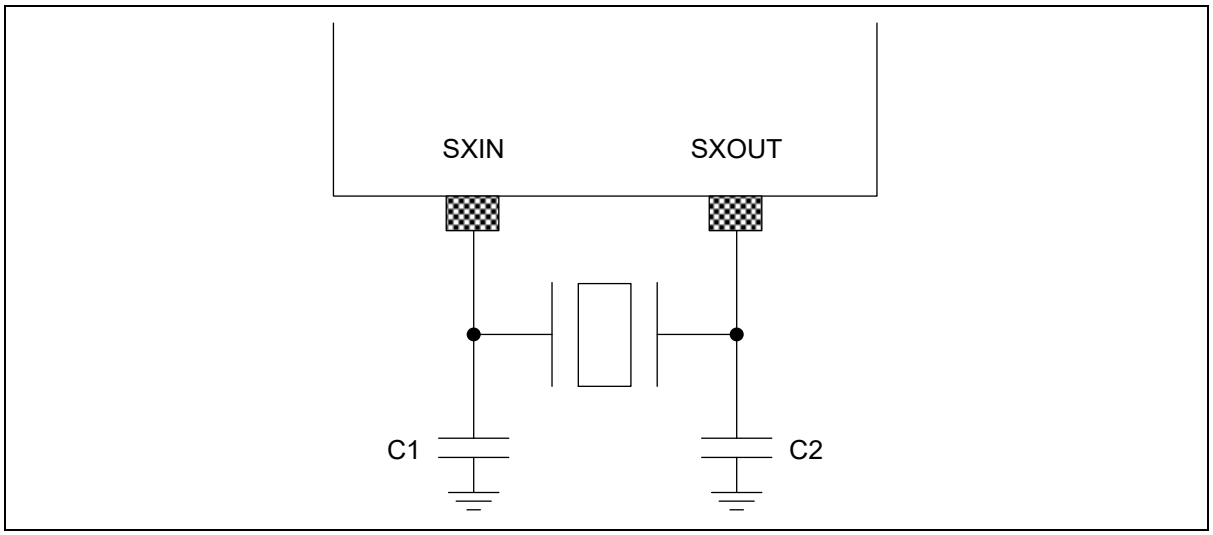
The Low-Speed External Oscillator (LSE) crystal is a 32.768 kHz crystal or ceramic resonator. It has the advantage of providing a low-power but highly accurate clock source to the clock peripheral for clock/calendar or other timing functions.

- LSE: 32.768 kHz

Table 12. LSE Clock Pins

Pin Name	I/O	Description
SXIN	Input	These pins are used to connect a 32.768 kHz crystal resonator
SXOUT	Output	

Figure 13. LSE External Clock



NOTE:

1. When not using LSE (32.768 kHz), If the LSECON bit is enabled (SCU_CSCR), if it is used as GPIO output or other functions, it may collide with SXOUT (output) and receive internal damage to the microcontroller. If external LSE (32.768 kHz) is not used, enable only the clock source to be used in the SCU_CSCR register.

4.4.7 LSI Clock

After the microcontroller is powered on, the LSI (500 kHz) is initially enabled by default in the system operation sequence.

For more details, refer to the electrical characteristics chapter in A34M420 series.

- LSI: 500 kHz
  - ±20% @ −40ºC to +85ºC

4.4.8 Two Main Operating Clocks: HCLK and PCLK

A34M420 has two main operating clocks. One is HCLK, which generates clock signals both for CPU and AHB system; the other is PCLK, which generates clock signals for peripheral systems.

4.4.8.1 HCLK Clock

The Cortex-M4F CPU requires two clocks, the HCLK and FCLK. The HCLK is fed to the core and AHB. The FCLK stays enabled except in DEEP-SLEEP mode, whereas the HCLK can be disabled in SLEEP mode.

The buses and memories are clocked by the HCLK. As the bus clock frequency is limited to a maximum of 140 MHz, the HCLK frequency must not exceed 140 MHz.

4.4.8.2 PCLK Clock

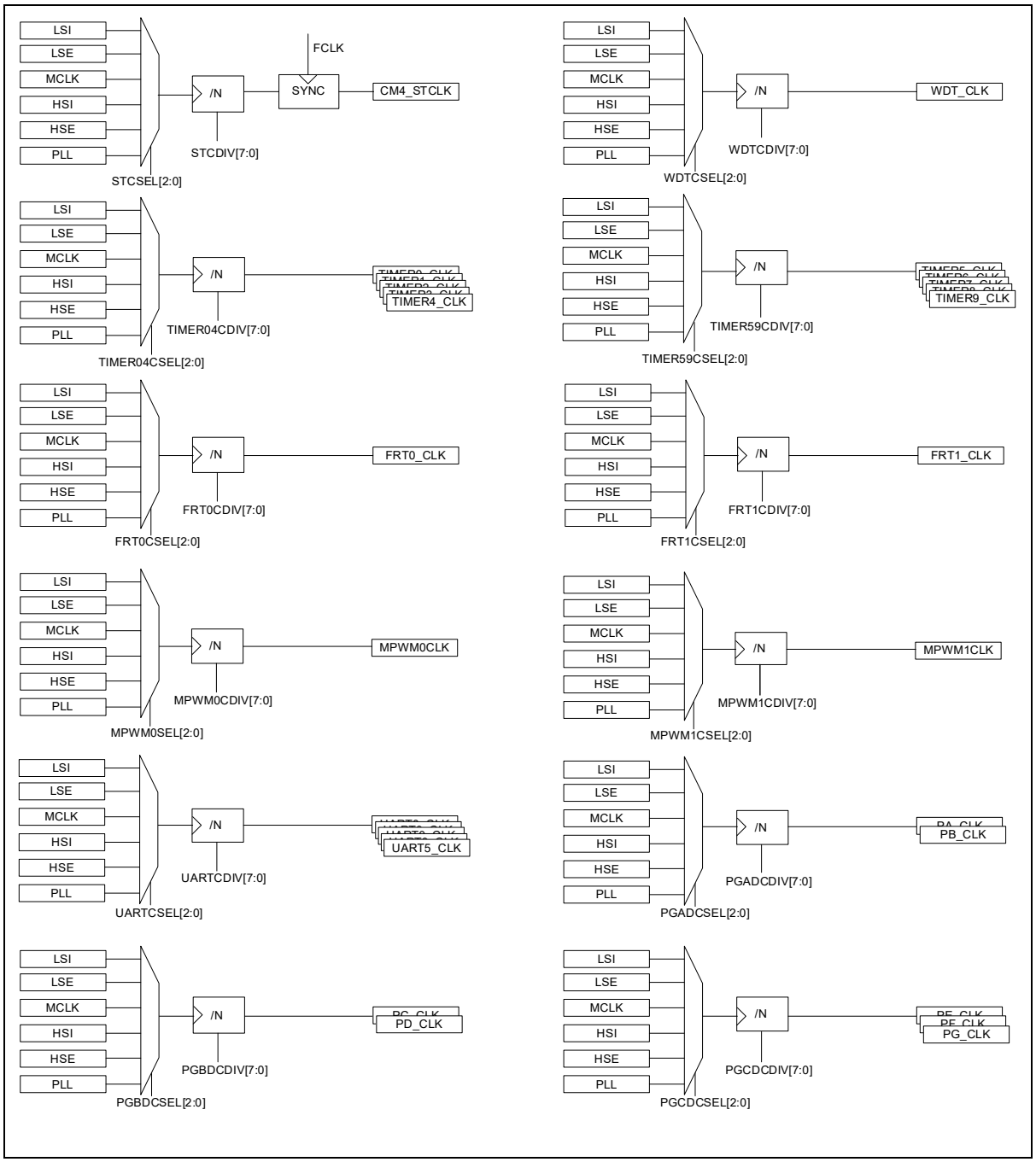
The PCLK can be used as a clock for any peripherals. Enable the PCLK for each peripheral is determined with the SCU_PCER registers; each peripheral block's registers cannot be read unless its PCLK input is enabled. And the PCLK stops operating in DEEP-SLEEP mode.

4.4.9 Configuration of Miscellaneous Clocks

The A34M420 supports the "Miscellaneous Clocks" feature, which allows each peripheral to be assigned with a different clock source (MCLK, HSE, LSE, PLL, or HSI) at a different frequency division ratio.

Users can set each peripheral's clock source and its frequency divider in the corresponding SCU_MCCRx register. The supported division ratio can be one ranging from 1 to 255.

Figure 14. Miscellaneous Clock Configuration



4.4.10 Peripheral Clocks

Table 13 describes the peripheral's clock and the register selection:

Table 13. Peripheral Clock Selection

Peripheral	SCU_MCCRx (x = 1 to 7)	PCLK
Systick	SCU_MCCR1	N/A
WDT		✓
MPWM0	SCU_MCCR2	N/A
MPWM1		N/A
TIMERn (n = 0 to 4)	SCU_MCCR3	✓
TIMERn (n = 5 to 9)		✓
ADC	SCU_MCCR4	✓
PGAD		N/A
PGBD	SCU_MCCR5	N/A
PGCD		N/A
FRT0	SCU_MCCR6	N/A
FRT1		N/A
UART	SCU_MCCR7	N/A

4.4.11 Clock Monitoring

The A34M420 provides clock monitoring functions used to detect whether the microcontroller's clock source is stopped. Using the clock monitoring function, if the microcontroller's clock source is detected to be stopped, it is possible to use interrupt events or reset the system. (For reference, when the external crystal such as the HSE and LSE is damaged or an external clock signal is stopped, the A34M420 generates error flags or interrupt events to check the status and enter recovery mode to put the application in a secure state.)

The speed of the internal clock and external clock may be different when using the clock monitoring function, so it is required to adjust the clock speed to meet the requirements for normal operation in the interrupt functions.

NOTES:

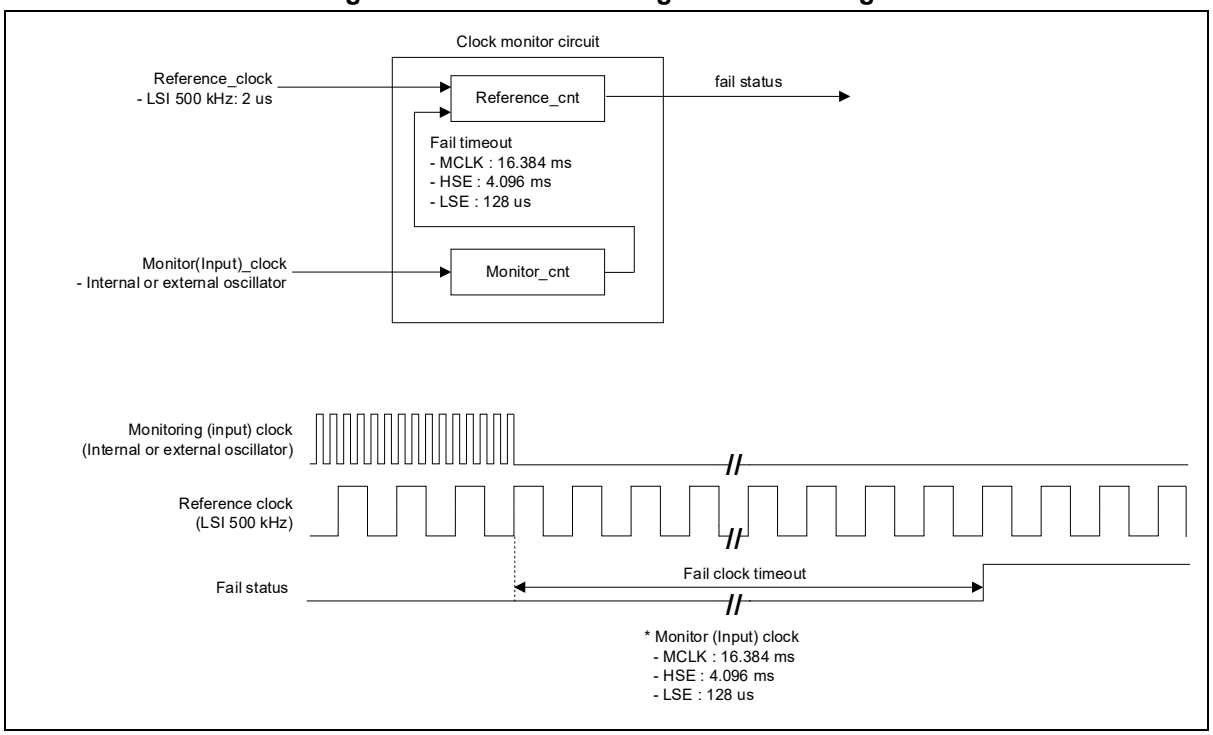
1. Clock monitoring for MCLK supports only the reset function.
2. When using MCLK as an HSE PLL, the MCLK fail reset function cannot be used. In this case, the MCLK fail reset function can be supplemented by activating the HSE fail reset function. In other words, reset occurs when HSE fail occurs.
3. The interrupt function of HSE and LSE clock monitoring is available only when HSE and LSE are not used as MCLK.

4.4.11.1 Detection Range

The clock monitoring functions detect bad oscillators by comparing the external or internal oscillator with the main clock monitoring sample clock. The LSI is used as a reference clock, and error signals are generated due to the reference counter timeout. The timeout period is set differently for MCLK, HSE, and LSE.

Figure 15 shows that the LSI is set as a reference clock and the timeout period required to detect the oscillation failure operation when performing the clock monitoring is different for each clock source.

Figure 15. Clock Monitoring Detection Range



4.4.11.2 Clock monitoring Functions and Clock Sources

Table 14 shows clock monitoring functions provided by the A34M420.

Table 14. Clock Monitoring Function

Clock Monitoring Function		Enable Bit Name
MCLK	Monitoring bit	MCLKMNT
HSE	Monitoring bit	HSEMNT
	Interrupt enable bit	HSEIE
LSE	Monitoring bit	LSEMNT
	Interrupt enable bit	LSEIE

NOTE:

1. Clock monitoring for MCLK supports only the reset function.

Table 15 shows clock sources and their frequency characteristics that the A34M420 can monitor. After each clock source is enabled, stabilization delay is required for stable frequency oscillation.

Table 15. Clock Source for Clock Monitoring

Clock Source	Frequency	Stabilization Delay
HSE	4 to 16 MHz	10 ms
PLL	8 to 140 MHz	500 µs
LSE	32.768 kHz	1 s
LSI	500 kHz	100 µs
his	32 MHz	100 µs

4.4.11.3 MCLK Monitoring

MCLK (main system clock) refers to the clock on which the microcontroller operates. The MCLK can be set to a clock generated by an external oscillating element, or a clock generated by a microcontroller internal oscillation circuit.

To monitor the MCLK clock signal, follow the procedure below:

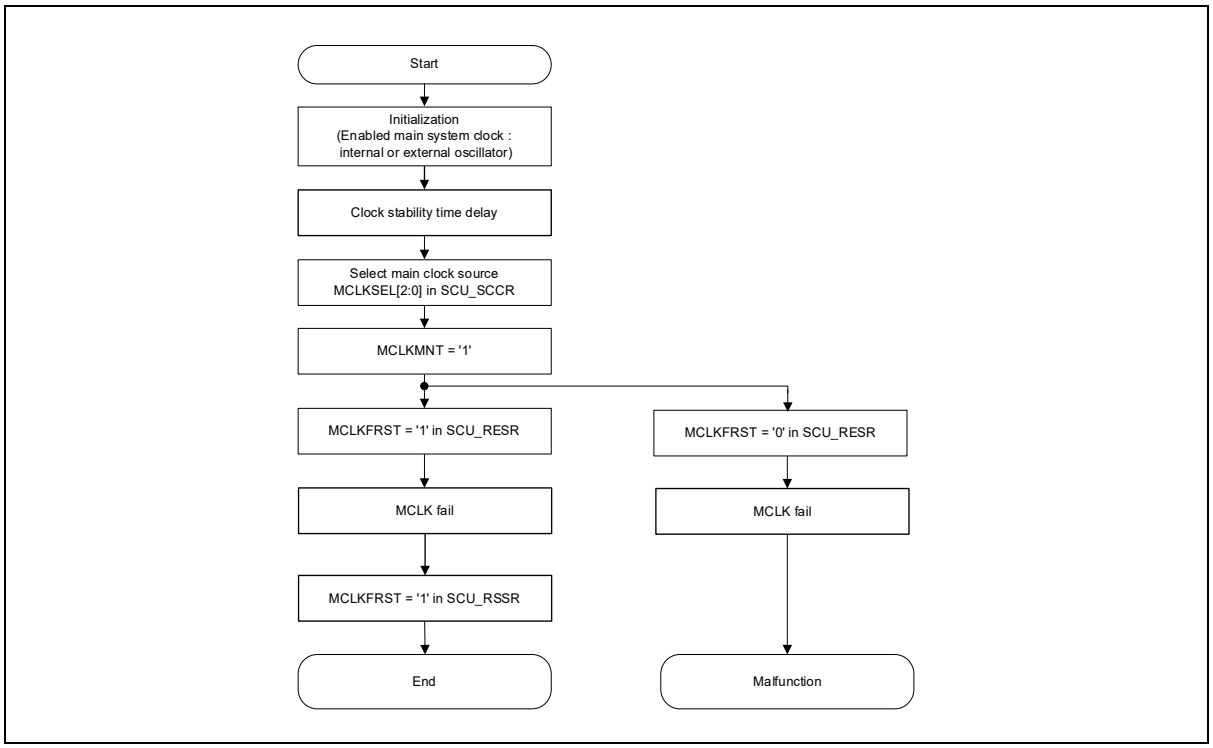
1. Set the HSECON and HSICON bits in the SCU_CSCR register to enable a clock source set to MCLK and wait for a stable time. (HSE: 10 ms, PLL: 500 μs, HSI: 100 μs)
2. Select the main system clock source by configuring the MCLKSEL[2:0] in the SCU_SCCR register. (1)(2)(3)
3. Set the MCLKMNT bit in the SCU_CMR register to '1' to activate the MCLK monitoring function. MCLK errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
4. Set the MCLKFRST bit in the SCU_RSER register to '1' to perform the microcontroller reset when the MCLK error flag occurs.
5. After the reset event occurs, the MCLKFRST flag in the SCU_RSSR register informs whether the reset event occurred in the microcontroller due to the MCLK error. The MCLKFRST flag is cleared by writing '1'.

NOTES:

1. When changing the MCLKSEL bits, both clock sources should be alive. For example, both HSI and HSE must be alive when MCLK is changed to HSE from HSI, otherwise the chip will malfunction.
2. When changing the PLLCLKSEL bits, both the HSE and HSI must be enabled. If the MCLKSEL bits are set to use the HSE, ports must be set for the functions of XIN and XOUT respectively.
3. If the MCLKSEL bits are set to use the LSE, ports must be set for the functions of SXIN and SXOUT respectively.
4. Clock monitoring for MCLK supports only the reset function.

Figure 16 shows the procedure in which the clock monitoring circuit on the A34M420 monitors the MCLK.

Figure 16. MCLK Monitoring Procedure



4.4.11.4 HSE Monitoring

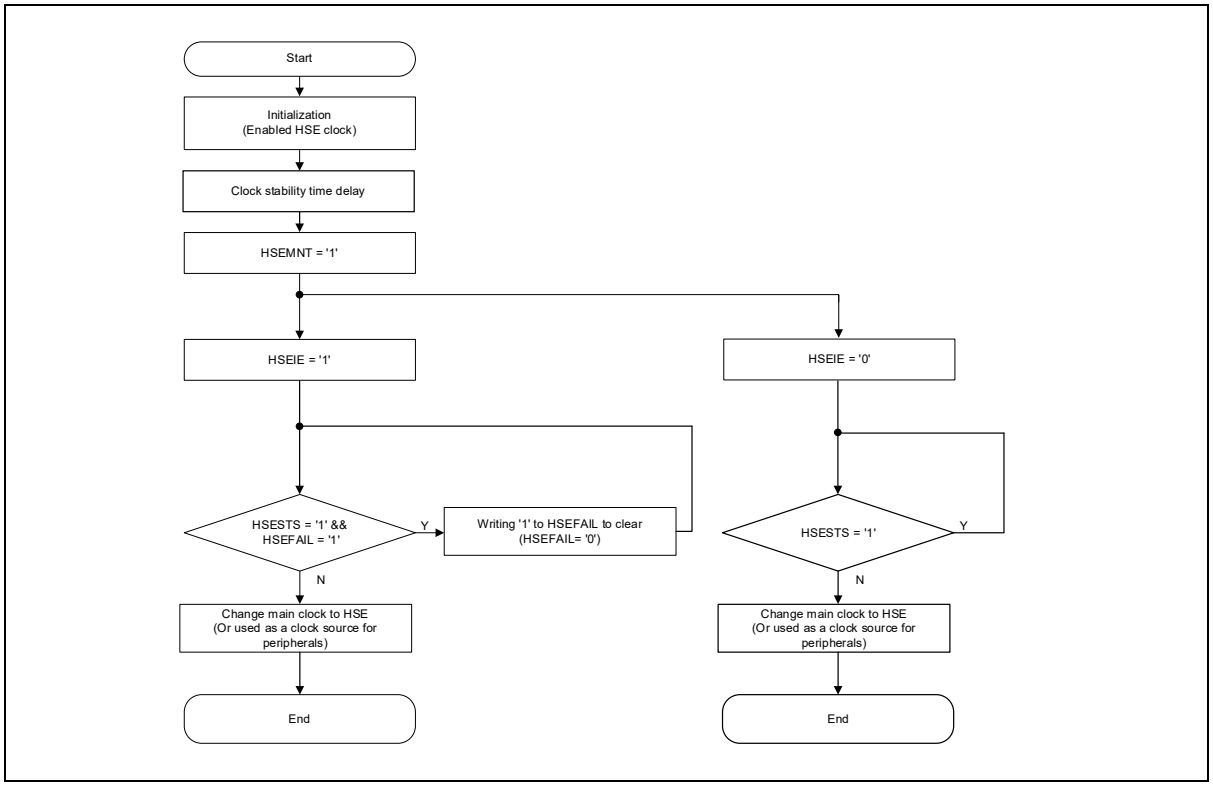
This section describes the HSE clock monitoring. For information on the HSE clock, see section 4.4.3.

To monitor the HSE clock signal, follow the procedure below:

1. Set the HSECON bit in the SCU_CSCR register to ‘1’ to enable the HSE clock source and wait approximately 10 ms for the clock to stabilize.
2. Set the HSEMNT bit in the SCU_CMR register to ‘1’ to activate the HSE monitoring function. HSE errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
3. To use the error detection interrupt function, set the HSEIE bit in the SCU_CMR register to ‘1’. If an error occurs with an interrupt enabled, the HSEFAIL interrupt event is generated. The interrupt error can be detected by the HSEFAIL flag in the SCU_CMR register. The HSEFAIL flag is cleared by writing ‘1’.
4. Set the HSEFRST bit in the SCU_RSER register to ‘1’ to perform the microcontroller reset event when the HSE clock error occurs.
5. After the reset event occurs, the HSEFRST flag in the SCU_RSSR register indicates whether the reset event occurred in the microcontroller due to the HSE error. The HSEFRST flag is cleared by writing ‘1’.

Figure 17 shows the procedure in which the clock monitoring circuit on the A31M420 monitors the HSE.

Figure 17.HSE Monitoring Procedure



4.4.11.5 LSE Monitoring

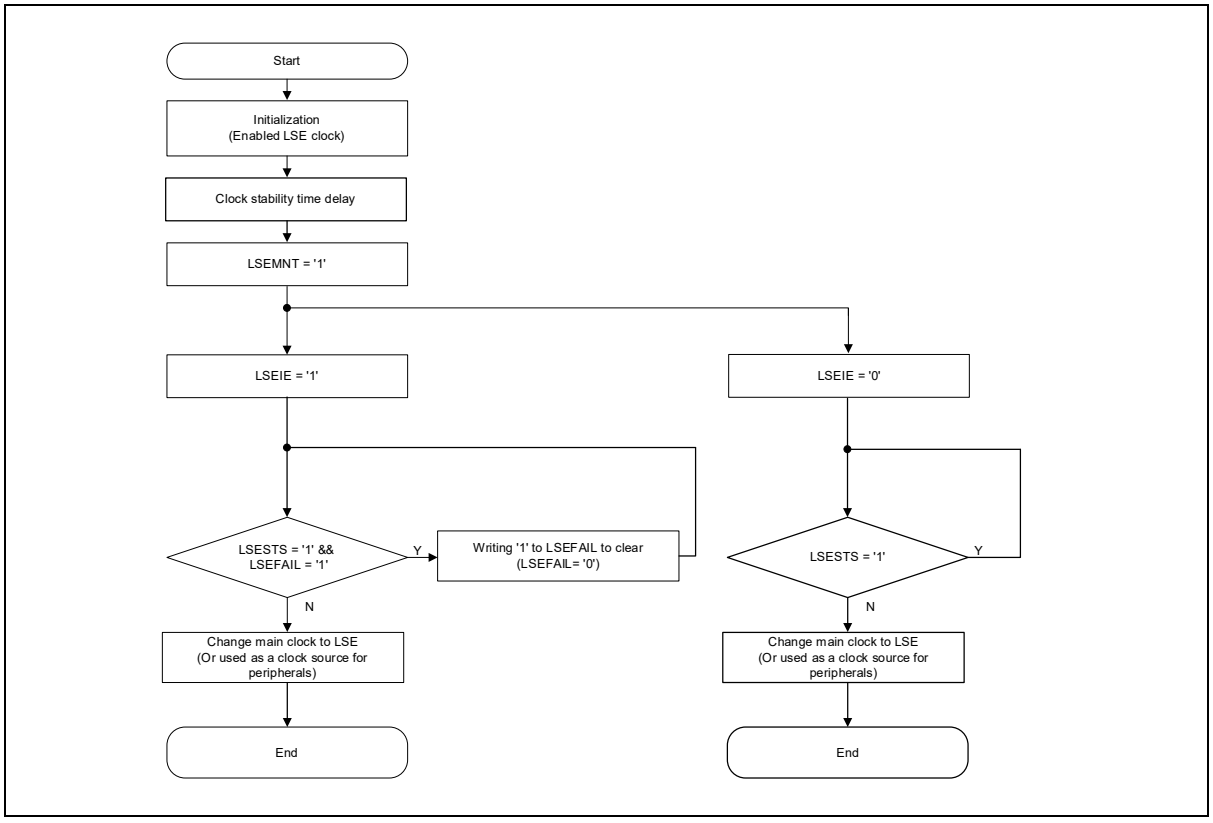
This section describes the LSE clock monitoring. For information on the LSE clock, see section 4.4.6.

To monitor the LSE clock signal, follow the procedure below:

1. Set the LSECON bit in the SCU_CSCR register to ‘1’ to enable the LSE clock source and wait approximately 1 s for the clock to stabilize.
2. Set the LSEMNT bit in the SCU_CMR register to ‘1’ to activate the LSE monitoring function. LSE errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
3. To use the error detection interrupt function, set the LSEIE bit in the SCU_CMR register to ‘1’. If an error occurs with an interrupt enabled, the LSEFAIL interrupt event is generated. The interrupt error can be detected by the LSEFAIL flag in the SCU_CMR register. The LSEFAIL flag is cleared by writing ‘1’.
4. Set the LSEFRST bit in the SCU_RSER register to ‘1’ to perform the microcontroller reset event when the LSE clock error occurs.
5. After the reset event occurs, the LSEFRST flag in the SCU_RSSR register informs whether the reset event occurs in the microcontroller due to the LSE error. The LSEFRST flag is cleared by writing ‘1’.

Figure 18 shows the procedure in which the clock monitoring circuit on the A31M420 monitors the LSE.

Figure 18.LSE Monitoring Procedure



4.4.12 Clock Configuration Procedure

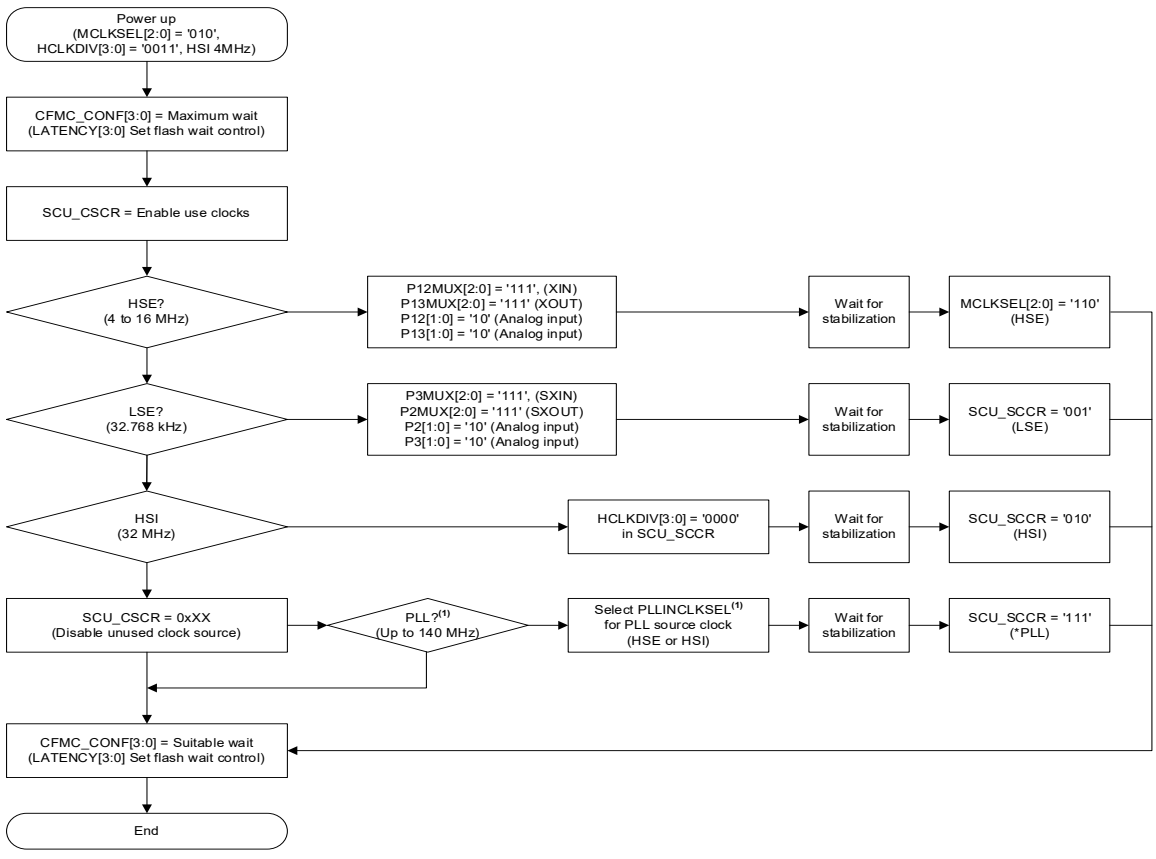
After powering on the device, the High-Speed Internal (HSI) clock of 32 MHz and the Low-Speed Internal (LSI) clock of 500 kHz are enabled by default. The HSI clock is divided by eight to give a system clock of 4 MHz. The other clock sources are set by software while the system is clocked by the HSI. Users can enable the clocks using the SCU_CSCR register, which is the clock source control register.

Before enabling the HSE block, the pin multiplexer configuration must be completed for XIN and XOUT. Users must be careful not to affect other bits of the PC_MR and PC_CR registers during this process. Once the HSE block has been enabled, the users must wait for the crystal oscillation to stabilize.

The secondary LSE (32.768 kHz) clock can be enabled with the SCU_CSCR register. Likewise with pins XIN and XOUT for the HSE, the LSE must be enabled after the pin mux configuration is completed for SXIN and SXOUT and then the stabilizing time is elapsed.

The MCLK can be changed with the SCU_SCCR register. Figure 19 shows an example sequence of how the system clock changes.

Figure 19. Clock Change Procedure



NOTE:

1. For details on the SCU Clock Control, see section 4.4.5 ‘PLL ’.

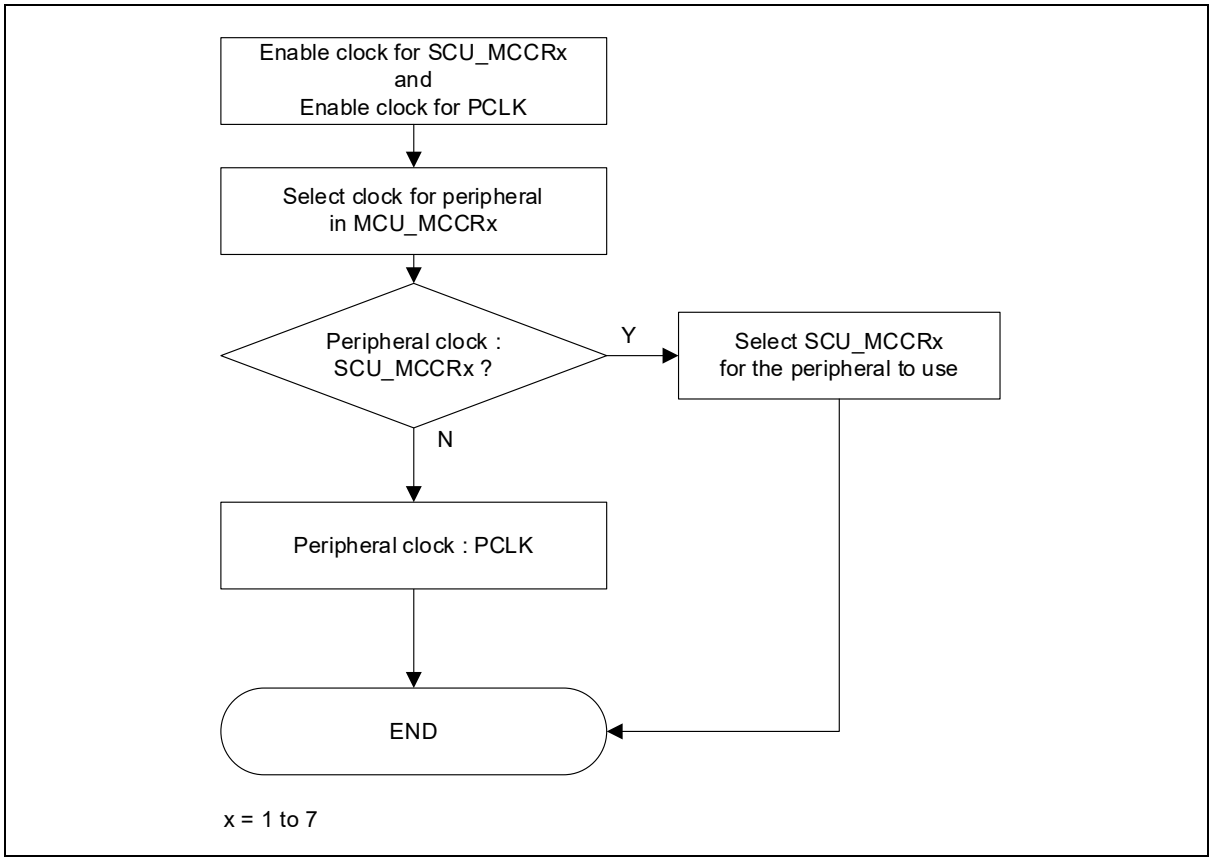
To speed up the system clock to the maximum operating frequency, users must check the flash memory wait control configuration. The flash memory read access time is a limitation factor for performance. The recommended wait control value is listed in Table 16.

Table 16. Flash Memory Wait Control Recommendation

LATENCY[3:0]	Flash Memory Access Wait	Available Max. System Clock Frequency
0000	0-clock wait	Up to 28 MHz
0001	1-clock wait	Up to 56 MHz
0010	2-clock wait	Up to 84 MHz
0011	3-clock wait	Up to 112 MHz
0100	4-clock wait	Up to 140 MHz
0101	5-clock wait	Up to 140 MHz
0110	6-clock wait	Up to 140 MHz
0111	7-clock wait	Up to 140 MHz
1000	8-clock wait	Up to 140 MHz
1001	9-clock wait	Up to 140 MHz
1010	10-clock wait	Up to 140 MHz
1011	11-clock wait	Up to 140 MHz
1100	12-clock wait	Up to 140 MHz
1101	13-clock wait	Up to 140 MHz
1110	14-clock wait	Up to 140 MHz
1111	15-clock wait	Up to 140 MHz

Figure 20 shows how to set the peripheral clock. The peripheral clocks are either set PCLK or selected by the SCU_MCCRx (x = 1 to 7) registers.

Figure 20. Peripheral Clock Select



4.5 Reset

4.5.1 Overview

The reset features are as follows:

- Power-On reset (POR)
- Low-Voltage Reset (LVR)
- nRESET pin reset
- WatchDog timer reset (WDTR)
- Software reset
- Clock oscillation error reset
- CPU request reset

The A34M420 provides two reset source types. One is a cold reset, which is effective during power up or down sequence; the other is a warm reset, which is generated by several reset sources.

The I/Os state under and after reset is “analog state” (for details, see section 2.2).

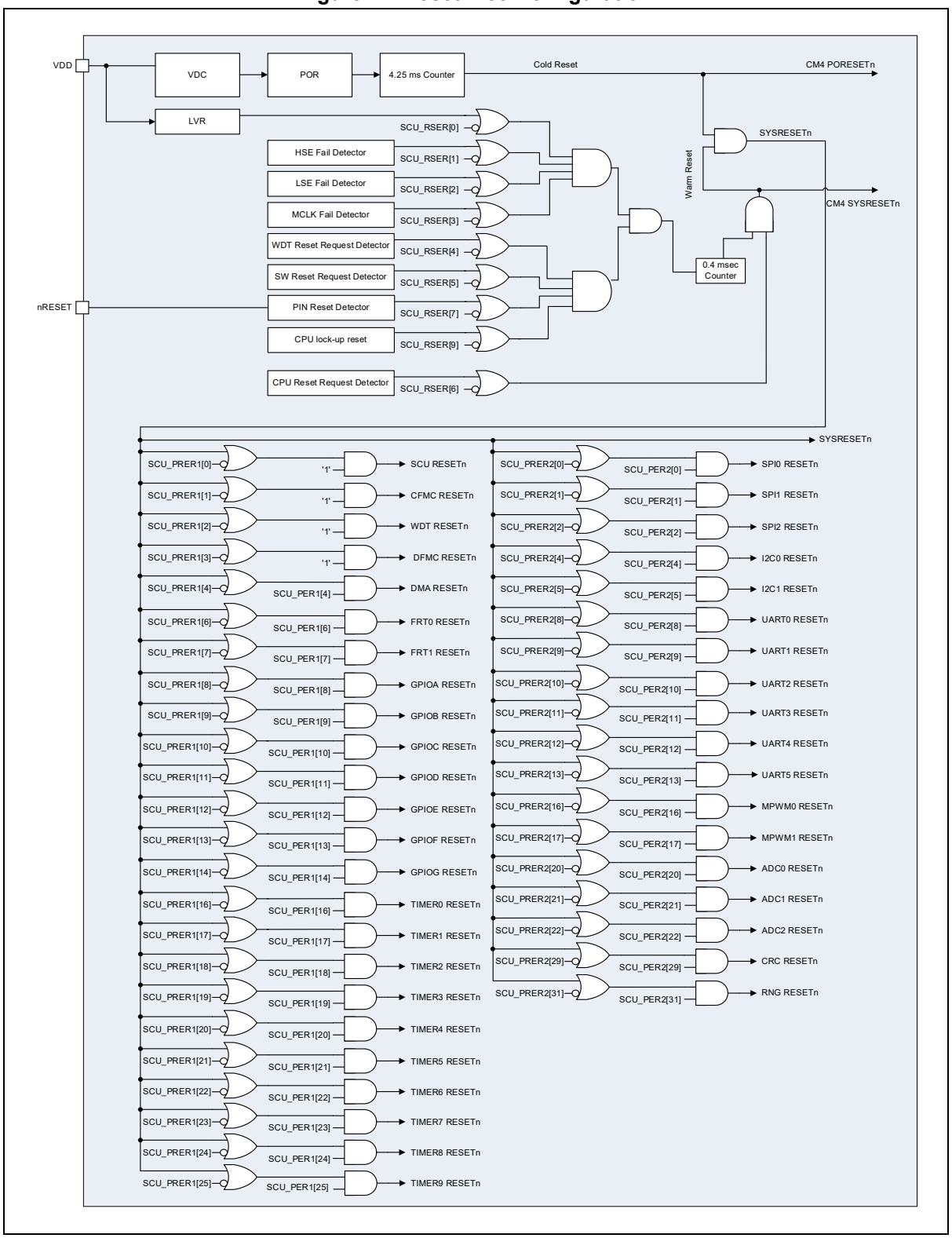
A reset event makes a chip to turn to an initial state. Reset sources of the cold reset and the warm reset are listed in Table 17.

Table 17. Reset Sources for Cold Reset and Warm Reset

Reset Type	Cold Reset	Warm Reset
Reset Source	POR (Power-On Reset)	• nRESET Pin reset • LVD reset • WatchDog Timer reset • Software reset • Clock oscillation error reset • CPU request reset

4.5.2 Reset Tree Configuration

Figure 21. Reset Tree Configuration



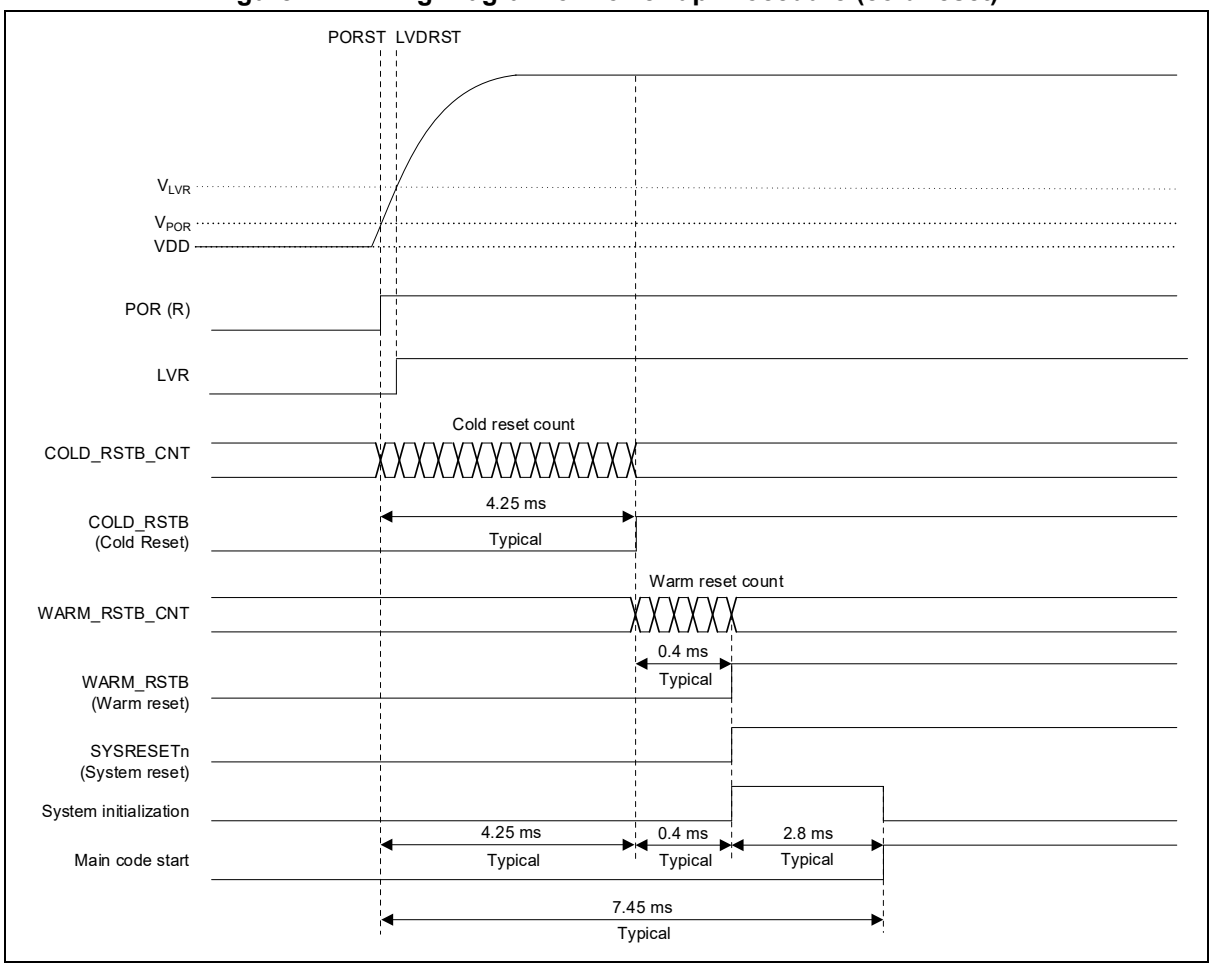
4.5.3 Cold Reset

A cold reset plays an important role during a power-up process and affects the entire process of system booting. The internal VDC becomes active as soon as the VDD is applied. The internal POR is triggered when the VDD is determined to reach 1.4 V based on the amount of the internal VDC output.

During the cold reset process, when the applied voltage exceeds 1.4 V, the LSI clock and the HSI clock are enabled. After the stabilization of the internal VDC level for 4.25 ms, the internal logic is initialized. And then, when the external VDD voltage rises above the LVD Reset voltage level (2.12 V), the cold reset is released and, after a waiting time of 0.4 ms for warm reset synchronization, booting begins.

Figure 22 shows waveform of power up sequence and initial reset.

Figure 22. Timing Diagram of Power-up Procedure (cold reset)



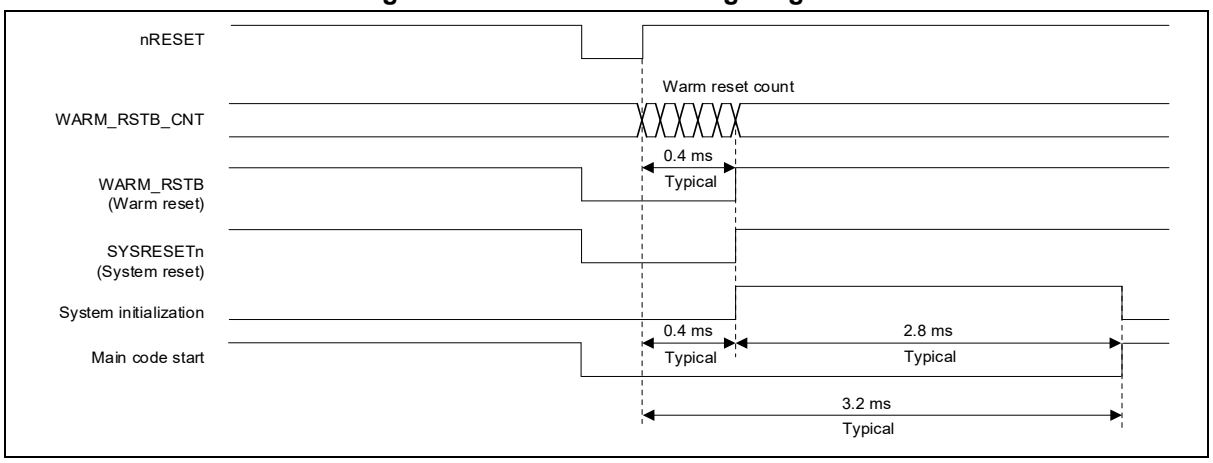
4.5.4 Warm Reset

A warm reset event is triggered for system initialization when the conditions of a reset source internally set are satisfied.

Warm reset sources are enabled or disabled by configuring the SCU_RSER register, and their occurrence is written to the SCU_RSSR register.

Which devices are to be initialized by the warm reset is determined by settings of the SCU_PRER register. Using this register, users can allow or disallow the initialization of individual peripheral.

Figure 23. Warm Reset Timing Diagram



4.5.5 LVD Reset (LVR)

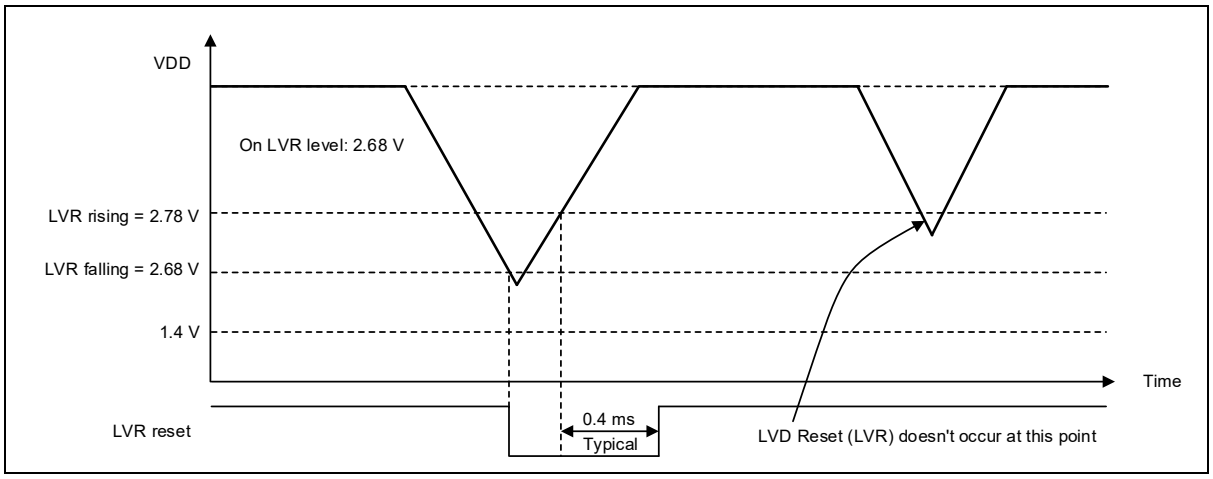
A Low-Voltage Detect Reset (LVR) event is triggered when the operating voltage drops below a certain level during the microcontroller's operation. Users can set the microcontroller whether to perform a reset or to perform an interrupt, when an LVR event is triggered.

Table 18 describes the range of LVR level and the register for LVR level control.

Table 18. LVR Level

LVR Level Range	Description
1.60 V to 4.48 V	LVR level (16 steps) This LVR level is set by the LVRVS[3:0] bits in the SCU_LVRCR register.

Figure 24. LVR Timing Diagram



4.5.6 nRESET Pin Reset

This reset signal is generated on the nRESET pin. When the nRESET pin is driven low, all the processing in progress is aborted and the microcontroller enters reset state.

The nRESET pin in Table 19 is assigned for RESET.

Table 19. nRESET Pin

Pin Name	Type	Description
nRESET	Input	External reset input

4.5.7 WatchDog Timer Reset

Watchdog timer (WDT) monitors the operation of the microcontroller and is typically used to detect software errors. When the microcontroller becomes uncontrollable due to a malfunction, the WDT resets the microcontroller to recover it.

The A34M420 has one WDT module built in, which functions as a 32 bit down-counter. Once the WDT counts down to zero while operating as a reset source, the microcontroller will restart.

For details on the watchdog timer reset, refer to chapter 8.

4.5.8 Software Reset

A system reset sets all registers to their reset values except the reset flags in the clock system reset control the SWRST bit in the SCU_SRCR register and the registers in the SCU block(1).

NOTE:

1. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

4.5.9 HSE Fail Reset

The HSEFRST in SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting the HSEFRST bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The HSEFRST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.5.10 LSE Fail Reset

The LSEFRST bit in the SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting a bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The LSEFRST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.5.11 CPU Request Reset

The CPURST bit in the SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting a bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The CPURST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.6 Power Supply

The device requires a 2.5 V to 5.5 V operating voltage supply (VDD).

Table 20. Power Supply of VDD

Parameter	Symbol	Min.	Max.	Unit
Supply Voltage	VDD	2.5	5.5	V
	AVDD	2.7	5.5	V

During a power-up process, a reset plays an important role and affects the entire process of system booting. A34M420 has two power-related reset options as described below:

- POR_RST (Power-On Reset) that controls the voltage less than 1.4 V.
- LVD_RST (Low-Voltage Detect Reset) that controls the voltage less than 2.21 V (default).

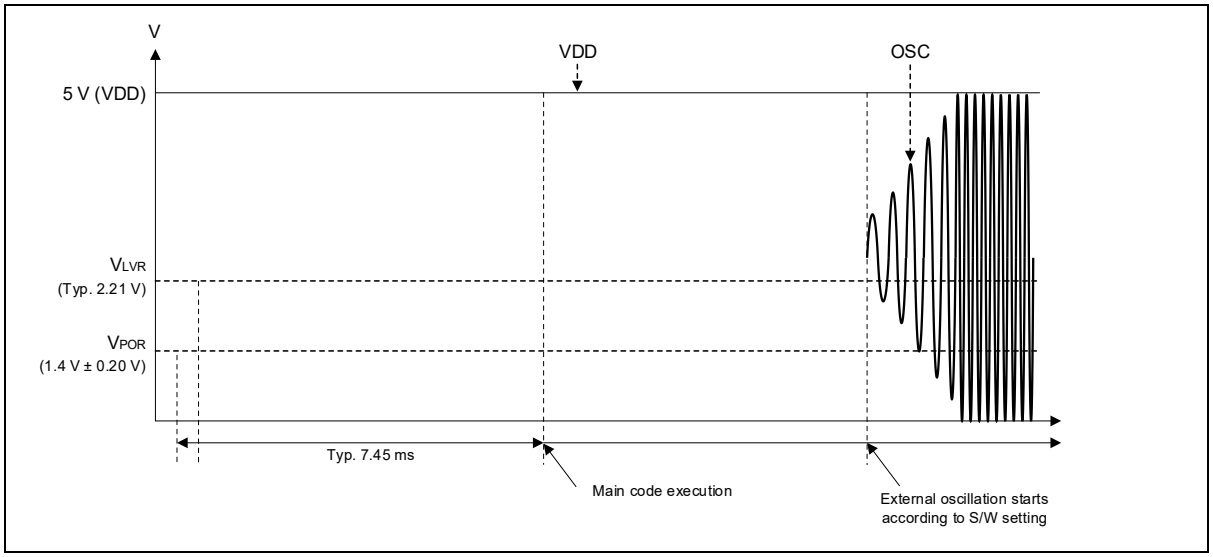
If the power level is higher than the POR and below the flash operating voltage (Min. 1.35 V), code read operation may malfunction. To prevent this abnormal code read operation, the LVD_RST operates to generate the SYSRESETn internal signal and enter the microcontroller into reset mode to prevent abnormal operation. The minimum level that can be set by the LVR is 1.6 V, which satisfies the Flash minimum operating voltage.

An LVR reset event is triggered when the operating voltage drops below the selected LVR level during the microcontroller's operation. Users can set the microcontroller to perform a LVD reset for LVR level setting. LVR is a warm reset. Refer to the description on section 4.5.4 for details.

Figure 25 shows a timing diagram for the operation sequence after the POR and LVR. As power is applied, it shows the waveform of main code execution and oscillator operation.

The main code is executed after the reset stabilization time has elapsed. In the case of A34M420, note that external oscillation is set to operate by software setting.

Figure 25. POR and LVR Timing Diagram

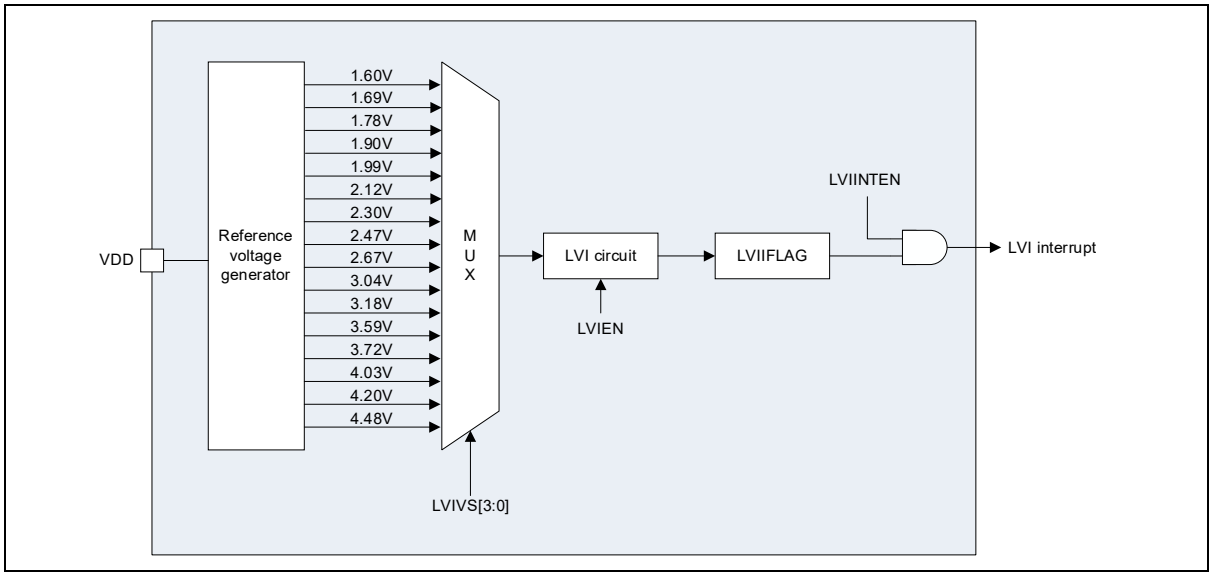


4.6.1 LVI Block Diagram

Using the LVI interrupt function, the LVI interrupt can be generated when the external VDD voltage reaches the voltage preset by users.

Figure 26 shows a block diagram of the LVI. For information on the LVI signal operation, refer to SCU_LVICR.

Figure 26. LVI block diagram

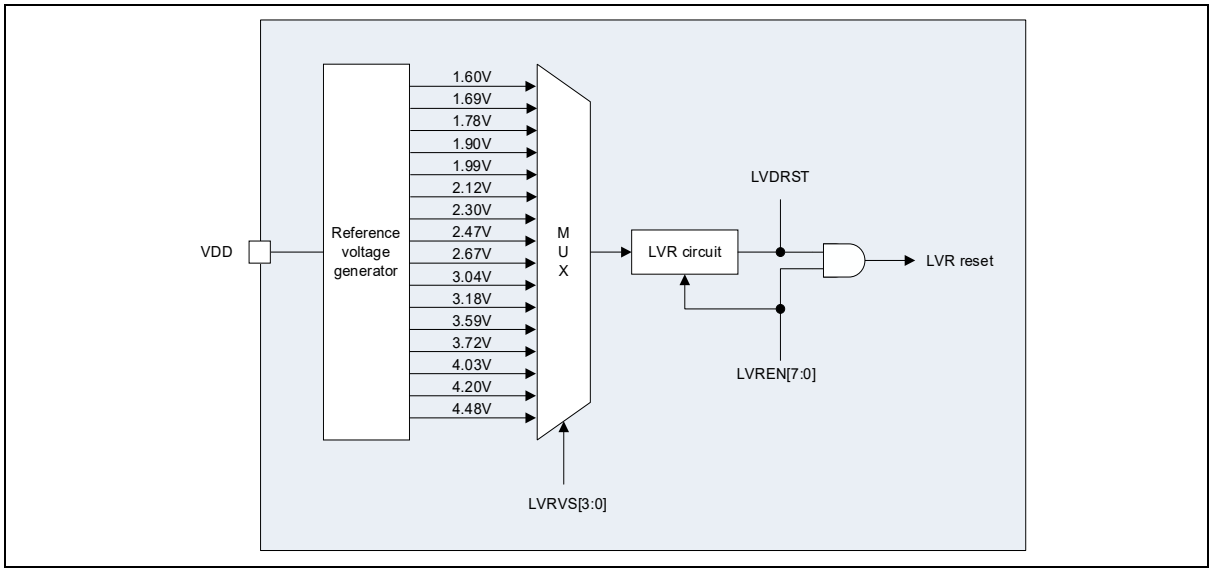


4.6.2 LVR Block Diagram

Using the LVR function, the microcontroller can be reset when the external VDD voltage reaches the voltage preset by users.

Figure 27 shows a block diagram of the LVR. For information on the LVR signal operation refer to SCU_LVRCR.

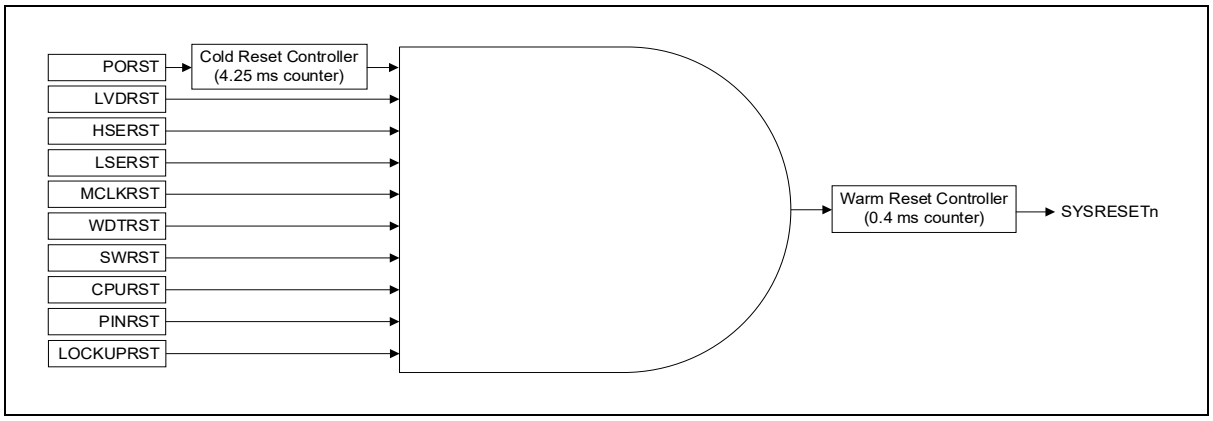
Figure 27. LVR Block Diagram



4.6.3 POR, LVR

For the case where the VDD input level is higher than the POR and lower than the minimum voltage (1.35 V) required for the Flash operation, the LVDRST (Min. 1.6 V) is set to improve stability and prevent operation errors such as code read. For more information, refer to section 4.5.

Figure 28. POR and LVR Block Diagram



4.7 Operation Modes

4.7.1 Overview

The microcontroller has several functions for reducing power consumption, such as setting clock dividers, stopping modules, selecting power control mode in RUN mode, and transitioning to low power modes (SLEEP mode, DEEP-SLEEP mode)

The A34M420 operates in three modes below:

- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP 1, 2) mode

4.7.2 Transition of Operation Mode

INIT mode is the initial state of the chip when a reset is asserted.

In RUN mode, the CPU shows the maximum performance with high-speed clock system.

SLEEP mode and DEEP-SLEEP mode can be used as a low power consumption mode. In the low power consumption mode, power is effectively managed to reduce power consumption by halting processor core and unused peripherals.

Figure 29 describes transition between the operation modes.

Figure 29. Transition between Operation Modes

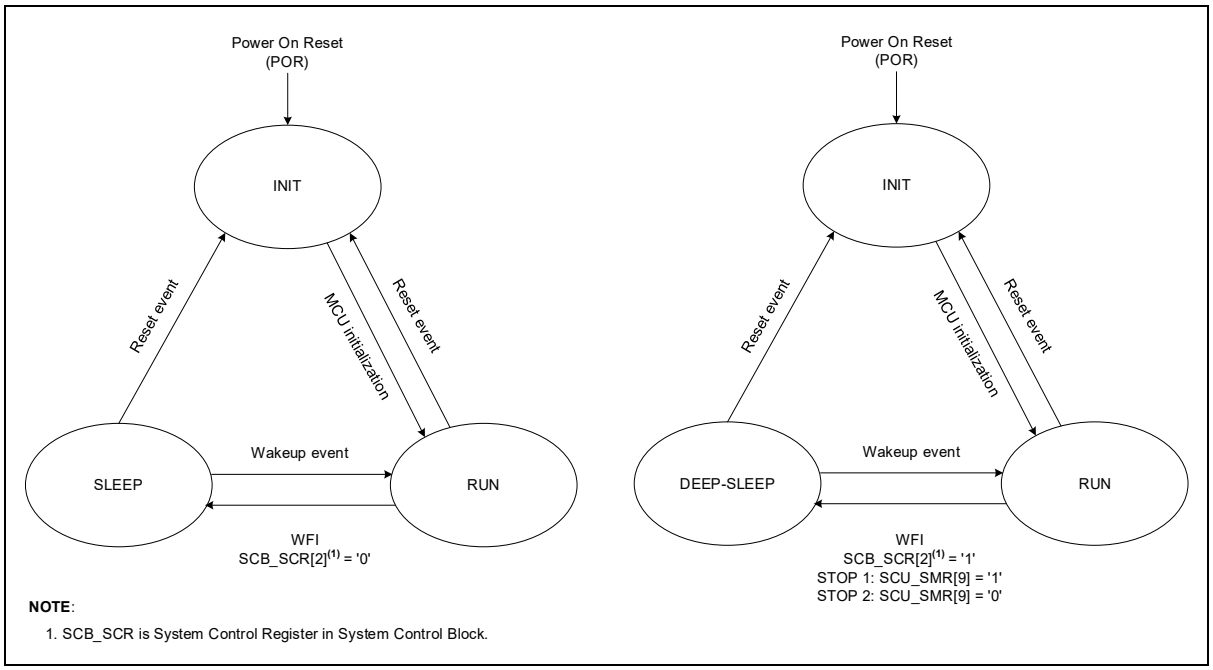


Table 21. Operation Mode

Mode	Conditions	After Wake-up Event	After Reset Event
RUN	POWER ON	N/A	INIT
SLEEP	WFI (wait for interrupt): SCB_SCR[2](1) = 0	RUN	INIT
DEEP-SLEEP (STOP 1)	WFI (wait for interrupt): SCB_SCR[2](1) = 1 SCU_SMR[9] = 1	RUN	INIT
DEEP-SLEEP (STOP 2)	WFI (wait for interrupt): SCB_SCR[2](1) = 1 SCU_SMR[9] = 0	RUN	INIT

NOTE:

1. SCB_SCR is the System Control Register in the System Control Block.

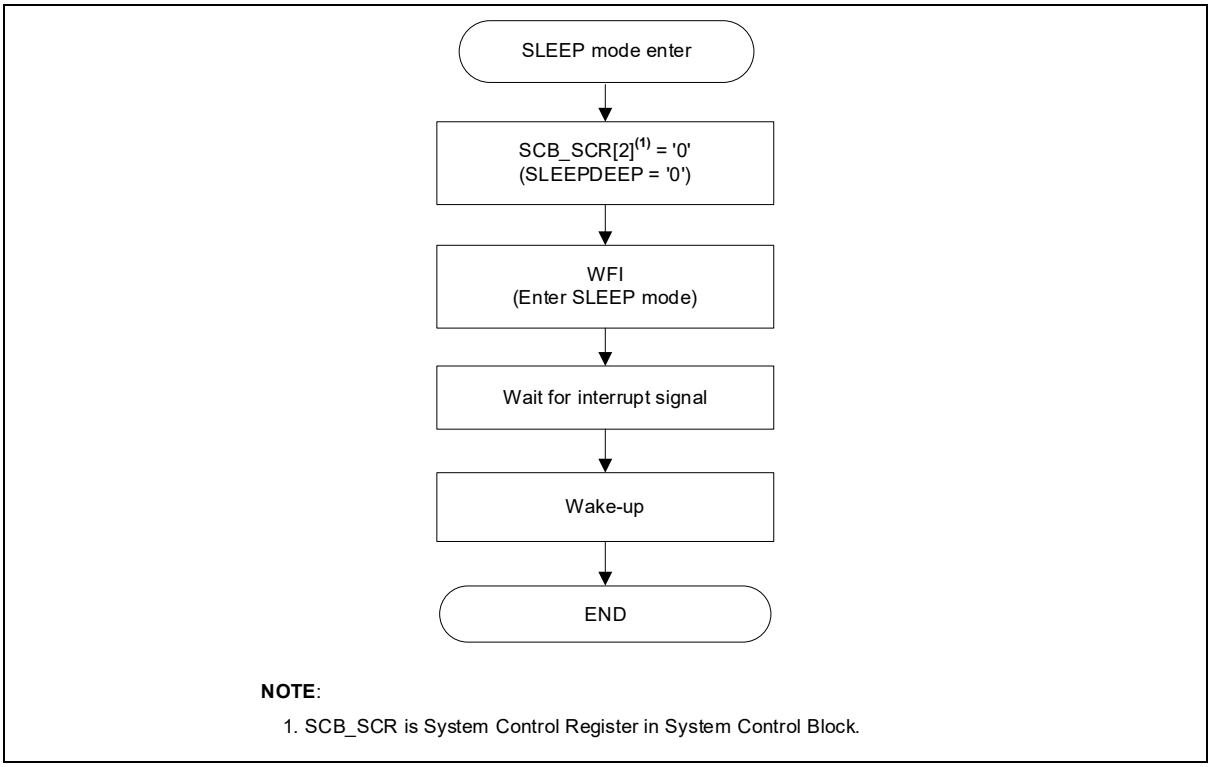
4.7.3 RUN Mode

In RUN mode, the CPU and peripheral hardware operate with a high-speed clock. If the INIT state occurs after a reset, the system enters RUN mode.

4.7.4 SLEEP Mode

Once the microcontroller enters SLEEP mode, the CPU becomes inactive. By setting the SCU_PER and SCU_PCER registers, users can determine which peripherals are to be inactive in SLEEP mode.

Figure 30. SLEEP Mode Operation Sequence



NOTE:

1. SCB_SCR is System Control Register in System Control Block.

4.7.5 DEEP-SLEEP (STOP) Mode

When the core goes under stop state using WFI instruction, the chip enters in DEEP-SLEEP mode. DEEP-SLEEP mode is divided into DEEP-SLEEP (STOP 1) and DEEP-SLEEP (STOP 2) modes.

To enter STOP 1 mode, first set SCU_SMR[9] = 1 and use the WFI command. In STOP 1 mode, all peripherals except WDT and FRT of internal VDC domain are in power off state.

To enter STOP 2 mode, first set SCU_SMR[9] = 0 and use the WFI command. In STOP 2 mode, all peripherals of internal VDC domain are stopped. To wake up in STOP 2 mode, an interrupt request must be generated and can be selected in the SCU_WUER register. Stabilization time is 4 ms after wakeup event occurs.

When entering DEEP-SLEEP mode, LSI (500 kHz) clock is automatically selected as the MCLK, HSE is selected as the PLL input clock, and all clock sources are automatically disabled. These are maintained after wake-up. If the SCU_SMR register is used, a function that automatically disables system clocks such as HSE, PLL, HSI, and LSI may not be used[1].

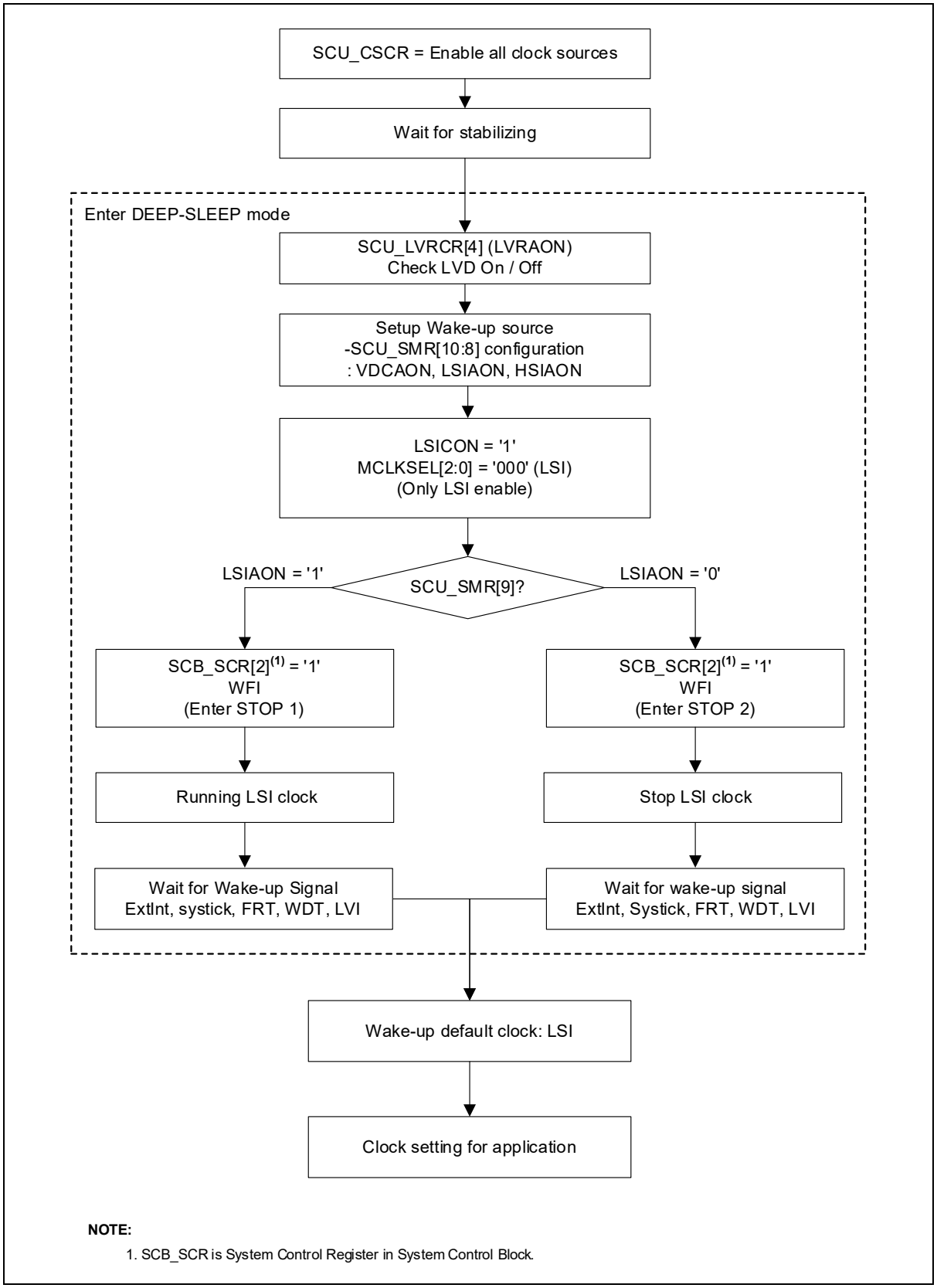
Table 22. DEEP-SLEEP Mode Configuration

Mode	Type	Conditions	Wakeup Source
DEEP-SLEEP	STOP 1	WFI SCB_SCR[2][2] = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '1' (LSIAON = '1')	Source included in WUER
	STOP 2	WFI SCB_SCR[2][2] = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '0' (LSIAON = '0')	Source included in WUER

NOTE:

1. When the WDT or FRT clock source is set to the PLL and the PLLAON bit in the SCU_SMR register enabled, the A34M420 cannot wake-up if it enters DEEP-SLEEP mode. In this case, the A34M420 must use SLEEP mode instead of DEEP-SLEEP mode.
2. SCB_SCR is the System Control Register in the System Control Block.

Figure 31. DEEP-SLEEP Mode Operation Sequence



NOTE:

1. SCB_SCR is System Control Register in System Control Block.

4.7.6 Operation Mode Summary

Table 23 and Table 24 are configurable clocks according to each operating mode.

Table 23. Configurable Clock in each RUN/SLEEP Mode

Mode	CPU	Peripheral	VDC	Peripheral Clock Source				
				PLL	LSI	HSI	HSE	LSE
RUN	Active	User-defined	ON	User-defined	User-defined	User-defined	User-defined	User-defined
SLEEP	Inactive	User-defined	ON	User-defined	User-defined	User-defined	User-defined	User-defined

Table 24. Configurable Clocks in each DEEP-SLEEP (STOP) Mode

Mode	CPU	Peripheral	SCU_SMR					
			VDCAON	PLLAON	LSIAON	HSIAON	HSEAON	LSEAON
RUN	Active	User-defined	Don't care	Don't care	Don't care	Don't care	Don't care	Don't care
SLEEP	Inactive	User-defined	Don't care	Don't care	Don't care	Don't care	Don't care	Don't care
DEEP-SLEEP (STOP 1)	Inactive	Only WDT, FRT	OFF	OFF	ON	OFF	OFF	OFF
DEEP-SLEEP (STOP 2)	Inactive	OFF	OFF	OFF	OFF	OFF	OFF	OFF

4.8 SCU Registers

The base address and register map of CHIPCONFIG are described in the followings:

Table 25. Base Address of CHIPCONFIG

Name	Base Address
CHIPCONFIG	0x4000_F000

Table 26. Register Map of CHIPCONFIG

Name	Offset	Type	Description	Reset Value	Reference
CHIPCONFIG_VENDORID	0x0000	RO(1)	Vendor ID Register	0x4142_4F56	4.8.1
CHIPCONFIG_CHIPID	0x0004	RO(1)	Chip ID Register	0x4D34_4200	4.8.2
CHIPCONFIG_REVNR	0x0008	RO(1)	Revision Number Register	0x0000_00XX	4.8.3

NOTE:

1. 'RO' means 'Read Only'.

The base address and register map of the SCU are described in the followings:

Table 27. Base Address of SCU

Name	Base Address
SCU	0x4000_0000

Table 28. Register Map of SCU

Name	Offset	Type	Description	Reset Value	Reference
SCU_SMR	0x0004	RW	System Mode Register	0x0000_0030	4.8.4
SCU_SRCR	0x0008	RW	System Reset Control Register	0x0000_0000	4.8.5
SCU_WUER	0x0010	RW	Wake-up Source Enable Register	0x0000_0000	4.8.6
SCU_WUSR	0x0014	RO	Wake-up Source Status Register	0x0000_0000	4.8.7
SCU_RSER	0x0018	RW	Reset Source Enable Register	0x0000_00D1	4.8.8
SCU_RSSR	0x001C	RWC1	Reset Source Status Register	0x0000_0101	4.8.9
SCU_PRER1	0x0020	RW	Peripheral Reset Enable Register 1	0x03FF_7FDF	4.8.10
SCU_PRER2	0x0024	RW	Peripheral Reset Enable Register 2	0xA073_3F37	4.8.11
SCU_PER1	0x0028	RW	Peripheral Enable Register 1	0x0000_000F	4.8.12
SCU_PER2	0x002C	RW	Peripheral Enable Register 2	0x0000_0101	4.8.13
SCU_PCER1	0x0030	RW	Peripheral Clock Enable Register 1	0x0000_000F	4.8.14
SCU_PCER2	0x0034	RW	Peripheral Clock Enable Register 2	0x0000_0101	4.8.15
SCU_CSCR	0x0040	RW	Clock Source Control Register	0x0000_0028	4.8.16

Table 27. Register Map of SCU (continued)

Name	Offset	Type	Description	Reset Value	Reference
SCU_SCCR	0x0044	RW	System Clock Control Register	0x0300_0002	4.8.17
SCU_CMR	0x0048	RW	Clock Monitor Register	0x0000_0000	4.8.18
SCU_COR	0x0050	RW	Clock Output Register	0x0000_000F	4.8.19
SCU_NMICR	0x0054	RW	Non-Maskable Interrupt Control Register	0x0000_0000	4.8.20
SCU_NMISR	0x0058	RWC1	Non-Maskable Interrupt Status Register	0x0000_0000	4.8.21
SCU_PLLCON	0x0060	RW	PLL Control Register	0x0600_0000	4.8.22
SCU_VDCCON	0x0064	RW	VDC Control Register	0x0000_007F	4.8.23
SCU_LVICR	0x0068	RW	LVI Control Register	0x0000_0000	4.8.24
SCU_LVISR	0x006C	RC	LVI Status Register	0x0000_0000	4.8.25
SCU_LVRCR	0x0070	RW	LVR Control Register	0x0000_0005	4.8.26
SCU_EOSCR	0x0080	RW	External Oscillator Control Register	0x0000_0000	4.8.27
SCU_MCCR1	0x0090	RW	MISC Clock Control Register 1	0x0001_0000	4.8.28
SCU_MCCR2	0x0094	RW	MISC Clock Control Register 2	0x0000_0000	4.8.29
SCU_MCCR3	0x0098	RW	MISC Clock Control Register 3	0x0000_0000	4.8.30
SCU_MCCR4	0x009C	RW	MISC Clock Control Register 4	0x0000_0000	4.8.31
SCU_MCCR5	0x00A0	RW	MISC Clock Control Register 5	0x0000_0000	4.8.32
SCU_MCCR6	0x00A4	RW	MISC Clock Control Register 6	0x0000_0000	4.8.33
SCU_MCCR7	0x00A8	RW	MISC Clock Control Register 7	0x0000_0000	4.8.34
SCU_SYSTEN	0x00F0	WO	System Access Enable	0x0000_0000	4.8.35

4.8.1 CHIPCONFIG_VENDORID: Vendor ID Register

The CHIPCONFIG_VENDORID register shows the vendor's identification information. This is a 32-bit read-only register.

CHIPCONFIG_VENDORID=0x4000_F000

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
VENDID[31:0]																															
0x4142_4F56																															
RO																															

31	VENDID[31:0]	Vendor identification bits.	
		0x4142_4F56	

4.8.2 CHIPCONFIG_CHIPID: Chip ID Register

The CHIPCONFIG_CHIPID register shows the chip's identification information. This is a 32-bit read-only register.

CHIPCONFIG_CHIPID=0x4000_F004

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[31:0]																															
0x4D34_4200																															
RO																															

15	CHIPID[31:0]	Chip ID bit.	
		0x4D34_4200	1024 KB flash memory / 64 KB RAM

4.8.3 CHIPCONFIG_REVNR: Revision Number Register

The CHIPCONFIG_REVNR register shows the revision number information of a chip. This register is a 32-bit read-only register.

CHIPCONFIG_REVNR=0x4000_F008

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
Reserved			REVNO[7:0]
0x000000			0xXX
.			RO

7 0	REVNO[7:0]	Chip revision number These bits are modified by a manufacturer.

4.8.4 SCU_SMR: System Mode Register

The SCU_SMR register contains bits controllable in DEEP-SLEEP(1) mode and informs which operating mode the microcontroller had been in before it returned to initial mode. Once a reset event occurs, the previous operating mode is recorded in this register. The register is 32 bits wide.

SCU_SMR=0x4000_0004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13	12	11	10	9	8	7 6	5 4	3 2 1 0
Reserved			LSEAON	HSEAON	PLLAON	HSIAON	LSIAON	VDCAON	Reserved	PREVMODE[1:0]	Reserved
-			0	0	0	0	0	0	-	11	-
-			RW	RW	RW	RW	RW	RW	-	RO	-

13	LSEAON(1)	LSE enablement in mode	
		0	Automatically disables the LSE in DEEP-SLEEP mode.
		1	Leaves the LSE enabled in DEEP-SLEEP mode.
12	HSEAON(1)	HSE enablement in DEEP-SLEEP mode	
		0	Automatically disables the HSE in DEEP-SLEEP mode.
		1	Leaves the HSE enabled in DEEP-SLEEP mode.
11	PLLAON(2)(3)	PLL enablement in DEEP-SLEEP mode	
		0	Automatically disables the PLL in DEEP-SLEEP mode.
		1	Leaves the PLL enabled in DEEP-SLEEP mode.
10	HSIAON(1)	HSI enablement in DEEP-SLEEP mode	
		0	Automatically disables the HSI in DEEP-SLEEP mode.
		1	Leaves the HSI enabled in DEEP-SLEEP mode.
9	LSIAON(1)	LSI enablement in DEEP-SLEEP mode	
		0	Automatically disables the LSI in DEEP-SLEEP mode.
		1	Leaves the LSI enabled in DEEP-SLEEP mode.

8	VDCAON	VDC enablement in DEEP-SLEEP mode	
		0	Automatically disables the VDC in DEEP-SLEEP mode.
		1	Leaves the VDC enabled in DEEP-SLEEP mode.
5 4	PREVMODE[1:0]	The operating mode when the last reset event occurred.	
		00	The microcontroller was in RUN mode.
		01	The microcontroller was in SLEEP mode.
		10	The microcontroller was in DEEP-SLEEP mode.
		11	The microcontroller was in INITIAL mode.

NOTES:

1. Even if you set the SCU_SMR register for a clock to stay enabled during DEEP-SLEEP mode, the clock will not run if it has been set disabled in the SCU_CSCR register.
2. If the PLLAON bit is set to '1', the VDC stays active regardless of the setting of the VDCAON bit.
3. When the WDT or FRT clock source is set to the PLL and the PLLAON bit in the SCU_SMR register enabled, the A34M420 cannot wakeup if it enters DEEP-SLEEP mode. In this case, the A34M420 must use SLEEP mode instead of DEEP-SLEEP mode.

4.8.5 SCU_SRCR: System Reset Control Register

It is possible to reset the microcontroller by setting the SWRST bit. This system reset control register is a 32-bit register.

SCU_SRCR=0x4000_0008

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
Reserved				STBYO	Reserved	SWRST
-				0	-	0
-				RW	-	WO

4	STBYO	Inversion selection for the STBYO pin's output	
		0	Low active when the chip is in power down.
		1	High active when the chip is in power down.
0	SWRST(1)	Internal soft reset activation bit (Check SWRST in SCU_RSER for reset)	
		0	Normal operation
		1	Internal soft resets can occur (The bit becomes automatically cleared).

NOTE:

1. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset.

4.8.6 SCU_WUER: Wake-up Source Enable Register

The SCU_WUER register is used when the microcontroller is in DEEP-SLEEP mode. To enable a signal used as a wake-up source, users must set the corresponding bit to '1' and the remaining bits not used as a wake-up source must be set to '0'. This register is 32 bits wide.

SCU_WUER=-0x4000_0010

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
Reserved			GPIOGWUE	GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	Reserved	FRT1WUE	FRT0WUE	WDTWUE	LVIWUE
-			0	0	0	0	0	0	0	-	0	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW

14	GPIOGWUE	Use the GPIOG port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
13	GPIOFWUE	Use the GPIOF port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
12	GPIOEWUE	Use the GPIOE port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
11	GPIODWUE	Use the GPIOD port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
10	GPIOCWUE	Use the GPIOC port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
9	GPIOBWUE	Use the GPIOB port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
8	GPIOAWUE	Use the GPIOA port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
3	FRT1WUE	Use the FRT1 event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
2	FRT0WUE	Use the FRT0 event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
1	WDTWUE	Use the WDT event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
0	LVIWUE	Use the LVI event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.

4.8.7 SCU_WUSR: Wake-up Source Status Register

When the microcontroller is woken up by a wake-up source, the corresponding bit in the SCU_WUSR register is flagged. A bit set to '1' means that its corresponding wake-up event has occurred. It is a read-only register. When an event source is cleared, the corresponding bit in this register is also cleared.

SCU_WUSR=-0x4000_0014

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
Reserved			GPIOGWU	GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	Reserved	FRT1WU	FRT0WU	WDTWU	LVIWU
-			0	0	0	0	0	0	0	-	0	0	0	0
-			R0	R0	R0	R0	R0	R0	R0	-	R0	R0	R0	R0

14	GPIOGWU	GPIOG port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
13	GPIOFWU	GPIOF port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
12	GPIOEWU	GPIOE port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
11	GPIODWU	GPIOD port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
10	GPIOCWU	GPIOC port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
9	GPIOBWU	GPIOB port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
8	GPIOAWU	GPIOA port event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
3	FRT1WU	FRT1 event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
2	FRT0WU	FRT0 event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
1	WDTWU	Watchdog Timer (WDT) event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.
0	LVIWU	LVI event Wake-up status
	0	Wake-up event is not occurred.
	1	Wake-up event occurs.

4.8.8 SCU_RSER: Reset Source Enable Register

The SCU_RSER allows users to configure the reset input signals that trigger a reset event. Setting a bit to '1' enables the corresponding reset input signal to trigger a reset; setting it to '0' disables the reset input signal, thereby masking the reset event.

SCU_RSER=0x4000_0018

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
Reserved			LOCKUPR	Reserved	PINRST	CPURST	SWRST	WDTRST	MCKFRST	LSEFRST	HSEFRST	LVDRST
-			0	-	1	1	0	1	0	0	0	1
-			RW	-	RW	RW	RW	RW	RW	RW	RW	RW

9	LOCKUPRST	Enable the CPU lock-up reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
7	PINRST	Enable the external pin reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
6	CPURST	Enable the CPU request reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
5	SWRST	Enable the software reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
4	WDTRST	Enable the WDT reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
3	MCKFRST	Enable the MCLK error reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
2	LSEFRST	Enable the LSE error reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
1	HSEFRST	Enable the HSE error reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.
0	LVDRST	Enable the LVD reset signal	
		0	Disables the signal to trigger a reset event.
		1	Enables the signal to trigger a reset event.

4.8.9 SCU_RSSR: Reset Source Status Register

The SCU_RSSR register records the occurrences of the reset events. A bit read as '1' means that its corresponding reset source has triggered a reset event. To clear the flag and the reset status of a reset source, write a '1' to the corresponding flag bit in this register. This register is 32 bits wide.

SCU_RSSR=0x4000_001C

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
Reserved			LOCKUPRST	PORST	PINRST	CPURST	SWRST	WDTRST	MCLKFRST	LSERST	HSERST	LVDRST
-			0	1	0	0	0	0	0	0	0	1
-			RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

9	LOCKUPRST	Indicates the CPU lock-up reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
8	PORST	Indicates the POR reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
7	PINRST	Indicates the external pin reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
6	CPURST	Indicates the CPU core reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
5	SWRST	Indicates the software reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
4	WDTRST	Indicates the WDT reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
3	MCLKFRST	Indicates the MCLK error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
2	LSEFRST	Indicates the LSE error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.

1	HSEFRST	Indicates the HSE error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
0	LVDRST	Indicates the LVD reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.

4.8.10 SCU_PRER1: Peripheral Reset Enable Register 1

The SCU_PRERx registers determine whether to allow each peripheral to be initialized when a warm reset event occurs. They consist of SCU_PRER1 and SCU_PRER2. Once a reset event occurs, the peripherals whose corresponding bits are set to '1' are initialized and those set to '0' remain uninitialized. However, the POR reset initializes all peripherals.

SCU_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
Reserved						TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	DFMC	WDT	CFMC	SCU
-						1	1	1	1	1	1	1	1	1	1	-	1	1	1	1	1	1	1	1	1	-	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	RW	RW

25	TIMER9	TIMER9 reset mask	
		0	Disabled
		1	Enabled
24	TIMER8	TIMER8 reset mask	
		0	Disabled
		1	Enabled
23	TIMER7	TIMER7 reset mask	
		0	Disabled
		1	Enabled
22	TIMER6	TIMER6 reset mask	
		0	Disabled
		1	Enabled
21	TIMER5	TIMER5 reset mask	
		0	Disabled
		1	Enabled
20	TIMER4	TIMER4 reset mask	
		0	Disabled
		1	Enabled
19	TIMER3	TIMER3 reset mask	
		0	Disabled
		1	Enabled
18	TIMER2	TIMER2 reset mask	
		0	Disabled
		1	Enabled

17	TIMER1	TIMER1 reset mask
		0 Disabled
		1 Enabled
16	TIMER0	TIMER0 reset mask
		0 Disabled
		1 Enabled
11	GPIOD	GPIOD reset mask
		0 Disabled
		1 Enabled
10	GPIOC	GPIOC reset mask
		0 Disabled
		1 Enabled
9	GPIOB	GPIOB reset mask
		0 Disabled
		1 Enabled
8	GPIOA	GPIOA reset mask
		0 Disabled
		1 Enabled
7	FRT1	FRT1 reset mask
		0 Disabled
		1 Enabled
6	FRT0	FRT0 reset mask
		0 Disabled
		1 Enabled
4	DMA	DMA reset mask
		0 Disabled
		1 Enabled
3	DFMC	DFMC reset mask
		0 Disabled
		1 Enabled
2	WDT	WDT reset mask
		0 Disabled
		1 Enabled
1	CFMC	CFMC reset mask
		0 Disabled
		1 Enabled
0	SCU	SCU reset mask
		0 Disabled
		1 Enabled

4.8.11 SCU_PRER2: Peripheral Reset Enable Register 2

The SCU_PRER2 register is 32 bits wide.

SCU_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
RNG	Reserved	CRC	Reserved	ADC2	ADC1	ADC0	Reserved	MPWM1	MPWM0	Reserved	UART5	UART4	UART3	UART2	UART1	UART0	Reserved	I2C1	I2C0	Reserved	SPI2	SPI1	SPI0
1		1	-	1	1	1	-	1	1	-	1	1	1	1	1	1	-	1	1	-	1	1	1
RW		RW	-	RW	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW	-	RW	RW	-	RW	RW	RW

31	RNG	RNG reset mask
		0 Disabled
		1 Enabled
29	CRC	CRC reset mask
		0 Disabled
		1 Enabled
22	ADC2	ADC2 reset mask
		0 Disabled
		1 Enabled
21	ADC1	ADC1 reset mask
		0 Disabled
		1 Enabled
20	ADC0	ADC0 reset mask
		0 Disabled
		1 Enabled
17	MPWM1	MPWM1 reset mask
		0 Disabled
		1 Enabled
16	MPWM0	MPWM0 reset mask
		0 Disabled
		1 Enabled
13	UART5	UART5 reset mask
		0 Disabled
		1 Enabled
12	UART4	UART4 reset mask
		0 Disabled
		1 Enabled
11	UART3	UART3 reset mask
		0 Disabled
		1 Enabled
10	UART2	UART2 reset mask
		0 Disabled
		1 Enabled
9	UART1	UART1 reset mask
		0 Disabled
		1 Enabled
8	UART0	UART0 reset mask
		0 Disabled
		1 Enabled
5	I2C1	I2C1 reset mask

		0	Disabled
		1	Enabled
4	I2C0	I2C0 reset mask	
		0	Disabled
		1	Enabled
2	SPI2	SPI2 reset mask	
		0	Disabled
		1	Enabled
1	SPI1	SPI1 reset mask	
		0	Disabled
		1	Enabled
0	SPI0	SPI0 reset mask	
		0	Disabled
		1	Enabled

4.8.12 SCU_PER1: Peripheral Enable Register 1

To enable a peripheral, users must write a '1' to its corresponding bit in SCU_PER1 or SCU_PER2. Before it is enabled, the peripheral is reset.

Upon initial reset, only a few specific peripherals will be enabled, so using other peripherals will require additional work to enable them. Peripherals can be reset by writing '0' to the bit representing an unused peripheral.

SCU_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
Reserved	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	Reserved
-	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	RW	RW	RW	RW	RW	RW	RW	-	RW	-

25	TIMER9	Enable TIMER9
		0 Disabled
		1 Enabled
24	TIMER8	Enable TIMER8
		0 Disabled
		1 Enabled
23	TIMER7	Enable TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Enable TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Enable TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Enable TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Enable TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Enable TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Enable TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Enable TIMER0
		0 Disabled
		1 Enabled
12	GPIOG	Enable GPIOG
		0 Disabled
		1 Enabled
11	GPIOF	Enable GPIOF

		0 Disabled
		1 Enabled
10	GPIOE	Enable GPIOE
		0 Disabled
		1 Enabled
11	GPIOD	Enable GPIOD
		0 Disabled
		1 Enabled
10	GPIOC	Enable GPIOC
		0 Disabled
		1 Enabled
9	GPIOB	Enable GPIOB
		0 Disabled
		1 Enabled
8	GPIOA	Enable GPIOA
		0 Disabled
		1 Enabled
7	FRT1	Enable FRT1
		0 Disabled
		1 Enabled
6	FRT0	Enable FRT0
		0 Disabled
		1 Enabled
4	DMA	Enable DMA
		0 Disabled
		1 Enabled

4.8.13 SCU_PER2: Peripheral Enable Register 2

The SCU_PER2 register is 32bits wide.

SCU_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
RNG	Reserved	CRC	Reserved	ADC2 ADC1 ADC0 Reserved MPWM1 MPWM0	Reserved UART5 UART4 UART3 UART2 UART1 UART0	Reserved	I2C1 I2C0 Reserved SPI2 SPI1 SPI0
0	-	0	-	0 0 0 - 0 0	- 0 0 0 0 0 1	-	0 0 - 0 0 1
RW	-	RW	-	RW RW RW - RW RW	- RW RW RW RW RW RW	-	RW RW - RW RW RW

31	RNG	Enable RNG
		0 Disabled
		1 Enabled
29	CRC	Enable CRC
		0 Disabled
		1 Enabled
22	ADC2	Enable ADC2
		0 Disabled
		1 Enabled
21	ADC1	Enable ADC1
		0 Disabled
		1 Enabled
20	ADC0	Enable ADC0
		0 Disabled
		1 Enabled
17	MPWM1	Enable MPWM1
		0 Disabled
		1 Enabled
16	MPWM0	Enable MPWM0
		0 Disabled
		1 Enabled
13	UART5	Enable UART5
		0 Disabled
		1 Enabled
12	UART4	Enable UART4
		0 Disabled
		1 Enabled
11	UART3	Enable UART3
		0 Disabled
		1 Enabled
10	UART2	Enable UART2
		0 Disabled
		1 Enabled
9	UART1	Enable UART1
		0 Disabled
		1 Enabled
8	UART0	Enable UART0
		0 Disabled
		1 Enabled
5	I2C1	Enable I2C1

		0	Disabled
		1	Enabled
4	I2C0	Enable I2C0	
		0	Disabled
		1	Enabled
2	SPI2	Enable SPI2	
		0	Disabled
		1	Enabled
1	SPI1	Enable SPI1	
		0	Disabled
		1	Enabled
0	SPI0	Enable SPI0	
		0	Disabled
		1	Enabled

4.8.14 SCU_PCER1: Peripheral Clock Enable Register 1

To enable a specific clock for a peripheral, users must write '1' to its corresponding bit in the SCU_PCER1 or SCU_PCER2 registers. A peripheral cannot reset or operate normally until it is clocked.

If a peripheral will not be used, the users must write '0' to its corresponding bit in the SCU_PCER1 or SCU_PECR2 registers to disable its clock. When the microcontroller is in SLEEP or DEEP-SLEEP mode, the clock is fed only to active peripherals.

SCU_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
Reserved	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	Reserved
-	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	-	0	1111
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RO

25	TIMER9	Enable the clock to TIMER9
		0 Disabled
		1 Enabled
24	TIMER8	Enable the clock to TIMER8
		0 Disabled
		1 Enabled
23	TIMER7	Enable the clock to TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Enable the clock to TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Enable the clock to TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Enable the clock to TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Enable the clock to TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Enable the clock to TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Enable the clock to TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Enable the clock to TIMER0
		0 Disabled
		1 Enabled
14	GPIOG	Enable the clock to GPIOG
		0 Disabled
		1 Enabled
13	GPIOF	Enable the clock to GPIOF

		0	Disabled
		1	Enabled
12	GPIOE	Enable the clock to GPIOE	
		0	Disabled
		1	Enabled
11	GPIOD	Enable the clock to GPIOD	
		0	Disabled
		1	Enabled
10	GPIOC	Enable the clock to GPIOC	
		0	Disabled
		1	Enabled
9	GPIOB	Enable the clock to GPIOB	
		0	Disabled
		1	Enabled
8	GPIOA	Enable the clock to GPIOA	
		0	Disabled
		1	Enabled
7	FRT1	Enable the clock to FRT1	
		0	Disabled
		1	Enabled
6	FRT0	Enable the clock to FRT0	
		0	Disabled
		1	Enabled
4	DMA	Enable the clock to DMA	
		0	Disabled
		1	Enabled

4.8.15 SCU_PCER2: Peripheral Clock Enable Register 2

To use peripheral unit, its clock should be activated by writing '1' to the corresponding bit in the SCU_PCER2 register.

SCU_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
RNG	Reserved	CRC	Reserved		ADC2	ADC1	ADC0	Reserved	MPWM1	MPWM0	Reserved	UART5	UART4	UART3	UART2	UART1	UART0	Reserved	I2C1	I2C0	Reserved	SPI2	SPI1	SPI0
0	-	0	-		0	0	0	-	0	0	-	0	0	0	0	0	1	-	0	0	-	0	0	1
RW	-	RW	-		RW	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW	-	RW	RW	-	RW	RW	RW

31	RNG	Enable the clock to RNG
		0 Disabled
		1 Enabled
29	CRC	Enable the clock to CRC
		0 Disabled
		1 Enabled
22	ADC2	Enable the clock to ADC2
		0 Disabled
		1 Enabled
21	ADC1	Enable the clock to ADC1
		0 Disabled
		1 Enabled
20	ADC0	Enable the clock to ADC0
		0 Disabled
		1 Enabled
17	MPWM1	Enable the clock to MPWM1
		0 Disabled
		1 Enabled
16	MPWM0	Enable the clock to MPWM0
		0 Disabled
		1 Enabled
13	UART5	Enable the clock to UART5
		0 Disabled
		1 Enabled
12	UART4	Enable the clock to UART4
		0 Disabled
		1 Enabled
11	UART3	Enable the clock to UART3
		0 Disabled
		1 Enabled
10	UART2	Enable the clock to UART2
		0 Disabled
		1 Enabled
9	UART1	Enable the clock to UART1
		0 Disabled
		1 Enabled
8	UART0	Enable the clock to UART0
		0 Disabled

		1	Enabled
5	I2C1	Enable the clock to I2C1	
		0	Disabled
		1	Enabled
4	I2C0	Enable the clock to I2C0	
		0	Disabled
		1	Enabled
2	SPI2	Enable the clock to SPI2	
		0	Disabled
		1	Enabled
1	SPI1	Enable the clock to SPI1	
		0	Disabled
		1	Enabled
0	SPI0	Enable the clock to SPI0	
		0	Disabled
		1	Enabled

4.8.16 SCU_CSCR: Clock Source control register

The A34M420 includes a number of different clock sources for generating internal operation clocks. By configuring the SCU_CSCR register, each of the clock sources can be enabled. This register is 32 bits wide.

SCU_CSCR=0x4000_0040

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
Reserved			LSECON	Reserved	LSICON	Reserved	HSICON	Reserved	HSECON	Reserved
-			0	-	1	-	1	-	0	-
-			RW	-	RW	-	RW	-	RW	-

7	LSECON[1]	Low-speed external oscillator control	
		0	Disables the low-speed external oscillator.
		1	Enables the low-speed external oscillator.
5	LSICON	Low-speed internal oscillator control	
		0	Disables the low-speed internal oscillator.
		1	Enables the low-speed internal oscillator.
3	HSICON	High-speed internal oscillator control	
		0	Disables the high-speed internal oscillator.
		1	Enables the high-speed internal oscillator.
1	HSECON[2]	High-speed external oscillator control	
		0	Disables the high-speed external oscillator.
		1	Enables the high-speed external oscillator.

NOTES:

1. For the LSE operation, the corresponding Pn_MR registers of the SXIN, SXOUT ports must be set to '111'. LSE (PD0, PD1)
2. For the HSE operation, the corresponding Pn_MR registers of the XIN, XOUT ports must be set to '111'. HSE (PC12, PC13).

4.8.17 SCU_SCCR: System Clock Control Register

By configuring the SCU_SCCR register, users can select one from many different clock sources as the system clock.

SCU_SCCR=0x4000_0044

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
Reserved	HCLKDIV[3:0]	Reserved	PCLKDIV[2:0]	Reserved	PLLCLKSEL	Reserved	PLLPREDIV[1:0]	Reserved	MCLKSEL[2:0]
-	0011	-	000	-	0	-	00	-	010
-	RW	-	RW	-	RW	-	RW	-	RW

27 24	HCLKDIV[3:0]	Clock divider for HCLK input	
		0000	HCLK = MCLK
		0001	HCLK = MCLK / 2
		0010	HCLK = MCLK / 4
		0011	HCLK = MCLK / 8
		0100	HCLK = MCLK / 16
		0101	HCLK = MCLK / 32
		0110	HCLK = MCLK / 64
		0111	HCLK = MCLK / 128
		1000	HCLK = MCLK / 256
		1001	HCLK = MCLK / 512
		Other	Reserved
18 16	PCLKDIV[2:0]	Clock divider for PCLK input	
		000	PCLK = HCLK
		001	PCLK = HCLK / 2
		010	PCLK = HCLK / 4
		011	PCLK = HCLK / 8
		100	PCLK = HCLK / 16
		Other	Reserved
12	PLLCLKSEL(2)	PLL clock selection	
		0	Selects the HSI as the clock source for the PLL.
		1	Selects the HSE as the clock source for the PLL.
9 8	PLLPREDIV[1:0]	Clock divider for PLL input	
		00	FXIN = PLLCLKSEL / 1
		01	FXIN = PLLCLKSEL / 2
		10	FXIN = PLLCLKSEL / 4
		11	FXIN = PLLCLKSEL / 8
2 0	MCLKSEL[2:0](1)	Main system clock selection	
		000	LSI
		001	LSE(3)
		010	HSI
		110	HSE
		111	PLL
		Others	Reserved

NOTES:

1. When changing the PLLCLKSEL bits, both the HSI and HSE must be enable. If the MCLKSEL[2:0] bits are set to use

the HSE, PC12 and PC13 must be set for the functions of XIN and XOUT respectively.

2. When changing the MCLKSEL[2:0] bits, both clock sources should be alive. Ex) Both HSI and HSE should be alive, otherwise the chip will malfunction.
3. If the MCLKSEL[2:0] bits are set to use the LSE, PD2 and PD3 must be set for the functions of SXIN and SXOUT respectively.

4.8.18 SCU_CMR: Clock Monitoring Register

The SCU_CMR register is used to monitor internal clocks for protection purposes by using the LSI oscillator. The SCU_CMR register is 32 bits wide.

SCU_CMR=0x4000_0048

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
Reserved			LSEMNT	LSEIE	LSEFAIL	LSESTS	MCLKMNT	Reserved	HSEMNT	HSEIE	HSEFAIL	HSESTS
-			0	0	0	0	0	-	0	0	0	0
-			RW	RW	RWC1	RO	RW	-	RW	RW	RWC1	RO

11	LSEMNT	Enable external LSE monitoring	
		0	Disables external LSE monitoring.
		1	Enables external LSE monitoring.
10	LSEIE	Enable the external LSE error interrupt	
		0	Disables the external LSE error interrupt.
		1	Enables the external LSE error interrupt.
9	LSEFAIL	External LSE error interrupt flag	
		0	The external LSE error interrupt has not occurred.
		1	Read: The external LSE error interrupt has occurred. Write: Clears the interrupt flag.
8	LSESTS	External LSE status flag	
		0	The external LSE is oscillating normally.
		1	The external LSE is not oscillating.
7	MCLKMNT	Enable MCLK monitoring	
		0	Disables MCLK monitoring.
		1	Enables MCLK monitoring.
3	HSEMNT	Enable external HSE monitoring	
		0	Disables external HSE monitoring.
		1	Enables external HSE monitoring.
2	HSEIE	Enable the external HSE error interrupt	
		0	Disables the external HSE error interrupt.
		1	Enables the external HSE error interrupt.
1	HSEFAIL	External HSE error interrupt flag	
		0	The external HSE error interrupt has not occurred.
		1	Read: The external HSE error interrupt has occurred. Write: Clears the interrupt flag.
0	HSESTS	External HSE status flag	
		0	The external HSE is oscillating normally.
		1	The external HSE is not oscillating.

NOTE:

1. Clock monitoring can be active only when the corresponding clock is enabled.

4.8.19 SCU_COR: Clock Output Register

The SCU_COR register defines the frequency division ratio at which the internal MCLK is fed to external devices. To use the CLKO pin in output mode, users must configure the CLKO to output mode in the pin mux. This register is 32 bits wide.

SCU_COR=0x4000_0050

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
Reserved			CLKOINSEL[2:0]	CLKOEN	CLKODIV[3:0]
-			000	0	1111
-			RW	RW	RW

7 5	CLKOINSEL[2:0]	Selection of the clock to send a signal from CLKO	
		000	LSI
		100	MCLK
		101	HSI
		110	HSE
		111	PLL
		Other	Reserved
4	CLKOEN	Clock output use	
		0	Not use for the CLKO (maintains the ‘Low’ signal output.)
		1	Use for the CLKO.
3 0	CLKODIV[3:0]	Clock output divider value.	
		CLKO = MCLK, when CLKODIV[3:0] = ‘0’ CLKO = MCLK / (CLKODIV[3:0] × 2), when CLKODIV[3:0] > 0	

4.8.20 SCU_NMICR: NMI Control Register

The SCU_NMICR register controls NMIs (Non-Maskable Interrupts), which are software configurable. Any interrupt can be an NMI source. Once an NMI event occurs, the microcontroller jumps to the NMI handler.

SCU_NMICR=0x4000_0054

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
Reserved	NMISRC[7:0]	NMIINEN	Reserved	DFEDEN	CFEDEN	Reserved	PROT1EN	OVP1EN	PROT0EN	OVP0EN	WDTINTEN	Reserved	LVIEN	
-	0x00	0	-	0	0	-	0	0	0	0	0	-	0	
-	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	-	RW	

23 16	NMISRC[7:0]	NMI source selection bit One of the interrupt priority levels can be used as the NMI source.
15	NMIINEN	Enable the NMIs (Interrupts that have been specified as NMI sources) 0 Disables. 1 Enables.
13	DFEDEN	Enable the Data Flash ECC Double error interrupt as an NMI 0 Disables 1 Enables.
12	CFEDEN	Enable the Code Flash ECC Double error interrupt as an NMI 0 Disables 1 Enables.
6	PROT1EN	Enable the MPWM1 protection interrupt as an NMI 0 Disables 1 Enables.
5	OVP1EN	Enable the MPWM1 overvoltage protection interrupt as an NMI 0 Disables
4	PROT0EN	Enable the MPWM0 protection interrupt as an NMI 0 Disables 1 Enables.
3	OVP0EN	Enable the MPWM0 overvoltage protection interrupt as an NMI 0 Disables 1 Enables.
2	WDTINTEN	Enable the WDT interrupt as an NMI 0 Disables 1 Enables.
0	LVIEN	Enable the LVI error interrupt as an NMI 0 Disables 1 Enables.

4.8.21 SCU_NMISR: NMI Status Register

The SCU_NMISR register is used to clear the NMI occurrence flags. Individual interrupt flags can be cleared after writing a specific key value to the identification key bit field.

SCU_NMISR=0x4000_0058

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
WTIDKY[7:0]	Reserved	NMIINTSTS	Reserved	DFEDSTS	CFEDSTS	Reserved		PRO1TSTS	OVP1STS	PROT0STS	OVP0STS	WDTINTSTS	Reserved	LVISTS
0x00	-	0	-	0	0	-		0	0	0	0	0	-	0
WO	-	RO	-	RWC1	RWC1	-		RWC1	RWC1	RWC1	RWC1	RWC1	-	RWC1

31 24	WTIDKY[7:0]	Write identification key Writing 0x8C to the bit field enables clearing flags in this register.
15	NMIINTSTS	Whether an NMI has occurred (the flag can be cleared by writing to the triggered NMI's bit.) 0 No NMI has occurred. 1 An NMI has occurred.
13	DFEDSTS	Indicates the Data Flash ECC Double error interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a 1 to the bit clears the flag).
12	CFEDSTS	Indicates the Code Flash ECC Double error interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a 1 to the bit clears the flag).
6	PROT1STS	Indicates the MPWM1 protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
5	OVP1STS	Indicates the MPWM1 overvoltage protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
4	PROT0STS	Indicates the MPWM0 protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
3	OVP0STS	Indicates the MPWM0 overvoltage protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
2	WDTINTSTS	Indicates the WDT interrupt has occurred (to ensure that the bit is flagged only once, the timer must be reloaded before the flag is cleared). 0 The interrupt has not occurred. 1 The interrupt has occurred (Writing a '1' to the bit clears the flag).
0	LVISTS	Indicates the LVI interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).

4.8.22 SCU_PLLCON: PLL Control Register

Integrated PLL will synthesize high speed clock for high performance of CPU. The PLL is controlled by setting the register. The PLL control register is 32-bit register.

SCU_PLLCON=0x4000_0060

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
PLLLOCK	Reserved	CTRLOTP[3:0]	PLLRSTB	PLLEN	BYPASSB	PLLMODE	Reserved	PREDIV[2:0]	POSTRIV1[7:0]	POSTDIV2[3:0]	OUTDIV[3:0]
0	-	0110	0	0	0	0	-	000	0x00	0000	0000
RO	-	RW	RW	RW	RW	RW	-	RW	RW	RW	RW

31	PLLLOCK	PLL Lock status bit			
		0	The PLL is in unlock state.		
		1	The PLL is in lock state.		
27 24	CTRLOTP[3:0]	PLL current option			
		[27:26]	Current option	[25:24]	VCO bias
		00	5 μA	00	× 1/4
		01	10 μA	01	× 1/2
		10	15 μA	10	× 1
		11	20 μA	11	× 2
23	PLLRSTB	PLL reset			
		0	PLL reset signal applied.		
		1	PLL reset signal released (PLL operation).		
22	PLLEN	PLL Enable bit			
		0	PLL is disabled		
		1	PLL is enabled		
21	BYPASSB	Decide whether to bypass the PLL input clock.			
		0	Bypasses the PLL input clock.		
		1	Uses the PLL multiplier clock (Does not bypass).		
20	PLLMODE	PLL voltage-controlled oscillator (VCO) mode selection			
		0	The VCO frequency is the same as FOUT.		
		1	The VCO frequency is twice FOUT.		
18 16	PREDIV[2:0]	FIN predivider bit			
		0 to 7	PLLINCLK divided by (PREDIV + 1), (PLLINCLK / 1 to PLLINCLK / 8)		
15 8	POSTDIV1[7:0]	Feedback control 1 (N1)			
		N1 value in the formula (N1 = 0x00 to 0xFF)			
7 4	POSTDIV2[3:0]	Feedback control 1 (N2)			
		N2 value in the formula (N2 = 0x0 to 0xF)			
3 0	OUTDIV[3:0]	Output divider control (P)			
		P value in the formula (P = 0x0 to 0xF)			

NOTES:

1. Refer to section 4.4.5 for how to set the PLL clock using the SCU_PLLCON register.
2. When setting the PLL current option, CTRLOTP[3:0], it is recommended to set the current option to '01' (10 μA) and the VCO bias to '10" (×1).

4.8.23 SCU_VDCCON: VDC Control Register

The SCU_VDCCON register controls the VDC, which represents the microcontroller's internal voltage. The VDCWDLY[7:0] bit field defines the length of the SCU's warm-up delay. This register is 32 bits wide.

SCU_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
Reserved			VDCWDLY[7:0]
-			0x7F
-			RW

7 0	VDCWDLY[7:0]	VDC Warm-up delay count value When the SCU wakes up from DEEP-SLEEP mode, the Warm-up delay is inserted after waiting until the VDC output is stabilized. Default delay value is 0x7F (2 ms). The amount of delay can be set with this register up to 0xFF (4 ms).

NOTE:

1. Reserved bits should never be modified.

4.8.24 SCU_LVICR: LVI (Low-Voltage Indicator) Control Register

The LVI block detects low voltage by 16 levels. When it detects a specified level of low voltage which was predefined in SCU_LVICR register, an LVI interrupt and a flag are generated. This function prevents system malfunction caused by unstable voltage supply and steadily supports to drive user applications.

SCU_LVICR=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
Reserved			LVIEN	Reserved	LVIINTEN	LVIAON	LVIVS[3:0]
-			0	-	0	0	0000
-			RW	-	RW	RW	RW

7	LVIEN	Enable the LVI	
		0	Disable
		1	Enable
5	LVIINTEN	Enable the LVI interrupt.	
		0	Disable
		1	Enable
4	LVIAON	Whether to deactivate the LVI in DEEP-SLEEP mode	
		0	Disables LVI auto-off.
		1	Enables the LVI to automatically turn off when the microcontroller enters DEEP-SLEEP mode.
3 0	LVIVS[3:0]	LVI interrupt voltage selection bits	
		0000	1.60 V
		0001	1.69 V
		0010	1.78 V
		0011	1.90 V
		0100	1.99 V
		0101	2.12 V
		0110	2.30 V
		0111	2.47 V
		1000	2.67 V
		1001	3.04 V
		1010	3.18 V
		1011	3.59 V
		1100	3.72 V
		1101	4.03 V
		1110	4.20 V
		1111	4.48 V

NOTE:

1. Refer to section 4.6.1 and Figure 26

4.8.25 SCU_LVISR: LVI (Low-Voltage Indicator) Status Register

The SCU_LVISR register is 32 bits wide and indicates the LVI's operating status.

SCU_LVISR=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
WTIDKY[7:0]	Reserved			LVIIFLAG	Reserved	LVIINTSTS
0x00	-			0	-	0
WO	-			RWC1	-	RO

31 24	WTIDKY[7:0]	Write identification key	
		Writing '0x7A' to the bit field enables clearing flags in this register.	
5	LVIFLAG	LVI interrupt flag	
		0	Not flagged
		1	Flagged (Writing a '1' to clear the flagged bit).
0	LVIINTSTS	Indicates the LVI interrupt is active (Raw data)	
		0	The LVI interrupt has ended.
		1	The LVI interrupt is active.

4.8.26 SCU_LVRCR: LVR (Low-Voltage Detect Reset) Control Register

The SCU_LVRCR register is 32 bits wide and defines the default mode for LVR reset operation.

SCU_LVRCR=0x4000_0070

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
Reserved		LVREN[7:0]	Reserved	LVRAON	LVRVS[3:0]
-		0x00	-	0	0101
-		RW	-	RW	RW

15 8	LVREN[7:0](1)	LVR reset operation control master configuration.	
		0xAA	Disable LVR reset operation
		Others	Enables LVR reset operation
4	LVRAON	Whether to deactivate the LVR reset in DEEP-SLEEP mode	
		0	Disables LVR reset auto-off.
		1	Enables the LVR reset to automatically turn off when the microcontroller enters DEEP-SLEEP mode.
0	LVRVS [3:0]	LVR voltage selection bits	
		0000	1.60 V
		0001	1.69 V
		0010	1.78 V
		0011	1.90 V
		0100	1.99 V
		0101	2.12 V
		0110	2.30 V
		0111	2.47 V
		1000	2.67 V
		1001	3.04 V
		1010	3.18 V
		1011	3.59 V
		1100	3.72 V
		1101	4.03 V
		1110	4.20 V
		1111	4.48 V

NOTES:

1. The LVDRST bit in SCU_RSER register must be enabled before the LVREN[7:0] bits are set to trigger a reset operation at low voltage.
2. See section 4.6.2.

4.8.27 SCU_EOSCR: External Oscillator Control Register

The SCU_EOSCR register defines the drive current of the external main crystal oscillator and the delay time to reduce noises. This register is 32-bit wide.

SCU_EOSCR=0x4000_0080

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
Reserved	LSEISEL[1:0]	Reserved	LSENFEN	Reserved	HSEISEL[1:0]	Reserved	HSENFEN	Reserved	HSENFSEL[1:0]
-	00	-	0	-	00	-	0	-	00
-	RW	-	RW	-	RW	-	RW	-	RW

25 24	LSEISEL[1:0]	LSE current capability selection	
		00	1.57 μA
		01	1.79 μA
		10	1.93 μA
		11	2.04 μA
16	LSENFEN	LSE noise filter enable bit	
		0	Disable
		1	Enable
9 8	HSEISEL[1:0]	HSE current capability selection	
		00	12 MHz < f_{OUT} ≤ 16 MHz
		01	8 MHz < f_{OUT} ≤ 12 MHz
		10	4 MHz < f_{OUT} ≤ 8 MHz
		11	1 MHz < f_{OUT} ≤ 4 MHz
4	HSENFEN	HSE noise filter enable bit	
		0	Disable
		1	Enable
1 0	HSENFSEL[1:0]	HSE noise filter selection	
		00	23 ns
		01	18 ns
		10	13 ns
		11	8 ns

Table 29. External Oscillator Control Settings

Frequency (MHz)	HSENFSEL[1:0]	HSEISEL[1:0]	Noise Filter Delay (ns)
4	00	11	25
8	01	10	20
12	10	01	15
16	11	00	10

4.8.28 SCU_MCCR1: Miscellaneous Clock Control Register 1

The SCU_MCCR1 register provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_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
Reserved	WDTCSEL[2:0]	WDTCDIV[7:0]	Reserved	STCSEL[2:0]	STCDIV[7:0]
-	000	0x01	-	000	0x00
-	RW	RW	-	RW	RW

25 24	WDTCSEL[2:0]	WDT clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	WDTCDIV[7:0]	WDT clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	STCSEL[2:0]	SYSTICK clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	STCDIV[7:0](1)	SYSTICK clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / (2 × N), N must be larger than zero.

NOTE:

1. **(Caution)** If the SYSTICK clock source register is set to '0' (external clock), STCDIV[7:0] cannot use a clock divider with '1', and the clock divider must use two or more. Otherwise, the SYSTICK clock will not work.

4.8.29 SCU_MCCR2: Miscellaneous Clock Control Register 2

The SCU_MCCR2 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_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
Reserved	MPWM1CSEL[2:0]	MPWM1CDIV[7:0]	Reserved	MPWM0CSEL[2:0]	MPWM0CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

25 24	MPWM1CSEL[2:0]	MPWM1 clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	MPWM1CDIV[7:0]	MPWM1 clock divider N (MPWM1 Clock ≤ 100 MHz)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	MPWM0CSEL[2:0]	MPWM0 clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	MPWM0CDIV[7:0]	MPWM1 clock divider N (MPWM0 Clock ≤ 100 MHz)	
		0x00	Disables the divider
		0xN	Clock source / (2 × N), N must be larger than zero.

4.8.30 SCU_MCCR3: Miscellaneous Clock Control Register 3

The SCU_MCCR3 register provides a different MCLK division ratio for each peripheral. If timer clock is set using the MCCR register, the timer clock must be slower than the main clock divided by two. This register is 32-bit wide.

SCU_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
Reserved	TIMER59CSEL[2:0]	TIMER59CDIV[7:0]	Reserved	TIMER04CSEL[2:0]	TIMER04CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	TIMER59CSEL[2:0]	Clock source selection for TIMER5 through TIMER9	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	TIMER59CDIV[7:0](1)	Clock divider (N) for TIMER5 through TIMER9	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than zero. If the timer clock (TCLK) is set in this register, the clock speed must satisfy PCLK ≥ TCLK × 2.
10 8	TIMER04CSEL[2:0]	Clock source selection for TIMER0 through TIMER4	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	TIMER04CDIV[7:0](1)	Clock divider (N) for TIMER0 through TIMER4	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than 0. If the timer clock (TCLK) is set in this register, the clock speed must satisfy PCLK ≥ TCLK × 2.

NOTE:

1. Because the PCLK is the default operation clock, the clock source set by the SCU_MCCRx register should be slower than the PCLK. Therefore, if the clock source set by the SCU_MCCRx register uses the same frequency as the PCLK, it should be divided more than two times. If the PCLK is 1/2 slower than the clock source set by the SCU_MCCRx register, frequency should be divided more than four times.

4.8.31 SCU_MCCR4: Miscellaneous Clock Control Register 4

The SCU_MCCR4 register provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_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
Reserved	PGADCSEL[2:0]	PGADCDIV[7:0]	Reserved	
-	000	0x00	-	
-	RW	RW	-	

26 24	PGADCSEL[2:0] (PA, PB)	Clock source for port group A (PA, PB) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGADCDIV[7:0] (PA, PB)	Clock divider N for port group A (PA, PB) debouncing	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than zero.

4.8.32 SCU_MCCR5: Miscellaneous Clock Control Register 5

The SCU_MCCR5 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_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
Reserved	PGCDCSEL[2:0]	PGCDCDIV[7:0]	Reserved	PGBDCSEL[2:0]	PGBDCDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

25 24	PGCDCSEL[2:0] (PE, PF, PG)	Clock source for port group C (PE, PF, PG) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGCDCDIV[7:0] (PE, PF, PG)	Clock divider (N) for port group C (PE, PF, PG) debouncing	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than zero.
10 8	PGBDCSEL[2:0] (PC, PD)	Clock source for port group B (PC, PD) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	PGBDCDIV[7:0] (PC, PD)	Clock divider (N) for port group B (PC, PD) debouncing	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than zero.

4.8.33 SCU_MCCR6: Miscellaneous Clock Control Register 6

The SCU_MCCR6 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_MCCR6=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
Reserved	FRT1CSEL[2:0]	FRT1CDIV[7:0]	Reserved	FRT0CSEL[2:0]	FRT0CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	FRT1CSEL[2:0]	FRT1 clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	FRT1CDIV[7:0]	FRT1 clock divider (N)	
		0x00	Disable the divider.
		0xN	Clock source / N, N must be larger than zero.
10 8	FRT1CSEL[2:0]	FRT clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	FRTCDIV[7:0]	FRT clock divider (N)	
		0x00	Disable the divider.
		other	Clock source / N, N must be larger than zero.

4.8.34 SCU_MCCR7: Miscellaneous Clock Control Register 7

The SCU_MCCR7 register provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_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
Reserved	UARTCSEL[2:0]	UARTCDIV[7:0]	Reserved
-	000	0x01	-
-	RW	RW	-

26 24	UARTCSEL[2:0]	UART clock source selection bit (UART0 through UART5)	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	UARTCDIV[7:0]	UART clock divider (N)	
		0x00	Disable the divider.
		other	Clock source / N, N must be larger than zero. If the UART clock is set in this register, the clock speed must satisfy PCLK ≥ UART clock set by SCU_MCCR7 × 2.

4.8.35 SCU_SYSTEN: System Access Enable Register

The SCU_SYSTEN register determines whether to allow the changes of the settings in all SCU registers.

SCU_SYSTEN=0x4000_00F0

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
Reserved			ENS	SYSTEN[7:0]
-			0	--
-			RO	WO

8	ENS	Enable SCU registers enable status bit	
		0	Disables the write access to SCU registers.
		1	Enables the write access to SCU registers.
7 0	SYSTEN[7:0]	SCU register access enable key. Writing ‘0x57’ and then ‘0x75’ to the bit field enables writing new values to SCU registers. After this, write a different value to this bit field to protect the SCU registers against being updated with new values. However, access to the NMISR and LVISR registers is not defined by the SYSTEM register. Their bits can be cleared regardless of SYSTEN enablement.	

4.8.36 SCU Register Map Summary

Table 30. SCU CHIPCONFIG Register Map Summary

Offset	Register name	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
0x00	CHIPCONFIG_ VENDORID	VENDID[31:0]																															
	Reset value	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	0	1	0	1	1	0
0x04	CHIPCONFIG_ CHIPID	CHIPID[31:0]																															
	Reset value	0	1	0	0	1	1	0	1	0	0	1	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x08	CHIPCONFIG_ REVNR																									REVNO[7:0]							
	Reset value																									x	x	x	x	x	x	x	x

Table 31. SCU Register Map Summary

Offset	Register name	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
0x04	SCU_SMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSEAON	HSEAON	PLLAON	HSIAON	LSIAON	VDCAON	Res.	Res.	PREVMODE[1:0]		Res.	Res.	Res.	Res.
	Reset value																			0	0	0	0	0	0			1	1				
0x08	SCU_SRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STBYO	Res.	Res.	Res.	SWRST
	Reset value																												0				0
0x10	SCU_WUER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOGWUE	GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	Res.	Res.	Res.	Res.	FRT1WUE	FRT0WUE	WDTWUE	LVIWUE
	Reset value																		0	0	0	0	0	0	0					0	0	0	0
0x14	SCU_WUSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOGWU	GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	Res.	Res.	Res.	Res.	FRT1WU	FRT0WU	WDTWU	LVIWU
	Reset value																		0	0	0	0	0	0	0					0	0	0	0
0x18	SCU_RSER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKUPRST	Res.	PINRST	CPURST	SWRST	WDTRST	MCLKRST	LSERST	HSERST	LVDRST
	Reset value																							0		1	1	0	1	0	0	0	1
0x1C	SCU_RSSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKUPRST	PORST	PINRST	CPURST	SWRST	WDTRST	MCLKRST	LSERST	HSERST	LVDRST
	Reset value																							0	1	0	0	0	0	0	0	0	1

Table 30. SCU Register Map Summary (continued)

Offset	Register name	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
0x20	SCU_PRER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	DFMC	WDT	CFMC	SCU
	Reset value							1	1	1	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1	1		1	1	1	1	1
0x24	SCU_PRER2	RNG	Res.	CRC	Res.	Res.	Res.	Res.	Res.	Res.	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	Res.	Res.	UART5	UART4	UART3	UART2	UART1	UART0	Res.	Res.	I2C1	I2C0	Res.	SPI2	SPI1	SPI0
	Reset value	1		1							1	1	1			1	1			1	1	1	1	1	1			1	1		1	1	1
0x28	SCU_PER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	Res.	Res.	Res.	Res.
	Reset value							0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0		0	1	1	1	1
0x2C	SCU_PER2	RNG	Res.	CRC	Res.	Res.	Res.	Res.	Res.	Res.	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	Res.	Res.	UART5	UART4	UART3	UART2	UART1	UART0	Res.	Res.	I2C1	I2C0	Res.	SPI2	SPI1	SPI0
	Reset value	0		0							0	0	0			0	0			0	0	0	0	0	1			0	0		0	0	1
0x30	SCU_PCER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	Res.	Res.	Res.	Res.
	Reset value							0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0		0	1	1	1	1
0x34	SCU_PCER2	RNG	Res.	CRC	Res.	Res.	Res.	Res.	Res.	Res.	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	Res.	Res.	UART5	UART4	UART3	UART2	UART1	UART0	Res.	Res.	I2C1	I2C0	Res.	SPI2	SPI1	SPI0
	Reset value	0		0							0	0	0			0	0			0	0	0	0	0	1			0	0		0	0	1
0x40	SCU_CSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSECON	Res.	LSICON	Res.	HSICON	Res.	HSECON	Res.
	Reset value																									0		1		1		0	
0x44	SCU_SCCR	Res.	Res.	Res.	Res.	HCLKDIV[3:0]				Res.	Res.	Res.	Res.	Res.	PCLKDIV[2:0]			Res.	Res.	Res.	PLLCLKSEL	Res.	Res.	PLLPREDIV[1:0]		Res.	Res.	Res.	Res.	Res.	MCLKSEL[2:0]		
	Reset value					0	0	1	1						0	0	0				0			0	0							0	1	0
0x48	SCU_CMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		LSEMNT	LSEIE	LSEFAIL	LSESTS	MCLKMNT	Res.	Res.	Res.	HSEMNT	HSEIE	HSEFAIL	HSESTS
	Reset value																					0	0	0	0	0				0	0	0	0
0x50	SCU_COR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKOINSEL[2:0]		CLKOEN		CLKODIV[3:0]			
	Reset value																									0	0	0	0	1	1	1	1

Table 30. SCU Register Map Summary (continued)

Offset	Register name	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
0x54	SCU_NMICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NMISRC[7:0]								NMIINEN	Res.	DFEDEN	CFEDEN	Res.	Res.	Res.	Res.	Res.	PROT1EN	OVP1EN	PROT0EN	OVP0EN	WDTEN	Res.	LVIEN
	Reset value									0	0	0	0	0	0	0	0	0		0	0						0	0	0	0	0		0
0x58	SCU_NMISR	WTIDKY[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NMIINTSTS	Res.	DFEDSTS	CFEDSTS	Res.	Res.	Res.	Res.	Res.	PROT1STS	OVP1STS	PROT0STS	OVP0STS	WDTINTSTS	Res.	LVISTS
	Reset value	0	0	0	0	0	0	0	0									0		0	0						0	0	0	0	0		0
0x60	SCU_PLLCON	PLLLOCK	Res.	Res.	Res.	CTRLOTP[3:0]				PLLRSTB	PLLEN	BYPASSB	PLLMODE	Res.	PREDIV[2:0]			POSTDIV1[7:0]								POSTDIV2[3:0]				OUTDIV[3:0]			
	Reset value	0				0	1	1	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	SCU_VDCCON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VDCWDLY[7:0]							
	Reset value																									0	1	1	1	1	1	1	1
0x68	SCU_LVICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVIEN	Res.	LVIINTEN	LVIAON	LVIVS[3:0]		
	Reset value																										0		0	0	0	0	0
0x6C	SCU_LVISR	WTIDKY[7:0]							Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVIIFLAG	Res.	Res.	Res.	Res.	Res.	LVIINTSTS
	Reset value	0	0	0	0	0	0	0																			0						0
0x70	SCU_LVRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVREN[7:0]								Res.	Res.	Res.	LVRAON	LVRVS[3:0]			
	Reset value																	0	0	0	0	0	0	0	0				0	0	1	0	1
0x80	SCU_EOSCR	Res.	Res.	Res.	Res.	Res.	Res.	LSEISEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSENFEN	Res.	Res.	Res.	Res.	Res.	Res.	HSEISEL[1:0]		Res.	Res.	Res.	HSENFEN	Res.	Res.	HSENFSEL[1:0]	
	Reset value							0	0								0							0	0				0			0	0
0x90	SCU_MCCR1	Res.	Res.	Res.	Res.	Res.	WDTCSEL[2:0]			WDTCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	STCSEL[2:0]			STCDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	1						0	0	0	0	0	0	0	0	0	0	0

Table 30. SCU Register Map Summary (continued)

Offset	Register name	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
0x94	SCU_MCCR2	Res.	Res.	Res.	Res.	Res.	MPWM1CSEL[2:0]			MPWM1CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	MPWM0CSEL[2:0]			MPWM0CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0x98	SCU_MCCR3	Res.	Res.	Res.	Res.	Res.	TIMER59CSEL[2:0]			TIMER59CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	IMER04CSEL[2:0]			TIMER04CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0x9C	SCU_MCCR4	Res.	Res.	Res.	Res.	Res.	PGADCSEL[2:0]			PGADCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value						0	0	0	0	0	0	0	0	0	0	0																
0xA0	SCU_MCCR5	Res.	Res.	Res.	Res.	Res.	PGCDCSEL[2:0]			PGCDCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	PGBDCSEL[2:0]			PGBDCDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0xA4	SCU_MCCR6	Res.	Res.	Res.	Res.	Res.	FRT1CSEL[2:0]			FRT1CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	FRT0CSEL[2:0]			FRT0CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0xA8	SCU_MCCR7	Res.	Res.	Res.	Res.	Res.	UARTCSE[2:0]			UARTCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value						0	0	0	0	0	0	0	0	0	0	1																
0xF0	SCU_SYSTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENS	SYSTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

5. Port Control Unit (PCU) and General-Purpose I/O (GPIO)

5.1 Introduction

5.1.1 PCU Introduction

A34M420 has a Port Control Unit (PCU) module that controls the external input and output (I/O) ports. By setting the PCU registers, users can configure the pins' uses, input/output direction, Pull-up / Pull-down resistor, and debouncing for their applications as needed.

5.1.2 GPIO Introduction

Pins except the VDD, GND, and certain specific-purpose pins can be used as General-Purpose Input/Output (GPIO) pins.

The GPIO module controls the general I/O ports. Output pins can be set to generate high- or low-level signals by configuring the corresponding bits of the GPIO control register, while logic input pins can be monitored for their input status in the control registers.

5.2 Main Features

5.2.1 PCU Features

The Port Control Unit (PCU) configures and controls external I/Os as listed below:

- MUX registers define the use of each pin.
  - Input / Output
  - Push-pull output
  - Open-drain output
  - Logic input
  - Analog input
- Internal Pull-up / down resistor and open-drain mode can be configured for each pin.
- Interrupts listed below can be set for each pin:
  - Input level interrupt
  - Input rising-edge interrupt
  - Input falling-edge interrupt
  - Input both-edge interrupt
- Up to seven GPIO interrupts are supported, including GPIOn (n = A to G).
- Each pin can be set for debouncing.

5.2.2 GPIO Features

The GPIO module controls the GPIO ports as listed below:

- Selects the output signal level.
- Enables or disables the pull-up and pull-down resistors for pins.
- External interrupt interfaces

5.3 GPIO Functional Description

Subject to the specific hardware characteristics of each I/O port listed in DC characteristics of Datasheet, each port bit of the general-purpose I/O (GPIO) ports can be individually configured by software in multiple modes:

- Input floating
- Input pull-up resistor
- Input pull-down resistor
- Analog input
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Alternative function

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in analog mode. The debug pins are in alternative function pull-up after reset:

- PC0: TCK/SWCLK with pull-up resistor
- PC1: TMS/SWDIO with pull-up resistor
- PC2: TDO/SWO with output low
- PC3: TDI with pull-up resistor
- PC4: nTRST with pull-up resistor
- PC10: nRESET with pull-up resistor
- PC11: nBOOT with pull-up resistor

Following figures (Figure 32, Figure 33 and Figure 34) show block diagrams of GPIO and external interrupt I/O pins, respectively.

Figure 32. PCU and GPIO Block Diagram

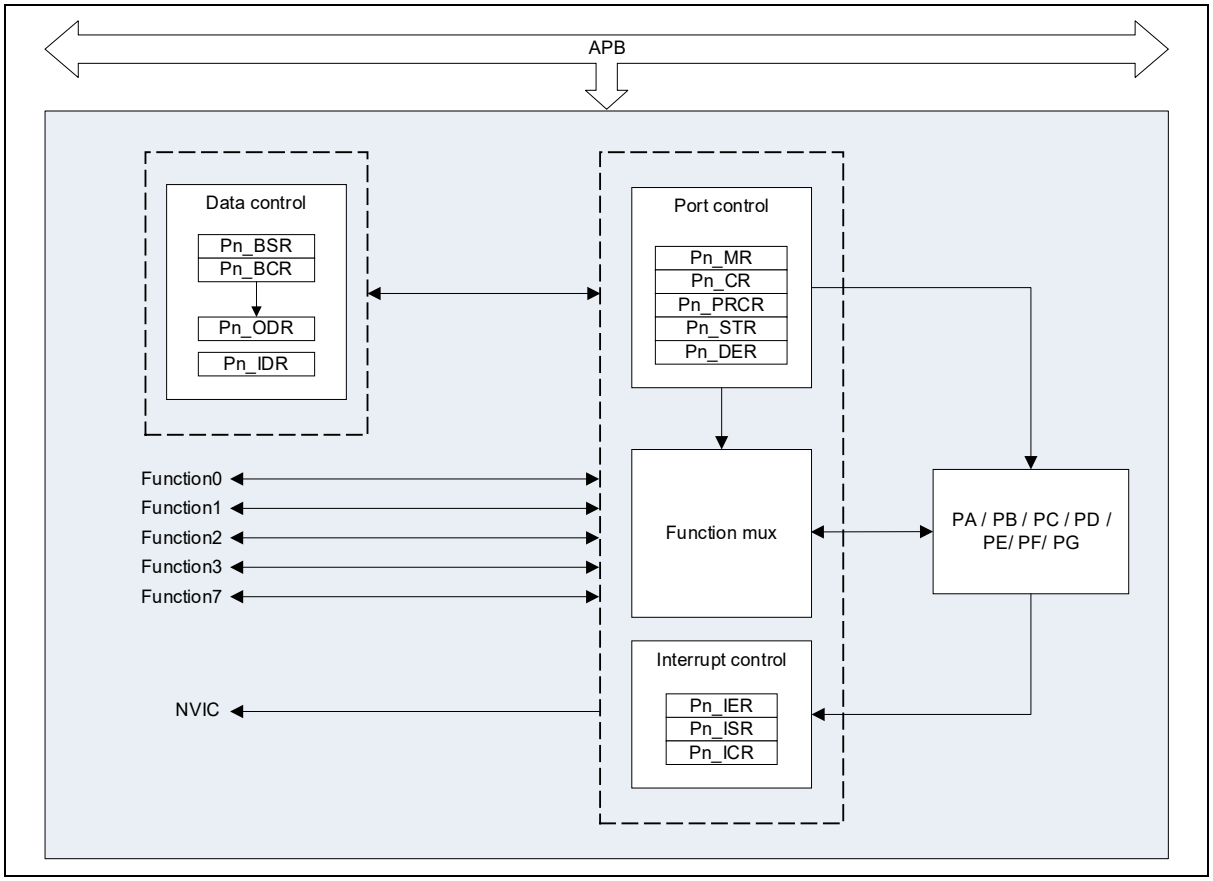


Figure 33. I/O Port Block Diagram (GPIO Pins)

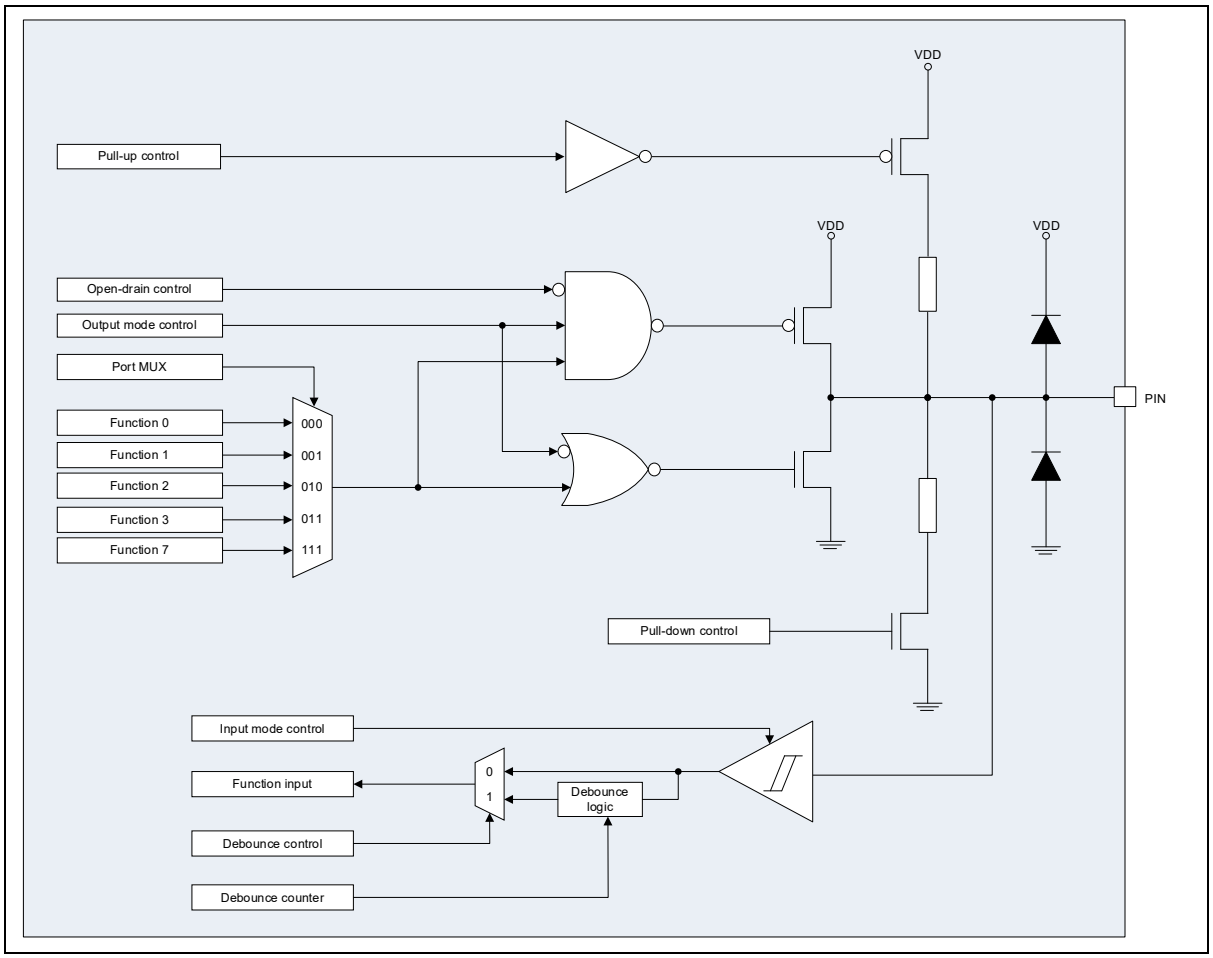
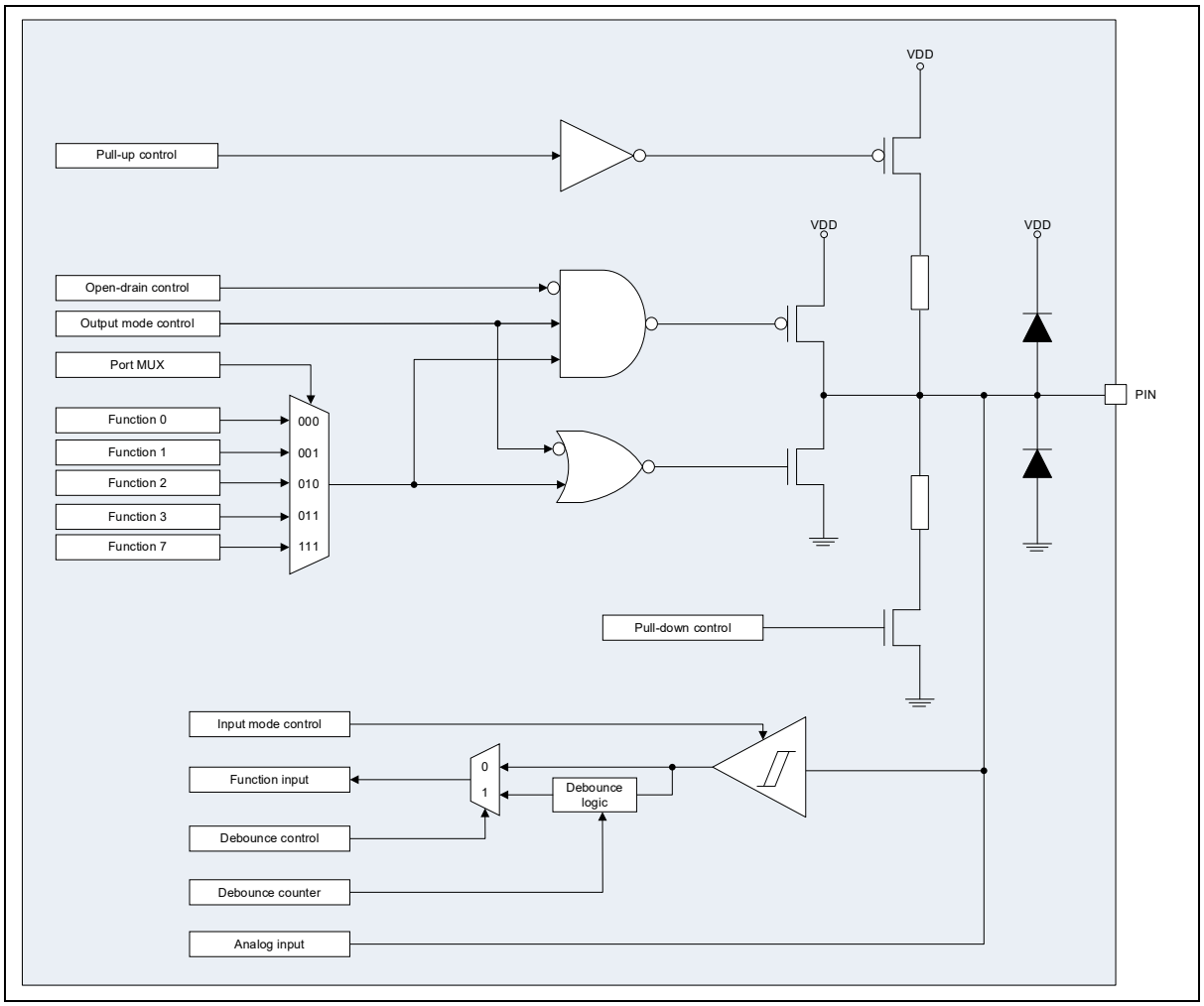


Figure 34. I/O Port Block Diagram (ADC and External Oscillator Pins)



5.3.1 GPIO Pins and Internal Signals

Table 32 describes the PCU and GPIO internal signal.

Table 32. PCU and GPIO Internal Signal

Signal Name	Signal Type	Description
PCLK	clock	APB clock

Table 33 describes seven pins assigned for PCU and GPIO module.

Table 33. PCU and GPIO Pins

Pin Name	Type	Description
PAx	IO	PA0 to PA15
PBx	IO	PB0 to PB15
PCx	IO	PC0 to PC15
PDx	IO	PD0 to PD15
PEx	IO	PE0 to PE15
PFx	IO	PF0 to PF15
PGx	IO	PG0 to PG10

5.3.2 I/O Port Control Registers

Each GPIO controls up to sixteen I/Os by using three 32-bit memory mapping control registers such as the Pn_CR and Pn_PRCR registers as shown below:

- The Pn_CR register selects the I/O mode from the Input, Output, and Output open-drain modes.
- The Pn_PRCR register controls the pull-up and pull-down resistors of all ports I/Os.

Some GPIO control registers can be updated with new data only when ‘0x15’ is written to the PORTEN[7:0](1) register and then ‘0x51’ is written again to the same register. After writing ‘0x00’ to the PORTEN[7:0] register, users cannot change the Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR register values.

NOTE:

1. Users must be careful when using PORTEN[7:0] control in other contexts such as interrupts. When an interrupt is occurred and the PORTEN[7:0] value is changed in interrupt service routine, the PORTEN[7:0] in original process could be unintentionally disabled.

5.3.3 I/O Port Data Registers

Each GPIO has two 16-bit memory mapping data registers that operate as shown below:

- The Pn_ODR register can be read and written when it has output data.
- The Pn_IDR register stores input data entered via the I/O pin. It can only be read.

5.3.4 I/O Data Bitwise Handling

The bit set/clear register (Pn_BSR) and the bit clear register (Pn_BCR) can be set or reset by the application that controls corresponding bit of the Pn_ODR register. These registers support bit banding.

5.3.5 I/O Alternative Function

Each GPIO has four 32-bit memory mapped alternative function(1) registers that operate as shown below:

- The Pn_MR1(2) register selects an alternative function for each port I/O pin (pin 0 to 7).
- The Pn_MR2(2) register selects an alternative function for each port I/O pin (pin 8 to 15).

The direction of the alternative function is automatically determined according to the purpose, whether it is input or output.

NOTES:

1. For the port that supports analog input/output, if a PORT mux on the port is set to GPIO with the corresponding analog module enabled, it doesn't operate correctly because of port conflicts during the operation. Therefore, if users want to use the PORT mux for GPIO, must disable the corresponding analog module.
2. The Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR registers can be updated with new data only when '0x15' is written to the PORTEN[7:0] register and then '0x51' is written again to the same register. After writing '0x00' to the PORTEN[7:0] register, users cannot change values of these registers.

Table 34 describes the GPIO alternative functions.

Table 34. GPIO Alternative Function

Pin Name	Alternative Function				
	AF0	AF1	AF2	AF3	AF7
PA0	PA0				AN0(1)
PA1	PA1				AN1(1)
PA2	PA2				AN2(1)
PA3	PA3				AN3(1)
PA4	PA4		T0IO		AN4(1)
PA5	PA5		T1IO	CAPEU	AN5(1)
PA6	PA6		T2IO	CAPEV	AN6(1)
PA7	PA7		T3IO	CAPEW	AN7(1)
PA8	PA8				AN8(1)
PA9	PA9				AN9(1)
PA10	PA10	RXD1		SCAPEU	AN10(1)
PA11	PA11	TXD1		SCAPEV	AN11(1)
PA12	PA12	SS0		SCAPEW	AN12(1)
PA13	PA13	SCK0			AN13(1)
PA14	PA14	MOSI0		PRTINEV	AN14(1)
PA15	PA15	MISO0		OVINEV	AN15(1)
PB0	PB0			MP0UH	(1)
PB1	PB1			MP0UL	(1)
PB2	PB2			MP0VH	(1)
PB3	PB3			MP0VL	(1)
PB4	PB4		T8IO	MP0WH	(1)
PB5	PB5		T9IO	MP0WL	(1)
PB6	PB6			PRTIN0U	(1)
PB7	PB7			OVIN0U	(1)
PB8	PB8	RXD3		PRTIN1U	(1)
PB9	PB9	TXD3		OVIN1U	(1)
PB10	PB10			MP1UH	(1)
PB11	PB11			MP1UL	(1)
PB12	PB12			MP1VH	(1)
PB13	PB13			MP1VL	(1)
PB14	PB14			MP1WH	(1)
PB15	PB15			MP1WL	(1)
PC0	PC0	RXD0		TCK/SWCLK(1)	
PC1	PC1	TXD0		TMS/SWDIO(1)	
PC2	PC2			TDO(1)	
PC3	PC3			TDI(1)	
PC4	PC4		T0IO	nTRST(1)	
PC5	PC5	RXD1	T1IO		(1)
PC6	PC6	TXD1	T2IO		(1)
PC7	PC7	SCL0	T3IO		(1)
PC8	PC8	SDA0	T4IO		(1)
PC9	PC9		T8IO	CLKO	(1)
PC10	PC10			nRESET(1)	
PC11	PC11		T9IO	nBOOT(1)	

Table 33. GPIO Alternative Function (continued)

Pin Name	Alternative Function				
	AF0	AF1	AF2	AF3	AF7
PC12	PC12				XIN(1)
PC13	PC13				XOUT(1)
PC14	PC14	MOSI0		RXD0	(1)
PC15	PC15	MISO0		TXD0	(1)
PD0	PD0	SS1			SXIN(1)
PD1	PD1	SCK1			SXOUT(1)
PD2	PD2	MOSI1			(1)
PD3	PD3	MISO1			(1)
PD4	PD4	SCL1			AN16(1)
PD5	PD5	SDA1			AN17(1)
PD6	PD6	TXD2			AN18(1)
PD7	PD7	RXD2			AN19(1)
PD8	PD8		T6IO	WDTO	(1)
PD9	PD9		T7IO	STBYO	(1)
PD10	PD10		T0IO		(1)
PD11	PD11		T1IO		(1)
PD12	PD12		T2IO		(1)
PD13	PD13		T3IO		(1)
PD14	PD14	SS0			(1)
PD15	PD15	SCK0			(1)
PE0	PE0				(1)
PE1	PE1				AN20(1)
PE2	PE2				AN21(1)
PE3	PE3	SCL0			(1)
PE4	PE4	SDA0			(1)
PE5	PE5		T5IO		(1)
PE6	PE6		T5IO		(1)
PE7	PE7		T6IO		(1)
PE8	PE8		T7IO		(1)
PE9	PE9		T8IO		(1)
PE10	PE10		T9IO		(1)
PE11	PE11	SCL1	T0IO		(1)
PE12	PE12	SDA1	T1IO		(1)
PE13	PE13	TXD4	T2IO		(1)
PE14	PE14	RXD4	T3IO		(1)
PE15	PE15				(1)
PF0	PF0				(1)
PF1	PF1				(1)
PF2	PF2				AN22(1)
PF3	PF3				AN23(1)
PF4	PF4	TXD5			(1)
PF5	PF5	RXD5			(1)
PF6	PF6			PRTINEW	(1)
PF7	PF7			OVINEW	(1)
PF8	PF8				(1)
PF9	PF9				(1)

Table 33. GPIO Alternative Function (continued)

Pin Name	Alternative Function				
	AF0	AF1	AF2	AF3	AF7
PF10	PF10				(1)
PF11	PF11				(1)
PF12	PF12				(1)
PF13	PF13				(1)
PF14	PF14				(1)
PF15	PF15				(1)
PG0	PG0	SS2			(1)
PG1	PG1	SCK2			(1)
PG2	PG2	MOSI2			(1)
PG3	PG3	MISO2			(1)
PG4	PG4				(1)
PG5	PG5				(1)
PG6	PG6				(1)
PG7	PG7				(1)
PG8	PG8	RXD3			(1)
PG9	PG9	TXD3			(1)
PG10	PG10				(1)

NOTES:

1. It means ‘selected pin function after reset condition’.
2. Unused pins must be set to output by software (low output is recommended).

5.3.6 Interrupt Functionality

The GPIO module has four interrupt sources, and each port can be an interrupt source. To set each GPIO pin as interrupt sources, users must configure the interrupt control register (Pn_ICR) and the interrupt enable register (Pn_IER).

A level-triggered or edge-triggered interrupt can be set for each pin. Once an interrupt occurs, the pin’s corresponding bit of the Pn_ISR register is flagged. This flag can be cleared by writing a ‘1’ to the bit.

Table 35. Interrupt Sources and Corresponding Pins in GPIO Module

Interrupt Name	Configurable Pins
GPIOA	Port A[15:0]
GPIOB	Port B[15:0]
GPIOC	Port C[15:0]
GPIOD	Port D[15:0]
GPIOE	Port E[15:0]
GPIOF	Port F[15:0]
GPIOG	Port G[10:0]

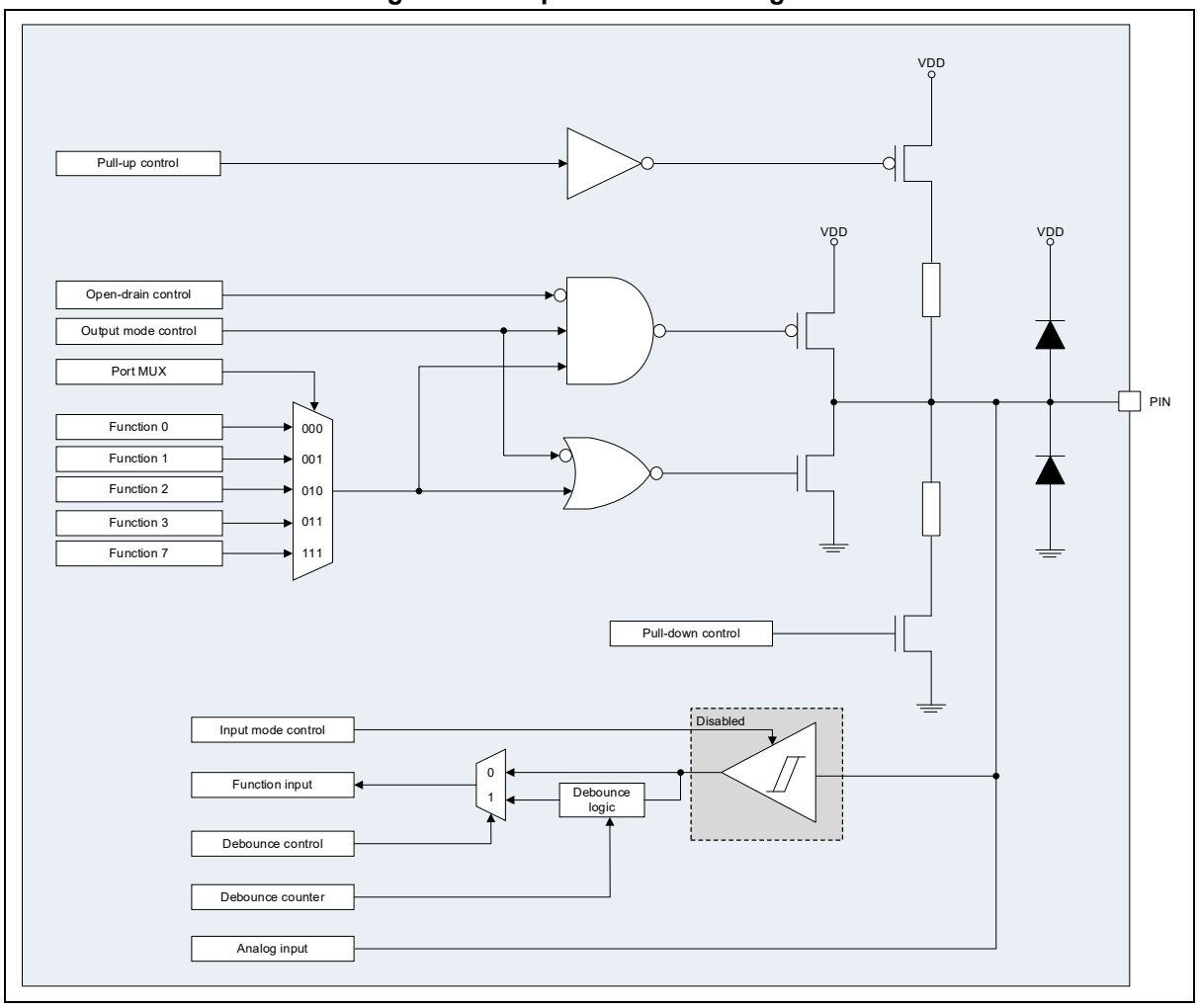
5.3.7 Output Configuration

When the I/O port is programed as output, it features the followings:

- The push-pull driver is turned on or off depending on the Pn_ODR register or function output value.
- Each pin's drive strength can be controlled by configuring the Pn_STR register.
- Open-drain mode is available.
- The use of the pull-up and pull-down resistors is determined by the Pn_PRCR register.
- The path of analog signal must be opened for the individual module control.

Figure 35 shows a block diagram of an output port.

Figure 35. Output Port Block Diagram



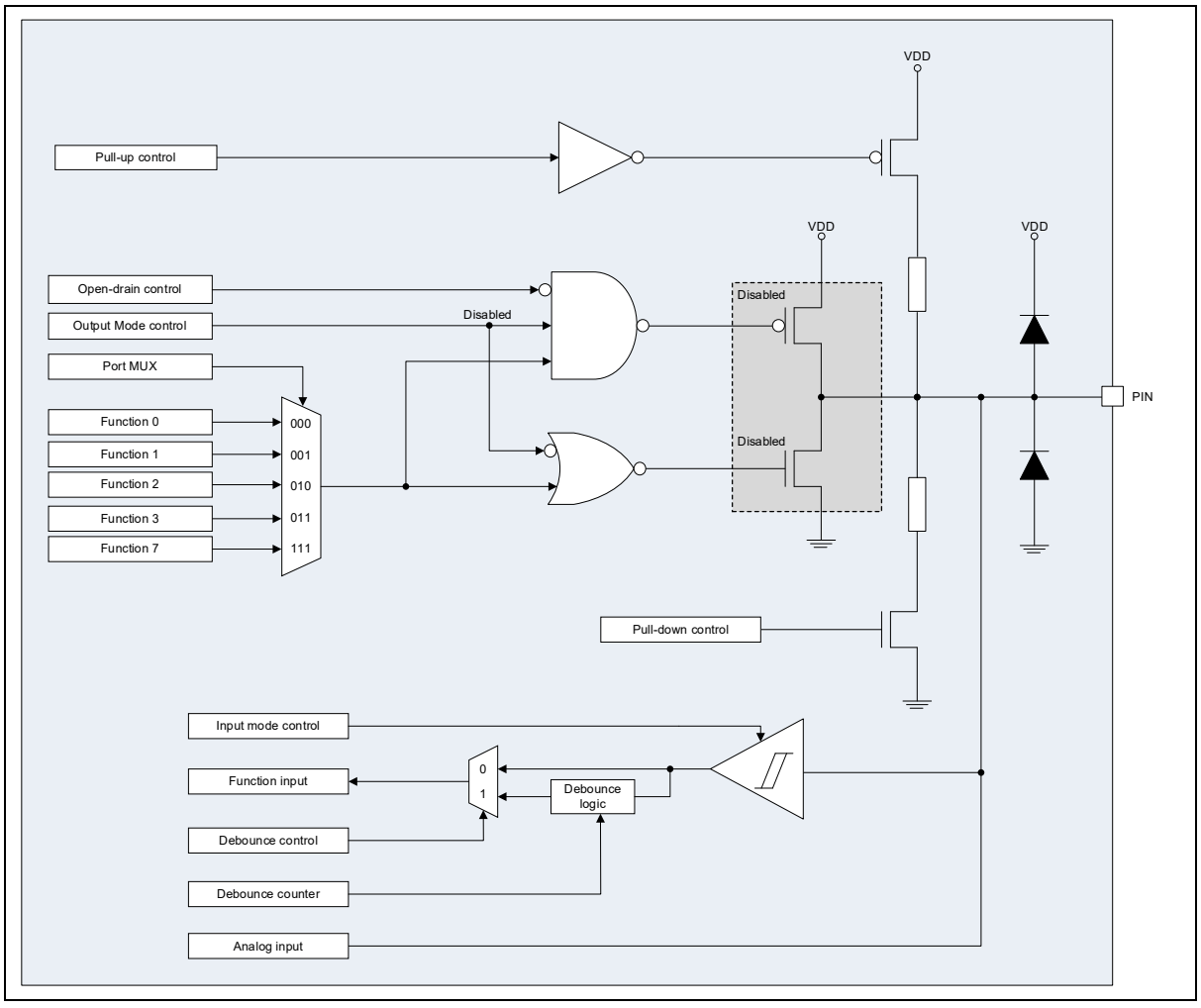
5.3.8 Input Configuration

When the I/O port is programed as input, it features the followings:

- All push-pull drivers are turned off, and Input Schmitt Trigger opens its input.
- Input data is stored in the Pn_IDR register or input signal enters toward functions.
- The path of analog signal must be opened for the individual module control.

Figure 36 shows a block diagram of an input port.

Figure 36. Input Port Block Diagram



5.3.9 Alternative Function Configuration

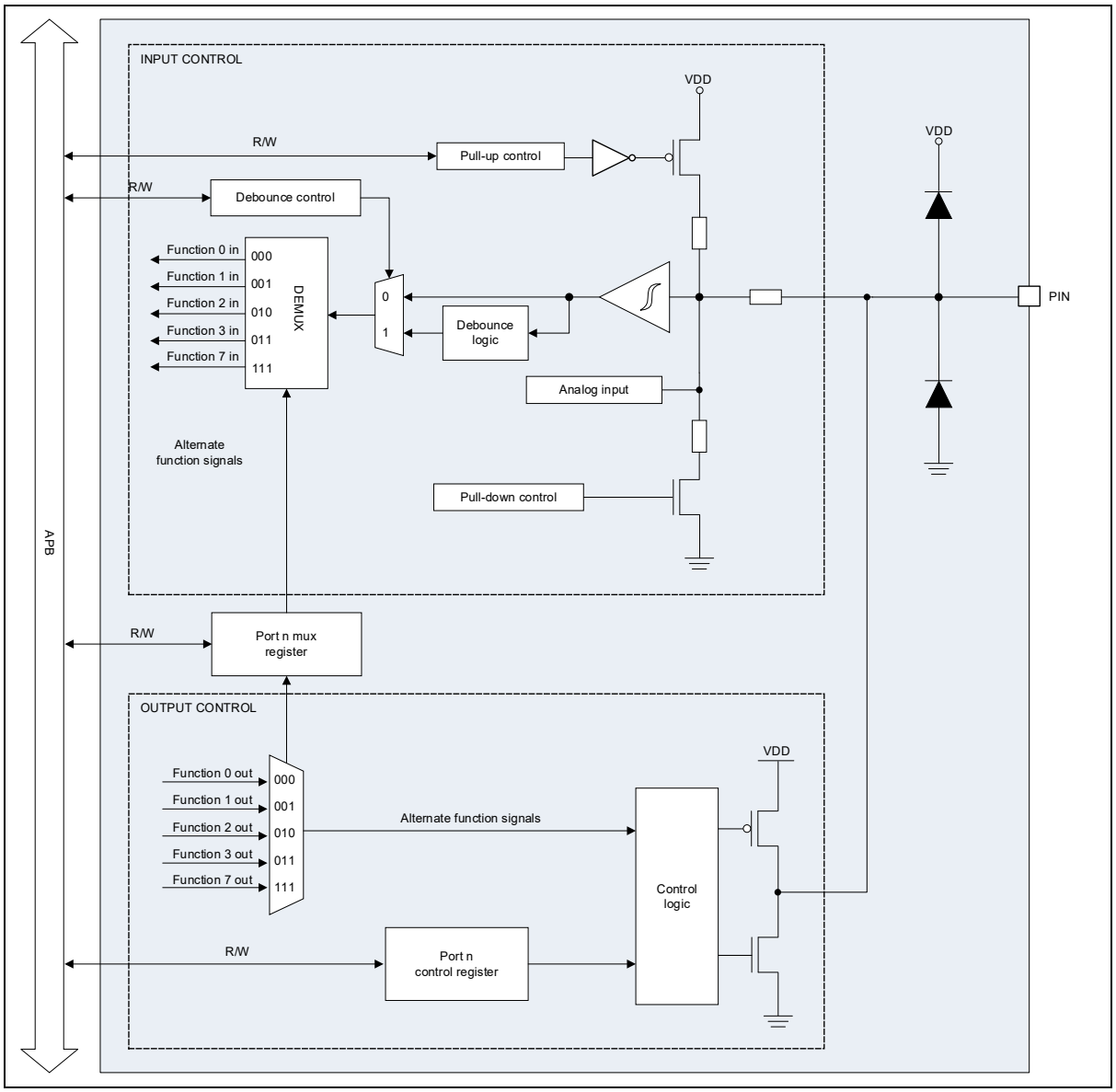
For the A34M420, all pins except those with specific purposes can be used as GPIO pins, and the GPIO function can be set by the pin multiplexer control register on each port.

If an I/O port's pin is set as an input pin by the pin control register, its output function becomes disabled.

Users can select different functions for each port' pins based on the settings of the alternative function selection register. The input data registers capture data entering each I/O pin, as either un-debounced or debounced, at every GPIO clock cycle.

Figure 37 shows a block diagram that describes alternative function selection on a port.

Figure 37. Port Function Block Diagram



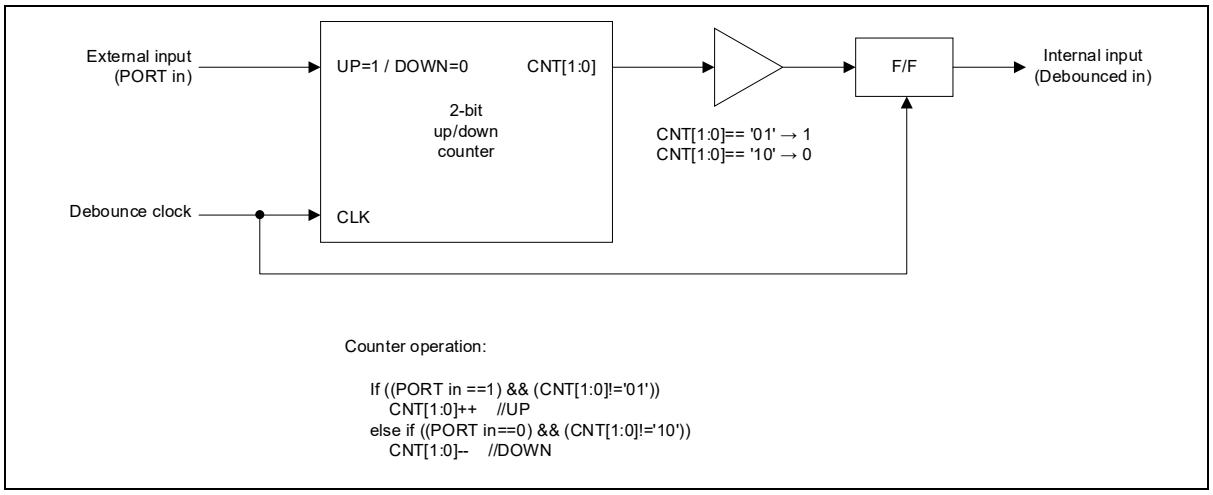
5.3.10 Debouncing Functionality

For the A34M420, each port is capable of debouncing input signals. Debouncing is to filter out noise that can interfere with the port's data input.

Filtering levels can be adjusted for each 16-pin port, and each individual pin is configured based on whether filtering is enabled or disabled. The debouncing clock used for each port is set in the SCU_MCCR4 and SCU_MCCR5 registers.

Figure 38 shows a block diagram of a simplified debouncing logic.

Figure 38. Debouncing Logic Block Diagram

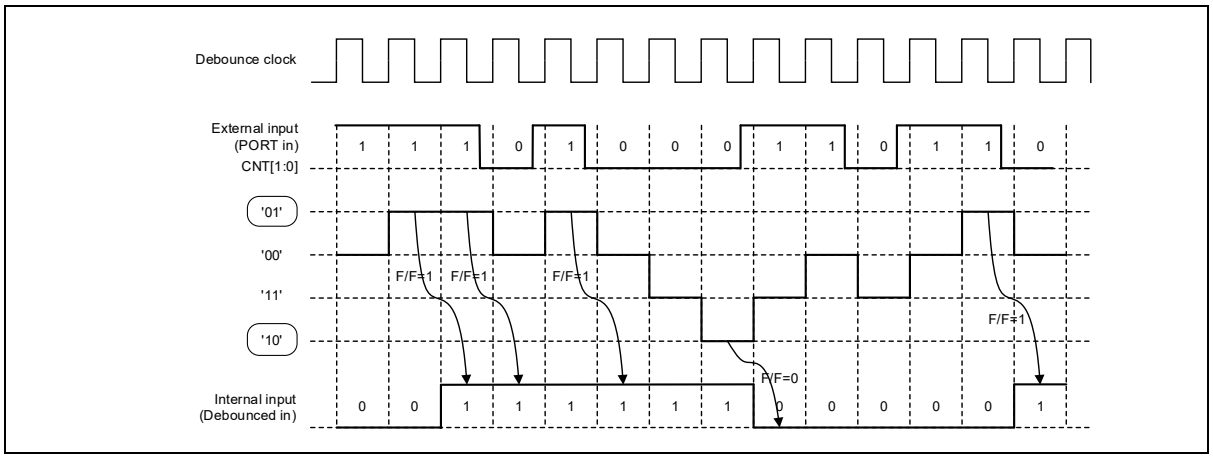


When the debouncing is enabled, the debouncing logic is clocked at the frequency division ratio set in the SCU_MCCR4 or SCU_MCCR5 registers, depending on which register the port belongs to.

A pin recognizes a change in the input if the changed value remains constant for three clock pulses. If the internal count (CNT[1:0]) is set to '01', the internal input value is set to '1'; while if the former is set to '10', the latter is set to '0'. Otherwise, the previous input value is maintained.

Figure 39 shows a timing diagram of a port debouncing.

Figure 39. Example Timing Diagram of Port Debouncing



5.4 GPIO Registers

The base addresses and register map of PCU are described in the followings:

Table 36. Base Address of PCU

Name	Base Address	Description
PA	0x4000_1000	GPIO Port A
PB	0x4000_1100	GPIO Port B
PC	0x4000_1200	GPIO Port C
PD	0x4000_1300	GPIO Port D
PE	0x4000_1400	GPIO Port E
PF	0x4000_1500	GPIO Port F
PG	0x4000_1600	GPIO Port G

Table 37. PCU and GPIO Register Map(1)

Name	Offset	Type	Description	Reset Value	Reference
Pn_MR1	0x0000	RW	Port n MUX1 Select Register	PA: 0x7777_7777 PB: 0x7777_7777 PC: 0x7773_3333 PD: 0x7777_7777 PF: 0x7777_7777 PG: 0x7777_7777	5.4.1 5.4.3
Pn_MR2	0x0004	RW	Port n MUX2 Select Register	PA: 0x7777_7777 PB: 0x7777_7777 PC: 0x7777_3377 PD: 0x7777_7777 PF: 0x7777_7777 PG: 0x7777_7777	5.4.2 5.4.4
Pn_CR	0x0008	RW	Port n Control Register	0xFFFF_FFFF	5.4.5
Pn_PRCR	0x000C	RW	Port n Pull-up/pull-down Control Register	PA: 0x0000_0000 PB: 0x0000_0000 PC: 0x00A0_028A PD: 0x0000_0000 PE: 0x0000_0000 PF: 0x0000_0000 PG: 0x0000_0000	5.4.6
Pn_DER	0x0010	RW	Port n Debouncing Enable Register	0x0000_0000	5.4.8
Pn_STR	0x0014	RW	Port n Strength Configuration Register	0x0000_0000	5.4.9
Pn_IER	0x0020	RW	Port n Interrupt Enable Register	0x0000_0000	5.4.10
Pn_ISR	0x0024	RWC1	Port n Interrupt Status Register	0x0000_0000	5.4.11
Pn_ICR	0x0028	RW	Port n Interrupt Control Register	0x0000_0000	5.4.12
Pn_ODR	0x0030	RW	Port n Output Data Register	0x0000_0000	5.4.13
Pn_IDR	0x0034	RO	Port n Input Data Register	-	5.4.14
Pn_BSR	0x0038	WO	Port n Set/Reset register	0x0000_0000	5.4.15
Pn_BCR	0x003C	WO	Port n Reset Register	0x0000_0000	5.4.16
PCU_PORTEN	0x0FF0	RW	Port Access Enable Register	0x0000_0000	5.4.17

NOTE:

1. n = A, B, C, D, E, and G.

5.4.1 Pn_MR1: Port n MUX Control Register 1

The Pn_MR1 register selects the mode of each pin (pin 0 to 7) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PA_MR1=0x4000_1000, PB_MR1=0x4000_1100, PD_MR1=0x4000_1300
PE_MR1=0x4000_1400, PF_MR1=0x4000_1500, PG_MR1=0x4000_1600

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
Reserved	P7MUX[2:0]	Reserved	P6MUX[2:0]	Reserved	P5MUX[2:0]	Reserved	P4MUX[2:0]	Reserved	P3MUX[2:0]	Reserved	P2MUX[2:0]	Reserved	P1MUX[2:0]	Reserved	P0MUX[2:0]
-	111	-	111	-	111	-	111	-	111	-	111	-	111	-	111
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4x+2 4x	PxMUX[2:0]	Port multiplexer selection bit, x=0 to 7	
		000	Alternative function 0 (AF0)
		001	Alternative function 1 (AF1)
		010	Alternative function 2 (AF2)
		011	Alternative function 3 (AF3)
		111	Alternative function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 1 to 7) bits of PD0 and PD1 cannot be modified while the LSE is set as the system clock (MCLK).
2. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the Pn_MR register settings, the I/O is automatically configured as an input or output of that function.)
3. If a port multiplexer selection is set to AF1, AF2, AF3, or AF7, the port is configured for the alternative function regardless of the Pn_CR register.
4. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
5. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.

5.4.2 Pn_MR2: Port n MUX Control Register 2

The Pn_MR2 register selects the mode of each pin (pin 8 to 15) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PA_MR2=0x4000_1004, PB_MR2=0x4000_1104, PD_MR2=0x4000_1304
PE_MR2=0x4000_1404, PF_MR2=0x4000_1504, PG_MR2=0x4000_1604

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
Reserved	P15MUX[2:0]	Reserved	P14MUX[2:0]	Reserved	P13MUX[2:0]	Reserved	P12MUX[2:0]	Reserved	P11MUX[2:0]	Reserved	P10MUX[2:0]	Reserved	P9MUX[2:0]	Reserved	P8MUX[2:0]
-	111	-	111	-	111	-	111	-	111	-	111	-	111	-	111
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4(x-8)+2 4(x-8)	PxMUX[2:0]	Port multiplexer selection bit, x = 8 to15	
		000	Alternative function 0 (AF0)
		001	Alternative function 1 (AF1)
		010	Alternative function 2 (AF2)
		011	Alternative function 3 (AF3)
		111	Alternative function 7 (AF7)
		Others	Reserved

NOTES:

1. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the Pn_MR register settings, the I/O is automatically configured as the input or output of that function.)
2. If a port multiplexer selection is set to AF1, AF2, AF3, or AF7, the port is configured for the alternative function regardless of the Pn_CR register.
3. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
4. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN[7:0] register is activated.

5.4.3 PC_MR1: Port C MUX Control Register 1

The PC_MR1 register selects the mode of each pin (pin 0 to 7) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PC_MR1=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
Reserved	P7MUX[2:0]	Reserved	P6MUX[2:0]	Reserved	P5MUX[2:0]	Reserved	P4MUX[2:0]	Reserved	P3MUX[2:0]	Reserved	P2MUX[2:0]	Reserved	P1MUX[2:0]	Reserved	P0MUX[2:0]
-	111	-	111	-	111	-	011	-	011	-	011	-	011	-	011
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4x+2 4x	PxMUX[2:0]	Port multiplexer selection bit, x=0 to 7	
		000	Alternative function 0 (AF0)
		001	Alternative function 1 (AF1)
		010	Alternative function 2 (AF2)
		011	Alternative function 3 (AF3)
		111	Alternative function 7 (AF7)
		Others	Reserved

NOTES:

1. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the PC_MR register settings, the I/O is automatically configured as the input or output of that function.)
2. If a port multiplexer selection is set to AF1, AF2, AF3, or AF7, the port is configured for the alternative function regardless of the PC_CR register.
3. If a port multiplexer selection is set to AF0, the PC_CR register configures the port type.
4. PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.

5.4.4 PC_MR2: Port C MUX Control Register 2

The PC_MR2 register selects the mode of each pin (pin 8 to 15) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PC_MR2=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
Reserved	P15MUX[2:0]	Reserved	P14MUX[2:0]	Reserved	P13MUX[2:0]	Reserved	P12MUX[2:0]	Reserved	P11MUX[2:0]	Reserved	P10MUX[2:0]	Reserved	P9MUX[2:0]	Reserved	P8MUX[2:0]
-	111	-	111	-	111	-	111	-	011	-	011	-	111	-	111
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4(x-8)+2 4(x-8)	PxMUX[2:0]	Port multiplexer selection bit, x = 8 to15	
		000	Alternative function 0 (AF0)
		001	Alternative function 1 (AF1)
		010	Alternative function 2 (AF2)
		011	Alternative function 3 (AF3)
		111	Alternative function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] bits of PC12, PC13 cannot be modified while the HSE is set as the system clock (MCLK).
2. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the PC_MR register settings, the I/O is automatically configured as the input or output of that function.)
3. If a port multiplexer selection is set to AF1, AF2, AF3, or AF7, the port is configured for the alternative function regardless of the PC_CR register.
4. If a port multiplexer selection is set to AF0, the PC_CR register configures the port type.
5. The PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.

5.4.5 Pn_CR: Port n Control Register

The Pn_CR register sets the pins in each port as input, open-drain output, or push-pull port pins.

PA_CR=0x4000_1008, PB_CR=0x4000_1108, PC_CR=0x4000_1208, PD_CR=0x4000_1308
PE_CR=0x4000_1408, PF_CR=0x4000_1508, PG_CR=0x4000_1608

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[1:0]	P14[1:0]	P13[1:0]	P12[1:0]	P11[1:0]	P10[1:0]	P9[1:0]	P8[1:0]	P7[1:0]	P6[1:0]	P5[1:0]	P4[1:0]	P3[1:0]	P2[1:0]	P1[1:0]	P0[1:0]
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

2x+1 2x	Px[1:0]	Port control, x = 0 to 15.	
		00	Push-pull output
		01	Open-drain output
		1x	Input

NOTES:

1. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
2. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN register is activated.

5.4.6 Pn_PRCR: Port n Pull-up/pull-down Resistor Control Register

All pins of each port include internal pull-up and pull-down resistors that can be determined by the Pn_PRCR register. This resistor can enable or disable the pull-up/pull-down resistors of each port pin.

PA_PRCR=0x4000_100C, PB_PRCR=0x4000_110C, PD_PRCR=0x4000_130C
PE_PRCR=0x4000_140C, PF_PRCR=0x4000_150C, PG_PRCR=0x4000_160C

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
PUE15[1:0]	PUE14[1:0]	PUE13[1:0]	PUE12[1:0]	PUE11[1:0]	PUE10[1:0]	PUE9[1:0]	PUE8[1:0]	PUE7[1:0]	PUE6[1:0]	PUE5[1:0]	PUE4[1:0]	PUE3[1:0]	PUE2[1:0]	PUE1[1:0]	PUE0[1:0]
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

2x+1 2x	PUEx[1:0]	Port pull-up and pull-down control, x = 0 to 15.	
		0x	Disables the pull-up and pull-down resistors.
		10	Enables the pull-up resistor and disables the pull-down resistor.
		11	Disables the pull-up resistor and/ enables the pull-down resistor.

NOTE:

1. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN register is activated.

5.4.7 PC_PRCR: Port C Pull-up/pull-down Resistor Control Register

All pins of each port include internal pull-up and pull-down resistors that can be determined by the PC_PRCR register. This resistor can enable or disable the pull-up/pull-down resistors of each port pin.

PC_PRCR=0x4000_120C

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
PUE15[1:0]	PUE14[1:0]	PUE13[1:0]	PUE12[1:0]	PUE11[1:0]	PUE10[1:0]	PUE9[1:0]	PUE8[1:0]	PUE7[1:0]	PUE6[1:0]	PUE5[1:0]	PUE4[1:0]	PUE3[1:0]	PUE2[1:0]	PUE1[1:0]	PUE0[1:0]
00	00	00	00	10	10	00	00	00	00	00	10	10	00	10	10
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	PUEx[1:0]	Port pull-up and pull-down control, x = 0 to 15.	
		0x	Disables the pull-up and pull-down resistors.
		10	Enables the pull-up resistor and disables the pull-down resistor.
		11	Disables the pull-up resistor and enables the pull-down resistor.

NOTE:

1. PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when PORTEN register is activated.

5.4.8 Pn_DER: Port n Debounce Enable Register

All pins in each port have a digital debouncing filter, which can be set in the Pn_DER register.

PA_DER=0x4000_1010, PB_DER=0x4000_1110, PC_DER=0x4000_1210, PD_DER=0x4000_1310
PE_DER=0x4000_1410, PF_DER=0x4000_1510, PG_DER=0x4000_1610

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
Reserved		PDE15	PDE14	PDE13	PDE12	PDE11	PDE10	PDE9	PDE8	PDE7	PDE6	PDE5	PDE4	PDE3	PDE2	PDE1	PDE0
-		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	RW	RW	RW	RW

x	PDEx	Enable pin debouncing, x = 0 to 15
		0 Disables the debouncing filter.
		1 Enables the debouncing filter.

NOTES:

1. To use the port debounce function, the debounce clock of SCU_MCCR4 and SCU_MCCR5 must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction.
2. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.9 Pn_STR: Port n Strength Configuration Register

The Pn_STR register configures each pin's drive strength. These settings affect the port's speed and current consumption.

PA_STR=0x4000_1014, PB_STR=0x4000_1114, PC_STR=0x4000_1214, PD_STR=0x4000_1314
PE_STR=0x4000_1414, PF_STR=0x4000_1514, PG_STR=0x4000_1614

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
Reserved		PST15	PST14	PST13	PST12	PST11	PST10	PST9	PST8	PST7	PST6	PST5	PST4	PST3	PST2	PST1	PST0
-		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	RW	RW	RW	RW

x	PSTx	Pin drive strength setting, x = 0 to 15.
		0 Disables the strength.
		1 Enables the strength.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0]

5.4.10 Pn_IER: Port n Interrupt Enable Register

All pins can be set as an external interrupt source which is either the edge-triggered interrupt or the level-triggered interrupt. The Pn_IER register enables or disables these interrupts for each pin.

PA_IER=0x4000_1020, PB_IER=0x4000_1120, PC_IER=0x4000_1220, PD_IER=0x4000_1320
PE_IER=0x4000_1420, PF_IER=0x4000_1520, PG_IER=0x4000_1620

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[1:0]	PIE14[1:0]	PIE13[1:0]	PIE12[1:0]	PIE11[1:0]	PIE10[1:0]	PIE9[1:0]	PIE8[1:0]	PIE7[1:0]	PIE6[1:0]	PIE5[1:0]	PIE4[1:0]	PIE3[1:0]	PIE2[1:0]	PIE1[1:0]	PIE0[1:0]
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

2x+1 2x	PIEx[1:0]	The interrupt source to be enabled for the pin, x = 0 to15	
		00	Disable the external interrupt source.
		01	Enables the level-triggered interrupt (Non-pending).
		10	Enables the level-triggered interrupt (Pending).
		11	Enables the edge-triggered interrupt.

NOTES:

1. In non-pending mode, users can check if the current state of the pin is an interrupt state. In high level non-pending interrupt mode, an interrupt signal is generated when the pin state is high.

   The interrupt status flag can be checked with the ISR register. In high level non-pending mode, the interrupt status is automatically cleared. Therefore, when the pin state changes from high to low, the interrupt state is automatically cleared.

2. In the pending mode, you can check whether an interrupt has occurred, not the current state of the pin. In high level pending interrupt mode, an interrupt signal is generated when the pin state is high.

   The interrupt status flag can be checked with the ISR register. In high level pending mode, the interrupt status is not automatically cleared. Therefore, when the state of the pin is changed from high to low, the interrupt will continue to occur unless the user clears the interrupt flag. In the pending mode, once an interrupt occurs, the user must clear the interrupt flag directly because the state is maintained until the user clears the interrupt flag directly.

3. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.11 Pn_ISR: Port n Interrupt Status Register

When an interrupt is received by the microcontroller, the status of the interrupt can be detected in the Pn_ISR register. This register informs what kind of interrupt occurred at each interrupt source pin. To clear the interrupt status register, write a '1' to the corresponding bit.

PA_ISR=0x4000_1024, PB_ISR=0x4000_1124, PC_ISR=0x4000_1224, PD_ISR=0x4000_1324
PE_ISR=0x4000_1424, PF_ISR=0x4000_1524, PG_ISR=0x4000_1624

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[1:0]			PIS14[1:0]			PIS13[1:0]			PIS12[1:0]			PIS11[1:0]			PIS10[1:0]			PIS9[1:0]			PIS8[1:0]			PIS7[1:0]			PIS6[1:0]			PIS5[1:0]			PIS4[1:0]			PIS3[1:0]			PIS2[1:0]			PIS1[1:0]			PIS0[1:0]	
00			00			00			00			00			00			00			00			00			00			00			00			00			00			00	
RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1			RWC1	

2x+1 2x	PISx[1:0]	Pin interrupt status, x = 0 to 15
	00	No interrupt event has occurred.
	01	The low-level interrupt or falling-edge interrupt event has occurred.
	10	The high-level interrupt or rising-edge interrupt event has occurred.
	11	Both the rising- and falling-edge interrupt events have been detected in edge-triggered interrupt mode. Level-triggered interrupt mode is not supported.

NOTES:

1. In rising and falling edge interrupt modes, the pin state becomes '10' when the rising edge occurs, and the pin state becomes '01' when the falling edge occurs.
2. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.12 Pn_ICR: Port n Interrupt Control Register

The Pn_ICR register controls each pin's interrupt modes that define interrupt-triggering signals.

PA_ICR=0x4000_1028, PB_ICR=0x4000_1128, PC_ICR=0x4000_1228, PD_ICR=0x4000_1328
PE_ICR=0x4000_1428, PF_ICR=0x4000_1528, PG_ICR=0x4000_1628

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[1:0]			PIC14[1:0]			PIC13[1:0]			PIC12[1:0]			PIC11[1:0]			PIC10[1:0]			PIC9[1:0]			PIC8[1:0]			PIC7[1:0]			PIC6[1:0]			PIC5[1:0]			PIC4[1:0]			PIC3[1:0]			PIC2[1:0]			PIC1[1:0]			PIC0[1:0]	
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	

2x+1 2x	PICx[1:0]	Pin interrupt mode, x = 0 to 15
	00	Disable the external interrupt source.
	01	Enables low-level or falling-edge interrupt mode.
	10	Enables high-level or rising-edge interrupt mode.
	11	Enables both rising- and falling-edge interrupt modes. Level-triggered interrupt mode is not supported.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.13 Pn_ODR: Port n Output Data Register

The Pn_ODR register defines the output data at each pin when the port is used as GPIO mode. The output level at each pin is decided by this register's settings.

PA_ODR=0x4000_1030, PB_ODR=0x4000_1130, PC_ODR=0x4000_1230, PD_ODR=0x4000_1330
PE_ODR=0x4000_1430, PF_ODR=0x4000_1530, PG_ODR=0x4000_1630

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
Reserved		POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
-		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	RW	RW	RW	RW

15 0	PODx	The pin's output data configuration, x = 0 to 15	
		0	Outputs a '0' if the port pin is in output mode.
		1	Outputs a '1' if the port pin is in output mode.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.14 Pn_IDR: Port n Input Data Register

The Pn_IDR register allows each pin level of the port to be read. This register's value represents the input at each pin in logic input mode even though the pin is currently in a different mode, except for in analog mode. If a pin is set to serve a special function, such as an ADC or clock Input / Output, the register bit always reads '1'. If a level interrupt has been enabled for a port's pin by the Pn_ICR register, the pin's input level can be checked by reading the Pn_IDR register.

PA_IDR=0x4000_1034, PB_IDR=0x4000_1134, PC_IDR=0x4000_1234, PD_IDR=0x4000_1334
PE_IDR=0x4000_1434, PF_IDR=0x4000_1534, PG_IDR=0x4000_1634

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
Reserved		PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
-		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-		RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

15 0	PIDx	The pin's input data (represents the port's status), x = 0 to 15	
		0	The current pin input is '0'.
		1	The current pin input is '1'.

NOTES:

1. If an external pin of the microcontroller is used as an input pin while it is floating, it will fall in an 'unknown' state, where the input signal level is identified as either '0' or '1', regardless of the input value.
2. If the microcontroller is programmed to perform a certain function based on the level identified by the Pn_IDR register in this floating state, this is highly likely to cause malfunctions. Therefore, it is recommended to pull up or down the

pins that receive external input signals, depending on their uses.

3. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.15 Pn_BSR: Port n Set/Reset Register

The Pn_BSR register sets or clears the level status flag of each pin of a Port n. The values written in this register are output to the Pn_ODR register. If values of the BCDx and BSDx bits become 1 at the same time, this register sets the corresponding pin level status to '1'.

PA_BSR=0x4000_1038, PB_BSR=0x4000_1138, PC_BSR=0x4000_1238, PD_BSR=0x4000_1338
PE_BSR=0x4000_1438, PF_BSR=0x4000_1538, PG_BSR=0x4000_1638

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
BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0	BSD15	BSD14	BSD13	BSD12	BSD11	BSD10	BSD9	BSD8	BSD7	BSD6	BSD5	BSD4	BSD3	BSD2	BSD1	BSD0
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
WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

X+16	BCDx	Pin clear bit, x = 16 to 31	
		0	No change
		1	Sets the bit if the corresponding bit is '1'.
X	BSDx	Pin set bit, x = 0 to 15	
		0	No change
		1	Sets the bit if the corresponding bit is '1'.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.16 Pn_BCR: Port n Reset Register

The Pn_BCR register clears the level status flag of each pin of a Port n. Setting a bit to '1' clears the corresponding bit in the Pn_ODR register.

PA_BCR = 0x4000_103C, PB_BCR=0x4000_113C, PC_BCR=0x4000_123C, PD_BCR=0x4000_133C
PE_BCR=0x4000_143C, PF_BCR=0x4000_153C, PG_BCR=0x4000_163C

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
Reserved																BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0
-																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-																WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

X	BCDx	Pin Clear bit, x = 0 to 15	
		0	Causes no changes.
		1	Resets the corresponding pin bit.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.17 PCU_PORTEN: Port Access Enable Register

The PORTEN register allows to change the values in all PCU registers.

PORTEN=0x4000_1FF0

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
Reserved		ENS	PORTEN[7:0]
-		0	--
-		RC	WO

8	ENS	Pin Clear bit, x = 0 to 15
		0 Causes no changes.
		1 Resets the corresponding pin bit.
7 0	PORTEN[7:0]	PCU and GPIO register access enable
		Writing ‘0x15’ and then ‘0x51’ to the bit field enables writing new values to GPIO registers. After this, write a different value to this bit field to protect the Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR registers against being updated with new values.

NOTE:

1. Some GPIO control registers can be updated with new data only when ‘0x15’ is written to the PORTEN[7:0] and then ‘0x51’ is written again to the same register. After writing ‘0x00’ to the PORTEN[7:0], users cannot change values. The corresponding registers are Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR.

5.4.18 PCU and GPIO Register Map Summary

Table 38. PCU and GPIO Register Map Summary

Offset	Register name	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
0x00	Pn_MR1	Res	P7MUX[2:0]			Res	P6MUX[2:0]			Res	P5MUX[2:0]			Res	P4MUX[2:0]			Res	P3MUX[2:0]			Res	P2MUX[2:0]			Res	P1MUX[2:0]			Res	P0MUX[2:0]		
	Reset value		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1
0x04	Pn_MR2	Res	P15MUX[2:0]			Res	P14MUX[2:0]			Res	P13MUX[2:0]			Res	P12MUX[2:0]			Res	P11MUX[2:0]			Res	P10MUX[2:0]			Res	P9MUX[2:0]			Res	P8MUX[2:0]		
	Reset value		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1		1	1	1
0x1200	PC_MR1	Res	P7MUX[2:0]			Res	P6MUX[2:0]			Res	P5MUX[2:0]			Res	P4MUX[2:0]			Res	P3MUX[2:0]			Res	P2MUX[2:0]			Res	P1MUX[2:0]			Res	P0MUX[2:0]		
	Reset value		1	1	1		1	1	1		1	1	1		0	1	1		0	1	1		0	1	1		0	1	1		0	1	1
0x1204	PC_MR2	Res	P15MUX[2:0]			Res	P14MUX[2:0]			Res	P13MUX[2:0]			Res	P12MUX[2:0]			Res	P11MUX[2:0]			Res	P10MUX[2:0]			Res	P9MUX[2:0]			Res	P8MUX[2:0]		
	Reset value		1	1	1		1	1	1		1	1	1		1	1	1		0	1	1		0	1	1		1	1	1		1	1	1
0x08	Pn_CR	P15[1:0]		P14[1:0]		P13[1:0]		P12[1:0]		P11[1:0]		P10[1:0]		P9[1:0]		P8[1:0]		P7[1:0]		P6[1:0]		P5[1:0]		P4[1:0]		P3[1:0]		P2[1:0]		P1[1:0]		P0[1:0]	
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	Pn_PRCR	PUE15[1:0]		PUE14[1:0]		PUE13[1:0]		PUE12[1:0]		PUE11[1:0]		PUE10[1:0]		PUE9[1:0]		PUE8[1:0]		PUE7[1:0]		PUE6[1:0]		PUE5[1:0]		PUE4[1:0]		PUE3[1:0]		PUE2[1:0]		PUE1[1:0]		PUE0[1:0]	
	Reset value	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
0x120C	PC_PRCR	PUE15[1:0]		PUE14[1:0]		PUE13[1:0]		PUE12[1:0]		PUE11[1:0]		PUE10[1:0]		PUE9[1:0]		PUE8[1:0]		PUE7[1:0]		PUE6[1:0]		PUE5[1:0]		PUE4[1:0]		PUE3[1:0]		PUE2[1:0]		PUE1[1:0]		PUE0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0
0x10	Pn_DER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PDE15	PDE14	PDE13	PDE12	PDE11	PDE10	PDE9	PDE8	PDE7	PDE6	PDE5	PDE4	PDE3	PDE2	PDE1	PDE0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	Pn_STR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PST15	PST14	PST13	PST12	PST11	PST10	PST9	PST8	PST7	PST6	PST5	PST4	PST3	PST2	PST1	PST0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	Pn_IER	PIE15[1:0]		PIE14[1:0]		PIE13[1:0]		PIE12[1:0]		PIE11[1:0]		PIE10[1:0]		PIE9[1:0]		PIE8[1:0]		PIE7[1:0]		PIE6[1:0]		PIE5[1:0]		PIE4[1:0]		PIE3[1:0]		PIE2[1:0]		PIE1[1:0]		PIE0[1:0]	
	Reset value	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
0x24	Pn_ISR	PIS15[1:0]		PIS14[1:0]		PIS13[1:0]		PIS12[1:0]		PIS11[1:0]		PIS10[1:0]		PIS9[1:0]		PIS8[1:0]		PIS7[1:0]		PIS6[1:0]		PIS5[1:0]		PIS4[1:0]		PIS3[1:0]		PIS2[1:0]		PIS1[1:0]		PIS0[1:0]	
	Reset value	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

Table 37. PCU and GPIO Register Map Summary (continued)

Offset	Register name	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
0x28	Pn_ICR	PIC15[1:0]		PIC14[1:0]		PIC13[1:0]		PIC12[1:0]		PIC11[1:0]		PIC10[1:0]		PIC9[1:0]		PIC8[1:0]		PIC7[1:0]		PIC6[1:0]		PIC5[1:0]		PIC4[1:0]		PIC3[1:0]		PIC2[1:0]		PIC1[1:0]		PIC0[1:0]	
	Reset value	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
0x30	Pn_ODR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	POD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	Pn_IDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PID[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	Pn_BSR	BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0	BSD15	BSD14	BSD13	BSD12	BSD11	BSD10	BSD9	BSD8	BSD7	BSD6	BSD5	BSD4	BSD3	BSD2	BSD1	BSD0
	Reset value	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
0x3C	Pn_BCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1FF0	PORTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENS	PORTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

6. Embedded Flash Memory

6.1 Introduction

The flash memory controller is an interface controller for embedded flash memory. It manages data stored on the flash memory.

The flash memory controller of A34M420 supports the followings:

- Code data and instruction cache in code flash area
- Erase, read, write functions in code flash areas
- Read protection function
- Write and erase protection function
- Data flash memory for programming user data

6.2 Embedded Flash Memory Main Features

The flash memory of the A34M420 has the features as listed below:

- Code flash memory:
  - Flash memory size: 1024 KB
    - Dual bank code flash memory (bank0: 512 KB, bank1: 512 KB) for OTA (Over-The-Air programming)
    - Bank swap with RWW (Read-While-Write)
  - Program unit: 1 word (4 bytes)
  - Erase unit:
    - Page (512 bytes)
    - Sector (2 KB)
    - Bulk (full-chip)
  - Zero-wait (when the clock is lower than 28 MHz), 1- to 15-wait, and cache (flash acceleration) access
  - Write protection
  - Read protection
  - Code data and instruction cache
  - Built-in 16-bit specific polynomial for code flash memory
    - ECC (Error Correct Code)
  - Self-program function for the code flash memory
    - Supports updating data in a code flash memory region while executing user program

in the code flash memory area.

- Data flash memory:
  - Flash memory size: 32 KB
  - Program unit: 1 word (4 bytes)
  - Erase unit:
    - Page (512 bytes)
    - Sector (2 KB)
    - Bulk (full-chip)
  - Zero wait (when the clock is lower than 28 MHz), 1- to 15-wait
  - Write protection
  - Built-in 16-bit specific polynomial for data flash memory
  - ECC (Error Correct Code)
  - Self-program function for the data flash memory
    - Supports updating data in the data flash memory region while executing user program in the code flash memory area.

6.3 Flash Functional Description

6.3.1 Flash Memory Organization

The flash memory consists of code and data flash memory. Code flash memory stores application codes or constant data, and data flash memory stores parameters, log data, etc.

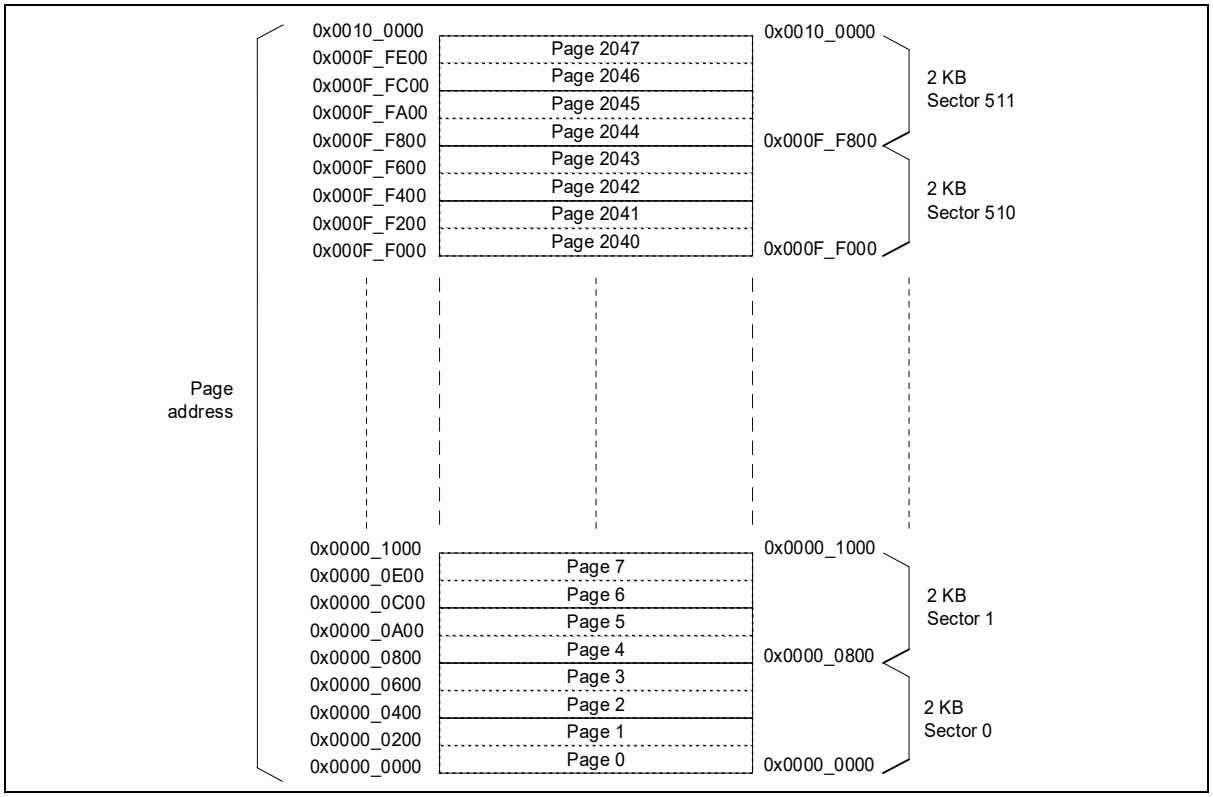
6.3.1.1 Code Flash Memory

Code flash memory has the features as shown below:

Table 39. Code Flash Memory Controller Features

Item	A34M420
Size	1,024 KB
Start address	0x0000_0000
End address	0x0010_0000
Page size	512 bytes
Total page count	2048 pages
PGM unit	4 bytes (1 word)
Erase unit	512 bytes 2 KB Bulk (full-chip)

Figure 40. Code Flash Memory Map



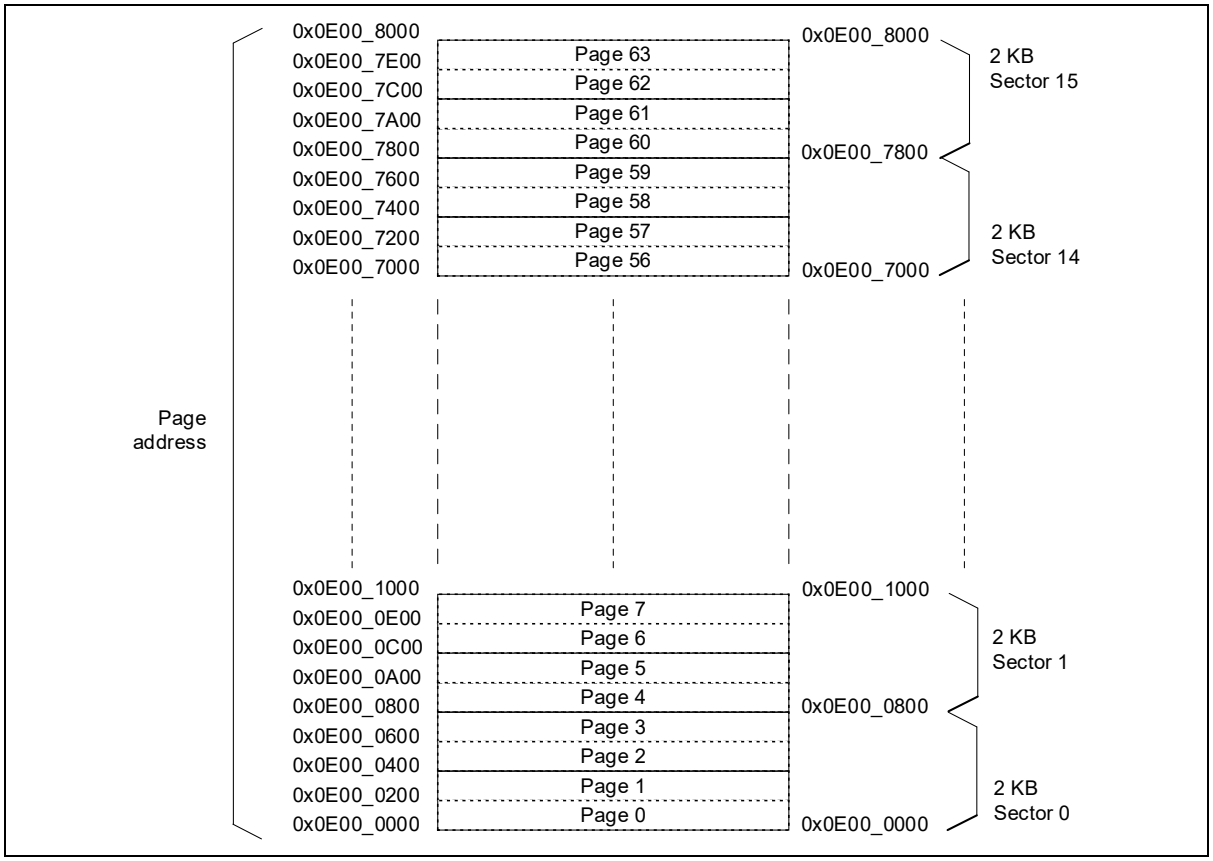
6.3.1.2 Data Flash Memory

Data flash memory has the features as shown below:

Table 40. Data Flash Memory Controller Features

Item	A34M420
Size	32 KB
Start address	0x0E00_0000
End address	0x0E00_8000
Page size	512 bytes
Total page count	64 pages
PGM unit	4 bytes (1 word)
Erase unit	512 bytes 2 KB Bulk (full-chip)

Figure 41. Data Flash Memory Map



6.3.2 Read Access Latency

Flash memory has a read latency that is a required wait time to access the flash memory, and it needs to be adjusted according to the microcontroller main clock.

How to adjust each latency of code flash memory is described in the following sections.

6.3.2.1 Code Flash Memory Wait Time

1. Determine the correct wait time corresponding to the microcontroller main clock by referring to Table 41.
2. Program the LATENCY[3:0] in CFMC_CONF register to have a new wait time.

Table 41. Internal Code Flash Memory Wait Time by Operating Clock

LATENCY[3:0]	Flash Memory Wait Time	Max. Available Clock Frequency
0000	0-clock wait	Up to 28 MHz
0001	1-clock wait	Up to 56 MHz
0010	2-clock wait	Up to 84 MHz
0011	3-clock wait	Up to 112 MHz
0100	4-clock wait	Up to 140 MHz
0101	5-clock wait	Up to 140 MHz
0110	6-clock wait	Up to 140 MHz
0111	7-clock wait	Up to 140 MHz
1000	8-clock wait	Up to 140 MHz
1001	9-clock wait	Up to 140 MHz
1010	10-clock wait	Up to 140 MHz
1011	11-clock wait	Up to 140 MHz
1100	12-clock wait	Up to 140 MHz
1101	13-clock wait	Up to 140 MHz
1110	14-clock wait	Up to 140 MHz
1111	15-clock wait	Up to 140 MHz

NOTE:

1. Incoming clock signal to the Flash must be less than the ‘Max. available clock frequency’ corresponding to the ‘Flash wait time’ set in the LATENCY[3:0] in CFMC_CONF. For more information of the incoming clocks, please refer to section 4.4.5.2.

6.3.2.2 Data Flash Memory Wait Time

1. Determine the correct wait time corresponding to the microcontroller main clock by referring to Table 42.
2. Program the LATENCY[3:0] bits in the DFMC_CONF register to have a new wait time.

Table 42. Internal Data Flash Memory Wait Time by Operating Clock

LATENCY[3:0]	Flash Memory Wait Time	Max. Available Clock Frequency
000	0-clock wait	Up to 28 MHz
001	1-clock wait	Up to 56 MHz
010	2-clock wait	Up to 84 MHz
011	3-clock wait	Up to 112 MHz
100	4-clock wait	Up to 140 MHz
101	5-clock wait	Up to 140 MHz
110	6-clock wait	Up to 140 MHz
111	7-clock wait	Up to 140 MHz

NOTE:

1. The clock signal to the flash memory must be less than the 'Max. Available Clock Frequency' corresponding to the 'Flash Memory Wait Time' set in the LATENCY[3:0] bits in the DFMC_CONF register. For more information of the incoming clocks, refer to section 4.4.5.2.

6.3.3 Usage of Cache

It is recommended that the code and instruction cache at code flash memory operate in the order described below:

For initial code operation:

1. Cache disable (CFMC_CONF[9:8] = '00')
2. Cache reset enable (CFMC_CTRL[25:24] = '11', Auto clear bit)
3. Cache enable (CFMC_CONF[9:8] = '11')

To disable cache:

1. Cache disable (CFMC_CONF[9:8] = '00')
2. Cache reset enable (CFMC_CTRL[25:24] = '11', Auto clear bit)

To enable cache:

1. Cache enable (CFMC_CONF[9:8] = '11')

If the active flash bank was erased or programmed, the data and instruction cache must be reset. See section 6.6 for a description of active and non-active banks.

6.3.4 Flash Memory Program and Erase Operations

Any interrupt must not be issued while erasing or programming the flash memory. The following routine must be executed before starting the flash memory control.

- All Interrupt disable
- Cache disable
- Interrupts are only available when programming and erasing a different bank area, and the Read-While-Write (RWW) function is also supported. However, to use the RWW function, bank mode must be enabled, and the flash memory must be programmed or erased in another bank.

For access to the flash memory, users must unlock the flash memory before starting the flash memory control, as described in the following section.

6.3.4.1 Access Control for Writing and Erasing Flash Memory

Before writing and erasing data on flash memory, an access enable process must be executed by setting the write protection registers. The access control of the flash memory follows the steps below:

1. Enter key values in the CFMC_FLSKEY or DFMC_FLSKEY register in the order described below:
   A. In normal input case:
      i. KEY1 → KEY2 → KEY3
   B. In non-consecutive input case:
      i. KEY1 → APB (No KEY) write → KEY2 → APB (No KEY) write → KEY3
2. Set the FLOCK in CFMC_CTRL or DFMC_CTRL register to ‘1’.

Read access of flash memory is always possible, regardless of whether read protection or write protection is enabled.

The flash erase operation must be proceeded before starting the flash memory programming.

6.3.5 Flash Memory Erase Sequences

6.3.5.1 Page Erase

Figure 42 shows a timing diagram for page erase operations in the flash memory.

Figure 42. Flash Memory Page Erase Timing Diagram

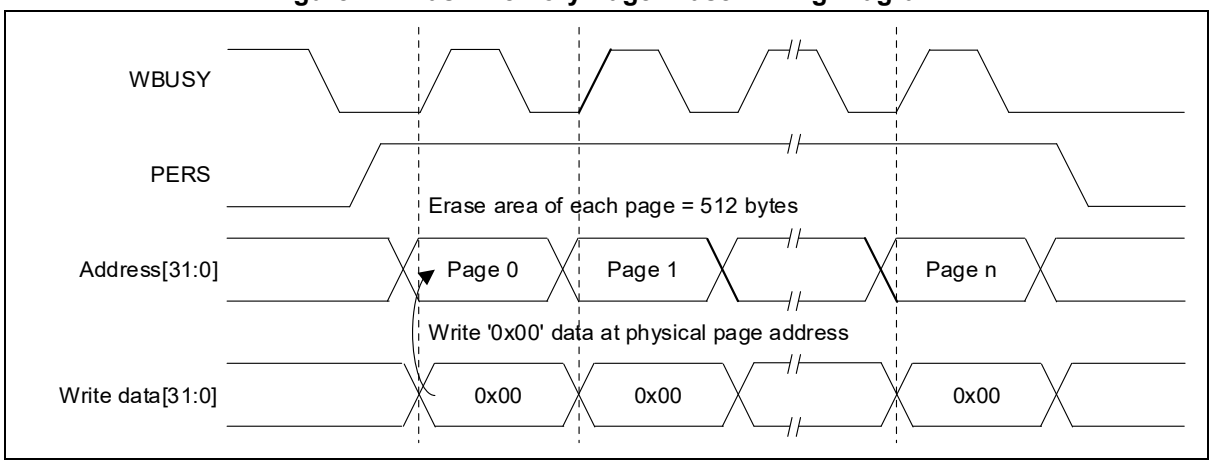


Figure 42 describes the following procedure for the page erase:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the PERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at physical page address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the PERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.5.2 2 KB Sector Erase

Figure 43 shows a timing diagram for 2 KB sector erase operations in the flash memory.

Figure 43. Flash Memory 2 KB Sector Erase Timing Diagram

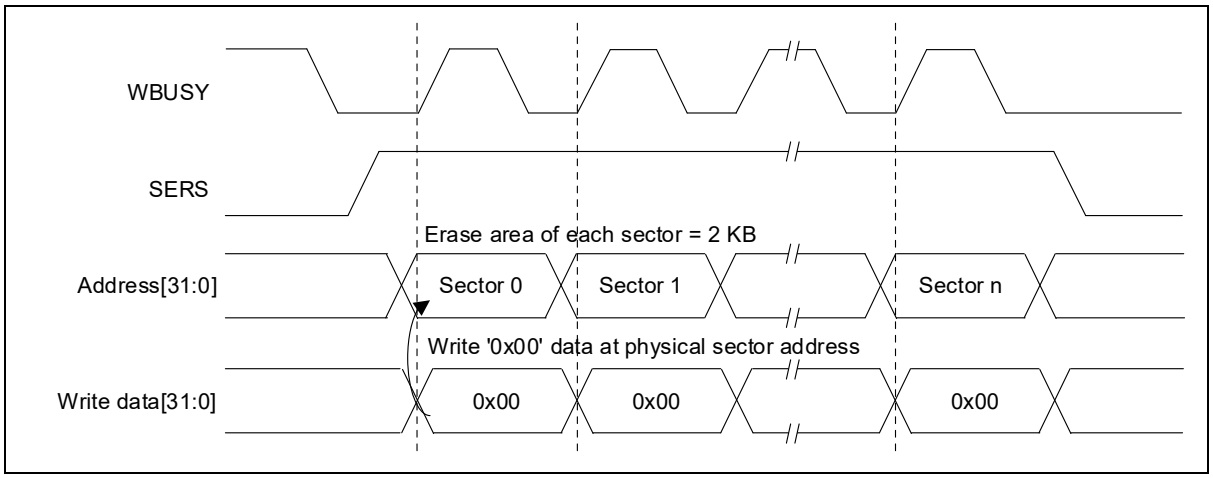


Figure 43 describes the following procedure for the 2 KB sector erase:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the SERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at physical 2 KB sector address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the SERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.5.3 Bulk Erase

Figure 44 shows a timing diagram for bulk (full-chip) erase operations in the flash memory.

Figure 44. Flash Memory Bulk (full-chip) Erase Timing Diagram

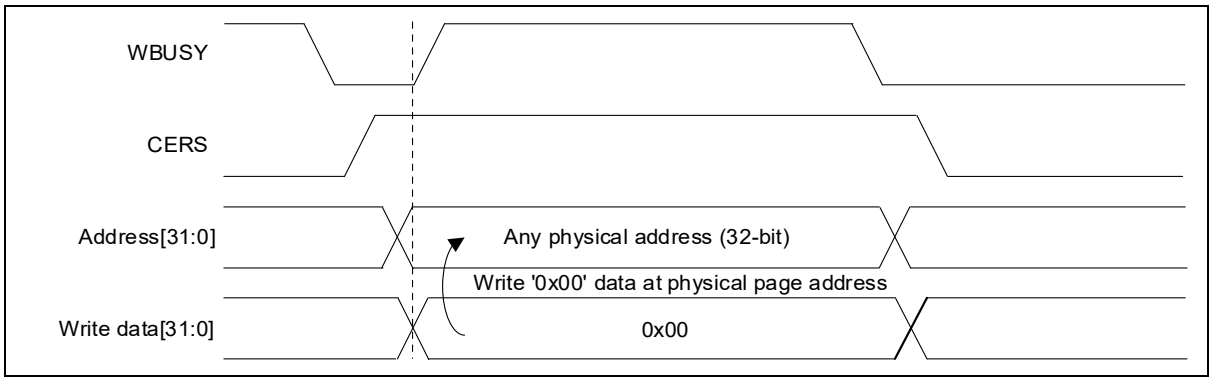


Figure 44 describes the following procedure for the bulk erase.

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the CERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at any physical address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the CERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.6 Flash Memory Programming Sequences

Before writing to the flash memory, it is important to unlock and unprotect the access area. Note that lock, protection, and all caches (instruction and data) must be disabled during the writing is complete.

6.3.6.1 Standard Programming

Figure 45. Flash Memory Program Timing Diagram

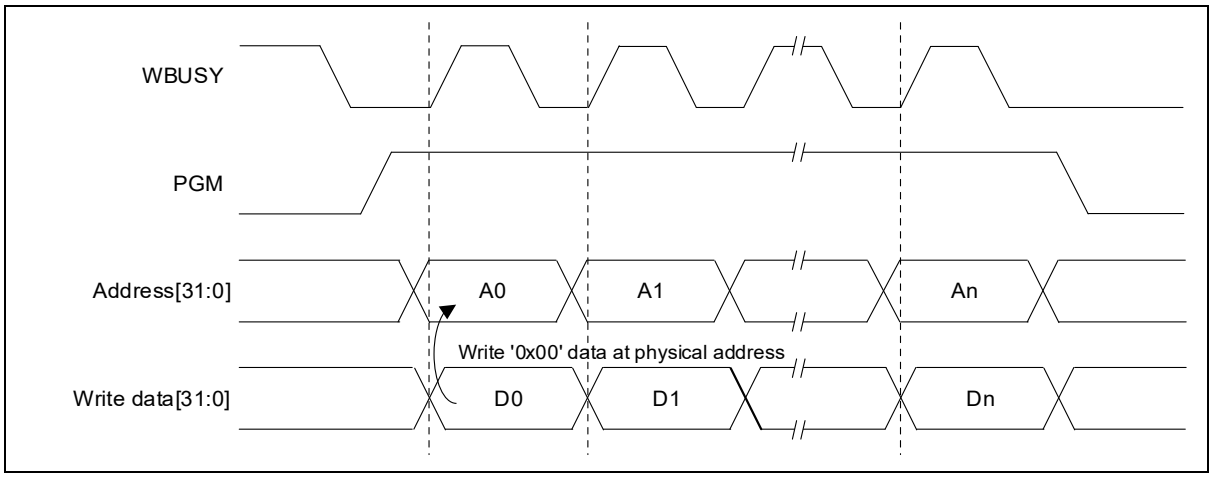


Figure 45 describes the following procedure for the standard programming:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the PGM bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write target data at physical address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the PGM bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.4 Flash Memory Protection

Read protection is a feature provided in the internal code flash memory and protects user code from external hacking. The read protection operates in three different protection mode levels such as unprotection (UNPROT), Level 1 (LVL1), and Level 2 (LVL2).

6.4.1 Read Protection

Table 43 describes three different protection mode levels of the read protection, such as UNPROT, LVL1, and LVL2.

Table 43. Available Operating Modes by Protection Level

Protection Level	Operation Mode	Main (code flash)		User Info0		User Info1 / 2 / 3		Main (data flash)	
		Read	Write	Read	Write	Read	Write	Read	Write
UNPROT	Normal mode	O	O	O	O	O	O	O	O
	Debug mode	O	O	O	O	O	O	O	O
	Boot mode	O	O	O	O	O	O	O	O
LVL1	Normal mode	O	O	O	N/A	O	O	O	O
	Debug mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Boot mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LVL2	Normal mode	O	O	O	N/A	O	O	O	O
	Debug mode	This mode cannot be entered.							
	Boot mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

NOTES:

1. Priority levels of the read protection mode are as follow:
   Normal protection to password protection
   A. Normal protection: LVL2 > LVL1 > UNPROT
   B. Password protection: LVL2 > LVL1 > UNPROT
2. A higher protection level cannot transition to a lower level without the bulk (chip) erase and User Info0 erase executed.
   e.g., procedure for transitioning from read protection LVL1 to UNPROT.
   A. Full-chip erase
   B. Check the CERSD flag in the CFMC_READPROT register.
   C. User Info0 erase
   D. Check the INFO0ERSD flag in the CFMC_READPROT register.
   E. Write '0xFF' to the RPROT[7:0] bits in the CFMC_READPROT register. After that, read protection LVL1 transition to UNPROT mode.

6.4.1.1 Read Protection: UNPROT (Unprotection)

Debug access is available, and there is no restriction on accessing the entire internal flash memory.

Table 44. Read Protection Operation in UNPROT Level

Normal Mode		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot / Code / SRAM	A	All	All	All
	User Info	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R/W	N/A
	Debug	R	All	All	All

NOTE:

1. R: read access, W: write access, All = read, write, erase access

6.4.1.2 Read Protection: LVL1 (Level-1)

Debug access is available, but the internal flash memory is read as '0x55AA_55AA' by external access. Therefore, the microcontroller cannot run while accessing the internal flash memory for debugging, and the flash memory read operation by DMA is not supported because the DMA settings can cause code leakage.

Table 45. Read Protection Operation in Level 1 (LVL1)

Read Protection Level-1		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot / Code / SRAM	R	All	All	R/W/(E)
	User Info	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R/W	N/A
	Debug	N/A	N/A	R/W	- /W/(E)

NOTE:

1. R: read access, W: write access, E: erase access, All = read, write and erase access

6.4.1.3 Read Protection: LVL2 (Level-2)

Debug access is not available, and the flash memory read operation by DMA is not supported because the DMA settings can cause code leakage.

Table 46. Read Protection Operation in Level-2 (LVL2)

Read Protection Level-2		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot / Code / SRAM	R	All	All	R/W/(E)
	User Info	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R/W	N/A
	Debug	N/A	N/A	N/A	N/A

NOTE:

1. R: read access, W: write access, E: erase access, All = read, write and erase access

6.4.1.4 Read Protection Setting

Users write data to the read protection area of the User Info0, and then the BootROM reads the data to enable the read protection when it runs on reset. After enabling the read protection, users must reset microcontroller to apply the read protection settings.

For the direct setup, users can write data in the read protection register to apply the settings immediately.

The read protection register is available for the change of the read protection mode level from low to high, and the change from high to low level is ignored. In addition, if a value other than the pre-set value is written to the read protection register, the highest level (LVL2) of the read protection mode level is applied.

6.4.1.5 Read Protection Removal

To remove the read protection, users must reset after deleting the related data in the User Info0. To delete the data of the User Info0 in the read protection mode, bulk erase must be executed for the code flash. The state when the bulk erase is complete for the code flash is stored and maintained until reset occurs.

6.4.1.6 Password Settings

Users can set the password only once by writing the password preset value to the CFMC_PWPRST register. If the input value of CFMC_PWIN register matches the preset value of the CFMC_PWPRST register, the read protection mode becomes normal mode.

For example, if users want to set the password to 0x1234_5678, the preset value 0x1234_5678 is written to the CFMC_PWPRST register as shown below:

```
CFMC_PWPRST = 0x12345678;
CFMC_PWPRST = 0x12345678;
```

6.4.2 Write Protection

Users can set up the write or erase protection for a certain area of the code and data flash. Before setting the protections, remember the followings:

- If any of the FPBY32_n or the FUBLP4K_y bit in the CFMC_FLS0PROT register and the CFMC_FLS1PROT register is ‘1’ (Protection), the Bulk Erase does not work because the protected area is included in the erase area.
- Similarly, the FUP512B_x bits in the CFMC_FLSPROT register take effect when the FPBY32K_15 bit is '1’. If the FPBY32_15 bit is set to ‘0’, the FUP512B_x bits are unavailable. For example, if the FPBY32K_15 bit is ‘1’ and the FUP512B_x bit is ‘0’, the FUP512B_x page becomes a Protected page. If the FPBY32K_15 bit is ‘1’ and the FUP512B_x bit is set to ‘1’, the corresponding page becomes Unprotected and allows page erase and writing, but sector erase and chip erase do not work for the region that includes Protected pages.
- Write Protection follows the physical address.

6.4.2.1 Write Protection for Code Flash Memory 0

The CFMC_FLS0PROT register is used to set the **Protection** and **Unprotection** as described in Table 47, Table 48, Table 49 and Table 50.

Table 47. Base Address of Protection in Code Flash Memory 0

Mode	Conditions	Base Address
Linear Mode	-	0x0000_0000
Bank Mode	No swap	0x0000_0000
	Swap	0x0008_0000

NOTES:

1. Code write protection is based on the physical address, so be careful when swapping banks.
   Write protection is set based on physical address, so in the above conditions, if 32 KB from 0x0000_0000 is write protected, 32 KB from 0x0008_0000 becomes write protection after bank swap.
2. Unprotection Selection bit in this table has a higher priority level than the Protection Selection bit.

Table 48. Code Flash Memory 0 Write Protection Area in 32 KB Unit

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_7FFF	1	0x0000_8000 ~ 0x0000_FFFF
2	0x0001_0000 ~ 0x0001_7FFF	3	0x0001_8000 ~ 0x0001_FFFF
4	0x0002_0000 ~ 0x0002_7FFF	5	0x0002_8000 ~ 0x0002_FFFF
6	0x0003_0000 ~ 0x0003_7FFF	7	0x0003_8000 ~ 0x0003_FFFF
8	0x0004_0000 ~ 0x0004_7FFF	9	0x0004_8000 ~ 0x0004_FFFF
10	0x0005_0000 ~ 0x0005_7FFF	11	0x0005_8000 ~ 0x0005_FFFF
12	0x0006_0000 ~ 0x0006_7FFF	13	0x0006_8000 ~ 0x0006_FFFF
14	0x0007_0000 ~ 0x0007_7FFF	15	0x0007_8000 ~ 0x0007_FFFF

Table 49. Code Flash Memory 0 Unprotection Area (bottom 32 KB) in 4 KB Unit

y	Offset	y	Offset
0	0x0000_0000 ~ 0x0000_0FFF	1	0x0000_1000 ~ 0x0000_1FFF
2	0x0000_2000 ~ 0x0000_2FFF	3	0x0000_3000 ~ 0x0000_3FFF
4	0x0000_4000 ~ 0x0000_4FFF	5	0x0000_5000 ~ 0x0000_5FFF
6	0x0000_6000 ~ 0x0000_6FFF	7	0x0000_7000 ~ 0x0000_7FFF

Table 50. Code Flash Memory 0 Unprotection Area (top 4 KB) in 512 Bytes Unit

x	Offset	x	Offset
0	0x0007_F000 ~ 0x0007_F1FF	1	0x0007_F200 ~ 0x0007_F3FF
2	0x0007_F400 ~ 0x0007_F5FF	3	0x0007_F600 ~ 0x0007_F7FF
4	0x0007_F800 ~ 0x0007_F9FF	5	0x0007_FA00 ~ 0x0007_FBFF
6	0x0007_FC00 ~ 0x0007_FDFF	7	0x0007_FE00 ~ 0x0007_FFFF

6.4.2.2 Write Protection for Code Flash Memory 1

The CFMC_FLS1PROT register is used to set the Protection and Unprotection as described in Table 47, Table 48, Table 49 and Table 50.

Table 51. Code Flash Memory 1 Base Address of Protection

Mode	Conditions	Base Address
Linear mode	-	0x0008_0000
Bank mode	No swap	0x0008_0000
	Swap	0x0000_0000

NOTES:

1. Code write protection is based on the physical address, so be careful when swapping banks.
   Write protection is set based on physical address, so in the above conditions, if the 32 KB area from 0x0000_0000 is write protected, the 32 KB area from 0x0008_0000 becomes write protection after bank swap.
2. Unprotection Selection bit in this table has a higher priority level than the Protection Selection bit.

Table 52. Code Flash Memory 1 Write Protection Area in 32 KB Unit

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_7FFF	1	0x0000_8000 ~ 0x0000_FFFF
2	0x0001_0000 ~ 0x0001_7FFF	3	0x0001_8000 ~ 0x0001_FFFF
4	0x0002_0000 ~ 0x0002_7FFF	5	0x0002_8000 ~ 0x0002_FFFF
6	0x0003_0000 ~ 0x0003_7FFF	7	0x0003_8000 ~ 0x0003_FFFF
8	0x0004_0000 ~ 0x0004_7FFF	9	0x0004_8000 ~ 0x0004_FFFF
10	0x0005_0000 ~ 0x0005_7FFF	11	0x0005_8000 ~ 0x0005_FFFF
12	0x0006_0000 ~ 0x0006_7FFF	13	0x0006_8000 ~ 0x0006_FFFF
14	0x0007_0000 ~ 0x0007_7FFF	15	0x0007_8000 ~ 0x0007_FFFF

Table 53. Code Flash Memory 1 Unprotection Area (bottom 32 KB) in 4 KB Unit

y	Offset	y	Offset
0	0x0000_0000 ~ 0x0000_0FFF	1	0x0000_1000 ~ 0x0000_1FFF
2	0x0000_2000 ~ 0x0000_2FFF	3	0x0000_3000 ~ 0x0000_3FFF
4	0x0000_4000 ~ 0x0000_4FFF	5	0x0000_5000 ~ 0x0000_5FFF
6	0x0000_6000 ~ 0x0000_6FFF	7	0x0000_7000 ~ 0x0000_7FFF

Table 54. Code Flash Memory 1 Unprotection Area (top 4 KB) in 512 Bytes Unit

x	Offset	x	Offset
0	0x0007_F000 ~ 0x0007_F1FF	1	0x0007_F200 ~ 0x0007_F3FF
2	0x0007_F400 ~ 0x0007_F5FF	3	0x0007_F600 ~ 0x0007_F7FF
4	0x0007_F800 ~ 0x0007_F9FF	5	0x0007_FA00 ~ 0x0007_FBFF
6	0x0007_FC00 ~ 0x0007_FDFF	7	0x0007_FE00 ~ 0x0007_FFFF

6.4.2.3 Write Protection for Data Flash Memory

The DFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 55 and Table 56.

Table 55. Data Flash Memory Write Protection Area in 2 KB Unit

N	Address	N	Address
0	0x0E00_0000 ~ 0x0E00_07FF	1	0x0E00_0800 ~ 0x0E00_0FFF
2	0x0E00_1000 ~ 0x0E00_17FF	3	0x0E00_1800 ~ 0x0E00_1FFF
4	0x0E00_2000 ~ 0x0E00_27FF	5	0x0E00_2800 ~ 0x0E00_2FFF
6	0x0E00_3000 ~ 0x0E00_37FF	7	0x0E00_3800 ~ 0x0E00_3FFF
8	0x0E00_4000 ~ 0x0E00_47FF	9	0x0E00_4800 ~ 0x0E00_4FFF
10	0x0E00_5000 ~ 0x0E00_57FF	11	0x0E00_5800 ~ 0x0E00_5FFF
12	0x0E00_6000 ~ 0x0E00_67FF	13	0x0E00_6800 ~ 0x0E00_6FFF
14	0x0E00_7000 ~ 0x0E00_77FF	15	0x0E00_7800 ~ 0x0E00_7FFF

Table 56. Data Flash Memory Unprotection Area (top 4 KB) in 512 Bytes Unit

X	Address	X	Address
0	0x0E00_7000 ~ 0x0E00_71FF	1	0x0E00_7200 ~ 0x0E00_73FF
2	0x0E00_7400 ~ 0x0E00_75FF	3	0x0E00_7600 ~ 0x0E00_77FF
4	0x0E00_7800 ~ 0x0E00_79FF	5	0x0E00_7A00 ~ 0x0E00_7BFF
6	0x0E00_7C00 ~ 0x0E00_7DFF	7	0x0E00_7E00 ~ 0x0E00_7FFF

6.4.2.4 Write Protection with Code Flash Memory Bank Swap

Write protection by the CFMC_FLS0PROT and CFMC_FLS1PROT registers varies in area depending on the swap status of the memory bank. Applications that require memory area data updates while user code is running should disable write protection to allow data updates only in inactive bank areas depending on memory swap status. Write protection follows the physical address.

Write protected coverage of Non-active block before Bank Swap

CFMC_FLS0PROT = 0x0000_0000

CFMC_FLS1PROT = 0x0000_FFFF

- Active block (0x00000 to 0x7FFFF) is write/erase accessible.
- Non-active block (0x80000 to 0xFFFFF) is write/erase inaccessible.

Write protected coverage of Non-active block after Bank Swap

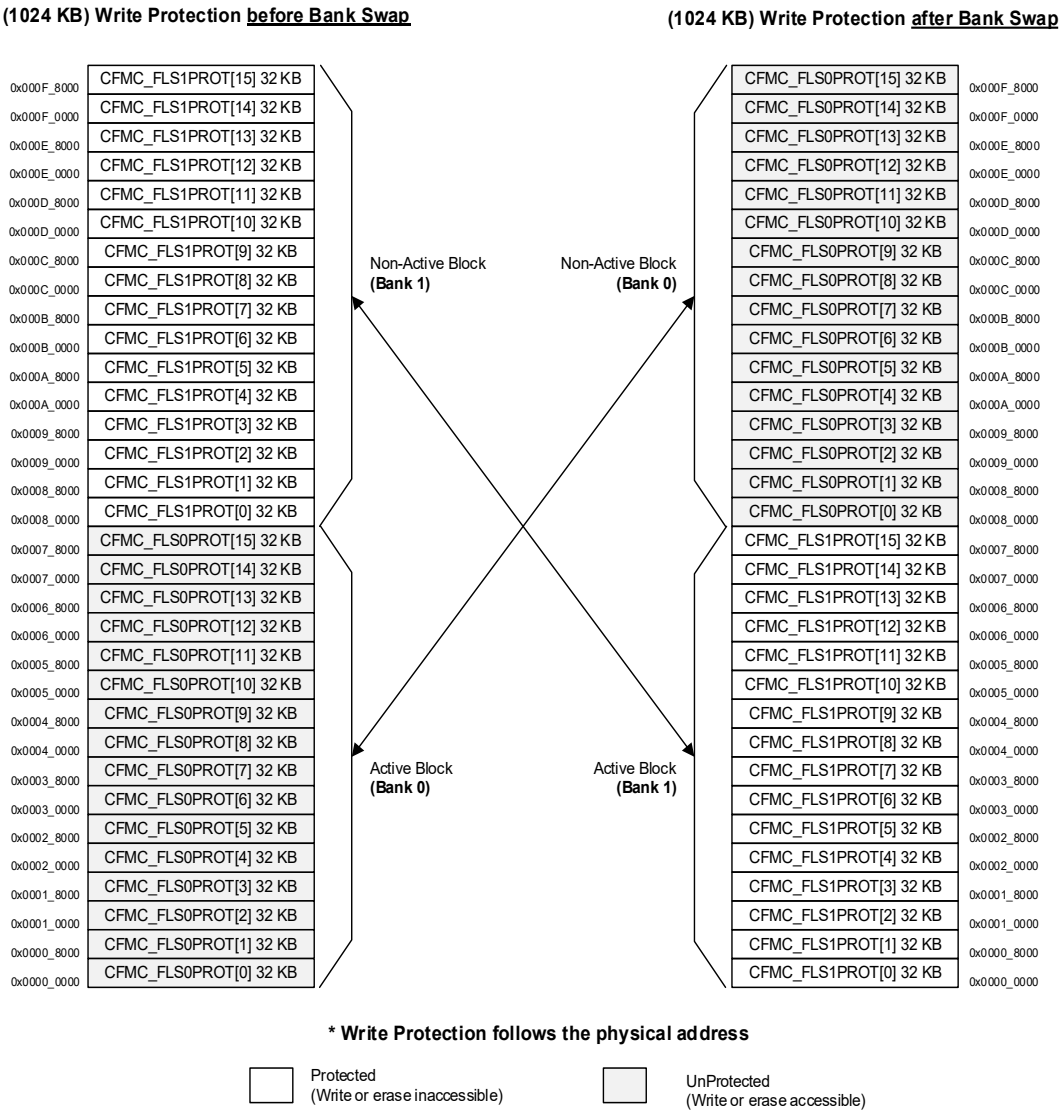
CFMC_FLS0PROT = 0x0000_0000

CFMC_FLS1PROT = 0x0000_FFFF

- Active block (0x00000 to 0x7FFFF) is write/erase inaccessible.
- Non-active block (0x80000 to 0xFFFFF) is write/erase accessible.

As above, the write protection area changes after the bank swap.

Figure 46. The Write-Protection Coverage of Code Flash Memory when using Memory Swap



6.4.3 Securable Memory Area (LVL1, LVL2)

Using the read protection, users can secure the flash memory area against being read by external devices (refer to 6.4.1).

6.4.4 Disabling Core Debug Access (LVL2)

Using the read protection, users can block the debugger connections (refer to 6.4.1).

6.5 Flash Memory CRC Check

The A34M420 uses the CRC-CCITT to calculate the CRC check for the code and data flash memories.

Its internal data input bus width is 128 bits, and the CRC check control sequence is as follows:

Background mode (CRC check is running at only flash memory idle state):

1. Set the CRC check start address.
2. Set the CRC check end address.
3. Reset the CRC check data.
4. Enable the BGEN (Background mode Enable) bit in the CFMC_CHKCTRL (or DFMC_CHKCTRL) register.
5. Check the CBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the CDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.

Burst mode (CRC Check is running continuously with microcontroller core halt state):

1. Set the CRC check start address.
2. Set the CRC check end address.
3. Reset the CRC check data.
4. Enable the BSTEN (Burst mode enable) bit in the CFMC_CHKCTRL (or DFMC_CHKCTRL) register.
5. Check the CBUSY flag in the CFMC_STAT register at polling mode or the CDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.

6.6 Bank Swap of Code Flash Memory

6.6.1 Bank Swap of Code Flash Memory

The A34M420 provides a flash memory bank selection function for user applications and stable update and execution of User Bootloader.

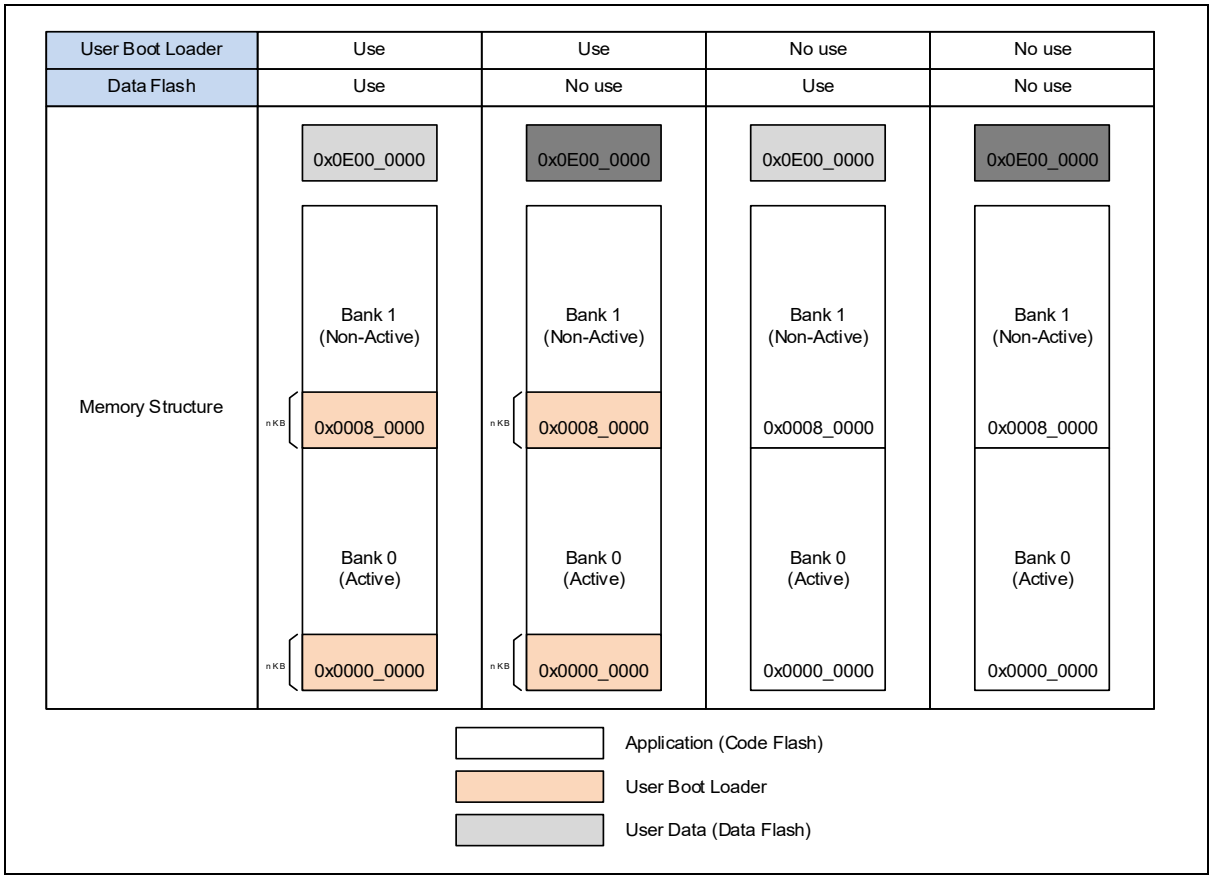
To enable Memory Swap function in the A34M420, users need to divide 1024 KB flash memory into two of 512 KB and develop them for programming.

To use the Memory Swap function in user applications, a user is required to acquire information of Bank Selection registers and memory structures.

Table 57. Memory Bank Usage Status of Code Flash Memory

CFMC_BANK <BM>	CFMC_BANK <BS>	Bank Status	Active block	Non-Active block
1	1	Disable	Single	
0	1	Enable	Bank0	Bank1
0	0	Swap	Bank1	Bank0

Figure 47. A34M420 Memory Structure with Bank Swap Available



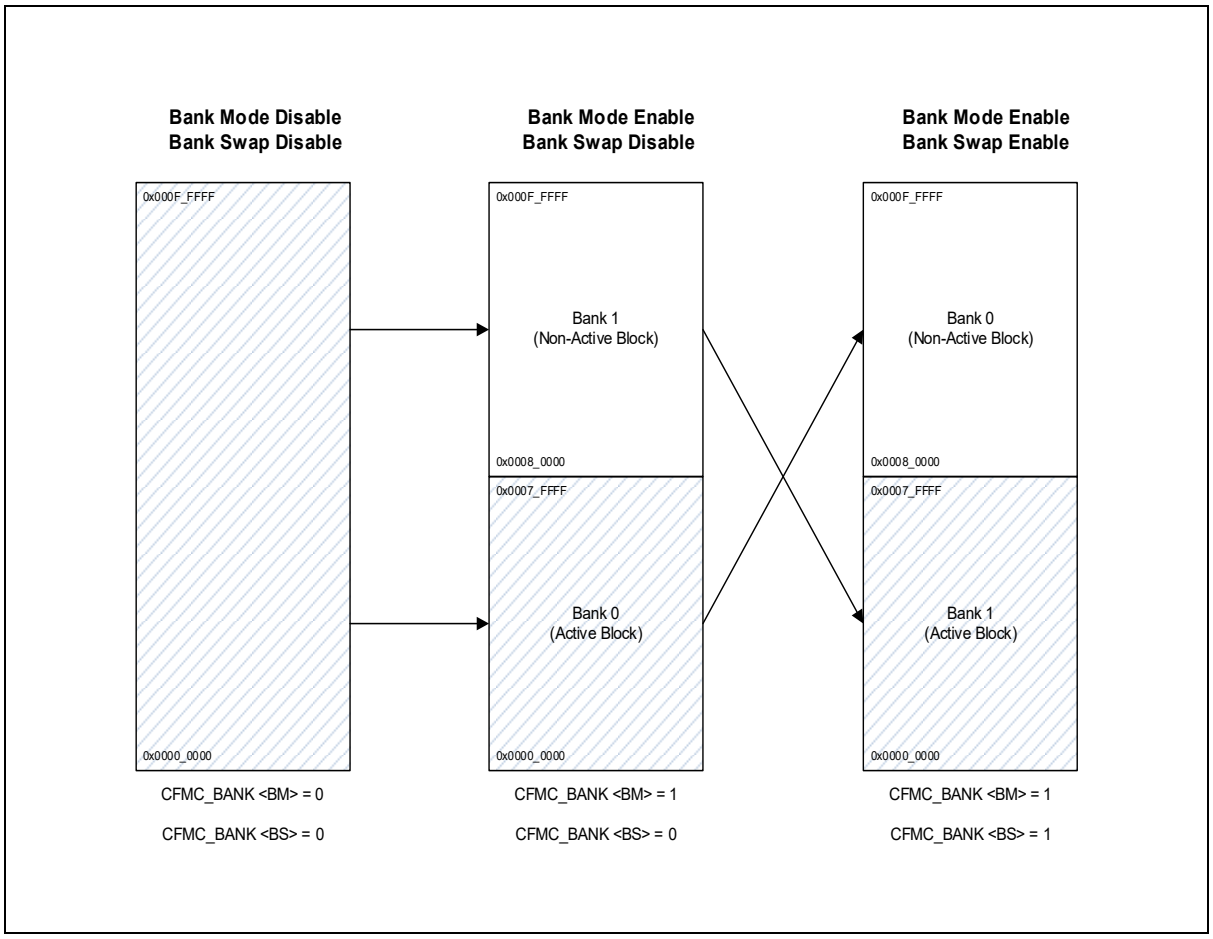
RWW function works when writing non-active bank in bank mode.

6.6.2 Bank Swap Operation

When the Memory Bank Swap function is enabled, memory block starting with address 0x0000_0000 is considered as Active block, where code will be executed after the BootROM operation. Next to the active block, non-active block is located where code will be updated or is used for back up operation.

When the Memory Bank Swap function is enabled, certain code is updated in the non-active area, Swap control bit is set, and the system restarts successively. Then the value of Swap Flag (SFLAG) is monitored in BootROM and switching between the Active block and the non-active block occurs. By the Swap operation, the non-active block where the code was updated switches to the Active block, while the Active block where the code was executed before the update operation switches to the non-active block.

Figure 48. Logical Address Change of the Code Flash Memory by Bank Swap Settings



NOTE:

1. Note that write protection must be set based on physical address during bank swap mode (see Figure 46).

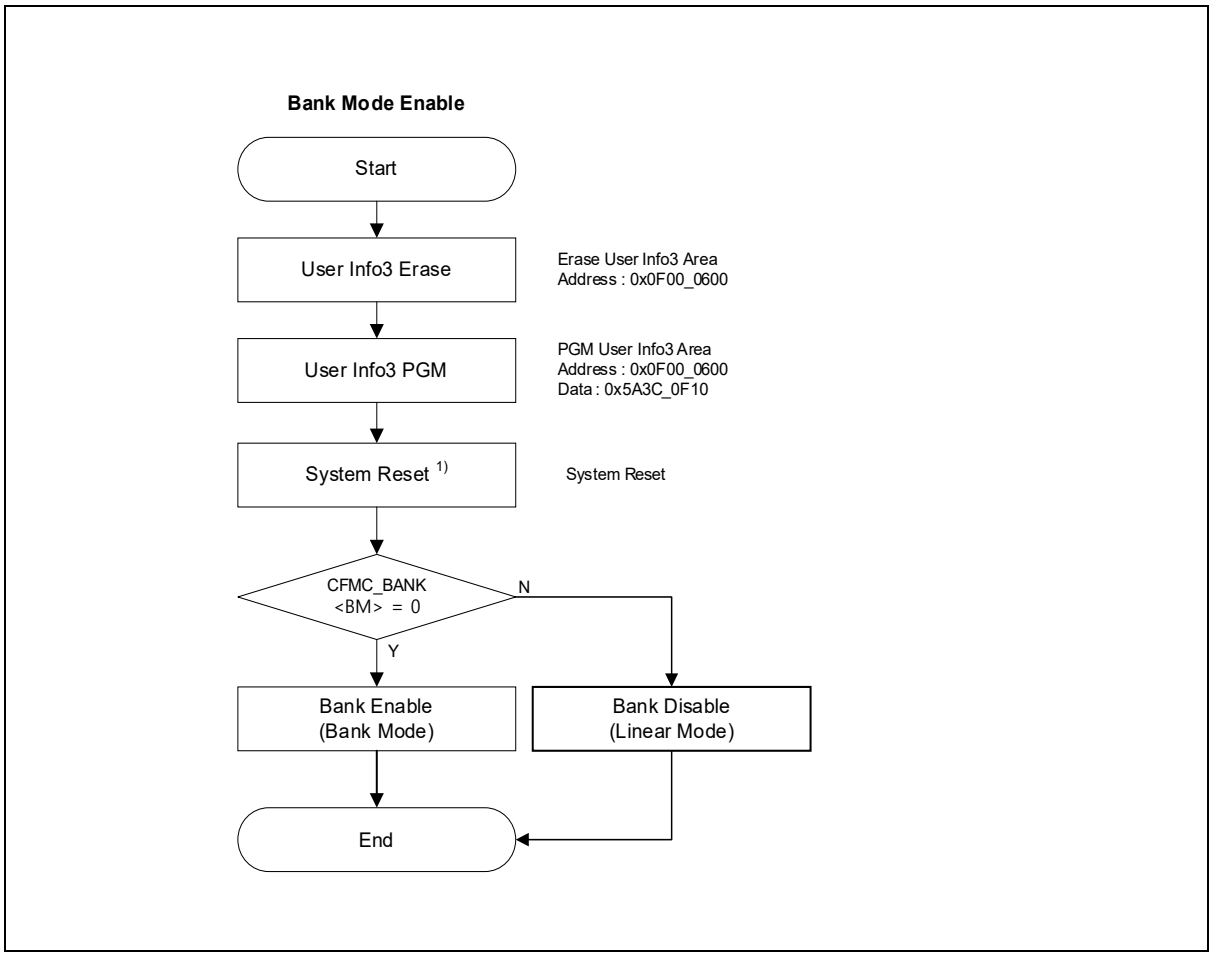
6.6.3 Limitations and Recommendations

When a main program is running in Bank1 area, where the read protection is applied, if full-chip erase is executed to disable only the read protection permanently, user data in whole area of Bank 0 and Bank1 is cleared. After system reset, the read protection is disabled. However, since Bank Swap function is still active, program in Bank1 is supposed to be executed but Bank1 doesn't have any command at all.

6.6.4 Sequence of Bank Mode Enable or Swap

Figure 49, Figure 50, and Figure 51 show how to enable Bank mode and use Bank swap in the code flash memory. Please note that to disable Bank mode or Bank swap, users must erase the User Info3 area. Refer to Table 57 for the memory bank usage status of the code flash memory.

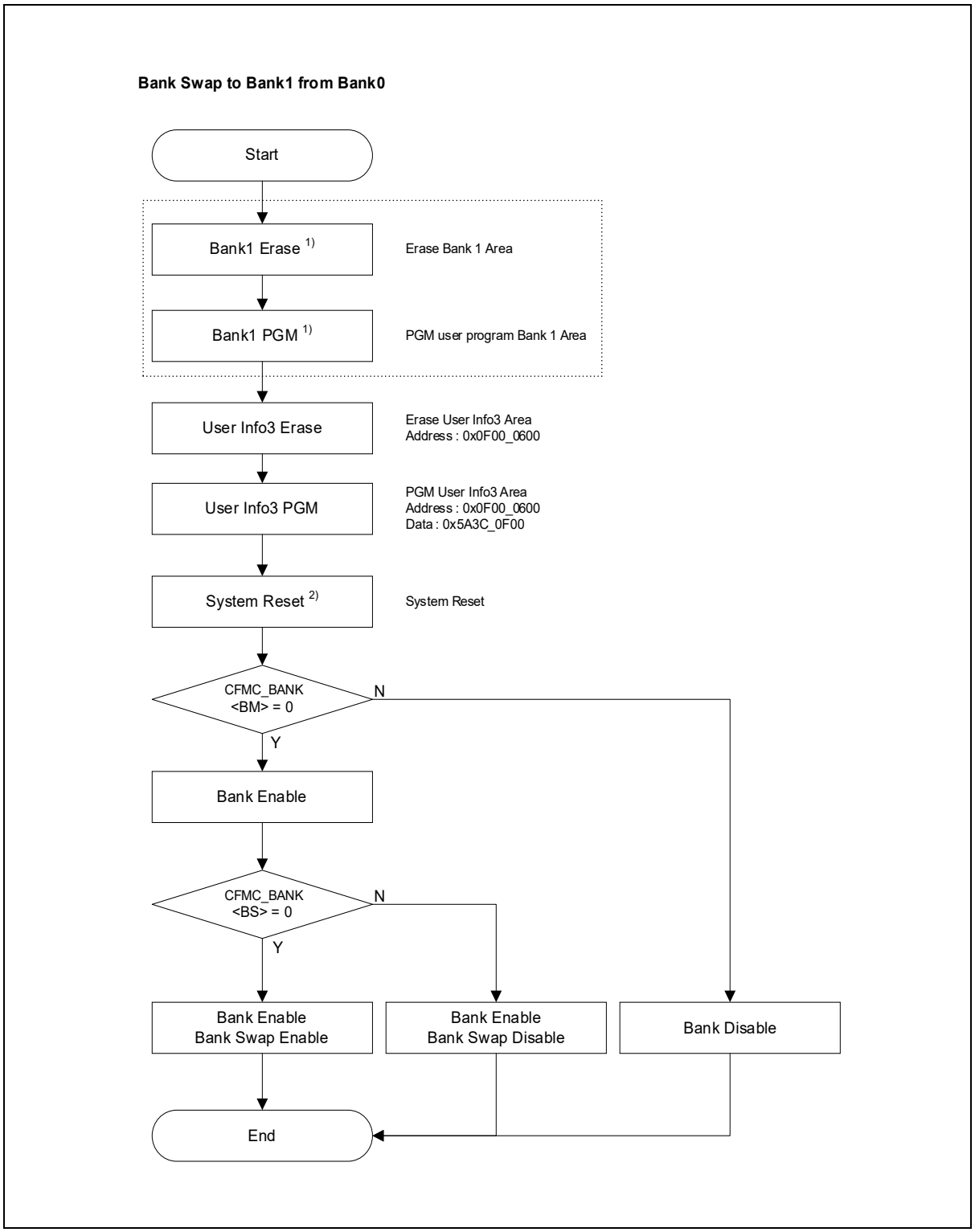
Figure 49. Sequence of Bank Mode Enable



NOTE:

1. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

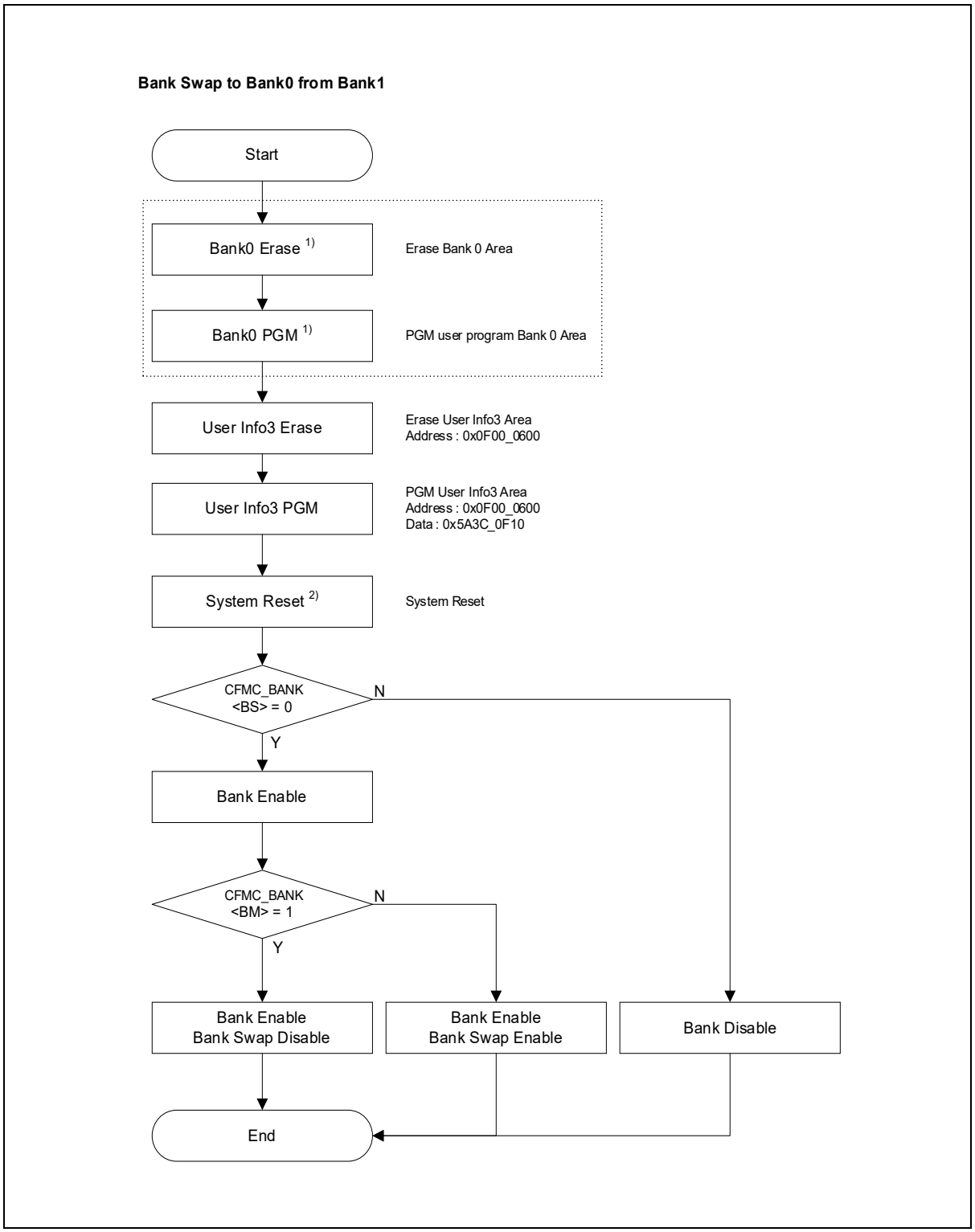
Figure 50. Sequence of Bank Swap: Bank0 → Bank1



NOTES:

1. When performing Bank Swap from Bank1 to Bank0, program can be executed in Bank0 after system reset only if there is a program in Bank0.
2. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

Figure 51. Sequence of Bank Swap: Bank1 → Bank0



NOTES:

1. When performing Bank Swap from Bank1 to Bank0, program can be executed in Bank0 after system reset only if there is a program in Bank0.
2. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

6.7 RWW (Read-While-Write)

The A34M420 supports RWW(Read-While-Write) function, which allows the microcontroller to be programmed while running from code flash memory. However, RWW(Read-While-Write) is only available under the following conditions.

Requires Bank enable or Bank swap. However, the same bank does not support RWW even if it is in Bank Enable or Bank Swap state. Refer to Figure 48

Bank Disable does not support RWW. Refer to the 6.10.16

Table 58. Code Flash Memory RWW Features (Bank Disable)

Bank Status: Enable	Bank0 (Active block)		Bank1 (Non-Active block)	
	Erase	Write	Erase	Write
Bank0 (Active block)	X	X	X	X
Bank1 (Non-Active block)	X	X	X	X

Table 59. Code Flash Memory RWW Features (Bank Enable)

Bank Status: Enable	Bank0 (Active block)		Bank1 (Non-Active block)	
	Erase	Write	Erase	Write
Bank0 (Active block)	X [1)]	X [1)]	O	O
Bank1 (Non-Active block)	O	O	X [1)]	X [1)]

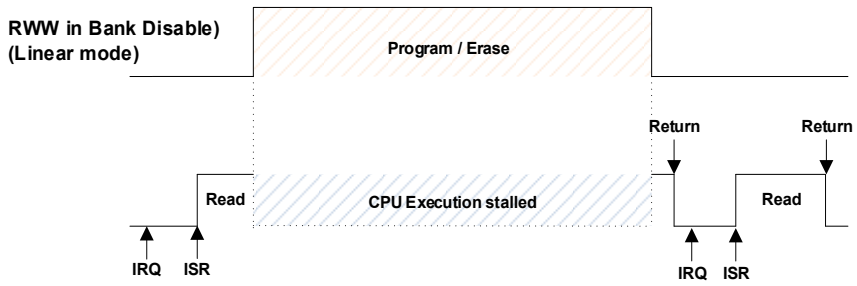
Table 60. Code Flash Memory RWW Features (Bank Swap)

Bank Status: Enable	Bank1 (Active block)		Bank0 (Non-Active block)	
	Erase	Write	Erase	Write
Bank1 (Active block)	X [1)]	X [1)]	O	O
Bank0 (Non-Active block)	O	O	X [1)]	X [1)]

NOTE:

1. CPU execution is stalled. Ex) Interrupt doesn't operate during erase and write.

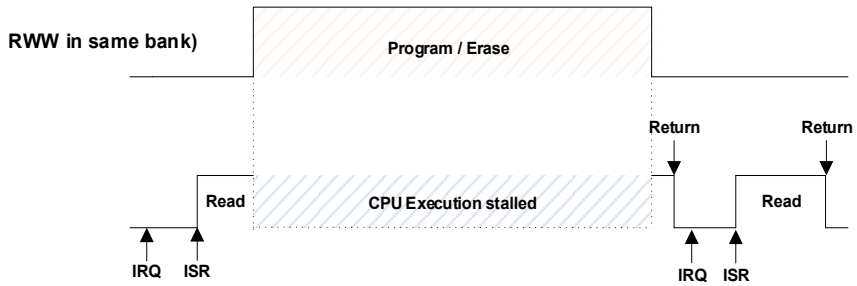
Figure 52. RWW in Code Flash Bank Disable



NOTE:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

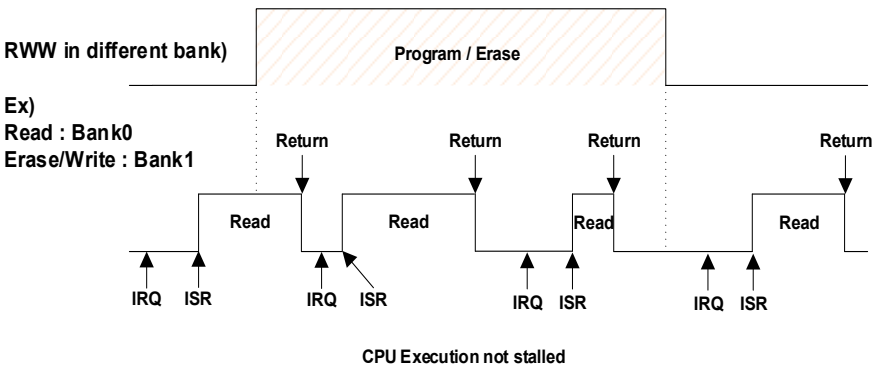
Figure 53. RWW in Code Flash Bank Enable and Bank Swap (Same Bank)



NOTE:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

Figure 54. RWW in Code Flash Bank Enable and Bank Swap (Different Bank)



NOTE:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

6.8 ECC (Error Correction Code)

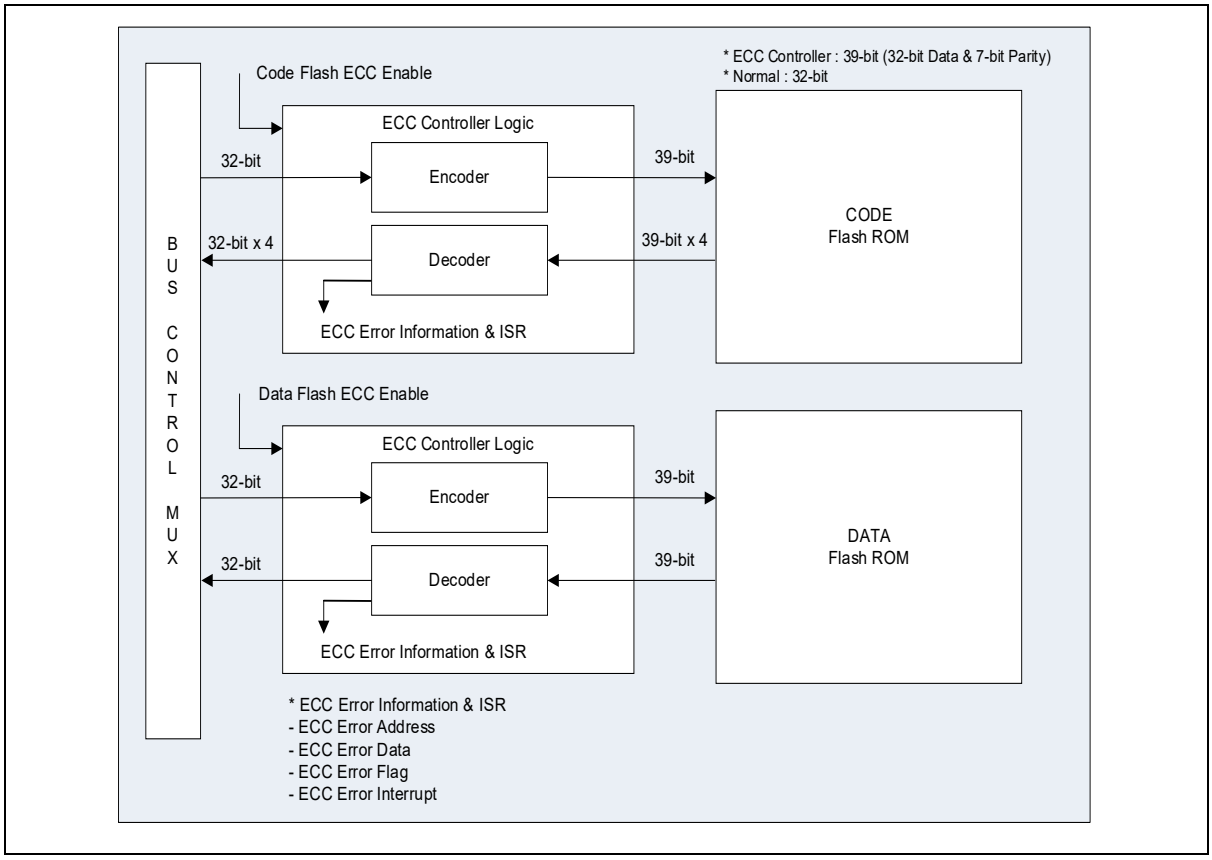
The ECC is based on an optimal minimum odd weight error parity code class that provides better performance than SECDED based on the Hamming code principle, and its use provides improved protection against bit flipping in flash memory. When writing in word (32-bit) units in the Code or Data Flash area, normal ECC parity (7-bit) is generated for each word, and the parity bit is impossible to read. Single-bit or double-bit errors can be detected by comparing the generated parity bit with the flash area word when reading code or data flash memory, and errors can be prevented using ECC. However, in the case of a single bit error, the ECC controller corrects it, but it is difficult to correct a double bit error, but it is possible to detect the error flag and handle it through software.

Please note that if you program in units of one byte, two bytes, etc. instead of words (four bytes), multiple ECC errors may occur. To use ECC, you must program in word units.

6.8.1 Block Diagram

When programming in code or data flash memory, 39-bit including 7-bit parity are stored in the flash memory through the encoder of the ECC block, and when reading the flash memory, the value stored in the flash memory is checked through the decoder to ensure that it does not change. Users can check the error status in the CFMC_STAT Register if an ECC error occurs.

Figure 55. ECC Block Diagram



6.8.2 ECC Interrupt Service Routine

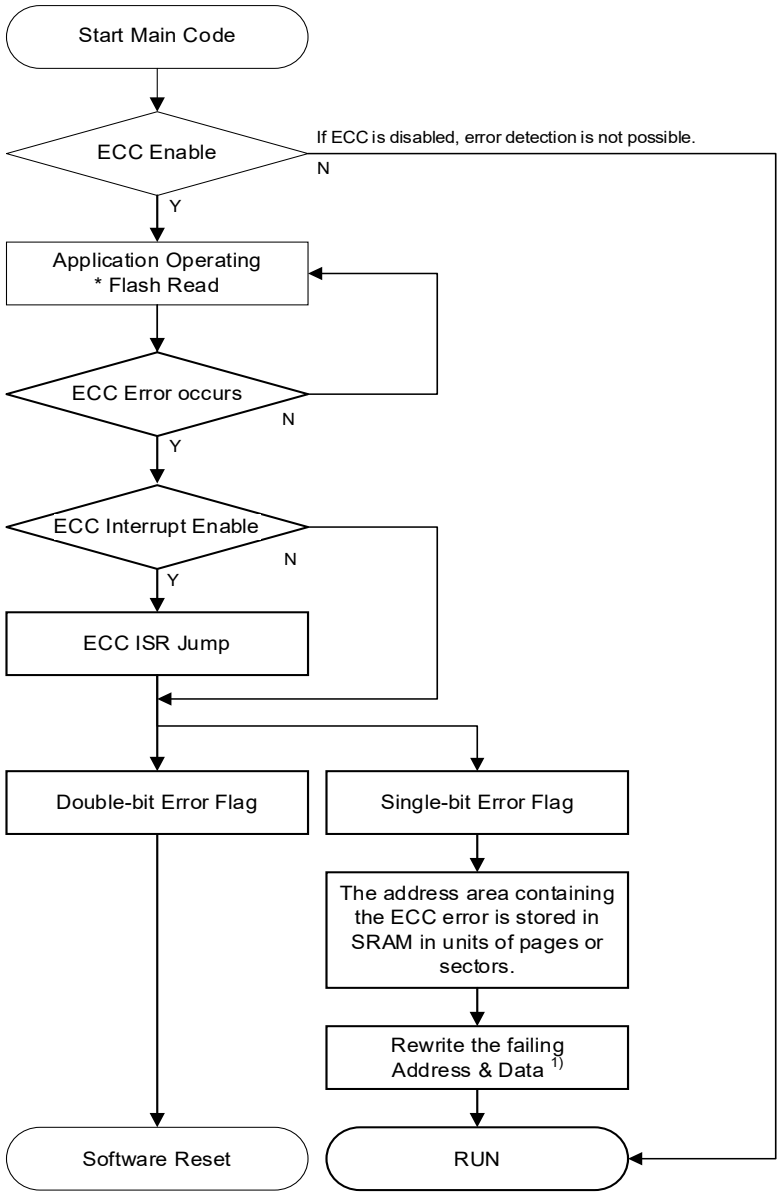
The ECC interrupt service routine provides an opportunity to take immediate action when an ECC error occurs. If an ECC error occurs when ECC interrupts are enabled, it will branch to the NMI interrupt or CFMC and DFMC interrupt routine.

In the case of single-bit error when reading the flash memory, a bit error occurs in the flash memory value, but it is automatically corrected by the ECC controller. If a single-bit error occurs, other bits in the same word may be additionally damaged later, resulting in a double-bit error. Therefore, it is recommended that the corrected data value be rewritten to the address where the ECC error occurred. When rewriting, it must be in a page or sector unit that contains the ECC error address (see Figure 56).

If it changes to a double-bit error, it is difficult to correct. The ECC controller can correct single-bit errors but not double-bit errors and reports the error only as an error flag.

6.8.3 Sequence of the ECC

Figure 56. Sequence of the ECC



NOTES:

1. If single bit error occurs, check the ECC error address and store the page or sector in the address area in SRAM. Then, users must erase and write the address where the error occurred.
   For example, if an ECC error occurs at 0x7300, when erasing and writing by page (ex, 512 bytes), 0x7200 to 0x73FF must be stored in SRAM, and then SRAM stored from 0x7200 to 0x73FF must be erased and written.
2. If double-bit error occurs, it is recommended to force a shutdown because it is impossible to know what application operation is being performed at the address where the error occurred without implementing it in software.

6.9 Flash Memory Interrupts

For the code and data flash control, A34M420 supports interrupt function. This interrupt can be activated by setting the CFMC_CTRL (or DFMC_CTRL) registers. To clear the error flag bit, Write “1” to this bit.

For the code and data flash control, A34M420 supports the write done interrupt. This interrupt can be activated by setting the WDIEN bit in the CFMC_CTRL (or DFMC_CTRL) register to ‘1’.

WDIEN (Write done interrupt)

- This interrupt occurs when word write is completed. The corresponding interrupt flag is the WDONE bit of CFMC_STAT (or DFMC_STAT) register.

ECCSI (ECC single-bit error interrupt)

- This interrupt occurs when one-bit changed value is read instead of the original written value of code (or data) flash memory. The corresponding interrupt flag is the ECCSE bit of the CFMC_STAT (or DFMC_STAT) register.

ECCDN (ECC double error interrupt)

- This interrupt occurs when a value that has been changed by more than two bits, other than the original written value of code (or data) flash memory, is read. The corresponding interrupt flag is the ECCDE bit of the CFMC_STAT (or DFMC_STAT) register.

6.10 Code Flash Memory Controller Registers

The base address and register map of the Code Flash Memory Controller (CFMC) are described in the following tables.

Table 61. Base Address of CFMC

Name	Base Address
CFMC	0x4100_0000

Table 62. CFMC Register Map

Name	Offset	Type	Description	Reset Value	Ref.
CFMC_CONF	0x0000	RW	Code Flash Control Register	0x0000_0000	6.10.1
CFMC_FLSKEY	0x0004	WO	Code Flash Access Key Register	0x0000_0000	6.10.2
CFMC_INFOKEY	0x0008	WO	Code Flash User Info Access Key Register	0x0000_0000	6.10.3
CFMC_FLS0PROT	0x000C	RW	Code Flash 0 Protection Register	0xFF00_0000	6.10.4
CFMC_FLS1PROT	0x0010	RW	Code Flash 1 Protection Register	0xFF00_0000	6.10.5
CFMC_INFOPROT	0x0014	RW	Code Flash User Info. Protection Register	0x0000_0000	6.10.6
CFMC_CTRL	0x0018	RW	Code Flash Access Control Register	0xC000_0000	6.10.7
CFMC_STAT	0x001C	RW	Code Flash Access Status Register	0x0000_0000	6.10.8
CFMC_READPROT	0x0020	RW	Code Flash Read Protection Register	0x0000_00FF	6.10.9
CFMC_PWIN	0x0024	WO	Code Flash Password Input Register	0x0000_0000	6.10.10
CFMC_CHKCTRL	0x0030	RW	Code Flash CRC Check Control Register	0x0000_0000	6.10.11
CFMC_CHKDOUT	0x0034	RO	Code Flash CRC Check Data Output Register	0x0000_FFFF	6.10.12
CFMC_CHKSADDR	0x0038	RW	Code Flash CRC Check Start Address Register	0x0000_0000	6.10.13
CFMC_CHKEADDR	0x003C	RW	Code Flash CRC Check End Address Register	0x0000_00FF	6.10.14
CFMC_WTOUT	0x0040	RW	Code Flash Write Timeout Register	0x0000_0000	6.10.15
CFMC_BANK	0x0050	RW	Code Flash Bank Control Register	0x0000_0011	6.10.16

Table 62. CFMC Register (continued)

Name	Offset	Type	Description	Reset Value	Ref.
CFMC_ECCEADDR	0x0060	RO	Code Flash ECC Error Address Register	0x0000_0000	6.10.17
CFMC_ECCEDATA0	0x0064	RO	Code Flash ECC Error Data0 Register	0x0000_0000	6.10.18
CFMC_ECCEDATA1	0x0068	RO	Code Flash ECC Error Data1 Register	0x0000_0000	6.10.19
CFMC_ECCEDATA2	0x006C	RO	Code Flash ECC Error Data2 Register	0x0000_0000	6.10.20
CFMC_ECCEDATA3	0x0070	RO	Code Flash ECC Error Data3 Register	0x0000_0000	6.10.21
CFMC_ECCEPARITY	0x0074	RO	Code Flash ECC Error Parity Register	0x0000_0000	6.10.22
CFMC_PWPRST	0x0F30	WO	Code Flash Password Preset Register	0x0000_0000	6.10.23

6.10.1 CFMC_CONF: Code Flash Control Register

The CFMC_CONF register is a control register for the internal flash memory. This register is 32 bits wide.

CFMC_CONF=0x4100_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
ECCDIS	Reserved	WTEN	Reserved	DCEN	ICEN	Reserved	LATENCY[3:0]
0	-	0	-	0	0	-	0000
RW	-	RW	-	RW	RW	-	RW

31	ECCDIS	Disable Code FLASH ECC
		0 ECC enable
		1 ECC disable
25	WTEN	Enable the code flash memory write operation.
		0 Disable
		1 Enable
9	DCEN	Code data cache enable bit
		0 Disable
		1 Enable
8	ICEN	Instruction cache enable bit
		0 Disable
		1 Enable
3 0	LATENCY[3:0][2]	Flash memory wait value bits The flash wait value between 0 and 15.

NOTES:

1. The following is how to set up a cache.
   A. Disable instruction and code data cache.
   B. Reset instruction and code data cache (auto-clear bit).
   C. Enable instruction and code data cache.
2. For detailed information of the LATENCY[3:0], refer to section 6.3.2.

6.10.2 CFMC_FLSKEY: Code Flash Access Key Register

The CFMC_FLSKEY register is the access key register for the internal flash memory. This register is 32bits wide.

CFMC_FLSKEY=0x4100_0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FKEY[31:0]																															
0x00000000																															
WO																															

31 0	EKEY[31:0]	These bits are the key values used to access the flash memory. FKEY[31:0] values must be written to this register in the following order to access the flash memory. (KEY1 → KEY2 → KEY3)
	KEY1	0x01234567
	KEY2	0x12345678
	KEY3	0x23456789

NOTE:

1. Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_FLSKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the CFMC_CTRL register is written as '1'.

6.10.3 CFMC_INFOKEY: Code Flash User Info Access Key Register

The CFMC_INFOKEY register is the access key register for the internal flash option area. This register is 32 bits wide.

CFMC_INFOKEY=0x4100_0008

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
INFOKEY[31:0]																															
0x00000000																															
WO																															

31 0	INFOKEY[31:0]	These bits are the key values used to access the User Info area. INFOKEY[31:0] values must be written to this register in the following order to access the flash option area. (KEY1 → KEY2 → KEY3)
	KEY1	0x3456789A
	KEY2	0x456789AB
	KEY3	0x56789ABC

NOTE:

1. User Info access control bit must be disabled before controlling User Info memory. To this end, the KEY values must be written in the CFMC_INFOKEY register in the order of KEY1 → KEY2 → KEY3. Once the User Info access control bit is disabled, the access enable status is maintained until the ILOCK bit of the CFMC_CTRL register is written as '1'.

6.10.4 CFMC_FLS0PROT: Code Flash 0 Protection Register

The CFMC_FLS0PROT register is an internal flash memory protection register. This register is 32 bits wide.

CFMC_FLS0PROT=0x4100_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
FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY32K_15	FPBY32K_14	FPBY32K_13	FPBY32K_12	FPBY32K_11	FPBY32K_10	FPBY32K_9	FPBY32K_8	FPBY32K_7	FPBY32K_6	FPBY32K_5	FPBY32K_4	FPBY32K_3	FPBY32K_2	FPBY32K_1	FPBY32K_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	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP512B_x (x = 0 to 7)	Unprotection of the 4 KB of Flash memory (0x0007_F000 ~ 0x0007_FFFF) (The settings of these bits have a higher priority level than the protection selection bits) NOTE) This bit is only meaningful if the FPBY32K_15 bit is '1'.	
		Unprotection area for 512 KB flash memory	
		0	Disable (protection)
		1	Enable (unprotection)
y+16	FUBLP4K_y	Protection for user boot loader area	
		0	Unprotection
		1	Protection
n	FPBY32K_n (n = 0 to 15)	Flash area protection selection bits for 512 KB flash memory.	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY32K_n bit or FUBLP4K_y is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 512 bytes, other erase commands (2 KB and chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY32K_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.
4. During bank swap mode, write protection follows the physical address. See section 6.4.2.4.

6.10.5 CFMC_FLS1PROT: Code Flash 1 Protection Register

The CFMC_FLS1PROT register is an internal flash memory protection register. This register is 32 bits wide.

CFMC_FLS1PROT=0x4100_0010

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
FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY32K_15	FPBY32K_14	FPBY32K_13	FPBY32K_12	FPBY32K_11	FPBY32K_10	FPBY32K_9	FPBY32K_8	FPBY32K_7	FPBY32K_6	FPBY32K_5	FPBY32K_4	FPBY32K_3	FPBY32K_2	FPBY32K_1	FPBY32K_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	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP512B_x (x = 0 to 7)	Unprotection of the 4 KB of flash memory (0x0007_F000 ~ 0x0007_FFFF) (The settings of these bits have a higher priority level than the protection selection bits) NOTE) This bit is only meaningful if the FPBY32K_15 bit is '1'.	
		Unprotection area for 512 KB flash memory	
		0	Disable (protection)
		1	Enable (unprotection)
y+16	FUBLP4K_y	Protection for user boot loader area	
		0	Unprotection
		1	Protection
n	FPBY32K_n (n = 0 to 15)	Flash area protection selection bit. (n = 0 to 15 for 512 KB flash memory)	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY32K_n bit or FUBLP4K_y is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 512 bytes, other erase commands (2 KB and chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY32K_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.
4. During bank swap mode, write protection follows the physical address. See section 6.4.2.4

6.10.6 CFMC_INFOPROT: Code Flash User Info Protection Register

The CFMC_ INFOPROT register is an internal flash memory User Info register. This register is 32 bits wide.

CFMC_FLS1PROT=0x4100_0014

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
Reserved				INFO3	INFO2	INFO1	INFO0
-				0	0	0	0
-				RW	RW	RW	RW

3	INFO3	User Info3 area protection (0x0F00_0600 ~ 0x0F00_07FF)	
		0	Disable
		1	Enable
2	INFO2	User Info2 area protection (0x0F00_0400 ~ 0x0F00_05FF)	
		0	Disable
		1	Enable
1	INFO1	User Info1 area protection (0x0F00_0200 ~ 0x0F00_03FF)	
		0	Disable
		1	Enable
0	INFO0	User Info0 area protection (0x0F00_0000 ~ 0x0F00_01FF)	
		0	Disable
		1	Enable

6.10.7 CFMC_CTRL: Code Flash Access Control Register

The CFMC_CTRL register is an internal flash memory access control register. This register is 32bits wide.

CFMC_CTRL=0x4100_0018

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
FLOCK	ILOCK	Reserved	DCRST	ICRST	Reserved		ECCSI	ECCDN	Reserved	WDIEN	WABORT	Reserved	CERS	SERS	PERS	PGM
1	1	-	0	0	-		0	0	-	0	0	-	0	0	0	0
RW	RW	-	RW	RW	-		RW	RW	-	RW	RW	-	RW	RW	RW	RW

31	FLOCK[1]	Code flash access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_FLSKEY register in the order of KEY1→KEY2→KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the CFMC_CTRL register is written as '1'.
	0	Disable (unlock status)
	1	Enable (lock status and flash memory access lock)
30	ILOCK	User Info access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_INFOKEY register in the order of KEY1→KEY2→KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the ILOCK bit of the CFMC_CTRL register is written as '1'.
	0	Disable (unlock status)
	1	Enable (lock status and flash memory access lock)
25	DCRST	Code data cache reset control bit (auto clear)
	0	No effect
	1	Reset
24	ICRST	Instruction cache reset control bit (auto clear)
	0	No effect
	1	Reset
11	ECCSI	ECC single error interrupt enable
	0	Disable
	1	Enable
10	ECCDN	ECC double error non-maskable interrupt enable
	0	Disable
	1	Enable
8	WDIEN[2]	Write done interrupt enable
	0	Disable
	1	Enable
7	WABORT	Abort write operation
	0	No effect
	1	Abort write operation.
3	CERS	Whole flash memory erase (bulk, chip)
	0	Disable
	1	Enable
2	SERS	Flash memory erase in 2 KB sector units

		0	Disable
		1	Enable
1	PERS	Flash memory erase in page units	
		0	Disable
		1	Enable
0	PGM	Flash memory write in word (4 bytes) units	
		0	Disable
		1	Enable

NOTES:

1. To disable the code flash or the User Info access control bit, the correct key values must be entered in the correct order by using the CFMC_FLSKEY or CFMC_INFOKEY register, respectively.
2. For detailed information of the write done interrupt, refer to section 6.9.

6.10.8 CFMC_STAT: Code Flash Access Status Register

The CFMC_STAT is an internal flash memory access status register. This register is 32 bits wide.

CFMC_STAT=0x4100_001C

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
Reserved		WTERR	RPERR	WSERR	IPERR	FPERR	ILERR	FLERR	Reserved	ECCSE	ECCDE	CDONE	WDONE	Reserved	CBUSY	WBUSY
-		0	0	0	0	0	0	0	-	0	0	0	0	-	0	0
-		RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RO	RO

22	WTERR	Write time out error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
21	RPERR	Read-protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
20	WSERR	Write sequence error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
19	IPERR	User Info protect error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
18	FPERR	Flash protect error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
17	ILERR	User Info lock error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
16	FLERR	Flash lock error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
11	ECCSE	Code flash ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a ‘1’)
10	ECCDE	Code flash ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a ‘1’)
9	CDONE	CRC check done check	
		0	Busy
		1	CRC check done (cleared by writing a ‘1’)
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a ‘1’)
1	CBUSY	CRC check busy check	
		0	Not busy
		1	Busy
0	WBUSY	Write busy check	
		0	Not busy
		1	Busy

6.10.9 CFMC_READPROT: Code Flash Read Protection Register

The CFMC_READPROT register is the internal flash memory's read protection register. It is a 32-bit register.

CFMC_READPROT=0x4100_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
DBGMOD	SRBOOT	Reserved	PWMATCH	INFO0ERSD	CERSD	Reserved	LVL2_STS	LVL1_STS	Reserved	LVL2_EN	LVL1_EN	RPROT[7:0]
0	0	-	0	0	0	-	0	0	-	0	0	0xFF
RO	RO	-	RO	RO	RO	-	RO	RO	-	RO	RO	RW

31	DBGMOD	Debug operating status bit
		0 Not operating
		1 Operating
30	SRBOOT	SRAM Boot mode status bit
		0 Normal mode
		1 SRAM Boot mode
26	PWMATCH	Password match flag bit
		0 Not matched
		1 Password preset value and password in value are matched.
25	INFO0ERSD	Chip erase and Info0 erase done flag bit
		0 Not occurred
		1 User Info0 erase done (RPROT erase / Write permit)
24	CERSD	Chip erase done flag bit
		0 Not occurred
		1 Chip erase done (User Info0 erase / Write permit)
17	LVL2_STS	Protection Level-2 status bit in debug mode This bit can be checked in **debug mode only**.
		0 Normal status
		1 Protection Level-2 status
16	LVL1_STS	Protection Level-1 status bit in debug mode This bit can be checked in **debug mode only**.
		0 Normal status
		1 Protection Level-1 status
9	LVL2_EN	Protection Level-2 status bit
		0 Normal status
		1 Protection Level-2 status
8	LVL1_EN	Protection Level-1 status bit
		0 Normal status
		1 Protection Level-1 status
7 0	RPROT[7:0]	Read protection control bit When the memory is in either protection mode, setting the 8th bit to '1' enables the password function for the protection mode. 0xB9: Protection 1 password mode 0x80: Protection 2 password mode
		0xFF Unprotection (UNPROT)
		0x39 Protection Level-1 (LVL1)
		others Protection Level-2 (LVL2)
NOTE: For detailed information of the Read Protection, refer to 6.4.1.		

6.10.10 CFMC_PWIN: Code Flash Password Input Register

The CFMC_PWIN register is used to enter the password value, while the microcontroller is in read protection password mode. If the password value matches the value fed to the CFMC_PWPRST register, the protected data can be read.

CFMC_PWIN=0x4100_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
PWIN[31:0]																															
0x00000000																															
WO																															

31 0	PWIN[31:0]	Password input data bit Writing is done twice[1] to PWIN[31:0] in the order of 'LSB → MSB'. Rewriting is restricted once the register is written.

NOTES:

1. Example 1) Password value is set to 0x1234_5678, 0x8765_4321

```
CFMC_PWIN = 0x12345678;
CFMC_PWIN = 0x87654321;
```

2. Example 2) Read protection password mode is set again when a user consecutively enters the same password value two times after the protection mode was released due to the password match in read protection password mode.

```
CFMC_PWIN = 0x12345678;
CFMC_PWIN = 0x87654321; // Protection mode is released due to password match.
CFMC_PWIN = 0x12345678;
CFMC_PWIN = 0x87654321; // Read protection password mode is set again.
```

3. Example 3) If a password that was written first is not correct, users must reset before writing a password again.

```
CFMC_PWIN = 0x11111111; // Wrong password value is input.
CFMC_PWIN = 0x12345678;
CFMC_PWIN = 0x87654321; // Password cannot disable the protection.
```

6.10.11 CFMC_CHKCTRL: Code Flash CRC Check Control Register

The CFMC_CHKCTRL register is an internal flash memory CRC control register. This register is 32 bits wide.

The internal CRC check function can operate while the read protection is applied.

CFMC_CHKCTRL=0x4100_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		
Reserved	CDRST	Reserved	CDIEN	Reserved	BSTEN	BGEN
-	0	-	0	-	0	0
-	RW	-	RW	-	RW	RW

16	CDRST	CRC data reset (automatically cleared)
	0	No effect
	1	Reset
8	CDIEN	CRC done interrupt
	0	Disable
	1	Enable
1	BSTEN	Burst mode
	0	Disable
	1	Enable
0	BGEN	Background mode
	0	Disable
	1	Enable

6.10.12 CFMC_CHKDOUT: Code Flash CRC Check Data Output Register

The CFMC_CHKDOUT register is an internal flash memory CRC data output register. This register is 32 bits wide.

CFMC_CHKDOUT=0x4100_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
Reserved	CDOUT[15:0]
-	0xFFFF
-	R0

15 0	CDOUT[15:0]	CRC data output value

6.10.13 CFMC_CHKSADDR: Code Flash CRC Check Start Address Register

The CFMC_CHKSADDR register is an internal flash memory CRC start address register. This register is 32 bits wide.

CFMC_CHKSADDR=0x4100_0038

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
SADDR[23:0]			FIXED_VALUE[7:0]
0x0000000			0x00
RW			RO

31 8	SADDR [23:0]	CRC start address value
6 0	FIXED_VALUE[7:0]	Fixed value

6.10.14 CFMC_CHKEADDR: Code Flash CRC Check End Address Register

The CFMC_CHKEADDR register is an internal flash memory CRC end address register. This register is 32 bits wide.

CFMC_CHKEADDR=0x4100_003C

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
EADDR[23:0]			FIXED VALUE[7:0]
0x000FFF			0xFF
RW			RO

31 8	EADDR[23:0]	CRC end address value
6 0	FIXED_VALUE[7:0]	Fixed value

6.10.15 CFMC_WTOUT: Code Flash Write Timeout Register

The CFMC_WTOUT register is an internal flash memory Write Timeout register. This register is 32 bits wide. After the Write operation (Program, Erase), if the Write operation does not end during $t_{Timeout}$, the Write operation is terminated.

CFMC_CHKDOUT=0x4100_0040

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
WTOUT[31:0]			
0x0000_0000			
RW			

31 0	WTOUT[31:0]	Write timeout count
		0 Disables.
		1 Enables. $t_{Timeout}$ = WTOUT / HCLK

6.10.16 CFMC_BANK: Code Flash Bank Control Register

The CFMC_BANK register is an internal flash memory bank control register. This register is 32 bits wide. This register must be accessed from BootROM or SRAM.

CFMC_BANK=0x4100_0050

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
BANK_WTIDKY[23:0]			Reserved	BS	Reserved	BM
0x000000			-	1	-	1
RW			-	RW	-	RW

31 8	BANK_WTIDKY	Bank register Identification key To control a bank, you must write to BS bit, BM bit and this ID key to CFMC_BANK register. Writing to a BS bit and BM without writing to BANK_WTIDKY[23:0] is prohibited.
		0x5A3C0F ID Key
		Othter No effect
1	BS	Bank swap (valid in bank mode)
		0 Swap mode
		1 No swap mode
0	BM	Bank mode
		0 Bank mode
		1 Linear mode

6.10.17 CFMC_ECCEADDR: Code Flash ECC Error Address Register

The CFMC_ECCEADDR register is an internal flash memory ECC error address register. This register is 32 bits wide.

CFMC_ECCEADDR=0x4100_0060

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
ECCEA[31:0]			
0x0000_0000			
R0			

31 0	ECCEA[31:0]	ECC error address

6.10.18 CFMC_ECCEDATA0: Code Flash ECC Error Data0 Address Register

The CFMC_ECCEDATA0 register is an internal flash memory ECC data0 address register. This register is 32 bits wide.

CFMC_ECCEDATA0=0x4100_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
ECCED0[31:0]			
0x0000_0000			
R0			

31 0	ECCED0[31:0]	ECC error data0 of 128 bits [31:0] ECC error data0 address: ECC error address + 0x0

6.10.19 CFMC_ECCEDATA1: Code Flash ECC Error Data1 Address Register

The CFMC_ECCEDATA1 register is an internal flash memory ECC data1 address register. This register is 32 bits wide.

CFMC_ECCEDATA1=0x4100_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
ECCED1[31:0]			
0x0000_0000			
R0			

31 0	ECCED1[31:0]	ECC error data1 of 128 bits [63:32] ECC error data1 address: ECC error address + 0x4

6.10.20 CFMC_ECCEDATA2: Code Flash ECC Error Data2 Address Register

The CFMC_ECCEDATA2 register is an internal flash memory ECC data2 address register. This register is 32 bits wide.

CFMC_ECCEDATA2=0x4100_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
ECCED2[31:0]			
0x0000_0000			
R0			

31 0	ECCED2[31:0]	ECC error data2 of 128 bits [95:64] ECC error data2 address: ECC error address + 0x8

6.10.21 CFMC_ECCEDATA3: Code Flash ECC Error Data3 Address Register

The CFMC_ECCEDATA3 register is an internal flash memory ECC data3 address register. This register is 32 bits wide.

CFMC_ECCEDATA3=0x4100_0070

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
ECCED3[31:0]			
0x0000_0000			
R0			

31 0	ECCED3[31:0]	ECC error data3 of 128 bits [127:96] ECC error data3 address: ECC error address + 0xC

6.10.22 CFMC_ECCEPARITY: Code Flash ECC Error Parity Register

The CFMC_ECCEPARITY register is an internal flash memory ECC parity register. This register is 32 bits wide.

CFMC_ECCEPARTIY=0x4100_0074

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	
Reserved	ECCEP3[6:0]	Reserved	ECCEP2[6:0]	Reserved	ECCEP1[6:0]	Reserved	ECCEP0[6:0]
-	0x00	-	0x00	-	0x00	-	0x00
-	R0	-	R0	-	R0	-	R0

31 24	ECCEP3[6:0]	ECC error parity 3 of 128 bits
22 16	ECCEP2[6:0]	ECC error parity 2 of 128 bits
14 8	ECCEP1[6:0]	ECC error parity 1 of 128 bits
6 0	ECCEP0[6:0]	ECC error parity 0 of 128 bits

6.10.23 CFMC_PWPRST: Code Flash Password Preset Register

The CFMC_PWPRST register is used to preset the password value. This register is 32 bits wide.

CFMC_PWPRST=0x4100_0F30

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
PWPRST[31:0]			
0x00000000			
WO			

31 0	PWPRST[31:0]	Password preset data bit Writing is done twice to the register in the order of 'LSB → MSB'. Rewriting is restricted once the register is written. For example, presetting 0x1234_5678 as a password. CFMC_PWPRST = 0x1234_5678 CFMC_PWPRST = 0x1234_5678

6.10.24 CFMC Register Map Summary

Table 63. CFMC Register Map Summary

Offset	Register name	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
0x00	CFMC_CONF	ECCDIS	Res	Res	Res	Res	Res	WTEN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DCEN	ICEN	Res	Res	Res	Res	LATENCY[3:0]			
	Reset value	0						0																0	0					0	0	0	0
0x04	CFMC_FLSKEY	FKEY[31:0]																															
	Reset value	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
0x08	FMC_OTPKEY	INFOKEY[31:0]																															
	Reset value	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
0x0C	CFMC_FLS0PROT	FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY32K_15	FPBY32K_14	FPBY32K_13	FPBY32K_12	FPBY32K_11	FPBY32K_10	FPBY32K_9	FPBY32K_8	FPBY32K_7	FPBY32K_6	FPBY32K_5	FPBY32K_4	FPBY32K_3	FPBY32K_2	FPBY32K_1	FPBY32K_0
	Reset value	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	CFMC_FLS1PROT	FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY32K_15	FPBY32K_14	FPBY32K_13	FPBY32K_12	FPBY32K_11	FPBY32K_10	FPBY32K_9	FPBY32K_8	FPBY32K_7	FPBY32K_6	FPBY32K_5	FPBY32K_4	FPBY32K_3	FPBY32K_2	FPBY32K_1	FPBY32K_0
	Reset value	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	CFMC_INFOPROT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	INFO3	INFO2	INFO1	INFO0
	Reset value																													0	0	0	0
0x18	CFMC_CTRL	FLOCK	ILOCK	Res	Res	Res	Res	DCRST	ICRST	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ECCSI	ECCDN	Res	WDIEN	WABORT	Res	Res	Res	CERS	SERS	PERS	PGM
	Reset value	1	1					0	0													0	0		0	0				0	0	0	0
0x1C	CFMC_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	WTERR	RPERR	WSERR	IPERR	FPERR	ILERR	FLERR	Res	Res	Res	Res	ECCSE	ECCDE	CDONE	WDONE	Res	Res	Res	Res	Res	Res	CBUSY	WBUSY
	Reset value										0	0	0	0	0	0	0							0	0							0	0
0x20	CFMC_READPROT	DBGMOD	SRBOOT	Res	Res	Res	PWMATCH	INFO0ERSD	CERSD	Res	Res	Res	Res	Res	Res	LVL2_STS	LVL1_STS	Res	Res	Res	Res	Res	Res	LVL2_EN	LVL1_EN	RPROT[7:0]							
	Reset value	0	0				0	0	0							0	0							0	0	1	1	1	1	1	1	1	1
0x24	CFMC_PWIN	PWIN[31:0]																															
	Reset value	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
0x30	CFMC_CHKCTRL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDRST	Res	Res	Res	Res	Res	Res	Res	CDIEN	Res	Res	Res	Res	Res	Res	BSTEN	BGEN
	Reset value																0								0							0	0
0x34	CFMC_CHKDOUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDOUT[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 63. CFMC Register Map Summary (continued)

Offset	Register name	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
0x38	CFMC_CHKSADDR	SADDR[23:0]																								FIXED_VALUE[7:0]							
	Reset value	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
0x3C	CFMC_CHKEADDR	EADDR[23:0]																								FIXED_VALUE[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x40	CFMC_WTOUT	WTOUT[31:0]																															
	Reset value	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
0x50	CFMC_BANK	BANK_WTIDKY[23:0]																								Res.	Res.	Res.	BS	Res.	Res.	Res.	BM
	Reset value	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
0x60	CFMC_ECCEADDR	ECCEA[31:0]																															
	Reset value	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
0x64	CFMC_ECCEDATA0	ECCED0[31:0]																															
	Reset value	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
0x68	CFMC_ECCEDATA1	ECCED1[31:0]																															
	Reset value	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
0x6C	CFMC_ECCEDATA2	ECCED2[31:0]																															
	Reset value	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
0x70	CFMC_ECCEDATA3	ECCED3[31:0]																															
	Reset value	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
0x74	CFMC_ECCEPARITY	Res.	ECCEP3[6:0]							Res.	ECCEP2[6:0]							Res.	ECCEP1[6:0]							Res.	ECCEP0[6:0]						
	Reset value		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
0xF30	CFMC_PWPRST	PWPRST[31:0]																															
	Reset value	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

6.11 Data Flash Registers

The base address and register map of the Data Flash Memory Controller (DFMC) are described in the following tables.

Table 64. Base Address of DFMC

Name	Base Address
DFMC	0x4100_1000

Table 65. DFMC Register Map

Name	Offset	Type	Description	Reset Value	Ref.
DFMC_CONF	0x0000	RW	Data Flash Control Register	0x0000_0000	6.11.1
DFMC_FLSKEY	0x0004	WO	Data Flash Access Key Register	0x0000_0000	6.11.2
DFMC_FLSPROT	0x000C	RW	Data Flash Protection Register	0x0000_0000	6.11.3
DFMC_CTRL	0x0014	RW	Data Flash Access Control Data	0x8000_0000	6.11.4
DFMC_STAT	0x0018	RW	Data Flash Access Status Register	0x0000_0000	6.11.5
DFMC_CHKCTRL	0x0030	RW	Data Flash CRC Check Control Register	0x0000_0000	6.11.6
DFMC_CHKDOUT	0x0034	RW	Data Flash CRC Check Data Output Register	0x0000_FFFF	6.11.7
DFMC_CHKSADDR	0x0038	RW	Data Flash CRC Check Start Address Register	0x0000_0000	6.11.8
DFMC_CHKEADDR	0x003C	RW	Data Flash CRC Check End Address Register	0x0000_7FFF	6.11.9
DFMC_WTOUT	0x0040	RW	Data Flash Write Timeout Register	0x0000_0000	6.11.10
DFMC_ECCEADDR	0x0050	RO	Data Flash ECC Error Address Register	0x0000_0000	6.11.11
DFMC_ECCEDATA	0x0054	RO	Data Flash ECC Error Data Register	0x0000_0000	6.11.12
DFMC_ECCEPARITY	0x0058	RO	Data Flash ECC Error Parity Register	0x0000_0000	6.11.13

6.11.1 DFMC_CONF: Data Flash Control Register

The DFMC_CONF register is a control register for the internal data flash memory. This register is 32 bits wide.

DFMC_CONF=0x4100_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
ECCDIS	Reserved							WTEN	Reserved																			LATENCY[3:0]			
0	-							0	-																			0000			
RW	-							RW	-																			RW			

31	ECCDIS	Disable Data FLASH ECC	
		0	ECC enable
		1	ECC disable
25	WTEN	Enable the data flash memory write operation	
		0	Disable
		1	Enable
3 0	LATENCY[3:0](2)	Flash memory wait value bits	
		The flash memory wait value between 0 and 15.	

NOTE:

1. For detailed information of the LATENCY[3:0], refer to section 6.3.2.

6.11.2 DFMC_FLSKEY: Data Flash Access Key Register

The DFMC_FLSKEY register is the access key register for the internal flash memory. This register is 32 bits wide.

DFMC_FLSKEY=0x4100_1004

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
FKEY[31:0]																															
0x00000000																															
WO																															

31 0	FKEY[31:0](1)	These bits are the key values used to access the flash memory. FKEY[31:0] values must be written to this register in the following order to access the flash memory. (KEY1 → KEY2 → KEY3)	
		KEY1	0x01234567
		KEY2	0x12345678
		KEY3	0x23456789

NOTE:

1. Flash lock must be disabled before controlling flash memory. To this end, the KEY values must be written in the DFMC_FLSKEY register in the order of KEY1 → KEY2 → KEY3. Once the flash lock is disabled, the unlock status is maintained until the FLOCK bit of the DFMC_CTRL register is written as '1'.

6.11.3 DFMC_FLSPROT: Data Flash Protection Register

The DFMC_FLSPROT register is an internal flash memory protection register. This register is 32 bits wide.

DFMC_FLSPROT=0x4100_100C

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
FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	Reserved	FPBY2K_15	FPBY2K_14	FPBY2K_13	FPBY2K_12	FPBY2K_11	FPBY2K_10	FPBY2K_9	FPBY2K_8	FPBY2K_7	FPBY2K_6	FPBY2K_5	FPBY2K_4	FPBY2K_3	FPBY2K_2	FPBY2K_1	FPBY2K_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	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP512B_x (x = 0 to 7)	Unprotection of the 4 KB of flash memory (0x0E00_7000 ~ 0x0E00_7FFF) (The settings of these bits have a higher priority level than the protection selection bits) NOTE: This bit is only meaningful if the FPBY2K_15 bit is '1'.
		Unprotection area for 512 KB flash memory x = 0: 0x0E00_7000 to 0x0E00_71FF x = 1: 0x0E00_7200 to 0x0E00_73FF x = 2: 0x0E00_7400 to 0x0E00_75FF x = 3: 0x0E00_7600 to 0x0E00_77FF x = 4: 0x0E00_7800 to 0x0E00_79FF x = 5: 0x0E00_7A00 to 0x0E00_7BFF x = 6: 0x0E00_7C00 to 0x0E00_7DFF x = 7: 0x0E00_7E00 to 0x0E00_7FFF
		0 Protection.
		1 Unprotection.
n	FPBY2K_n (n = 0 to 15)	Flash area protection selection bit n = 0 to 15 values for 32 KB flash memory
		n = 0: 0x0E00_0000 to 0x0E00_07FF n = 1: 0x0E00_0800 to 0x0E00_0FFF n = 2: 0x0E00_1000 to 0x0E00_17FF n = 3: 0x0E00_1800 to 0x0E00_1FFF n = 4: 0x0E00_2000 to 0x0E00_27FF n = 5: 0x0E00_2800 to 0x0E00_2FFF n = 6: 0x0E00_3000 to 0x0E00_37FF n = 7: 0x0E00_3800 to 0x0E00_3FFF n = 8: 0x0E00_4000 to 0x0E00_47FF n = 9: 0x0E00_4800 to 0x0E00_4FFF n = 10: 0x0E00_5000 to 0x0E00_57FF n = 11: 0x0E00_5800 to 0x0E00_5FFF n = 12: 0x0E00_6000 to 0x0E00_67FF n = 13: 0x0E00_6800 to 0x0E00_6FFF n = 14: 0x0E00_7000 to 0x0E00_77FF n = 15: 0x0E00_7800 to 0x0E00_7FFF
		0 Unprotection
		1 Protection

NOTES:

1. When any of the FPBY2K_n bit is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 512 bytes, other erase commands (2 KB and full chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY2K_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.

6.11.4 DFMC_CTRL: Data Flash Access Control Register

The DFMC_CTRL register is an internal flash memory access control register. This register is 32bits wide.

DFMC_CTRL=0x4100_1014

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
FLOCK	Reserved																			ECCSI	ECCDN	Reserved	WDIEN	WABORT	Reserved					CERS	SERS	PERS	PGM
1	-																			0	0	-	0	0	-					0	0	0	0
RW	-																			RW	RW	-	RW	RW	-					RW	RW	RW	RW

31	FLOCK[1]	Data flash memory access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the DFMC_FLSKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the DFMC_CTRL register is written as '1'.
		0 Disable Access
		1 Enable Access
11	ECCSI	ECC single error interrupt enable
		0 Disable
		1 Enable
10	ECCDN	ECC double error non-maskable interrupt enable
		0 Disable
		1 Enable
8	WDIEN[2]	Write done interrupt enable
		0 Disable
		1 Enable
7	WABORT	Abort write operation
		0 No effect
		1 Abort write operation
3	CERS	Whole flash memory erase
		0 Disable
		1 Enable
2	SERS	Flash memory erase in 2KB sector units
		0 Disable
		1 Enable
1	PERS	Flash memory erase in page units
		0 Disable
		1 Enable
0	PGM	Flash memory write in word (4 bytes) units
		0 Disable
		1 Enable

NOTES:

1. To disable the data flash memory access, the correct key values must be entered in the correct order by using the DFMC_FLSKEY register, respectively.
2. For detailed information of the write done interrupt, refer to section 6.9.

6.11.5 DFMC_STAT: Data Flash Access Status Register

The DFMC_STAT is an internal flash memory access status register. This register is 32 bits wide.

DFMC_STAT=0x4100_1018

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
Reserved		WTERR	RPERR	WSERR	Reserved	FPERR	Reserved	FLERR	Reserved	ECCSE	ECCDE	CDONE	WDONE	Reserved	CBUSY	WBUSY
-		0	0	0	-	0	0	0	-	0	0	0	0	-	0	0
-		RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RO	RO

22	WTERR	Write time out error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
21	RPERR	Read-protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
20	WSERR	Write sequence error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
18	FPERR	Flash protect error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
16	FLERR	Flash lock error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
11	ECCSE	Code flash ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a ‘1’)
10	ECCDE	Code flash ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a ‘1’)
9	CDONE	CRC check done check	
		0	Busy
		1	CRC check done (cleared by writing a ‘1’)
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a ‘1’)
1	CBUSY	CRC check busy check	
		0	Not busy
		1	Busy
0	WBUSY	Write busy check	
		0	Not busy
		1	Busy

6.11.6 DFMC_CHKCTRL: Code Flash CRC Check Control Register

The DFMC_CHKCTRL register is an internal flash memory CRC control register. This register is 32 bits wide.

The internal CRC check function can operate while the read protection is applied.

DFMC_CHKCTRL=0x4100_1030

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							
Reserved							CDRST	Reserved		CDIEN	Reserved			BSTEN	BGEN
-							0	-		0	-			0	0
-							RW	-		RW	-			RW	RW

16	CDRST	CRC data reset (automatically cleared) 16-bit specific polynomial (internal use only)
		0 No effect
		1 Reset
8	CDIEN	CRC done interrupt
		0 Disable
		1 Enable
1	BSTEN	Burst mode
		0 Disable
		1 Enable
0	BGEN	Background mode
		0 Disable
		1 Enable

6.11.7 DFMC_CHKDOUT: Data Flash CRC Check Data Output Register

The DFMC_CHKDOUT register is an internal flash memory CRC data output register. This register is 32 bits wide.

DFMC_CHKDOUT=0x4100_1034

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
Reserved		CDOUT[15:0]	
-		0xFFFF	
-		R0	

15 0	CDOUT[15:0]	CRC data output value

6.11.8 DFMC_CHKSADDR: Data Flash CRC Check Start Address Register

The DFMC_CHKSADDR register is an internal flash memory CRC start address register. This register is 32 bits wide.

DFMC_CHKSADDR=0x4100_1038

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
SADDR[25:0]			FIXED_VALUE[5:0]
0x0000000			0x00
RW			RO

31 6	SADDR[25:0]	CRC start address value (physical address) Address range: 0x000 ~ 0x1FF
5 0	FIXED_VALUE[5:0]	Fixed value

6.11.9 DFMC_CHKEADDR: Data Flash CRC Check End Address Register

The DFMC_CHKEADDR register is an internal flash memory CRC end address register. This register is 32 bits wide.

DFMC_CHKEADDR=0x4100_103C

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
EADDR[25:0]			FIXED_VALUE[5:0]
0x00001FF			0x3F
RW			RO

31 6	EADDR[25:0]	CRC end address value (physical address) Address range: 0x000 ~ 0x1FF
5 0	FIXED_VALUE[5:0]	Fixed value

6.11.10 DFMC_WTOUT: Data Flash Write Timeout Register

The DFMC_WTOUT register is an internal flash memory Write Timeout register. This register is 32 bits wide. After the Write operation (Program, Erase), if the Write operation does not end during $t_{Timeout}$, the write operation is terminated.

DFMC_WTOUT=0x4100_1040

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
WTOUT[31:0]			
0x0000_0000			
RW			

31 0	WTOUT[31:0]	Write timeout count
		0 Disable
		1 Enable $t_{Timeout}$ = WTOUT / HCLK

6.11.11 DFMC_ECCEADDR: Data Flash ECC Error Address Register

The DFMC_ECCEADDR register is an internal flash memory ECC error address register. This register is 32 bits wide.

DFMC_ECCEADDR=0x4100_1050

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
Reserved		ECCEA[15:0]	
-		0x0000	
-		R0	

31 0	ECCEA[15:0]	ECC error address (Offset: 0x0E00_0000)

6.11.12 DFMC_ECCEDATA: Data Flash ECC Error Address Register

The DFMC_ECCEDATA register is an internal flash memory ECC data address register. This register is 32 bits wide.

DFMC_ECCEDATA=0x4100_1054

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
Reserved		ECCED[15:0]	
-		0x0000	
-		R0	

31 0	ECCED[15:0]	ECC error data (Offset: 0x0E00_0000)

6.11.13 DFMC_ECCEPARITY: Data Flash ECC Error Parity Register

The DFMC_ECCEPARITY register is an internal flash memory ECC parity register. This register is 32 bits wide.

DFMC_ECCEPARTIY=0x4100_1058

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
Reserved			ECCEP[6:0]
-			0x00
-			R0

6 0	ECCEP[6:0]	ECC error parity

6.11.14 DFMC Register Map Summary

Table 66. DFMC Register Map Summary

Offset	Register name	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
0x00	DFMC_CONF	Res	Res	Res	Res	Res	Res	WTEN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	LATENCY[3:0]			
	Reset value							0																						0	0	0	0
0x04	DFMC_FLSKEY	FKEY[31:0]																															
	Reset value	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
0x0C	DFMC_FLSPROT	FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	Res	Res	Res	Res	Res	Res	Res	Res	FPBY2K_15	FPBY2K_14	FPBY2K_13	FPBY2K_12	FPBY2K_11	FPBY2K_10	FPBY2K_9	FPBY2K_8	FPBY2K_7	FPBY2K_6	FPBY2K_5	FPBY2K_4	FPBY2K_3	FPBY2K_2	FPBY2K_1	FPBY2K_0
	Reset value	0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	DFMC_CTRL	FLOCK	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ECCSI	ECCDN	Res	WDIEN	WABORT	Res	Res	Res	CERS	SERS	PERS	PGM
	Reset value	1																				0	0		0	0				0	0	0	0
0x18	DFMC_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	WTERR	RPERR	WSERR	Res	FPERR	Res	FLERR	Res	Res	Res	Res	ECCSE	ECCDE	CDONE	WDONE	Res	Res	Res	Res	Res	Res	CBUSY	WBUSY
	Reset value										0	0	0		0		0					0	0	0	0						0	0
0x30	DFMC_CHKCTRL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDRST	Res	Res	Res	Res	Res	Res	Res	CDIEN	Res	Res	Res	Res	Res	Res	BSTEN	BGEN
	Reset value																0								0							0	0
0x34	DFMC_CHKDOUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDOUT[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x38	DFMC_CHKSADDR	SADDR[25:0]																												FIXED_VALUE[5:0]			
	Reset value	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
0x3C	DFMC_CHKEADDR	EADDR[25:0]																												FIXED_VALUE[5:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 66. DFMC Register Map Summary

Offset	Register name	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
0x40	DFMC_WTOUT	WTOUT[31:0]																															
	Reset value	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
0x50	DFMC_ECCEADDR	ECCEA[31:0]																															
	Reset value	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
0x54	DFMC_ECCEDATA	ECCED[31:0]																															
	Reset value	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
0x58	DFMC_ECCEPARITY	ECCEP[31:0]																															
	Reset value	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

7. Direct Memory Access Controller (DMA)

7.1 DMA Introduction

The direct memory access (DMA) controller is used for high-speed data transfers between peripherals and memories. DMA enables quick data transfers having memory to memory copying or moving of data within memory.

- 16 channels
- Only single-ended signaling supported
- 8- / 16- / 32-bit data transfers supported
- Various buffers with the same size supported
- DMA transfers are triggered through peripheral interrupts

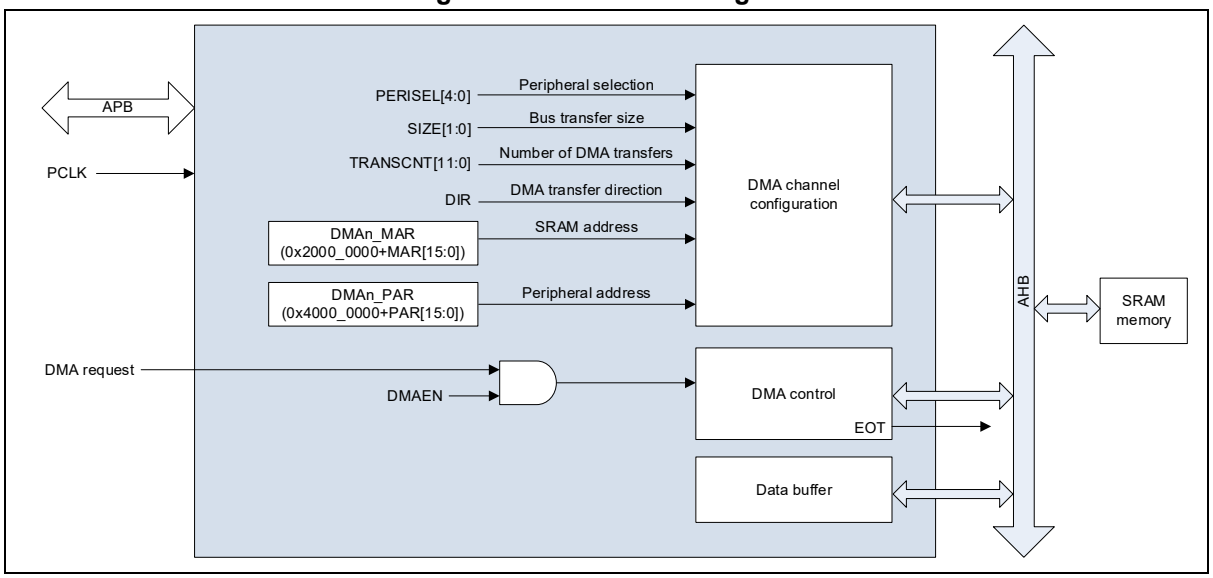
7.1.1 Modules supported by DMA

A34M420 DMA supports the following 4 modules. Refer to this link for details.

- UART
  - UART functional description
    - Character transmission procedure / character reception
  - Continuous communication using DMA and UART
    - Transmission and reception using DMA
  - UART interrupts
- SPI
  - Configuration of SPI
  - Procedure for enabling SPI
  - Data transmission and reception procedures
    - DMA handshake
  - SPI status flags
    - DMA transmit operation complete flag (DMA to SPI, TXDMAF).
    - DMA receive operation complete flag (SPI to DMA, RXDMAF).
- ADC
  - A/D conversion using the DMA
  - ADC overrun (OVR) in DMA mode
  - ADC interrupts
- CRC
  - CRC using DMA

Figure 57 shows a DMA block diagram.

Figure 57. DMA Block Diagram



7.2 DMA Functional Description

The DMA controller directly transfers data to memories by sharing the AHB with the microcontroller core. The AHB shares two masters operating in a round-robin fashion. Therefore, the DMA controller shares only half the system bandwidth with the microcontroller core.

The DMA controller is triggered only by requests made by peripherals. Once a peripheral request a transfer to the DMA controller, the connected channel becomes enabled. Then, the bus is accessed to transfer the requested data from the memory buffer to the peripheral's data buffer (Or the other way around).

The DMA controller operates in the following sequence:

1. The programmer sets the address of the peripheral to use in the DMAn_PAR register and the memory address in the DMAn_MAR register.
2. In the DMAn_CR register, the programmer sets the DMA transfer count (0 to 4095), transferring direction, and bus transfer size (8-bit, 16-bit and 32-bit).
3. In the DMAn_SR register, the programmer sets the DMAEN to '1' to enable the DMA channel.
4. A DMA request is made by the peripheral set in the PERISEL[4:0] bits of the DMAn_CR register and this triggers the requested peripheral channel to become active.
5. Data received from the source DMA address is read and stored in the internal buffer. The DMA operation writes this data to the target address.
6. Each time data is written to the target address, the DMA transfer count decreases by '1'. When the DMA transfer count reaches zero, the EOT bit of the DMAn_SR register is set to '1'. This signals an interrupt to the connected peripheral(1).

NOTE:

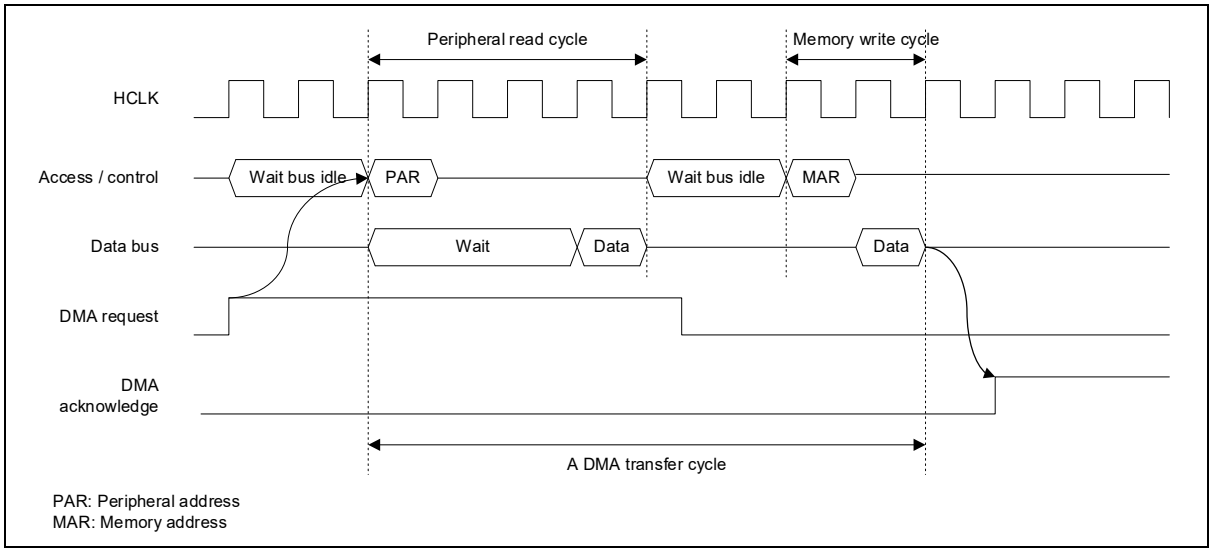
1. DMA itself has no interrupt sources; instead, each peripheral has flag bits that display the status of the connected DMA channel's transfer interrupts.

7.2.1 Peripheral-to-Memory DMA Transfer Timing

The diagram below shows the functional timing of the DMA controller. A transfer request from a peripheral is internally held off while the AHB calls the address of the data read transfer source.

The data read from the source address is stored in the internal buffer. Then, once the AHB becomes available, the stored data is transferred to the target address. As shown in the diagram below, a waiting time of four clock cycles is required until the peripheral is accessed. If the bus is occupied by another master, additional wait cycles will be taken.

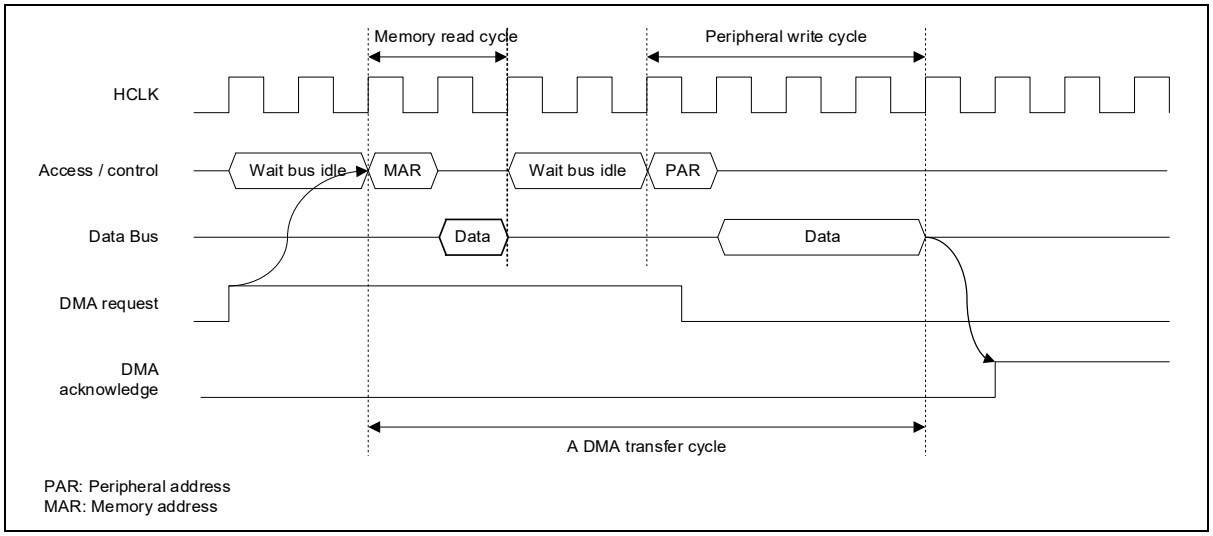
Figure 58. Peripheral-to-Memory DMA Transfer Timing



7.2.2 Memory-to-Peripheral DMA Transfer Timing

The diagram below depicts the timing of DMA transfer from memory to peripheral. A waiting time of four clock cycles is required until the peripheral is accessed. If the bus is occupied by another master, additional wait cycles will be taken.

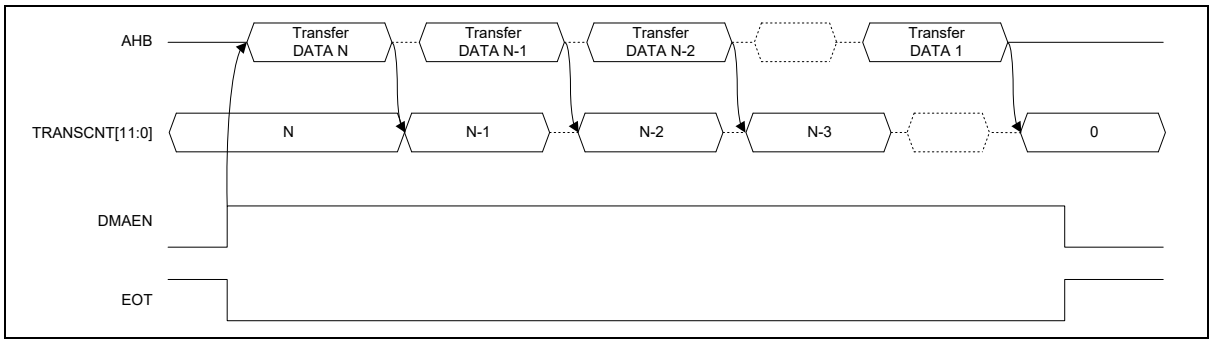
Figure 59. Memory-to-Peripheral DMA Transfer Timing



7.2.3 DMA Transfer

The diagram below depicts N number of DMA transfers in the queue as an example. DMA transferring starts when the DMAEN bit in the DMAn_SR register is set to ‘1’. And the bit is cleared to ‘0’ when all transfers have been completed.

Figure 60. N Number of DMA Transfers



7.3 DMA Registers

The base addresses of DMA is described in the following tables.

Table 67. Base Address of DMA

Name	Base Address
DMA0	0x4000_0400
DMA1	0x4000_0410
DMA2	0x4000_0420
DMA3	0x4000_0430
DMA4	0x4000_0440
DMA5	0x4000_0450
DMA6	0x4000_0460
DMA7	0x4000_0470
DMA8	0x4000_0480
DMA9	0x4000_0490
DMA10	0x4000_04A0
DMA11	0x4000_04B0
DMA12	0x4000_04C0
DMA13	0x4000_04D0
DMA14	0x4000_04E0
DMA15	0x4000_04F0

Table 68. DMA Register Map

Name	Offset	Type	Description	Reset Value	Reference
DMAn_CR	0x0000	RW	DMA Channel n Control Register	0x0000_0000	7.3.1
DMAn_SR	0x0004	RW	DMA Channel n Status Register	0x0000_0080	7.3.2
DMAn_PAR	0x0008	RW	DMA Channel n Peripheral Address Register	0x4000_0000	7.3.3
DMAn_MAR	0x000C	RW	DMA Channel n Memory Address Register	0x2000_0000	7.3.4

NOTE:

1. n = 0 to 15.

7.3.1 DMAn_CR: DMA Channel n Control Register

The DMAn_CR registers are DMA operation control registers. Each register is 32 bits wide.

DMA0_CR=0x4000_0400, DMA1_CR=0x4000_0410, DMA2_CR=0x4000_0420, DMA3_CR=0x4000_0430, DMA4_CR=0x4000_0440, DMA5_CR=0x4000_0450, DMA6_CR=0x4000_0460, DMA7_CR=0x4000_0470, DMA8_CR=0x4000_0480, DMA9_CR=0x4000_0490, DMA10_CR=0x4000_04A0, DMA11_CR=0x4000_04B0, DMA12_CR=0x4000_04C0, DMA13_CR=0x4000_04D0, DMA14_CR=0x4000_04E0, DMA15_CR=0x4000_04F0

1 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
Reserved	TRANSCNT[11:0]	Reserved	PERISEL[4:0]	Reserved	SIZE[1:0]	DIR	Reserved
-	0x000	-	0x0	-	00	0	-
-	RW	-	RW	-	RW	RW	-

27 16	TRANSCNT[11:0]	Number of DMA transfers Before enabling DMA transfer, the number of transfers must be written to TRANSCNT[11:0].
	0	All DMA transfers have been completed.
	N	There are N transfers remaining.
11 8	PERISEL[4:0](1)	Peripheral selection The selected peripheral is connected to the DMA channel. PERISEL[4:0] must be written with the number representing the peripheral to be connected to the DMA interface.
	N	Selects the peripheral to use. Refer to Table 69.
3 2	SIZE[1:0]	Bus transfer size
	00	Sets the DMA transfer size to 1 byte.
	01	Sets the DMA transfer size to half-word (2 bytes) size.
	10	Sets the DMA transfer size to one-word (4 bytes) size.
	11	Holds off the size setting.
1	DIR	Transfer direction selection
	0	Memory-to-peripheral (TX)
	1	Peripheral-to-memory (RX)

NOTE:

1. A DMA channel will be connected with selected peripheral. Below table shows peripheral selection numbers. This PERISEL[4:0] field should be set with proper number of peripherals which will be connected with DMA interface.

Table 69. DMA PERISEL[4:0] Selection

PERISEL[4:0]	Associate Peripheral	PERISEL[4:0]	Associate Peripheral
0	CHANNEL IDLE	13	SPI0 RX
1	UART0 RX	14	SPI0 TX
2	UART0 TX	15	SPI1 RX
3	UART1 RX	16	SPI1 TX
4	UART1 TX	17	SPI2 RX
5	UART2 RX	18	SPI2 TX
6	UART2 TX	19	ADC0 RX
7	UART3 RX	20	ADC1 RX
8	UART3 TX	21	ADC2 RX
9	UART4 RX	22	Reserved
10	UART4 TX	23	Reserved
11	UART5 RX	24	CRC Tx
12	UART5 TX	-	-

The PERISEL[4:0] bit field cannot have the same value in different channels. If the same PERISEL[4:0] value is written in more than one channel, proper operation cannot be guaranteed. Unused channel should have CHANNEL IDLE value in the PERISEL[4:0] bit positions.

7.3.2 DMAn_SR: DMA Channel n Status Register

The DMAn_SR is a 32-bit register. It shows the status of the DMA controller and whether the DMA channel is enabled or disabled.

DMA0_SR=0x4000_0404, DMA1_SR=0x4000_0414, DMA2_SR=0x4000_0424, DMA3_SR=0x4000_0434, DMA4_SR=0x4000_0444, DMA5_SR=0x4000_0454, DMA6_SR=0x4000_0464, DMA7_SR=0x4000_0474, DMA8_SR=0x4000_0484, DMA9_SR=0x4000_0494, DMA10_SR=0x4000_04A4, DMA11_SR=0x4000_04B4, DMA12_SR=0x4000_04C4, DMA13_SR=0x4000_04D4, DMA14_SR=0x4000_04E4, DMA15_SR=0x4000_04F4

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
Reserved			EOT	Reserved	DMAEN
-			1	-	0
-			R0	-	RW

7	EOT	End-of-Transfer flag.	
		0	There are remaining transfers. TRANSCNT[11:0] has a value other than zero.
		1	All data has been transferred. The TRANSCNT[11:0] value is zero.
0	DMAEN(1)	DMA enable	
		0	Disables DMA or fix the status.
		1	Enables DMA or available to use.

7.3.3 DMAn_PAR: DMA Channel n Peripheral Address Register

The DMAn_PAR shows the address value of the connected peripheral.

DMA0_PAR=0x4000_0408, DMA1_PAR=0x4000_0418, DMA2_PAR=0x4000_0428, DMA3_PAR=0x4000_0438, DMA4_PAR=0x4000_0448, DMA5_PAR=0x4000_0458, DMA6_PAR=0x4000_0468, DMA7_PAR=0x4000_0478, DMA8_PAR=0x4000_0488, DMA9_PAR=0x4000_0498, DMA10_PAR=0x4000_04A8, DMA11_PAR=0x4000_04B8, DMA12_PAR=0x4000_04C8, DMA13_PAR=0x4000_04D8, DMA14_PAR=0x4000_04E8, DMA15_PAR=0x4000_04F8

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 address offset(1)		PAR[15:0]	
0x4000		0x0000	
RW		RW	

15 0	PAR[15:0]	Address of the transfer / receive buffer's target peripheral The address does not change until the ongoing transfer is completed.

NOTE:

1. Peripheral base address in DMAn_PAR register must be set to '0x4000' or '0x4100'.

7.3.4 DMAn_MAR: DMA Channel n Memory Address Register

The DMAn_MAR registers show the DMA transfer's target memory address.

DMA0_MAR=0x4000_040C, DMA1_MAR=0x4000_041C, DMA2_MAR=0x4000_042C, DMA3_MAR=0x4000_043C, DMA4_MAR=0x4000_044C, DMA5_MAR=0x4000_045C, DMA6_MAR=0x4000_046C, DMA7_MAR=0x4000_047C, DMA8_MAR=0x4000_048C, DMA9_MAR=0x4000_049C, DMA10_MAR=0x4000_04AC, DMA11_MAR=0x4000_04BC, DMA12_MAR=0x4000_04CC, DMA13_MAR=0x4000_04DC, DMA14_MAR=0x4000_04EC, DMA15_MAR=0x4000_04FC

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
Memory base address offset(1)		MAR[15:0]	
0x2000		0x0000	
RW		RW	

15 0	MAR[15:0]	Target memory address of the data transfer On the completion of each transfer, the address automatically increases, depending on the setting of the SIZE[1:0] bits.

NOTE:

1. Memory Base Address in DMAn_MAR register must be set to '0x2000'.

7.3.5 DMA Register Map Summary

Table 70. DMA Register Map Summary

Offset	Register name	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
0x00	DMAn_CR	Res.	Res.	Res.	Res.	TRANSCNT[11:0]												Res.	Res.	Res.	PERISEL[4:0]					Res.	Res.	Res.	Res.	SIZE[1:0]		DIR	Res.
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	0					0	0	0	
0x04	DMAn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EOT	Res.	Res.	Res.	Res.	Res.	Res.	DMAEN
	Reset value																									1							0
0x08	DMAn_PAR	Peripheral Base Address																PAR[15:0]															
	Reset value	0	1	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
0x0C	DMAn_MAR	Memory Base Address																MAR[15:0]															
	Reset value	0	0	1	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

8. WatchDog Timer (WDT)

8.1 WDT Introduction

The WatchDog Timer (WDT) is used to detect microcontroller errors due to external interference or unexpected logical condition. These errors cause the application program to give up the normal sequence. If the microcontroller is out of control, the WDT resets the microcontroller and returns it to normal.

The WDT of the A34M420 is a 32-bit down counter. If the WDT is set as a reset source, the microcontroller restarts when the down counter reaches zero.

The WDT can be used as a cycle timer along with an interrupt.

8.2 WDT Main Features

The WDT module has key features as listed below:

- 32-bit down counter
- WDT underflow reset functionality
- Cycle timer and underflow interrupt functionalities
- WDT input clock sources selectable
  - PCLK
  - The clock source can be selected by configuring the WDTCSEL[2:0] bits in the SCU_MCCR1 register : LSI, LSE, MCLK, HSI, HSE, and PLL.
- Eight levels prescaler for the WDT input clock
- In debug mode, the WDT counter can be set not to operate.

8.3 WDT Functional Description

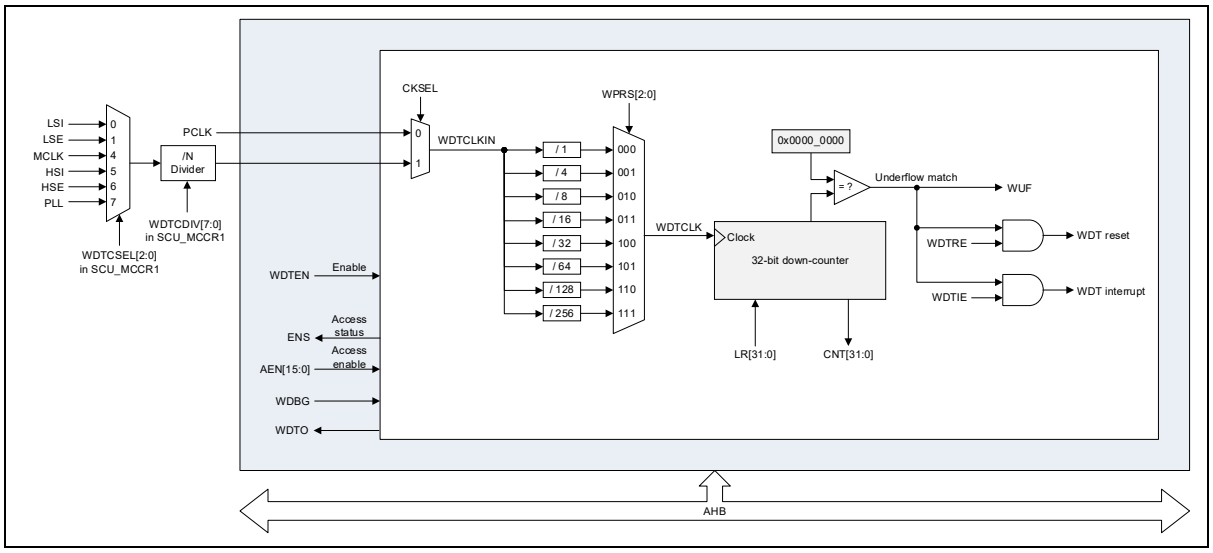
Once the WDT module is enabled and the WDTEN bit in the WDT_CON register is set to '1', the 32-bit down counter decrements. The application program writes a value in the WDT_LR register regularly to prevent an instant reset or an interrupt. Users must write certain values in the WDT_LR register to update WDT_CNT value.

Timeout cycle of the WDT is set to 2,000 ms by default.

8.3.1 WDT Block Diagram

Figure 61 shows a block diagram for the WDT.

Figure 61. WDT Block Diagram



8.3.2 Enabling the WDT

The WDT is always enabled after powered on, and it operates in WDT reset mode.

The PCLK (Peripheral Clock) or an external clock source can be set as a clock source for the WDT. The external clock source can be selected as the WDT's clock source when the CKSEL bit in the WDT_CON register is set to '1' and the SCU_MCCR1 register is set to select the external clock source. When changing the clock source, set it after disabling the WDTEN bit in the WDT_CON register.

8.3.3 Controlling the Down Counter

When the WDTEN in WDT_CON register is set to '1', the 32-bit down counter that is the WDT_CNT register starts counting down. To prevent an instant reset or an interrupt during the down counter operation, the WDT_CNT register must be set to a certain value.

To change the value of the WDT_CNT register, users need to follow the procedure below:

1. Write a number larger than '0' to the WDT_LR register with the WDTEN bit in the WDT_CON register set to '1'.
2. Reload the counter value using the WDT_LR register while the down counter operates. It is important that the operating down counter must have a value larger than zero.

The WDT includes a programmable 32-bit down counter prescaler that enables users to set the timeout intervals using many different methods. Users can generate the WDT base clock by configuring the WPRS[2:0] bits in the WDT_CON register, and the minimum frequency value of the prescaler is 1/256 of the clock source frequency.

Table 71 describes the prescaled frequencies of various clock sources in the WDT.

Table 71. Prescaled WDT Counter Clock Frequency

Clock Source	WDTCLKIN / 1	WDTCLKIN / 4	WDTCLKIN / 16	WDTCLKIN / 64	WDTCLKIN / 256
LSI	500 kHz	125 kHz	31.25 kHz	7.8125 kHz	1.953125 kHz
LSE	32.768 kHz	8.192 kHz	2.048 kHz	512Hz	128Hz
MCLK	MCLK	MCLK / 4	MCLK / 16	MCLK / 64	MCLK / 256
HSI	32 MHz	8 MHz	2 MHz	500 kHz	125 kHz
HSE	XTAL	XTAL / 4	XTAL / 16	XTAL / 64	XTAL / 256
PLL	PLL	PLL / 4	PLL / 16	PLL / 64	PLL / 256

NOTE:

1. Time-out period = (Load Value + 1) × (1 / pre-scaled WDT counter clock frequency) at interrupt operation
   Time-out period = (Load Value) × (1 / pre-scaled WDT counter clock frequency) at reset operation
   // At the load value reaches '0', underflow flag is set to '1'

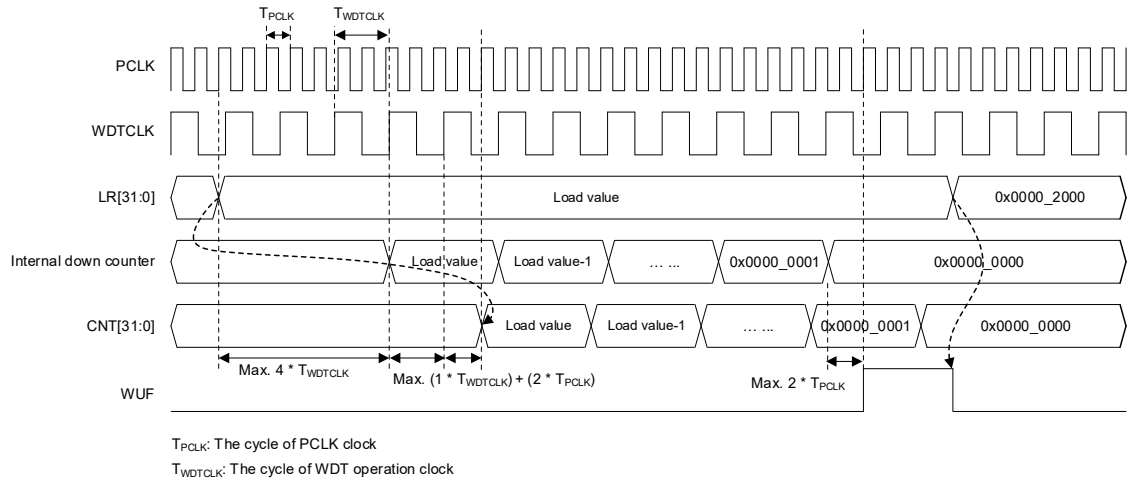
8.3.4 Interrupt Feature

The WDT interrupt can be used to perform safety tasks or data logging before triggering an actual reset.

The WDT interrupt is enabled by setting the WDTIE bit in the WDT_CON register to '1'. When the value of the WDT_CNT down counter changes from one to zero, the interrupt is triggered and the WUF flag in the WDT_CON register is set to '1'. To clear the WUF flag, users must write a value other than zero in the WDT_LR register.

8.3.4.1 Interrupt Mode Operation Timing Diagram

Figure 62. Interrupt Mode Operation Timing Diagram with External Clock



NOTE:

1. The internal WDT 32-bit down counter uses the WDTCLK as a reference clock, and the WDT registers operate based on the PCLK. To synchronize these two clocks, the delay is introduced as shown in the above timing diagram.

In WDT interrupt mode, to prevent the WDT's next interrupt in short time after the WDT underflows, specific counter value should be reloaded. This reload operation can use only when the WDT timer counter is set for Interrupt Mode.

It takes five cycles from the loaded value to the count value. The WDT interrupt signal for the Internal down counter value can be delayed up to two system bus clocks (For the case of synchronized logic).

8.3.5 How to Program the WatchDog Timeout

8.3.5.1 Basic Reset Operation

To reset the WDT, users must set the WDTRE bit in the WDT_CON register and the WDTRST bit in the SCU_RSER register to '1'. When the down counter reaches zero (the moment the value of the down counter changes from one to zero), the WDT generates the reset signal to reset microcontroller.

8.3.6 Debug Mode

If the WDBG bit in the WDT_CON register is set to '1', users can stop the WDT in Debug Mode. Users can see that the down counter value stops without decreasing, when pressing the Stop in Debug Mode. If the users execute the run again, the down counter value starts to decrease by one.

Conversely, if the WDBG bit in the WDT_CON register is set to '0', the WDT counter continues to operate without stopping in Debug Mode. If users press Stop in Debug Mode, the down counter value appears to be stationary. However, when executing Run again, the users can check that the down counter value is reduced because the down counter was not stationary.

8.4 WDT Registers

The base address of WDT and register map are described in the followings. Initial setting of WDT's time-out period is 2,000 ms.

Table 72. Base Address of WDT

Name	Base Address
WDT	0x4000_0200

Table 73. WDT Register Map

Name	Offset	Type	Description	Reset Value	Reference
WDT_LR	0x0000	RW	WDT Load Register	0x0000_0000	8.4.1
WDT_CNT	0x0004	RO	WDT Current Count Register	0x0000_7A12	8.4.2
WDT_CON	0x0008	RW	WDT Control Register	0x0000_805C	8.4.3
WDT_AEN	0x00F0	RW	WDT Access Enable Register	0x0000_0000	8.4.4

8.4.1 WDT_LR: WDT Load Register

The WDT_LR is used to update the value of the WDT_CNT register. To change the WDT_CNT's value, two conditions must be satisfied: the WDTEN in WDT_CON register must be set to '1', and the WDT_LR must be written.

If the WDTEN bit in the WDT_CON register is set to '0', the value written to the WDT_LR register will remain unrepresented in the WDT_CNT register until the WDTEN bit is changed to '1'.

If the WDT is being used as a reset source, the WDT_LR register must be written before the WDT_CNT value becomes '0' to prevent a reset.

The WDT triggers an event when the WDT_CNT value is changed from '1' to '0'. Therefore, when the WDT is used in reset mode, the WDT's count value is written to the WDT_LR as it is; whereas when the WDT is used in interrupt mode, '1' must be subtracted from the count value before the WDT_LR is written to.

At least five WDT counter clock cycles are required to update the WDT_CNT with the WDT_LR value.

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
LR[31:0]																															
0x0000_0000																															
RW																															

31 0	LR[31:0]	WDT load value register If the WDTEN remains at '1', the WDT_CNT register will be updated with the WDT_LR value. Since the initial value of WDT_LR is zero, it must be set to the desired value before use.

8.4.2 WDT_CNT: WDT Current Count Register

The WDT_CNT register is a 32-bit down counter that shows the WDT's current value. It is a read-only register. Its value can be changed by writing to the WDT_LR register while the WDTEN bit in the WDT_CON register is '1'.

Thus, when the WDT's count value reaches zero, an interrupt or reset is triggered.

To use the WDT as a reset source, both the WDTRE bit in the WDT_CON register and the WDTRST bit in the SCU_RSER register must be set to '1'.

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
CNT[31:0]																															
0x0000_7A12																															
R0																															

31 0	LR[31:0]	WDT current counter register The 32-bit down-counter counts down from the value written to the WDT_LR register.

8.4.3 WDT_CON: WDT Control Register

The microcontroller's WDT module must be set appropriately before it is enabled. The WDT module can be programmed to trigger a reset event or interrupt signal.

Instead of being used as a reset source or an interrupt source, the WDT can also function as a countdown timer starting from the set value of the down counter.

The WUF bit is a flag that is set when the WDT counts down to zero.

WDT_CON=0x4000_0208

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
Reserved		WDBG	Reserved	WUF	WDTIE	WDTRE	Reserved	WDTEN	CKSEL	WPRS[2:0]
-		1	-	0	0	1	-	1	1	100
-		RW	-	RW	RW	RW	-	RW	RW	RW

15	WDBG	Enable the WDT in debug mode	
		0	Enables the WDT counter stop in debugging mode
		1	Disables the WDT counter stop in debugging mode
8	WUF	WDT underflow flag (The bit is cleared when WDT_LR is written.)	
		0	There is no underflow.
		1	Underflow is pending.
7	WDTIE	Enable the WDT counter underflow interrupt	
		0	Disables the interrupt.
		1	Enables the interrupt.
6	WDTRE	Enable the WDT counter underflow reset	
		0	Disables the WDT counter underflow reset.
		1	Enables the WDT counter underflow reset.
4	WDTEN	Enable the WDT counter	
		0	Disables the WDT counter.
		1	Enables the WDT counter.
3	CKSEL	WDTCLKIN clock source selection	
		0	PCLK
		1	External clock (SCU_MCCR1)
2 0	WPRS[2:0]	Counter clock prescaler	
		000	WDTCLK = WDTCLKIN
		001	WDTCLK = WDTCLKIN / 4
		010	WDTCLK = WDTCLKIN / 8
		011	WDTCLK = WDTCLKIN / 16
		100	WDTCLK = WDTCLKIN / 32
		101	WDTCLK = WDTCLKIN / 64
		110	WDTCLK = WDTCLKIN / 128
		111	WDTCLK = WDTCLKIN / 256

NOTE:

1. WDTO output is output according to WUF (WDT underflow flag) in the WDT_CON Register.
   If WUF is underflow pending with '1', WDTO outputs high-level, and if WUF is '0', WDTO outputs low-level.

8.4.4 WDT_AEN: WDT Access Enable Register

The WDT_AEN register enables or disables writing to the settings of all WDT registers. In addition, when the WDT_AEN register is written '0x555A', WDT_CNT value is updated as current WDT_LR value instantly. After using the instant reload, WDT access is disabled automatically.

WDT_AEN=0x4000_02F0

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
Reserved	ENS	AEN[15:0]	
-	0	-	
-	R0	W0	

16	ENS	Enable status bit
	0	Disables AEN.
	1	Enables AEN (WDT register access possible state)
15 0	AEN[15:0](1)	Enable the write access of WDT registers
		Writing '0xA55A' to the bit field enables writing new values to WDT module's registers. User can write a different value to this bit field to protect the WDT registers against being updated with new values. Additionally, writing 0x555A to the bit field triggers an instant reload. When using the instant reload, the WDT_LR register should be set appropriate value before.

NOTES:

1. Example code for using the WDT_AEN register

```
WDT_AEN = 0xA55A; // Enables WDT access.
                  // WDT_LR and WDT_CON, etc. become settable.
WDT_AEN = 0;      // Disables WDT access.
```

2. To use the reload feature, a value greater than 0 must be set in the WDT_LR register.

```
WDT_AEN = 0xA55A; // Enable WDT access.
WDT_LR = 0x7A12;  // Set timer load value.
WDT_AEN = 0;      // Disable WDT access.
```

3. The instant reload feature

```
WDT_AEN = 0x555A; // Triggers an instant reload.
```

4. When using the instant reload, the WDT_LR register should be set appropriate value before. After using the instant reload, WDT access is disabled automatically.

8.4.5 WDT Register Map Summary

Table 74. WDT Register Map Summary

Offset	Register name	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
0x00	WDT_LR	LR[31:0]																															
	Reset value	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
0x04	WDT_CNT	CNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0
0x08	WDT_CON	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	WDBG	Res	Res	Res	Res	Res	Res	WUF	WDTIE	WDTRE	Res	WDTEN	CKSEL	WPRS[2:0]		
	Reset value																	1							0	0	1		1	1	1	0	0
0xF0	WDT_AEN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ENS	AEN[15:0]															
	Reset value																0	-															

9. 16-bit Timer

9.1 16-bit Timer Introduction

The A34M420 has a TIMER module each configured with 10 units. This 16-bit Timer supports four operating modes such as Periodic mode, PWM mode, One-shot mode, and Capture mode.

Users can use a divided PCLK or an external clock as an input clock source for the 16-bit Timer. An internal 10-bit prescaler allows to generate a variety of timer base clocks.

Interrupts can be triggered at regular intervals when the timer is used in Periodic Mode. Users can set the period and duty to form a PWM signal that is used in PWM Mode.

In One-shot Modes, the timer can generate one PWM waveform. In Capture Mode, the external input signal's pulse intervals can be measured based on the preset condition. Moreover, the timer can export signals to other devices to control them. This timer is primarily used as periodic tick timer or wake-up sources.

9.2 16-bit Timer Main Features

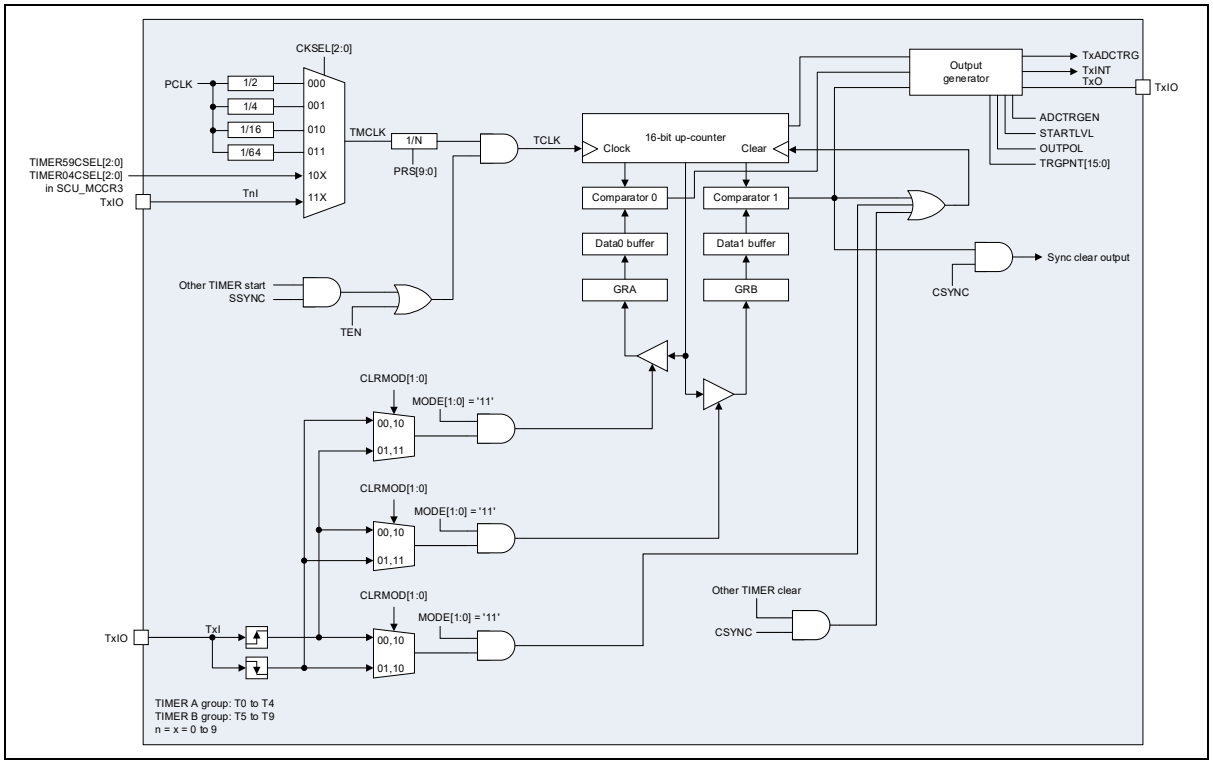
TIMERn (n = 0 to 9) includes the features below:

- 16-bit up-counter timers
- Four operating modes:
  - Periodic mode
  - One-shot mode
  - PWM mode
  - Capture mode
- Various interrupts:
  - Match interrupt
  - Overflow interrupt
- Various clock sources selectable as a timer input:
  - Four PCLK prescaler levels (1/2, 1/4, 1/16, and 1/64)
  - External clock sources selectable using the SCU_MCCR3 register: LSI, LSE, MCLK, HSI, HSE, and PLL
  - Timer clock source on port TnIO used as input. (n = 0 to 9)
- A built-in 10-bit prescaler supporting the timer input clock
- PWM synchronization
  - Start delay and clear synchronization

9.3 16-bit Timer Functional Description

9.3.1 Block Diagram

Figure 63. 16-bit Timer Block Diagram (n = 0 to 9)



9.3.2 Pins and Internal Signals

Table 75 summarizes the I/O information of the TIMERn (n = 0 to 9).

Table 75. Input and Output Pins for TIMERn (n = 0 to 9)

Pin Name	Type	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
T0IO	IO	TIMER0 capture input signal / external clock input TIMER0 periodic / pwm / one-shot output	O	O	O
T1IO	IO	TIMER1 capture input signal / external clock input TIMER1 periodic / pwm / one-shot output	O	O	O
T2IO	IO	TIMER2 capture input signal / external clock input TIMER2 periodic / pwm / one-shot output	O	O	O
T3IO	IO	TIMER3 capture input signal / external clock input TIMER3 periodic / pwm / one-shot output	O	O	O
T4IO	IO	TIMER4 capture input signal / external clock input TIMER4 periodic / pwm / one-shot output	O	O	O
T5IO	IO	TIMER5 capture input signal / external clock input TIMER5 periodic / pwm / one-shot output	O	O	N/A
T6IO	IO	TIMER6 capture input signal / external clock input TIMER6 periodic / pwm / one-shot output	O	O	N/A
T7IO	IO	TIMER7 capture input signal / external clock input TIMER7 periodic / pwm / one-shot output	O	O	N/A
T8IO	IO	TIMER8 capture input signal / external clock input TIMER8 periodic / pwm / one-shot output	O	O	O
T9IO	IO	TIMER9 capture input signal / external clock input TIMER9 periodic / pwm / one-shot output	O	O	O

Table 76. Internal Input and Output Signals for TIMERn (n = 0 to 9)

Internal Signal Name	Signal Type	Description
PCLK	Input	Peripheral system clock
EXT_CLK	Input	Timer external clock (Selectable with the setting of SCU_MCCR3)
TnI	Input	Timer n input to TnIO for external clock or capture input
CKSEL[2:0] in TIMERn_CR1	Input	Counter clock source selection bit in the Timer/counter Control Register 1
TIMERn_PRS	Input	Prescaler value for the counter clock
TEN in TIMERn_CR2	Input	Enable timer signal in the Timer/counter Control Register 2
SSYNC in TIMERn_SYNC	Input	Synchronized counter start mode signal in the Timer/counter Sync Configuration Register
CSYNC in TIMERn_SYNC	Input	Synchronized counter clear mode signal in the Timer/counter Sync Configuration Register
ADCTRGEN in TIMERn_CR1	Input	Uses the timer as an ADC trigger source signal in the Timer/counter Control Register 1
STARTLVL in TIMERn_CR1	Input	Starting output value signal in the Timer/counter Control Register 1
OUTPOL in TIMERn_CR1	Input	Output polarity signal in the Timer/counter Control Register 1
TnADTRG	Output	Timer n ADC trigger source signal
TnINT	Output	Timer n interrupt signal
TnO	Output	Timer n output signal to TnIO
SYNC CLEAR OUT	Output	Synchronized counter clear signal

NOTE:

1. n = 0 to 9.

9.3.3 Clock selection

The counter clock can be provided by the following clock sources:

- Peripheral system clock (PCLK)
- Timer external clock (SCU_MCCR3 register)
- Input to pin TnIO (n = 0 to 9) for timer clock source

Users can select a clock source by configuring the CKSEL[2:0] in TIMERn_CR (n = 0 to 9).

9.3.3.1 Peripheral System Clock (PCLK)

Users can select a timer input clock source using four PCLK divide levels (PCLK / 2, PCLK / 4, PCLK / 16, PCLK / 64). If the CKSEL[2:0] bit in the TIMERn_CR1 register is set to a number between 0 and 3, the timer clock is provided by the PCLK divided level.

Table 77. PCLK Divide Levels and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
000	PCLK / 2
001	PCLK / 4
010	PCLK / 16
011	PCLK / 64

9.3.3.2 Timer External Clock (SCU_MCCR3 Register)

Users can select a timer input clock source by configuring the SCU_MCCR3 register. The selectable clock sources include LSI, LSE, MCLK, HSI, HSE and PLL. If the CKSEL[2:0] bits in the TIMERn_CR1 is set to ‘10X’, the timer clock is provided by the SCU_MCCR3 register.

Table 78. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
10X	EXT_CLK (SCU_MCCR3 register)

The EXT_CLK can be set by the SCU_MCCR3 register, Miscellaneous Clock Control Register 3. Because the PCLK is the standard of operation, the SCU_MCCR3 value must be set slower than the PCLK, and therefore, it must be divided at least two times if the EXT_CLK clock source uses the same frequency as the PCLK. If the PCLK is half as slow as the EXT_CLK clock source, the EXT_CLK clock source must be divided at least four times.

9.3.3.3 Input to Pin TnIO for Timer Clock Source

Users can select a timer input clock source using input on the port TnIO (n = 0 to 9). If the IOSEL bit in the TIMERn_CR1 is set to ‘0’, the TnIO pin is set as input.

Table 79. TnIO and IOSEL in TIMERn_CR1

IOSEL in TIMERn_CR1	Description
0	Sets the pin as an input port.
1	Sets the pin as an output port.

If the CKSEL[2:0] bits in the TIMERn_CR1 is set to ‘11X’, the timer clock is provided by input on the port TnIO.

Table 80. TnIO and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
11X	Input to pin TnIO (n = 0 to 9)

9.3.4 Time-base Unit

The main block of the programmable 16-bit timer is a timer/counter with its related timer/counter count register. The counter increases based on the specified input clock. The counter clock can be divided by a prescaler.

The time-base unit includes the following registers:

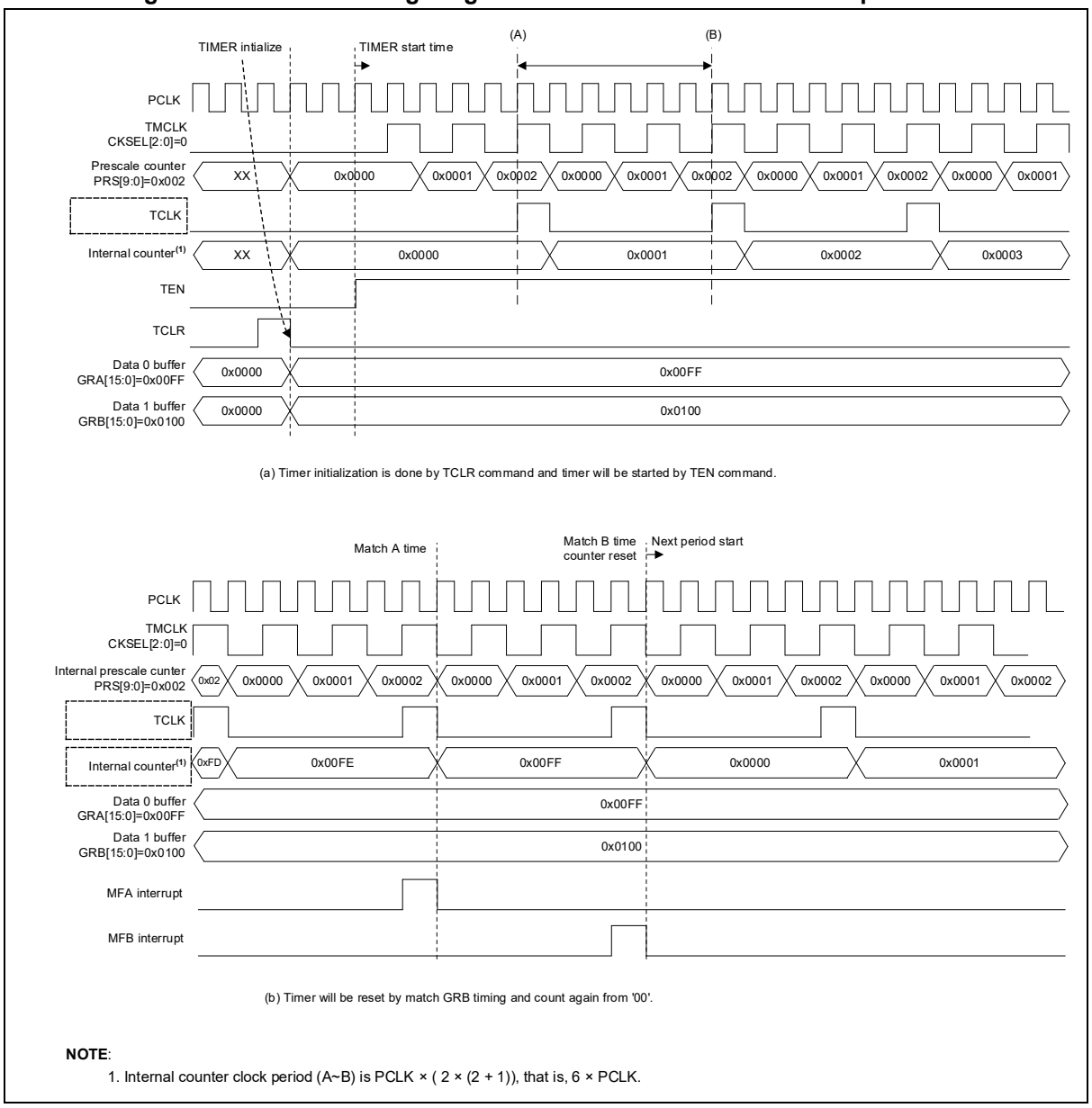
- Timer Counter Register: TIMERn_CNT (n = 0 to 9)
- Timer Prescaler Register: TIMERn_PRS (n = 0 to 9)
- Timer General Data Register A: TIMERn_GRA (n = 0 to 9)
- Timer General Data Register B: TIMERn_GRB (n = 0 to 9)
- Miscellaneous Clock Control Register 3: SCU_MCCR3

9.3.4.1 Prescaler Description

The prescaler is used to set the timer input frequency divider. It is 10-bit wide. Users can generate precise and varied timer base clocks by applying the prescaler to the timer clock source that has been selected in TIMERn_CR1 register.

The TMCLK shown in Figure 64 is a reference clock for operating the timer. The frequency of this clock can be divided by setting the prescaler to operate a counting clock. This figure shows the start and end points of a counter in normal periodic mode. When changing the timer settings or restarting the timer with a new value, it is recommended that users set the TCLR bit in the TIMERn_CR2 before setting the TEN bit in the TIMERn_CR2 register.

Figure 64. Counter Timing Diagram with Basic Start and Match Operations



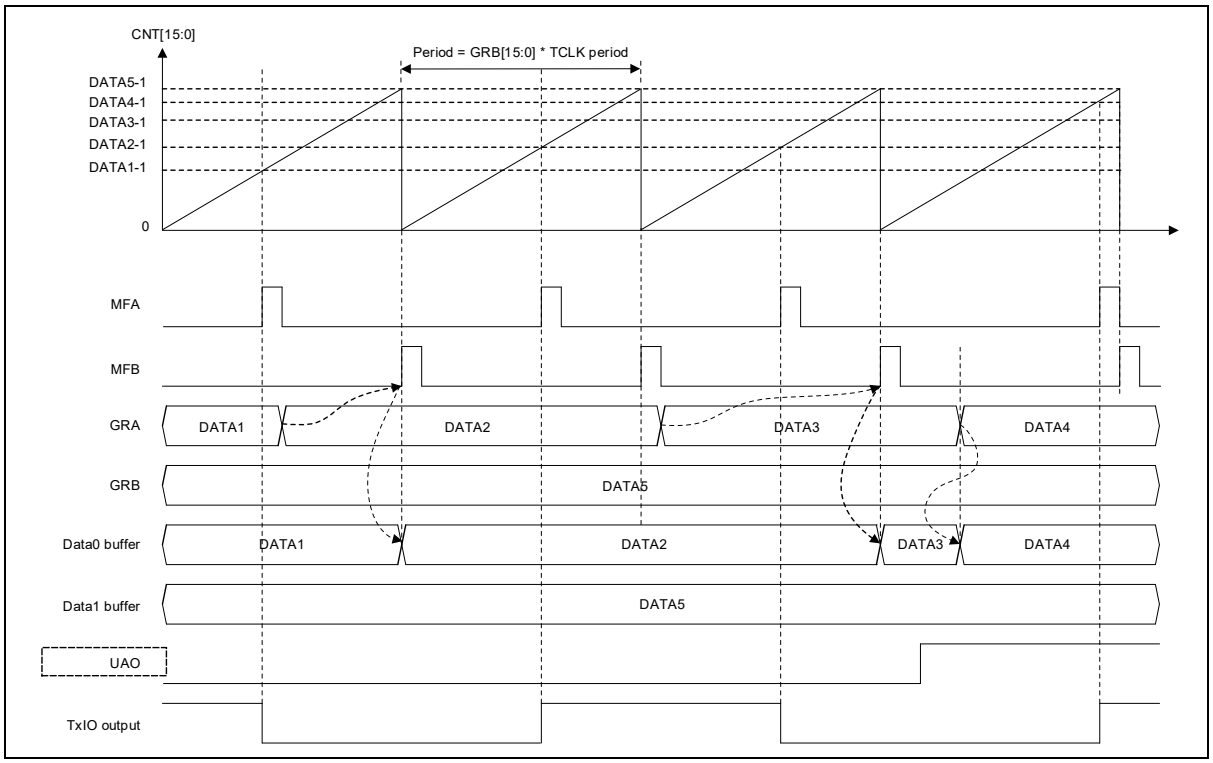
NOTE:

1. Internal counter clock period (A~B) is PCLK × (2 × (2 + 1)), that is, 6 × PCLK.

9.3.5 Periodic Mode

Figure 65 shows a timing diagram for the normal periodic mode operation. The TIMERn_GRA value determines the timer period. At this time, the TIMERn_GRB value must be set to be equal to the TIMERn_GRA value.

Figure 65. Normal Periodic Mode Operation



The timer's count period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRA value
- Match A interrupt time = TCLK period × TIMERn_GRA value

If the TIMERn_GRA value is '0, the timer cannot be started even if the TEN bit in the TIMERn_CR2 is set to '1' because the period is zero.

The values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this periodic mode with the UAO bit in the TIMERn_CR1 set value of '0', it will load the TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to '1', the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated. The TnIO output signal will be toggled at every match A condition time.

If the TIMERn_GRA value is zero and the TIMERn_GRB value is not zero, the TnIO output does not change its previous level. If the TIMERn_GRA value is the same as the TIMERn_GRB value, the TnIO output will toggle simultaneously depending on the counter clear time. The initial level of the TnIO signal is decided by the STARTLVL bit in the TIMERn_CR1 register.

9.3.5.1 Operation according to TIMERn_GRA and TIMERn_GRB Settings

- For normal counter mode operation, users must enter the same value in the TIMERn_GRA and TIMERn_GRB registers.
  - Please refer to Table 81 according to the TIMERn_GRA and TIMERn_GRB register settings.
- To use the Timer Overflow function, users must write '0' to both the TIMERn_GRA and TIMERn_GRB registers.
  - If the TIMERn_GRA register or the TIMERn_GRB register has a value other than '0', the Match flag occurs.
  - To generate the normal OverFlow flag, write '0' to both the TIMERn_GRA and TIMERn_GRB registers.

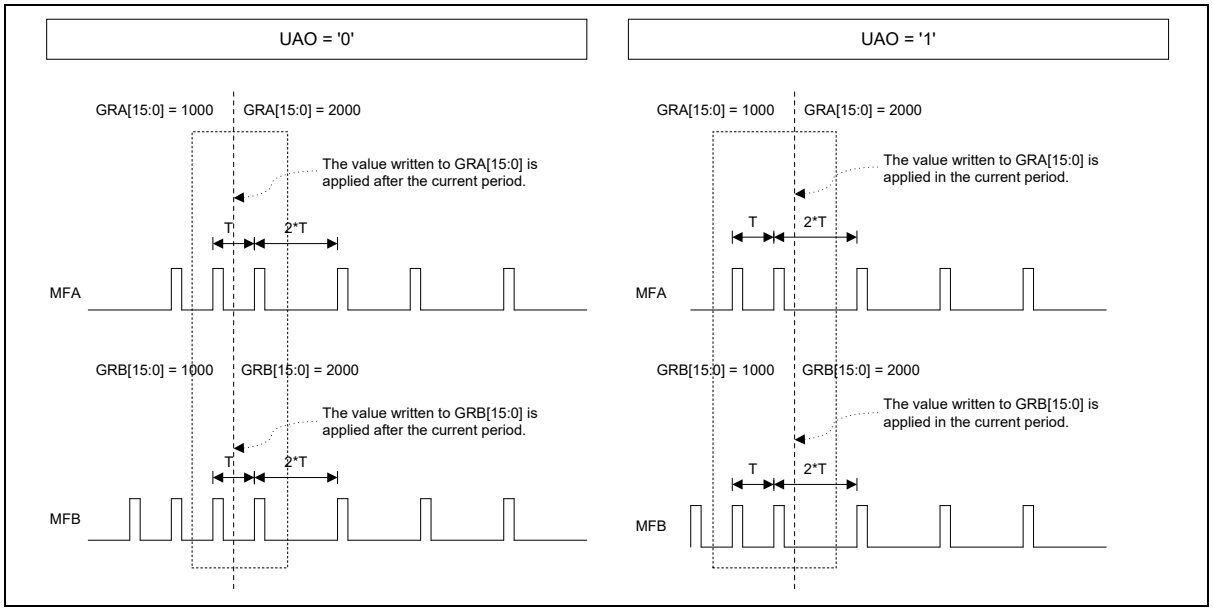
Table 81. Operations Comparison according to TIMERn_GRA/GRB Settings

Case	GRA[15:0]	GRB[15:0]	UAO	OUTPOL	IOSEL	STARTLVL	TnIO Pin Output
1	1,000	1,000	0/1	0	1	0	Normal operation
2	1,000	2,000	0/1	0	1	0	Outputs the GRB[15:0] setting value.
3	1,000	0	0/1	0	1	0	Outputs the GRB[15:0] setting value.
4	1,000	1 to 999	0/1	0	1	0	No output
5	1,000	65,535	0/1	0	1	0	Outputs the GRB[15:0] setting value.

9.3.5.2 Operation according to UAO, OUTPOL, and STARTLVL

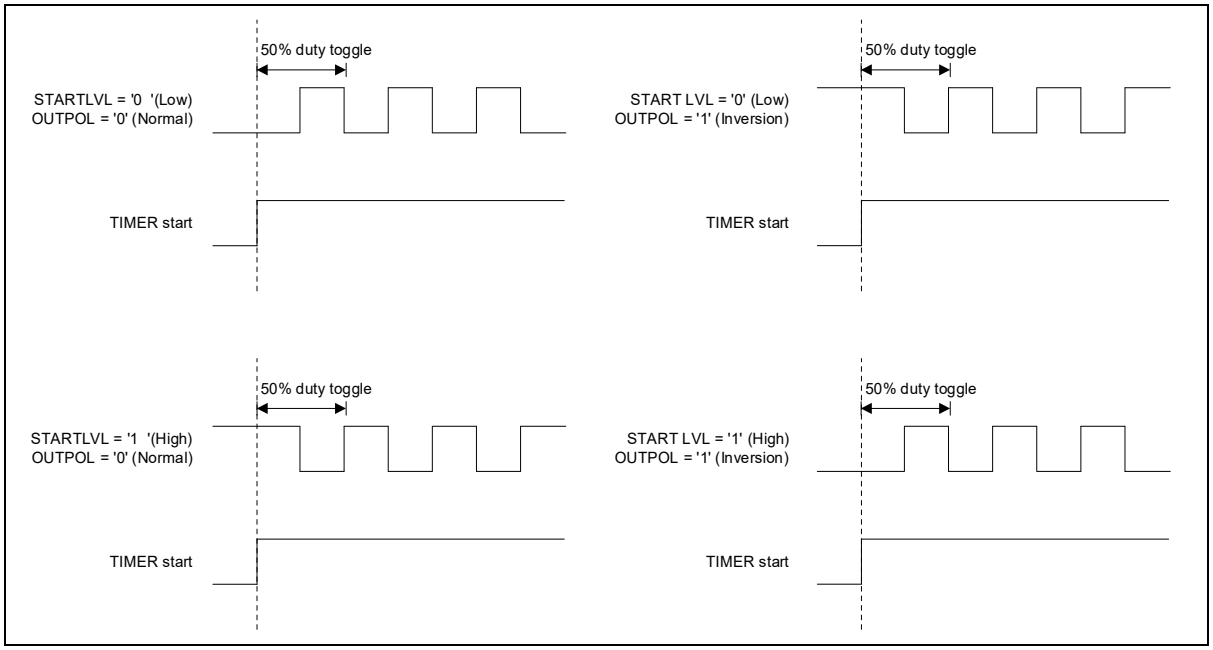
- Refer to the Figure 66 for the operation waveform according to the UAO bit in the TIMERn_CR1 settings.
  - If the UAO bit in the TIMERn_CR1 value is ‘0’, the TIMERn_GRA / GRB value which has been updated after the current cycle operation is applied.
  - If the UAO bit in the TIMERn_CR1 value is ‘1’, the TIMERn_GRA / GRB value is applied within the current cycle operation.

Figure 66. Operation Waveform depending on UAO Bit



- Refer to the Figure 67 below for the operation waveform according to the STARTLVL bit in the TIMERn_CR1 register, OUTPOL bit in the TIMERn_CR1 register settings (n = 0 to 9).
  - The STARTLVL bit in the TIMERn_CR1 register is reflected in TnIO output by the TCLR bit in the TIMERn_CR2 register.
  - The OUTPOL bit in the TIMERn_CR1 register is applied when the timer counter is cleared after setting. (Applied after the current cycle operation.)

Figure 67. Operation Waveform according to STARTLVL and OUTPOL Settings



9.3.5.3 Additional Precautions over Counter Mode Operation

1. If the timer counter is cleared during timer operation, the output of the timer pin (TnIO, n = 0 to 9) is changed.
2. The timer counter is cleared when the TIMERn_CNT register is forcibly cleared or when the TIMERn_GRB register is matched.
3. When the timer stops, the Timer pin (TnIO, n = 0 to 9) output maintains the status at the time of stopping, and when the timer counter is cleared, it is changed to the setting value of the STARTLVL bit in the TIMERn_CR1 register.

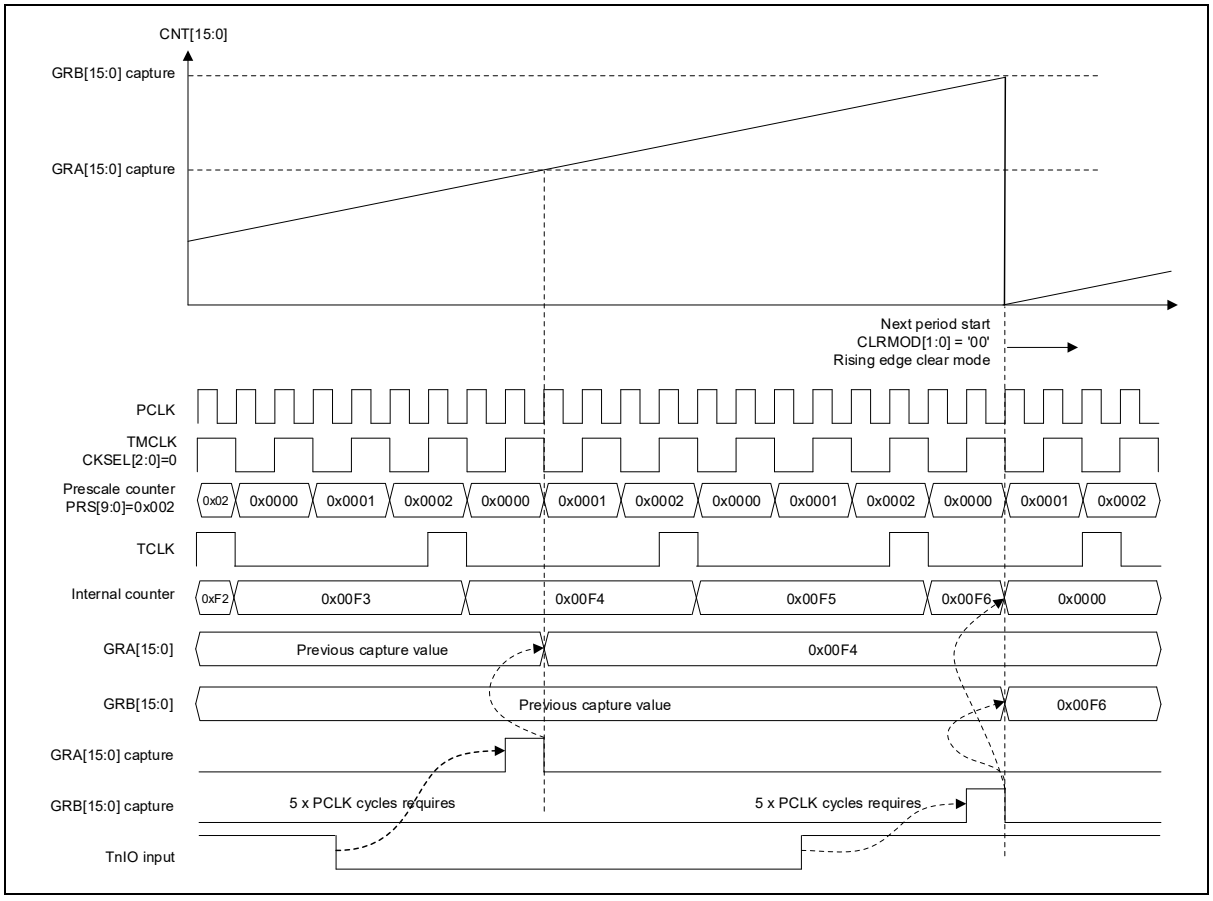
9.3.5.4 Counter Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the Peripheral Enable Register 1 (SCU_PER1) and the Peripheral Clock Enable Register 1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write ‘1’ to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the same set value in the TIMERn_GRA and TIMERn_GRB registers for period timer operation.
9. Write ‘1’ to the corresponding bit of the TIMERn_SR register to clear the pending interrupt.
10. Write ‘1’ to the MAIE bit in the TIMERn_IER register to set the timer match interrupt.
11. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
12. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

9.3.6 Capture Mode

Figure 68 shows a timing diagram for the capture mode operation. The TnIO input signal is used for capturing the pulse. Rising and falling edges can capture the counter value in each capture condition.

Figure 68. Timing Diagram of Capture Mode Operation



The actual capture point is five PCLK clock cycles after the rising or falling edge of the TnIO input signal. The internal counter can be cleared in multiple modes. The CLRMOD[1:0] bits in the TIMERn_CR1 controls counter clearing in capture mode. The supported modes include rising-edge clear mode, falling-edge clear mode, both-edge clear mode, and non-clear mode.

Example timing diagram in Figure 68 describes rising-edge clear mode. On the falling edge of the input signal on the TnIO, the TIMERn_GRA register captures the CNT value; on the rising edge, the TIMERn_GRB register captures the CNT value.

NOTE:

1. Since the timer input operates as a functional pin of a GPIO port, users must first set the GPIO port to function pin mode. When the GPIO port is set to function pin mode, it operates as an input port or an output port according to the timer operation mode.

To use the capture mode, set the GPIO port to timer function pin mode (TnIO, n = 0 to 9) and then set the IOSEL bit in the TIMERn_CR1 register to ‘0’.

Table 82. Operation Status by Capture Mode

CLRMOD[1:0] in TIMERn_CR1	Clear Mode	TIMERn_GRA data	TIMERn_GRB data
00	Rising edge clear	Falling edge capture	Rising edge capture
01	Falling edge clear	Rising edge capture	Falling edge capture
10	Both edge clear	Falling edge capture	Rising edge capture
11	Disable clear	Rising edge capture	Falling edge capture

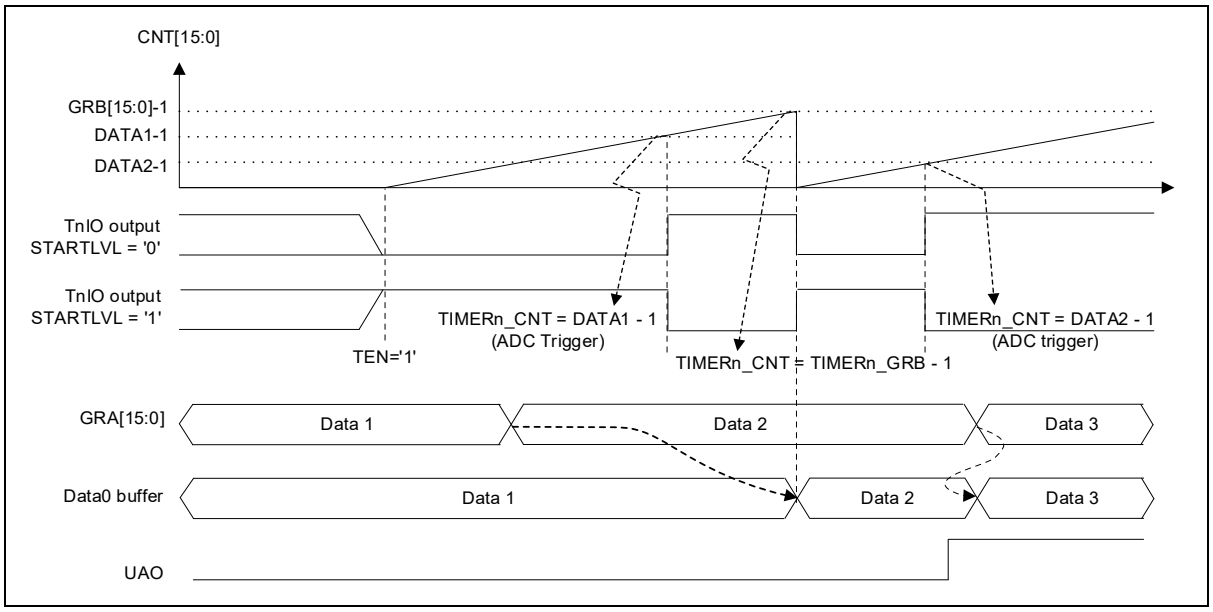
9.3.6.1 Capture Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Initialize the TIMERn_GRA and TIMERn_GRB registers to zero.
4. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
5. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
6. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
7. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
8. Write ‘0’ to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
9. For the clear selection in capture mode, enter the corresponding set value in the CLRMOD[1:0] bits in the TIMERn_CR1 register.
10. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
11. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

9.3.7 PWM Mode

Figure 69 shows a timing diagram for the PWM output mode operation. The TIMERn_GRB value decides the PWM pulse period. An additional comparison point is provided by the TIMERn_GRA register value which defines the pulse width of PWM output. (n = 0 to 9)

Figure 69. Timing Diagram of PWM Output Operation



The PWM pulse period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is zero, the timer cannot be started even if the TEN bit in the TIMERn_CR2 register is set to ‘1’ because the period is zero.

The values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this periodic mode with the UAO bit in the TIMERn_CR1 register set value of ‘0’, it will load TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to ‘1’, the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated.

The TnIO output signal generates a PWM pulse.

The TIMERn_GRB value defines the output pulse period and the TIMERn_GRA value defines the pulse width of PWM pulse. The active level of the PWM pulse can be controlled by the STARTLVL bit in the TIMERn_CR1 register.

ADC trigger generation is available at Match A interrupt time.

9.3.7.1 PWM Mode Operation Characteristics and Precautions

Precautions for setting the TIMERn_GRA register are shown below:

- The TIMERn_GRA value must be greater than '0' and less than or equal to the TIMERn_GRB value. (0 < TIMERn_GRA ≤ TIMERn_GRB)
- If the TIMERn_GRA value is zero, the GRA matching event does not occur, and the output does not change.

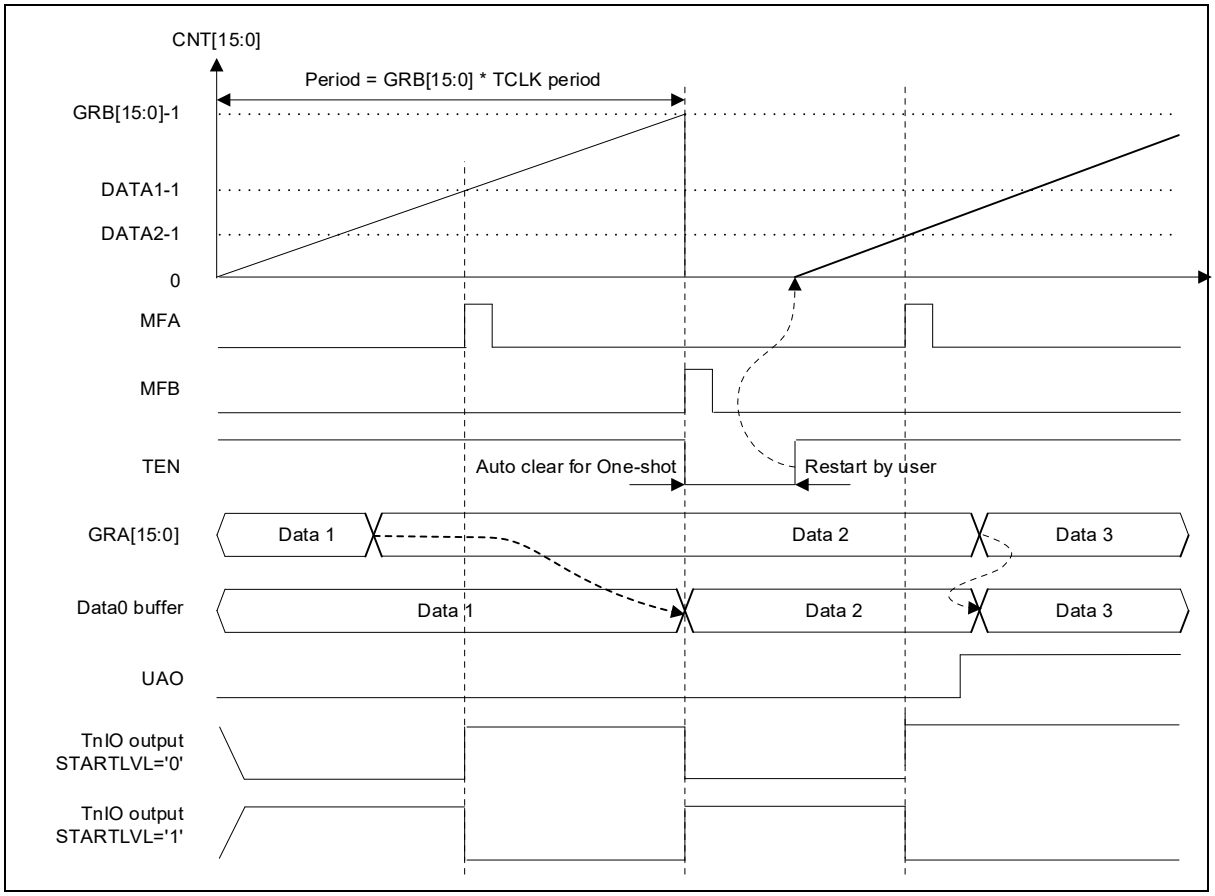
9.3.7.2 PWM Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the peripheral enable register1 (SCU_PER1) and the Peripheral Clock Enable Register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bit in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write '1' to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
9. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
10. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer operation.

9.3.8 One-shot Mode

Figure 70 shows a timing diagram for the one-shot mode operation. The TIMERn_GRB value determines the one-shot period, and the TIMERn_GRA value provides another comparative point.

Figure 70. Timing Diagram of One-shot Mode Operation



The one-shot count period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is zero, the timer cannot be started even if the TEN bit in the TIMERn_CR2 register is set to ‘1’ because the period is zero.

The values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this one-shot mode with the UAO bit in the TIMERn_CR1 register set value of ‘0’, it will load TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to ‘1’, the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated.

The TnIO output signal format is the same as in PWM mode. The TIMERn_GRB value defines the output pulse period and the TIMERn_GRA value defines the one-shot pulse width.

9.3.8.1 One-Shot Mode Operation Characteristics and Precautions

- Precautions when setting the TIMERn_GRA register:
  - The TIMERn_GRA value must be greater than zero and less than or equal to the TIMERn_GRB value. (0 < TIMERn_GRA ≤ TIMERn_GRB)
  - If the TIMERn_GRA value is zero, the GRA Matching event does not occur, and the output does not change.
- Operation sequence according to the STARTLVL in TIMERn_CR1 settings:
  - If the STARTLVL is '1', GRA (Duty) High output → GRB-GRA Low output → High output
  - If the STARTLVL is '0', GRA (Duty) Low output → GRB-GRA High output → Low output
  - Refer to the table below that compares the operations according to the STARTLVL and OUTPOL settings.

Table 83. One-Shot Mode Operation Setting Table

Case	GRA[15:0]	GRB[15:0]	UAO	POL	IOSEL	STARTLVL	Remarks
#1	500	2,000	0	0 (N)	1 (Out)	0	Initial output: Low High pulse output between GRA[15:0] and GRB[15:0]
#2	500	2,000	0	0 (N)	1 (Out)	1	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#3	500	2,000	0	1 (I)	1 (Out)	0	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#4	500	2,000	0	1 (I)	1 (Out)	1	Initial output: Low High pulse output between GRA[15:0] and GRB[15:0]

9.3.8.2 One-Shot Mode Register Setting Sequence

1. To enable system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bit in the TIMERn_CR1 register.

6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write ‘1’ to the IOSEL bit in the TIMERn_CR1 to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for the one-shot operation.
9. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
10. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

9.3.9 ADC Trigger

The timer module can generate ADC start trigger signals. One timer can be one trigger source of the ADC module. The trigger source is controlled by the ADC control registers.

Figure 71, Figure 72 and Figure 73 show how the ADC triggering works in each mode. The conversion rate must be shorter than the timer period; otherwise, an overrun can occur. ADC acknowledge is not required because the trigger signal is automatically cleared three PCLK clock pulses later.

Figure 71. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘00’)

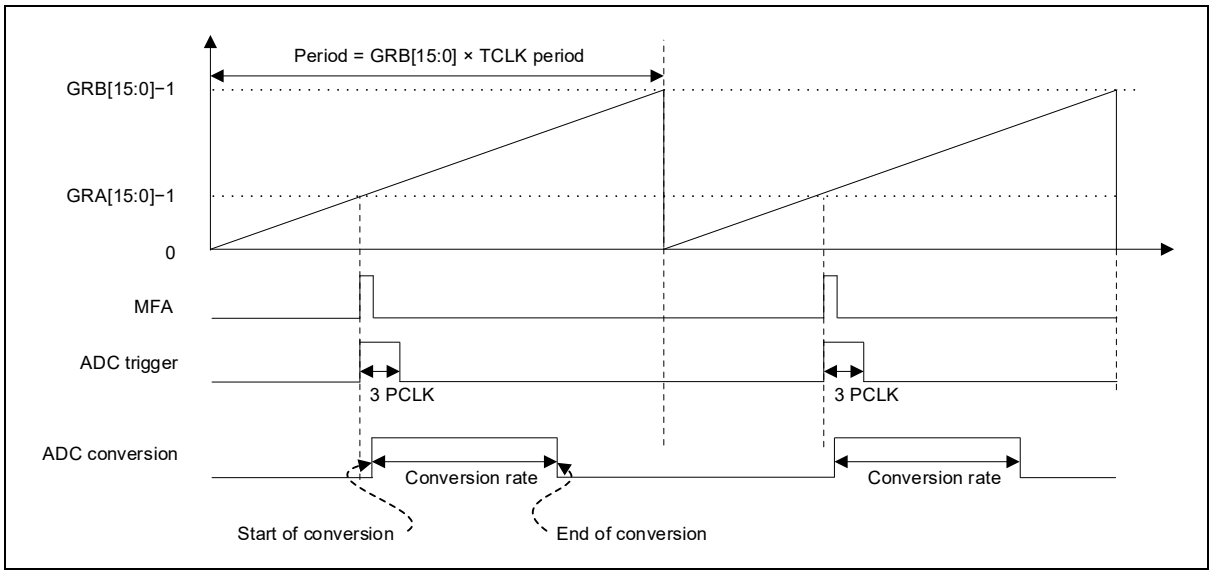


Figure 72. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '10')

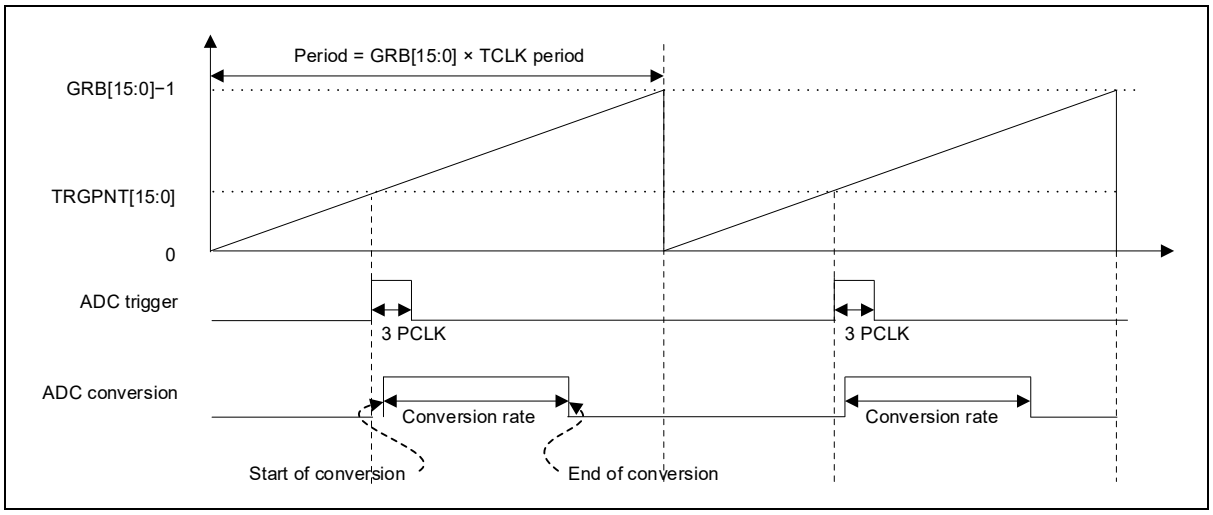
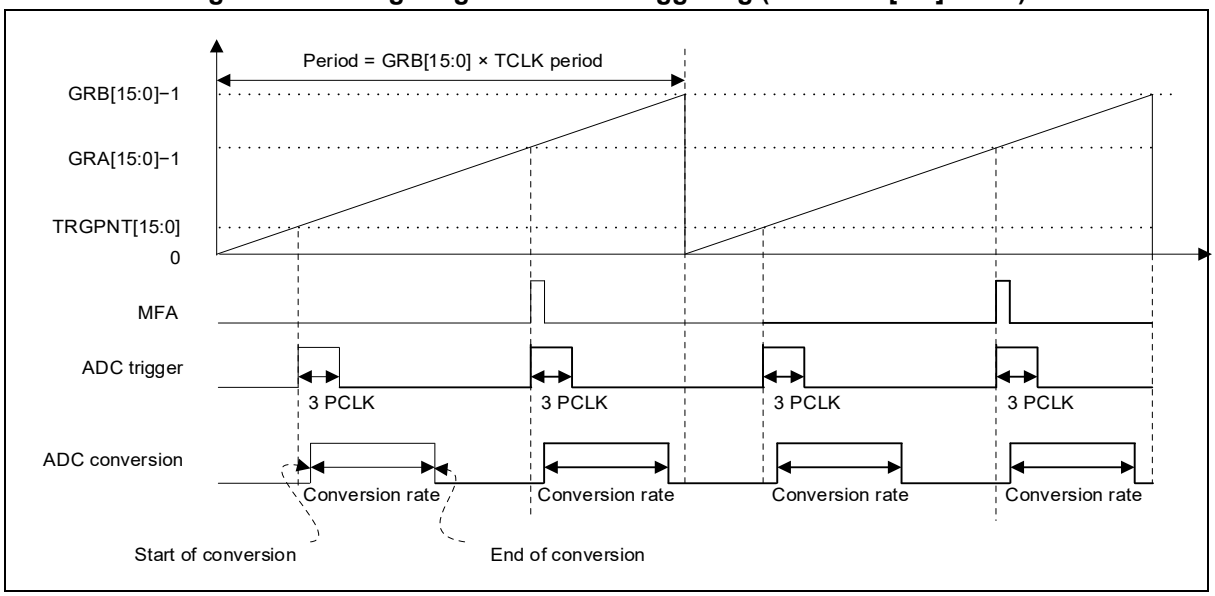


Figure 73. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '11')



9.3.9.1 ADC Trigger Operation Characteristics and Precautions

If the TIMERn_GRB value is less than the TIMERn_GRA value or the TIMERn_TRGPNT value, the timer counter is cleared whenever timer counter matches with TIMERn_GRB value. So, the matching event does not occur.

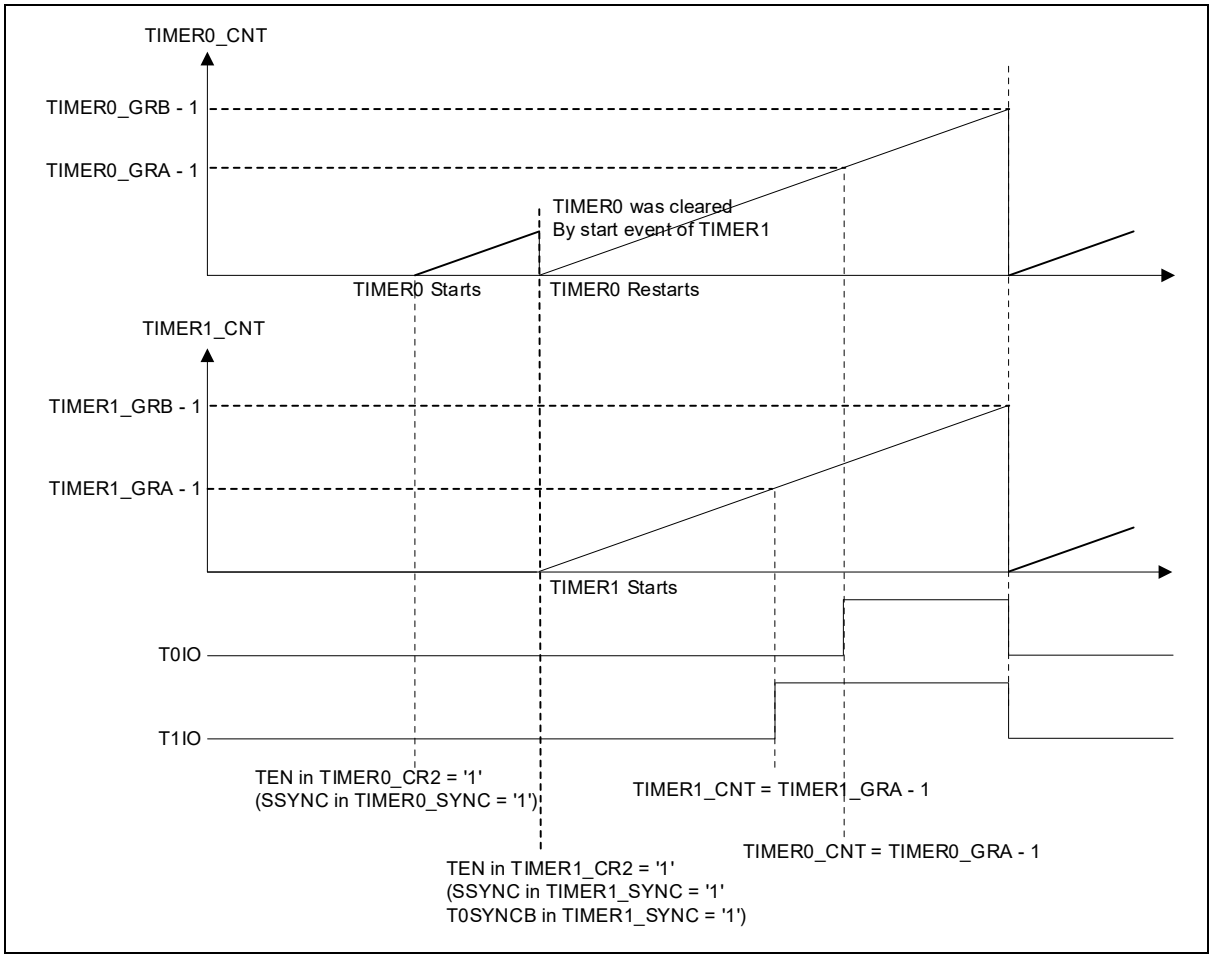
9.3.9.2 Register Setting Sequence for the ADC Trigger Operation

1. To enable system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 regsiter.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Enter the corresponding set values in the TIMERn_GRB and TIMERn_GRA registers to set the timer period and trigger point.
8. Write ‘1’ to the ADCTRGEN bit in the TIMERn_CR1 register to use the trigger source.
9. Enter the corresponding mode set value in the TRGMOD[1:0] bits in the TIMERn_CR1 register to select the trigger mode.
10. Enter the set value in the TIMERn_TRGPNT register to set the additional trigger point.
    A. If the TIMERn_GRA register or the TIMERn_TRGPNT register is selected by the TRGMOD[1:0] bits in the TIMERn_CR1 register , when the value set in the selected register matches the TIMERn_CNT value, the ADC is triggered and its value can be read.
11. Initialize the ADC registers. (Set the TRGSEL[1:0] bits in the ADCn_MR register to ‘01’.)
12. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
13. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

9.3.10 Timer Synchronization

PWM outputs are usually used as the synchronous PWM signal control. This function is provided with synchronous start. Figure 74 shows synchronous PWM generation.

Figure 74. Timer Synchronization Example (when SSYNC = '1', CSYNC = '1')



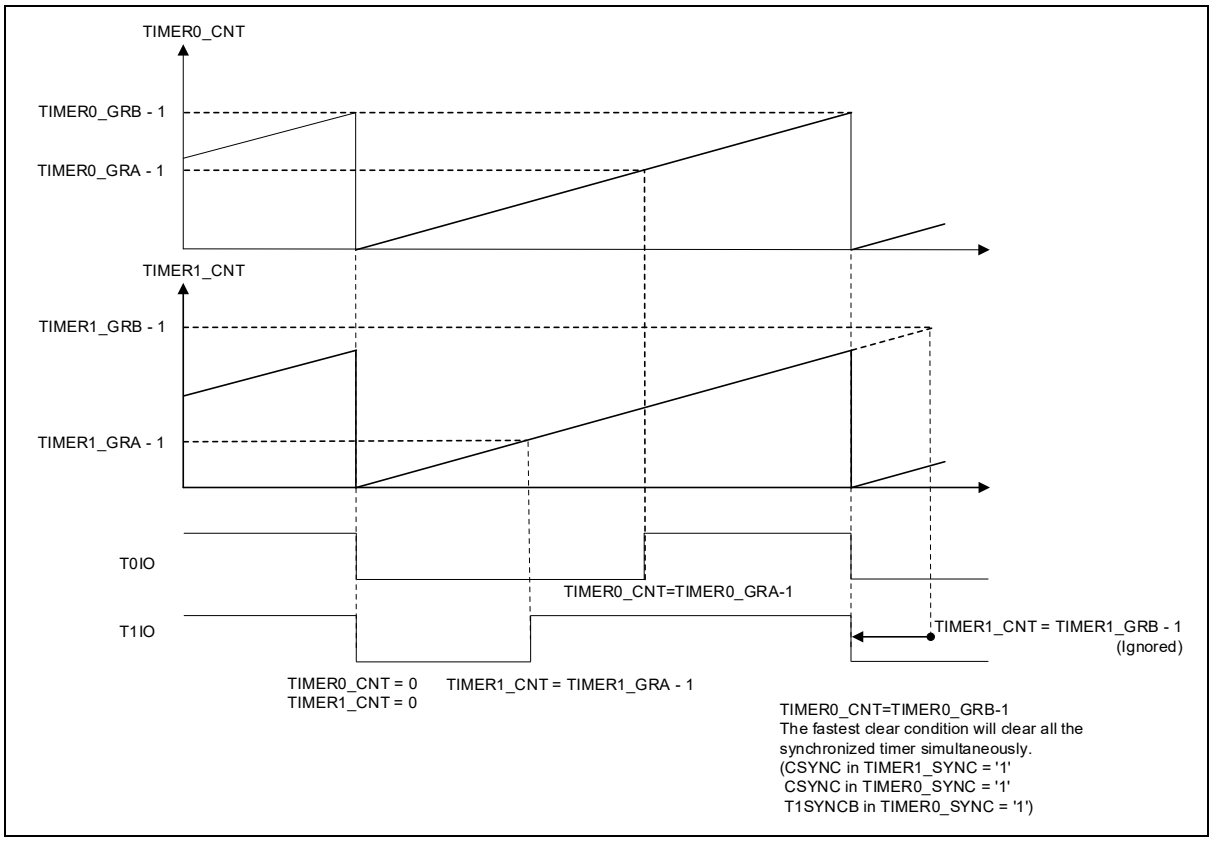
The SSYNC bit in the TIMERn_SYNC register controls start synchronization with other timer modules. In the above figure, the TIMER0 (slave timer) waveform shows that the timer's start is synchronized to the start of the signal from TIMER1 (master timer).

For this SYNC function in the figure above, both the master and slave timers' synchronous start bits must be set enabled: In the master timer's TIMER1_SYNC register, the T0SYNCB and SSYNC bits must be set to '1'. In the slave timer's TIMER0_SYNC register, the SSYNC bit must be set to '1'.

If only the master timer's synchronous start bit is set enabled for the slave timer while the slave's synchronous start bit is disabled, the slave timer can run independently without being affected by the master timer's start synchronization signal.

Additionally, because TIMER0 (slave) operates as synchronized to TIMER1's (master's) starts synchronization signal, it runs even if the TEN bit in the TIMER0_CR2 register is set to '0'.

Figure 75. Timer Synchronization Example (when CSYNC = '1')



The CSYNC bit in the TIMERn_SYNC register controls clear synchronization with other timer modules. In the above figure, the TIMER1 (slave timer) waveform shows that the timer's clear is synchronized to the clear of the signal from the TIMER0 (master timer).

For this CSYNC function in the figure above, both the Master and Slave Timers' synchronous clear bits must be set enabled: In the master timer's TIMER0_SYNC register, the T1SYNCB and CSYNC bits must be set to '1'. In the slave timer's TIMER1_SYNC register, the CSYNC bit must be set to '1'.

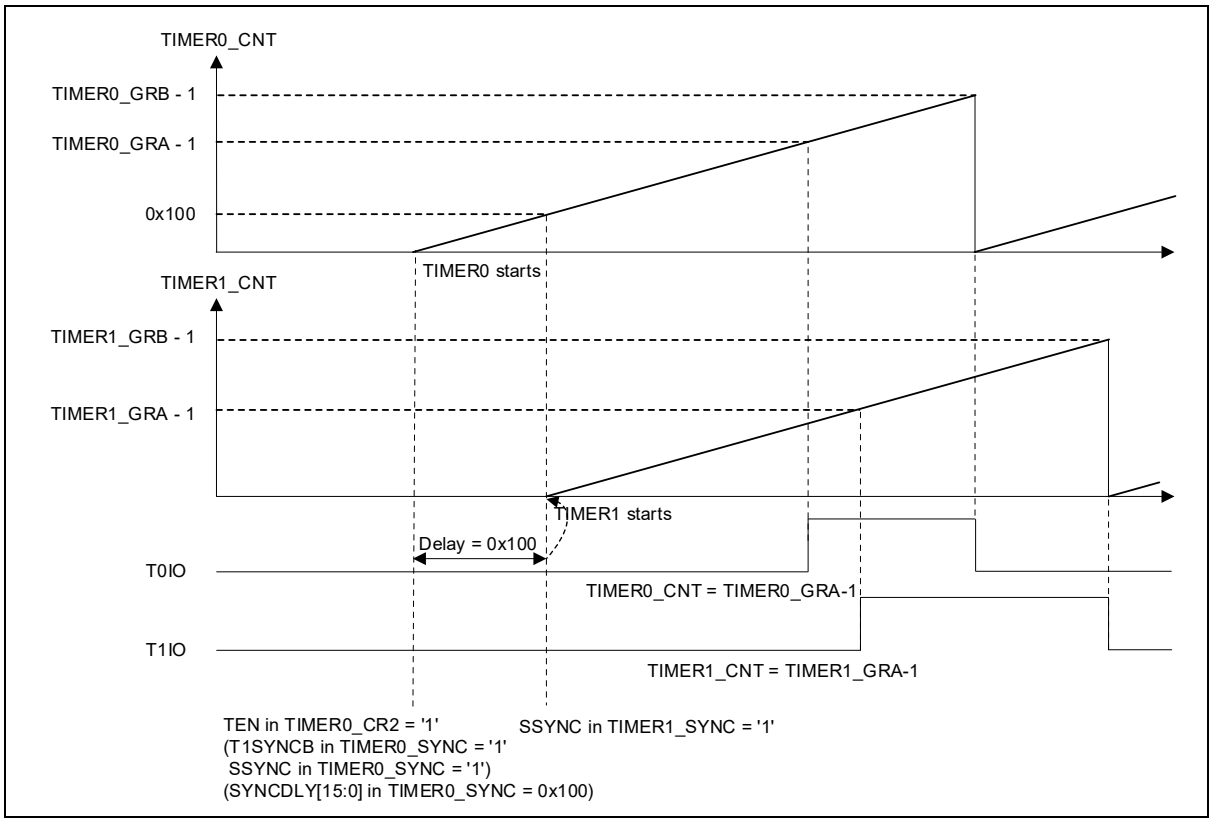
If only the master timer's synchronous clear bit is set enabled for the slave timer while the slave's synchronous clear bit is disabled, the slave timer can run independently without being affected by the master timer's clear synchronization signal.

9.3.10.1 Timer Delayed Synchronization

The PWM delayed synchronization function is used between timers within the same group (group 1: T0, T1, ..., T3 and group 2: T5, T6, …, T9); synchronization is delayed by the amount of time written to the SYNCDLY[15:0] bits in the TIMERn_SYNC.

Using this function, users can have slave timers start sequentially after a certain amount of delay time since the master timer started.

Figure 76. Timer Delayed Synchronization Example (when SSYNC = '1')



For example as shown in Figure 59, to start the TIMER1 (slave) after 0x100 counts from the TIMER0's (master's) synchronization signal output, users must set the master timer's TIMER0_SYNC register to T1SYNCB=1, SSYNC=1, and SYNCDLY[15:0]=0x100. And then, the users must set the slave timer's SSYNC bit in the TIMER1_SYNC register to '1'.

Additionally, because the TIMER1 (slave) operates as synchronized to the TIMER0's (master's) start signal, it runs even if the TEN bit in the TIMERn_CR2 register is set to '0'.

The Start SYNC can be used in cascaded structure, but the Clear SYNC cannot be used in cascaded structure.

9.3.10.2 Timer Synchronization Operation Characteristics and Precautions

1. To use SSYNC, the TEN bit in the TIMERn_CR2 register must be set to '0' for the slave timer. (This synchronizes to the initial operation and can create a sync-delay by cascading multiple timers.)
2. To use CSYNC, the TEN bit in the TIMERn_CR2 register must be set to '1' for the slave timer. (This clears the slave timer counter using the clear signal of the master timer, and timers cannot be cascaded.)
3. The SSYNC is a signal that is generated once at the initial stage, and the CSYNC is a signal that is generated at every cycle.
4. If the SSYNC and CSYNC are set at the same time, the timer operates based on the SSYNC at the initial operation, and afterwards, the timer counter is cleared by the CSYNC.

Figure 77 and Figure 78 show the operation waveform of the SSYNC and CSYNC signals in cascaded structure. The timers in the figures below are cascaded in the order of TIMER0 → TIMER1 → TIMER3.

Figure 77. SSYNC Operation Example

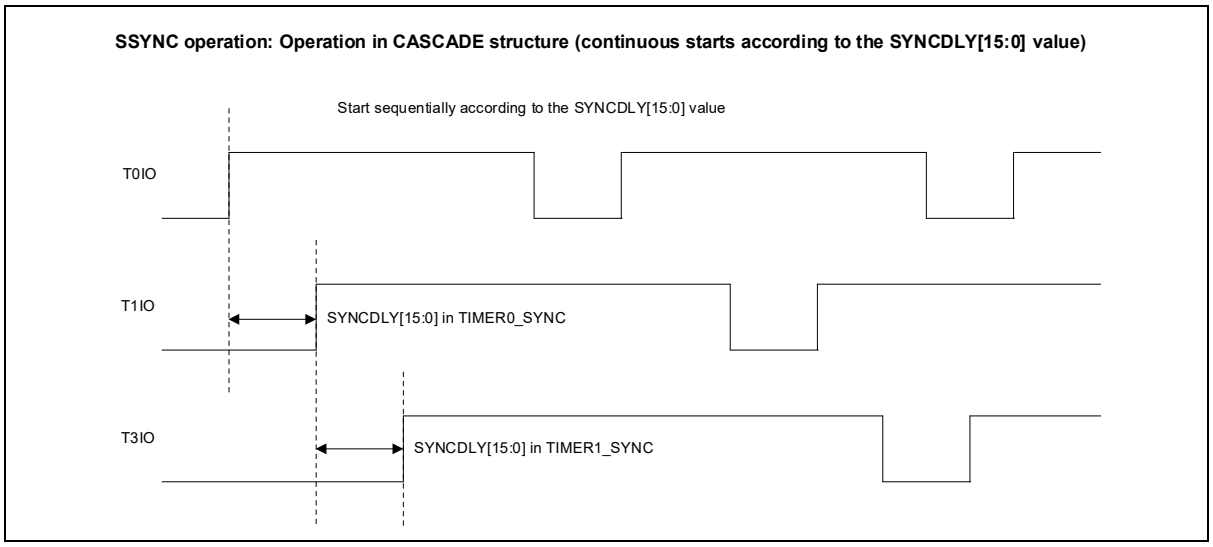
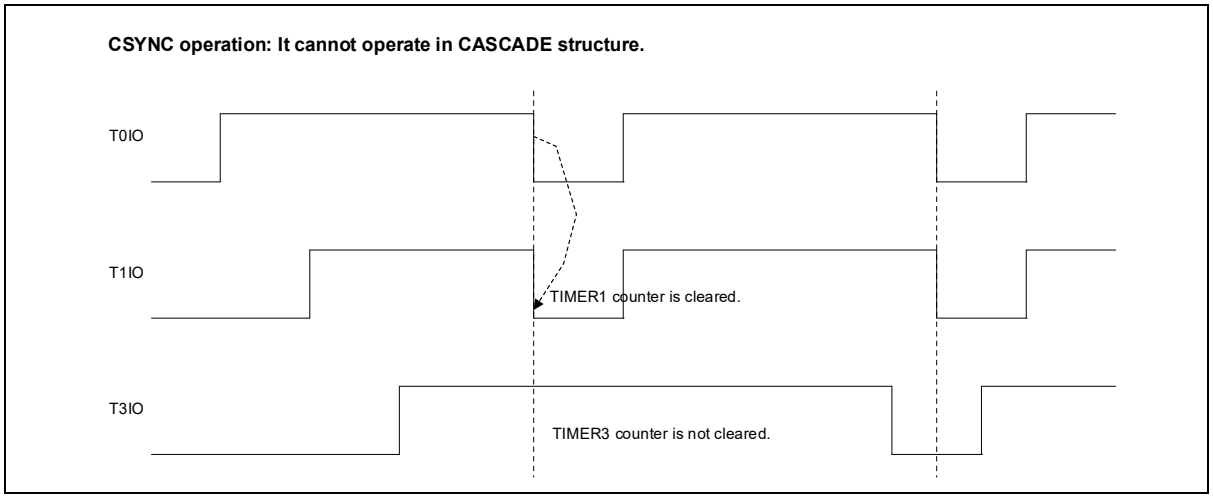


Figure 78. CSYNC Operation Example



9.3.10.3 Register Setting Sequence for the Synchronous Function Operation

1. To enable system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the Peripheral Clock Enable Register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write '1' to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
9. To use the synchronization function, write '1' to the SSYNC bit in the TIMERn_SYNC register or the CSYNC bit in the TIMERn_SYNC register (n = 0 to 9).
    A. When using a synchronized counter start mode, set the SSYNC bit in the TIMERn_SYNC register.
    B. When using a synchronized counter clear mode, set the CSYNC bit in the TIMERn_SYNC register.
10. To use a sync delay, enter the corresponding set value in the SYNCDLY[15:0] bits in the TIMERn_SYNC register.
11. Repeat steps 1 through 10 above to set all timers that use the synchronization function. And write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
12. For the timer operating as the master timer, enter the corresponding set value in the TnSYNCB bit in the TIMERn_SYNC register to select the slave timer (n = 0 to 9).
13. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
14. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer that operates as the Master.

9.3.11 Direction Bit Output

Since the timer outputs through a function pin of a GPIO port, users must set the GPIO port as a function pin. When the GPIO port is set to function pin mode, it operates as an input port or an output port depending on the timer operation mode.

Users can set the timer output value by configuring the IOSEL, OUTPOL, and STARTLVL bits in the TIMERn_CR1 register. Setting the IOSEL bit determines the timer input/output pin and setting the OUTPOL bit selects the polarity between the normal output and inversion output. Setting the STARTLVL

bit allows users to select the initial output value and select whether to set the start output value to Low or High.

9.3.12 Debug Mode

When the microcontroller enters debug mode, the timer counter continues to work. Therefore, the outputs (TnIO, n = 0 to 9) are enabled, although the core halted, when the TEN bit in the TIMERn_CR2 register is set. When the TEN bit in the TIMERn_CR2 register is cleared, the outputs (TnIO, n = 0 to 9) are disabled.

9.3.13 Interrupts

The TIMERn can generate interrupts, as shown in Table 84.

Table 84. Interrupt Requests

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
Overflow interrupt	OVF in TIMERn_SR	OVIE in TIMERn_IER	Writing a '1' to OVF in TIMERn_SR
GRB match interrupt	MFB in TIMERn_SR	MBIE in TIMERn_IER	Writing a '1' to MFB in TIMERn_SR
GRA match interrupt	MFA in TIMERn_SR	MAIE in TIMERn_IER	Writing a '1' to MFA in TIMERn_SR

NOTE:

1. When using overflow interrupt, set the TIMERn_GRA value and TIMERn_GRB value to zero. (Especially if the TIMERn_GRB value is not set to zero, OVF in TIMERn_SR will not be generated.)

9.4 Timer Registers

The base addresses of the 16-bit timers are described in the followings:

Table 85. Base Address of 16-bit Timer

Name	Base Address
TIMER0	0x4000_3000
TIMER1	0x4000_3040
TIMER2	0x4000_3080
TIMER3	0x4000_30C0
TIMER4	0x4000_3100
TIMER5	0x4000_3140
TIMER6	0x4000_3180
TIMER7	0x4000_31C0
TIMER8	0x4000_3200
TIMER9	0x4000_3240

Table 86. 16-bit Timer Register Map

Name	Offset	Type	Description	Reset Value	Ref.
TIMERn_CR1	0x0000	RW	Timer n Control Register 1	0x0000_0000	9.4.1
TIMERn_CR2	0x0004	RW	Timer n Control Register 2	0x0000_0000	9.4.2
TIMERn_PRS	0x0008	RW	Timer n Prescaler Register	0x0000_0000	9.4.3
TIMERn_GRA	0x000C	RW	Timer n General Data Register A	0x0000_0000	9.4.4
TIMERn_GRB	0x0010	RW	Timer n General Data Register B	0x0000_0000	9.4.5
TIMERn_CNT	0x0014	RW	Timer n Counter Register	0x0000_0000	9.4.6
TIMERn_SR	0x0018	RWC1	Timer n Status Register	0x0000_0000	9.4.7
TIMERn_IER	0x001C	RW	Timer n Interrupt Enable Register	0x0000_0000	9.4.8
TIMERn_TRGPNT	0x0020	RW	Timer n Trigger Point Register	0x0000_0000	9.4.9
TIMERn_SYNC	0x0024	RW	Timer n Sync Configuration Register	0x0000_0000	9.4.10

NOTE:

1. n = 0 to 9.

9.4.1 TIMERn_CR1: Timer/Counter n Control Register 1

The TIMERn_CR1 is a 32-bit register. The timer module must be set appropriately before running the timer. After the use of the timer is specified, the timer can be set in TIMERn_CR1. After this register is set, users can enable or disable the timer by setting the TIMERn_CR2.

TIMER0_CR1=0x4000_3000, TIMER1_CR1=0x4000_3040, TIMER2_CR1=0x4000_3080, TIMER3_CR1=0x4000_30C0, TIMER4_CR1=0x4000_3100, TIMER5_CR1=0x4000_3140, TIMER6_CR1=0x4000_3180, TIMER7_CR1=0x4000_31C0, TIMER8_CR1=0x4000_3200, TIMER9_CR1=0x4000_3240

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
Reserved		TRGMOD[1:0]	UAO	OUTPOL	IOSEL	Reserved	ADCTRGEN	STARTLVL	CKSEL[2:0]	CLRMOD[1:0]	MODE[1:0]
-		00	0	0	0	-	0	0	000	00	00
-		RW	RW	RW	RW	-	RW	RW	RW	RW	RW

15 14	TRGMOD[1:0]	ADC trigger mode selection	
		0X	Selects TIMERn_GRA value trigger mode (normal mode).
		10	Selects TIMERn_TRGPNT value trigger mode.
		11	Triggers both TIMERn_GRA and TIMERn_TRGPNT values.
13	UAO	GRA / GRB update mode selection	
		0	The value written to TIMERn_GRA or TIMERn_GRB is applied after the current period.
		1	The value written to TIMERn_GRA or TIMERn_GRB is applied in the current period.
12	OUTPOL	Timer output polarity	
		0	General output
		1	General output inversion
11	IOSEL	TnIO pin configuration (n = 0 to 9)	
		0	Sets the pin as an input port
		1	Sets the pin as an output port
8	ADCTRGEN	Use the timer as an ADC trigger source	
		0	Does not use the timer as an ADC trigger source.
		1	Uses the timer as an ADC trigger source.
7	STARTLVL	Starting output value in periodic / PWM / one-shot modes	
		0	Sets the starting output value to “L”.
		1	Sets the starting output value to “H”.
6 4	CKSEL[2:0]	Counter clock source selection	
		000	PCLK / 2
		001	PCLK / 4
		010	PCLK / 16
		011	PCLK / 64
		10X	EXT0 (SCU_MCCR3)
		11X	Input to pin TnIO (n = 0 to 9)
3 2	CLRMOD[1:0]	Clear mode selection in capture mode	
		00	Rising-edge clear mode
		01	Falling-edge clear mode
		10	Both-edge clear mode
		11	Disables clearing.

1 0	MODE[1:0]	Timer operation mode control	
		00	Normal periodic mode
		01	PWM mode
		10	One-shot mode
		11	Capture mode

9.4.2 TIMERn_CR2: Timer/Counter n Control Register 2

The TIMERn_CR2 is a 32-bit register used to control the timer's operation. Before starting the timer, users must set the TCLR in TIMERn_CR2 to '1' to clear the timer count register.

TIMER0_CR2=0x4000_3004, TIMER1_CR2=0x4000_3044, TIMER2_CR2=0x4000_3084, TIMER3_CR2=0x4000_30C4, TIMER4_CR2=0x4000_3104, TIMER5_CR2=0x4000_3144, TIMER6_CR2=0x4000_3184, TIMER7_CR2=0x4000_31C4, TIMER8_CR2=0x4000_3204, TIMER9_CR2=0x4000_3244

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
Reserved				TCLR	TEN
-				0	0
-				WO	RW

1	TCLR	Clearing of the timer count register[1]	
		0	No action
		1	Initializes the timer. Writing a '1' to TCLR clears the counter register (automatically cleared to '0' after operation).
0	TEN	Enable the timer	
		0	Stop the timer.
		1	Start the timer.

NOTE:

1. Users must set the TCLR bit to '1' before starting the timer to prevent a time error.

9.4.3 TIMERn_PRS: Timer/Counter n Prescaler Register

The TIMERn_PRS register is used to set the timer input frequency divider. It is 10 bits wide. Users can generate precise and varied timer base clocks by applying the prescaler to the timer clock source that has been selected in the TIMERn_CR1 register.

TIMER0_PRS=0x4000_3008, TIMER1_PRS=0x4000_3048, TIMER2_PRS=0x4000_3088, TIMER3_PRS=0x4000_30C8, TIMER4_PRS=0x4000_3108, TIMER5_PRS=0x4000_3148, TIMER6_PRS=0x4000_3188, TIMER7_PRS=0x4000_31C8, TIMER8_PRS=0x4000_3208, TIMER9_PRS=0x4000_3248

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
Reserved			PRS[9:0]
-			000
-			RW

9 0	PRS[9:0]	Prescaler value for the counter clock TCLK = TMCLK / (PRS[9:0] + 1) (TMCLK = Timer input clock selected by the setting of CKSEL[2:0] in TIMERn_CR1)

9.4.4 TIMERn_GRA: Timer/Counter n General Data Register A

The TIMERn_GRA register is 16 bits wide.

TIMER0_GRA=0x4000_300C, TIMER1_GRA=0x4000_304C, TIMER2_GRA=0x4000_308C, TIMER3_GRA=0x4000_30CC, TIMER4_GRA=0x4000_310C, TIMER5_GRA=0x4000_314C, TIMER6_GRA=0x4000_318C, TIMER7_GRA=0x4000_31CC, TIMER8_GRA=0x4000_320C, TIMER9_GRA=0x4000_324C

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
Reserved		GRA[15:0]	
-		0x0000	
-		RW	

15 0	GRA[15:0]	Timer general register A This register is used for different purposes depending on the operating mode. **Period / PWM / One-shot Modes:** The target count value is stored in the register. If the counter value matches the TIMERn_GRA value, the counter is cleared to restart or stop running. When the timer is cleared, TnIO outputs the STARTLVL in TIMERn_CR1 value. If the counter value matches the register's value, the MFA in TIMERn_SR is triggered. When the GRA interrupt is triggered or clearing is demanded, the counter value is copied to internal data buffer 0. In PWM mode, the GRA[15:0] value represents the duty value. **Capture mode:** - In rising-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. - In falling-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO.

9.4.5 TIMERn_GRB: Timer/Counter n General Data Register B

The TIMERn_GRB register is 16 bits wide.

TIMER0_GRB=0x4000_3010, TIMER1_GRB=0x4000_3050, TIMER2_GRB=0x4000_3090, TIMER3_GRB=0x4000_30D0, TIMER4_GRB=0x4000_3110, TIMER5_GRB=0x4000_3150, TIMER6_GRB=0x4000_3190, TIMER7_GRB=0x4000_31D0, TIMER8_GRB=0x4000_3210, TIMER9_GRB=0x4000_3250

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
Reserved		GRB[15:0]	
-		0x0000	
-		RW	

15 0	GRB[15:0]	Timer general register B This register is used for different purposes depending on the operating mode. **Periodic mode:** It should be set the same as the TIMERn_GRA **PWM / One-shot mode:** The target count value is stored in the register. If the counter value matches the register's value, the MFB in TIMERn_SR is triggered. In PWM mode, the TIMERn_GRB value represents the period value. **Capture mode:** In rising-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA) In falling-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA)

9.4.6 TIMERn_CNT: Timer/Counter n Count Register

The TIMERn_CNT register is 16 bits wide. The count is incremented based on the specified input clock. This register can be both read and written to.

TIMER0_CNT=0x4000_3014, TIMER1_CNT=0x4000_3054, TIMER2_CNT=0x4000_3094, TIMER3_CNT=0x4000_30D4, TIMER4_CNT=0x4000_3114, TIMER5_CNT=0x4000_3154, TIMER6_CNT=0x4000_3194, TIMER7_CNT=0x4000_31D4, TIMER8_CNT=0x4000_3214, TIMER9_CNT=0x4000_3254

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
Reserved		CNT[15:0]	
-		0x0000	
-		RW	

15 0	CNT[15:0]	Timer count value	
		R	Reads the current timer count.
		W	Sets the count value.

9.4.7 TIMERn_SR: Timer/Counter n Status Register

The TIMERn_SR register shows a timer module's status. This register is 8 bits wide.

TIMER0_SR=0x4000_3018, TIMER1_SR=0x4000_3058, TIMER2_SR=0x4000_3098, TIMER3_SR=0x4000_30D8, TIMER4_SR=0x4000_3118, TIMER5_SR=0x4000_3158, TIMER6_SR=0x4000_3198, TIMER7_SR=0x4000_31D8, TIMER8_SR=0x4000_3218, TIMER9_SR=0x4000_3258

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
Reserved				MFA	MFB	OVF
-				0	0	0
-				RWC1	RWC1	RWC1

2	MFA	TIMERn_GRA match flag	
		0	No TIMERn_GRA match has been detected.
		1	A TIMERn_GRA match has been flagged. (Writing a '1' to the bit clears the flag)
1	MFB	TIMERn_GRB match flag	
		0	No TIMERn_GRB match has been detected.
		1	A TIMERn_GRB match has been flagged. (Writing a '1' to the bit clears the flag)
0	OVF	Counter overflow flag	
		0	No overflow event has occurred.
		1	A counter overflow event has been flagged. (Writing a '1' to the bit clears the flag)

9.4.8 TIMERn_IER: Timer/Counter n Interrupt Enable Register

The TIMERn_IER register is 8 bits wide. Each status flag in the timer module can generate an interrupt. To trigger an interrupt, users must write '1' to the corresponding bit in this register.

TIMER0_IER=0x4000_301C, TIMER1_IER=0x4000_305C, TIMER2_IER=0x4000_309C, TIMER3_IER=0x4000_30DC, TIMER4_IER=0x4000_311C, TIMER5_IER=0x4000_315C, TIMER6_IER=0x4000_319C, TIMER7_IER=0x4000_31DC, TIMER8_IER=0x4000_321C, TIMER9_IER=0x4000_325C

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
Reserved				MAIE	MBIE	OVIE
-				0	0	0
-				RW	RW	RW

2	MAIE	Enable the TIMERn_GRA match interrupt
	0	Disables the TIMERn_GRA match interrupt.
	1	Enables the TIMERn_GRA match interrupt.
1	MBIE	Enable the TIMERn_GRB match interrupt
	0	Disables the TIMERn_GRB match interrupt.
	1	Enables the TIMERn_GRB match interrupt.
0	OVIE	Enable the counter overflow interrupt
	0	Disables the counter overflow interrupt.
	1	Enables the counter overflow interrupt.

9.4.9 TIMERn_TRGPNT: Timer/Counter n Trigger Point Register

The TIMERn_TRGPNT register is used to trigger the ADC at the desired timer counter point. This register is 16 bits wide.

TIMER0_TRGPNT=0x4000_3020, TIMER1_TRGPNT=0x4000_3060, TIMER2_TRGPNT=0x4000_30A0, TIMER3_TRGPNT=0x4000_30E0, TIMER4_TRGPNT=0x4000_3120, TIMER5_TRGPNT=0x4000_3160, TIMER6_TRGPNT=0x4000_31A0, TIMER7_TRGPNT=0x4000_31E0, TIMER8_TRGPNT=0x4000_3220, TIMER9_TRGPNT=0x4000_3260

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
Reserved		TRGPNT[15:0]	
-		0x0000	
-		RW	

15 0	TRGPNT[15:0]	Timer trigger point value
		Read: Reads the current trigger point value.
		Write: Sets the trigger point value.

NOTE:

1. The ADC value can be read when the timer count and TPGPNT[15:0] value are matched.

9.4.10 TIMERn_SYNC: Timer/Counter n Sync Configuration Register

The TIMERn_SYNC regiser is used to synchronize Group 1 (TIMER0 to TIMER4) and Group 2 (TIMER5 to TIMER9) to start or clear the respective timers. For example, it needs to set '15' to SYNCDLY[15:0] bits in the TIMER0_SYNC register and set '1' to T1SYNCB in TIMER0_SYNC register if user wants to start TIMER1 after 15 counts after starting TIMER0.

It has no effect to set '1' to T0SYNCB in TIMER0_SYNC register. And Syncing TIMER0 from TIMER1 after TIMER0 to TIMER1 has started syncing has no effect.

TIMER0_SYNC=0x4000_3024, TIMER1_SYNC=0x4000_3064, TIMER2_SYNC=0x4000_30A4 TIMER3_SYNC=0x4000_30E4, TIMER4_SYNC=0x4000_3124, TIMER5_SYNC=0x4000_3168, TIMER6_SYNC=0x4000_31A8, TIMER7_SYNC=0x4000_31E8, TIMER8_SYNC=0x4000_3228 TIMER9_SYNC=0x4000_3268

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
Reserved		T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Reserved		SSYNC	CSYNC	SYNCDLY[15:0]															
-		0	0	0	0	0	0	0	0	0	0	-		0	0	0x0000															
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW															

20+n	TnSYNCB (n=0 to 9)	Synchronization settings to the Timer n: As a master, the timer must select a slave timer that is enabled to synchronize. The slave timer must not select a master timer. (GROUP1: TIMER0 to TIMER4, GROUP2: TIMER5 to TIMER9)
		0 Disables synchronization.
		1 Enables synchronization.
17	SSYNC(1)	Synchronizes other synchronization timer to start counter. (The slave timer needs this configuration.)
		0 Disables synchronization.
		1 Counter start mode that is synchronized.
16	CSYNC	Synchronizes other synchronization timer to clear counter. (The slave timer needs this configuration.)
		0 Disables synchronization.
		1 Counter clear mode that is synchronized.
15 0	SYNCDLY[15:0](2)	SYNC delay start count value setting bit (based on the master counter)
		The SYNCDLY[15:0] is used to set the interval between the starting points of two synchronized timers. If this field is set to zero, the timers start simultaneously. (In this case, the two timers must not be running. If a value other than zero is written while the timers are running, the interval is applied from the next cycle onward.)

NOTES:

1. If synchronized slave timer is not enabled and SSYNC = 1, the synchronized slave timer starts counting when the master counter's count value reaches the value written to SYNCDLY[15:0].
2. If you enable synchronization with TIMER1 in TIMER0_SYNC and synchronization with TIMER2 in TIMER1_SYNC, TIMER0, TIMER1 and TIMER2 will start successively with the delay time set at SYNCDLY[15:0] between them.

9.4.11 Timer Register Map Summary

Table 87. TIMER Register Map Summary

Offset	Register name	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
0x00	TIMERn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRGMOD[1:0]		UAO	OUTPOL	IOSEL	Res.	Res.	ADCTRGEN	STARTLVL	CKSEL[2:0]			CLRMOD[1:0]		MODE[1:0]	
	Reset value																	0	0	0	0	0			0	0	0	0	0	0	0	0	0
0x04	TIMERn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCLR	TEN
	Reset value																															0	0
0x08	TIMERn_PRS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRS[9:0]									
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x0C	TIMERn_GRA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GRA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	TIMERn_GRB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GRB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	TIMERn_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIMERn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MFA	MFB	OVF
	Reset value																														0	0	0
0x1C	TIMERn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MAIE	MBIE	OVIE
	Reset value																														0	0	0
0x20	TIMERn_TRGPNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRGPNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	TIMERn_SYNC	Res.	Res.	T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Res.	Res.	SSYNC	CSYNC	SYNCDLY[15:0]															
	Reset value			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

10. Free-Run Timer (FRT)

10.1 FRT Introduction

The A34M420 has two built-in Free-Run Timer (FRT) that is 32-bit up count timer. The FRT can run with an overflow interrupt or a match interrupt according to their register setting and can remain active in DEEP-SLEEP STOP 1 mode.

10.2 FRT Main Features

The FRT of the A34M420 has the features shown below:

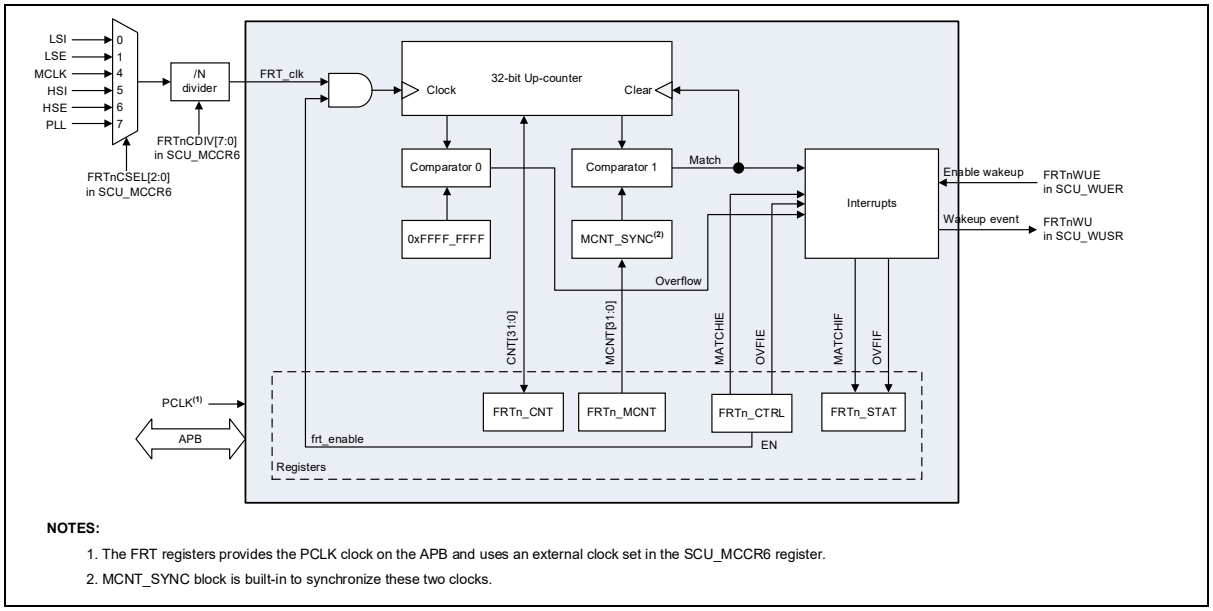
- Two FRT units
- 32-bit up count timer‘s two operations
  - Free-run timer mode operation
  - Periodic timer mode operation based on the setup period
- FRT overflow interrupt and match interrupt functions
- Selectable FRT input clocks
  - Input clock source can be selected using the SCU_MCCR6 register: LSI, LSE, MCLK, HSI, HSE, or PLL.

10.3 FRT Functional Description

10.3.1 FRT Block Diagram

Figure 79 shows a block diagram of the FRT.

Figure 79. FRT block diagram



10.3.2 FRT Clock Selection

The FRT can be enabled by configuring the SCU_PER1 and SCU_PCER1 registers that are peripheral enable register and peripheral clock enable register, respectively. Users must set the FRTn (n = 0 to 1) in the SCU_PER1 and SCU_PCER1 register to '1'. Until the clock is provided to the FRT module, it is not reset nor operates normally.

An external clock source can be set as a clock source for the FRT. The external clock source can be selected as the FRT's clock source when the FRTnCDIV[7:0] bits in the SCU_MCCR6 register is set to a certain value larger than zero and the FRTnCSEL[2:0] bits in the SCU_MCCR6 register is set to select the external clock source.

There is an important thing to remember. When the FRT main clock is set to the MCLK, the A34M420 cannot wake up from DEEP-SLEEP (STOP 1) mode. To enable the system to wake from Deep Sleep mode, the FRT main clock must be set to a clock other than the MCLK.

10.3.3 FRT Operation Mode

The MODE bit in the FRTn_CTRL register determines whether the FRT operates in Periodic Timer Mode or Free-run Timer Mode (n = 0 to 1).

When the FRT runs in Periodic Timer Mode, the MATCHIE bit in the FRTn_CTRL register determines whether to enable or disable the match interrupt. This bit does not affect the overflow interrupt.

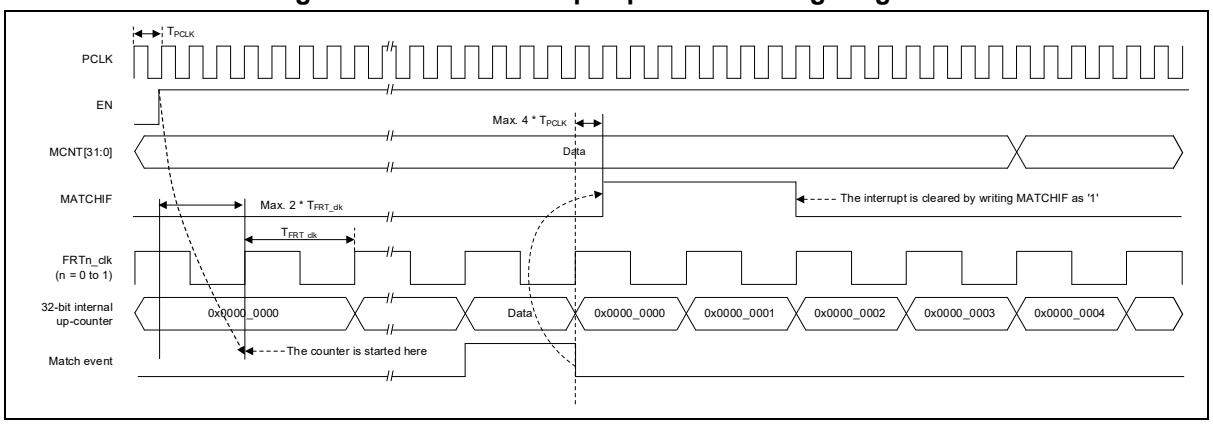
When the FRT runs in Free-run Timer Mode, the OVFIE bit in the FRTn_CTRL register determines whether to enable or disable the overflow interrupt. This bit does not affect the match interrupt.

10.3.3.1 Match Interrupt Operation

Figure 80 shows a timing diagram of the match interrupt operation. To enable the match interrupt, the MATCHIE bit in the FRTn_CTRL register must be set to ‘1’.

When the EN bit in the FRTn_CTRL register is set to ‘1’, the FRT counter starts counting. Once the counter value reaches the FRTn_MCNT value, the FRT counter is clear to zero and the interrupt and wake-up signal are triggered. An interrupt signal can be delayed by up to approximately four system clock (PCLK) pulses.

Figure 80. Match Interrupt Operation Timing Diagram



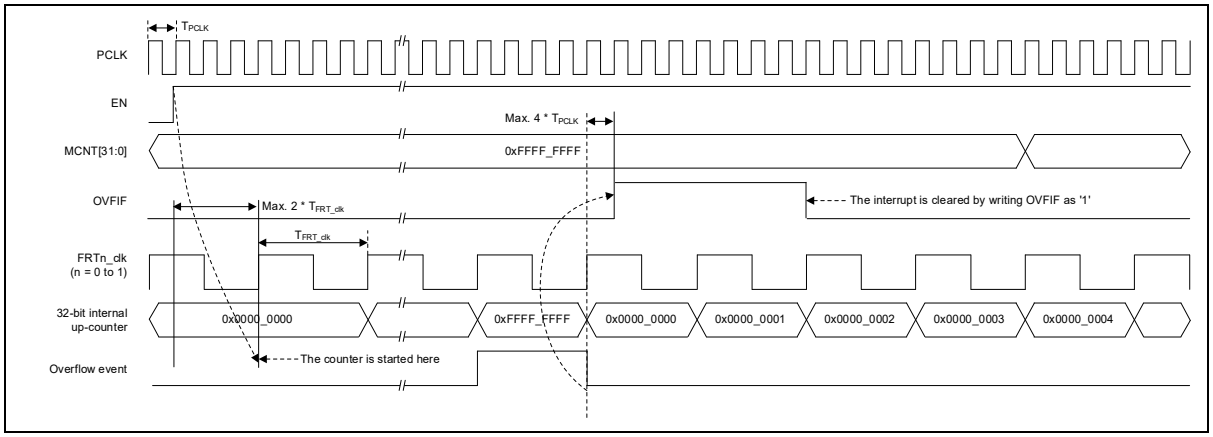
The following formula calculates the FRT’s count period in Periodic Timer Mode:

- Period = FRTn_clk period × (MCNT + 1)
- Match interrupt time = FRTn_clk period × (MCNT + 1)

10.3.3.2 Overflow Interrupt Operation

Figure 81 shows a timing diagram of the overflow interrupt operation. The overflow interrupt operates almost in the same way as the match interrupt. It is triggered when the FRT counter value matches 0xFFFF_FFFF.

Figure 81. Overflow Interrupt Operation Timing Diagram



10.3.3.3 Register Setting Procedure by FRT Operation Mode

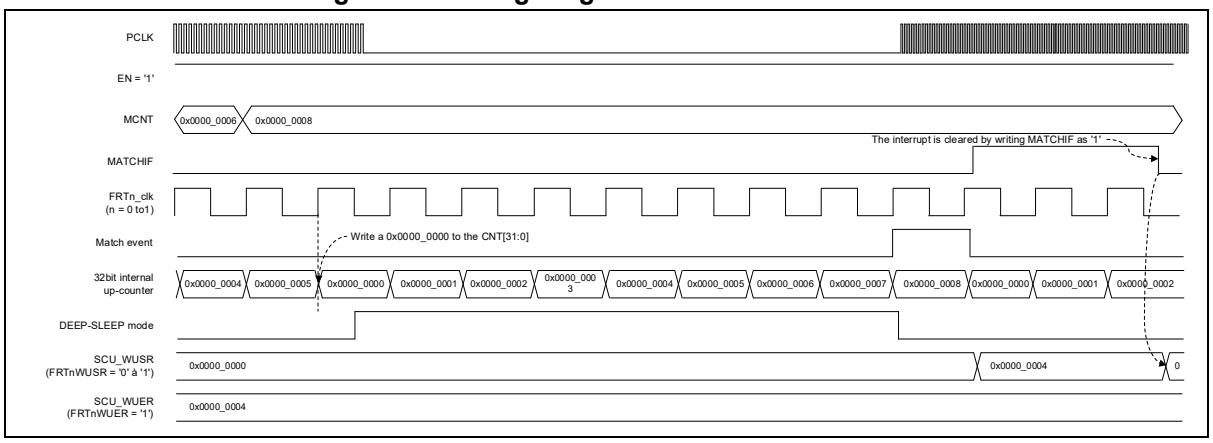
1. First, write the predefined value to the SCU_SYSTEN register. This is the system access enable register and writing predefined value to this register can enable users to access the system.
2. To enable the FRT module, set each corresponding bit of the SCU_PER1 and SCU_PCER1 registers to '1'. The SCU_PER1 and SCU_PCER1 registers are the peripheral enable register and peripheral clock enable register, respectively.
3. To select an external clock for the FRT module, set the FRTnCDIV[7:0] bits in the SCU_MCCR6 register to a value larger than '0'. Then set the FRTnCSEL[2:0] bits in the SCU_MCCR6 register to a value of the desired clock.
4. Set the MODE in FRTn_CTRL register to select a FRT operation mode:
   A. '0': Free-run Timer Mode
   B. '1': Match Interrupt Mode
5. To enable interrupts for each mode, set the corresponding bit of the FRTn_CTRL register to '1' depending on the mode that was selected in the previous step:
   A. OVFIE bit: FRT counter overflow interrupts are enabled in Free-run Timer Mode.
   B. MATCHIE bit: FRT counter match interrupts are enabled in Match Interrupt Mode.
6. In Match Interrupt Mode, use the FRTn_MCNT register to specify the match counter value.

   Match interrupt time = FRTn_clk period × (MCNT + 1)
7. Set the NVIC registers that are assigned for the FRT interrupts.

8. To initialize the FRT counter, write ‘0’ to the FRTn_CNT register.
9. To enable the FRT counter, set the EN in FRTn_CTRL register to ‘1’.
10. Read the FRTn_STAT register to check if the FRT interrupt event occurs. If the corresponding bit in the register is set, clear it by writing ‘1’ to the bit field.

10.3.4 FRT Low-Power Modes

Figure 82. Timing Diagram in Low-Power Mode



10.3.4.1 Description of the Steps of Low-Power Mode

1. To enable the signal to be used as a wake-up source, set the FRTnWUE bit in the SCU_WUER register to ‘1’.
2. Set the MCNT[31:0] bits in the FRTn_MCNT register, which triggers the match interrupt. (In the diagram above, it is set to 8 as an example.)
3. Clear the CNT[31:0] bits in the FRTn_CNT register to zero, which initializes 32-bit internal up-counter.
4. Enter low-power mode (DEEP-SLEEP mode)
5. When 32-bit internal up-counter reaches the MCNT[31:0] bits in the FRTn_MCNT register, the Match Event is generated. Then MACHIF flag in the FRTn_STAT register and FRTnWU flag in SCU_WUSR register are triggered after delay of up to four system clocks.
6. And writing as ‘1’ to MATHIF bit in the FRTn_STAT register to clear triggered flags. At this time the FRTnWU flag in the SCU_WUSR register is also cleared.

10.4 FRT Registers

The base addresses of the FRTs and register map are described in the followings:

Table 88. Base Address of FRT

Name	Base Address
FRT0	0x4000_0600
FRT1	0x4000_0700

Table 89. FRT Register Map

Name	Offset	Type	Description	Reset Value	Reference
FRTn_CTRL	0x0000	RW	FRT n Control Register	0x0000_0000	10.4.1
FRTn_MCNT	0x0004	RW	FRT n Match Counter Register	0xFFFF_FFFF	10.4.2
FRTn_CNT	0x0008	RW	FRT n Counter Register	0x0000_0000	10.4.3
FRTn_STAT	0x000C	RWC1	FRT n Status Register	0x0000_0000	10.4.4

NOTE:

1. n = 0 and 1.

10.4.1 FRTn_CTRL: FRT n Control Register

The FRT_CTRL register has control bits to the FRT operation. This register is 32 bits wide.

FRT0_CTRL=0x4000_0600, FRT1_CTRL=0x4000_0700

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
Reserved			OVFIE	MATCHIE	Reserved	MODE	EN
-			0	0	-	0	0
-			RW	RW	-	RW	RW

9	OVFIE	Enable the FRT counter overflow interrupt in Free-run timer mode (MODE = '0')	
		0	Disable the overflow interrupt
		1	Enable the overflow interrupt
8	MATCHIE	Enable the FRT counter match interrupt in Periodic timer mode (MODE = '1')	
		0	Disable the match interrupt
		1	Enable the match interrupt
1	MODE	FRT mode selection	
		0	Free-run timer mode
		1	Periodic timer mode
0	EN	Enable the FRT	
		0	Disable
		1	Enable

10.4.2 FRTn_MCNT: FRT n Match Counter Register

The FRTn_MCNT register is a 32-bit register. It is used to specify the period value when the FRT operates in Periodic timer mode. In Periodic Timer Mode, the value of the FRTn_CNT register counts up until it reaches the FRTn_MCNT value, which triggers the match interrupt if it has been set enabled.

FRT0_MCNT=0x4000_0604, FRT1_MCNT=0x4000_0704

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
MCNT[31:0]			
0xFFFF_FFFF			
RW			

31 0	MCNT[31:0]	FRT's match counter value	
		The match interrupt is triggered when the current counter value reaches the set match counter value. Match interrupt time = FRTn_clk period × (MCNT + 1)	

10.4.3 FRTn_CNT: FRT n Counter Register

The FRTn_CNT register is a 32-bit register that shows the FRT's current counting value. The register can be both read and written as an up-count timer whose count value is incremented.

If the EN bit in the FRTn_CTRL register is set to '1', this register can be read and written.

FRT0_CNT=0x4000_0608, FRT1_CNT=0x4000_0708

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[31:0]			
0x0000_0000			
RW			

31 0	CNT[31:0]	FRT count data Represents the current count value. (Only the initializing value 0x0 can be written to the bit field.)

10.4.4 FRTn_STAT: FRT n Status Register

The FRTn_STAT register is a 32-bit register that shows the FRTn status. It can be read the status of each of FRTn events and can be clear the status bit by writing '1' to the corresponding bit when the event occurs.

FRT0_STAT=0x4000_060C, FRT1_STAT=0x4000_070C

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
Reserved			OVFIE	MATCHIF	Reserved
-			0	0	-
-			RW	RW	-

9	OVFIF	FRT counter overflow interrupt flag	
		0	The overflow interrupt event has not occurred.
		1	The overflow interrupt event has occurred. (Writing a '1' to the bit clears this flag)
8	MATCHIF	FRT counter match interrupt flag	
		0	The match interrupt event has not occurred.
		1	The match interrupt event has occurred. (Writing a '1' to the bit clears this flag)

10.4.5 FRT Register Map Summary

Table 90. FRT Register Map Summary

Offset	Register name	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
0x00	FRTn_CTRL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVFIE	MATCHIE	Res	Res	Res	Res	Res	Res	MODE	EN
	Reset value																							0	0							0	0
0x04	FRTn_MCNT	MCNT[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x08	FRTn_CNT	CNT[31:0]																															
	Reset value	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
0x0C	FRTn_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVFIF	MATCHIF	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value																							0	0								

11. Universal Asynchronous Receiver/Transmitter (UART)

11.1 UART Introduction

The A34M420 has six channels UART module. This built-in UART module transmits and receives data according to user-specified settings and reads the current UART status.

The UART status information includes type and conditions of the current UART transmission / reception process and can be used to check for errors (parity, overrun, framing, or break interrupts) that may occur during data reception.

Each UART channel has a programmable baud-rate generator that generates an internal clock for the corresponding UART unit by dividing the prescaled clock using a baud-rate divisor (ranging from 1 to 65,535) and then dividing the result by 16.

Users can program interrupts that can control the UART communication.

Using Direct Memory Access (DMA), users can perform high-speed data communication.

11.2 UART Main Features

The UART of A34M420 features the followings:

- Six 16450 full-duplex asynchronous serial communication ports
- Configurable standard asynchronous communication bits (start, stop, and parity) are supported.
- User-programmable serial communication is available.
  - 5, 6, 7, or 8 data bits
  - Even, odd, or no parity generation and checking
  - 1-, 1.5-, or 2-stop bit generation and checking
- A 16-bit baud-rate generator and an 8-bit fractional compensator are included.
- Single or multi-sampling for start and data bits
- Delay between data frames is possible.
- Separate signal polarity control for transmission and reception
- Transfer status indicated by the interrupt ID and line status registers
  - Stop bit error detection
  - Display of information about the current status
  - Line break generation and checking
  - Receive error diagnosis
- A priority-based interrupt system
- Continuous communication is possible using DMA.
- Received / transmitted bytes are buffered in reserved SRAM using centralized DMA.

11.2.1 UART Implementation

Table 91 describes the UART implementation on A34M420.

Table 91. UART Features

UART Modes / Features	UART
Burst transfer using DMA	O
UART data length	5, 6, 7 and 8 bits

11.3 UART Functional Description

The UART module is compatible with 16450 UART. It provides DMA channels and a fractional compensation logic to obtain baud-rates. Because it does not include a FIFO block, data transfers are performed either interactively or with DMA support.

The DMA operates as follows:

A UART can be linked with two DMA channels, of which one is responsible for Tx transfers and the other is responsible for Rx transfers. Each channel has a 32-bit memory address register and a 32-bit peripheral address register. Users must set these two registers before enabling the DMA.

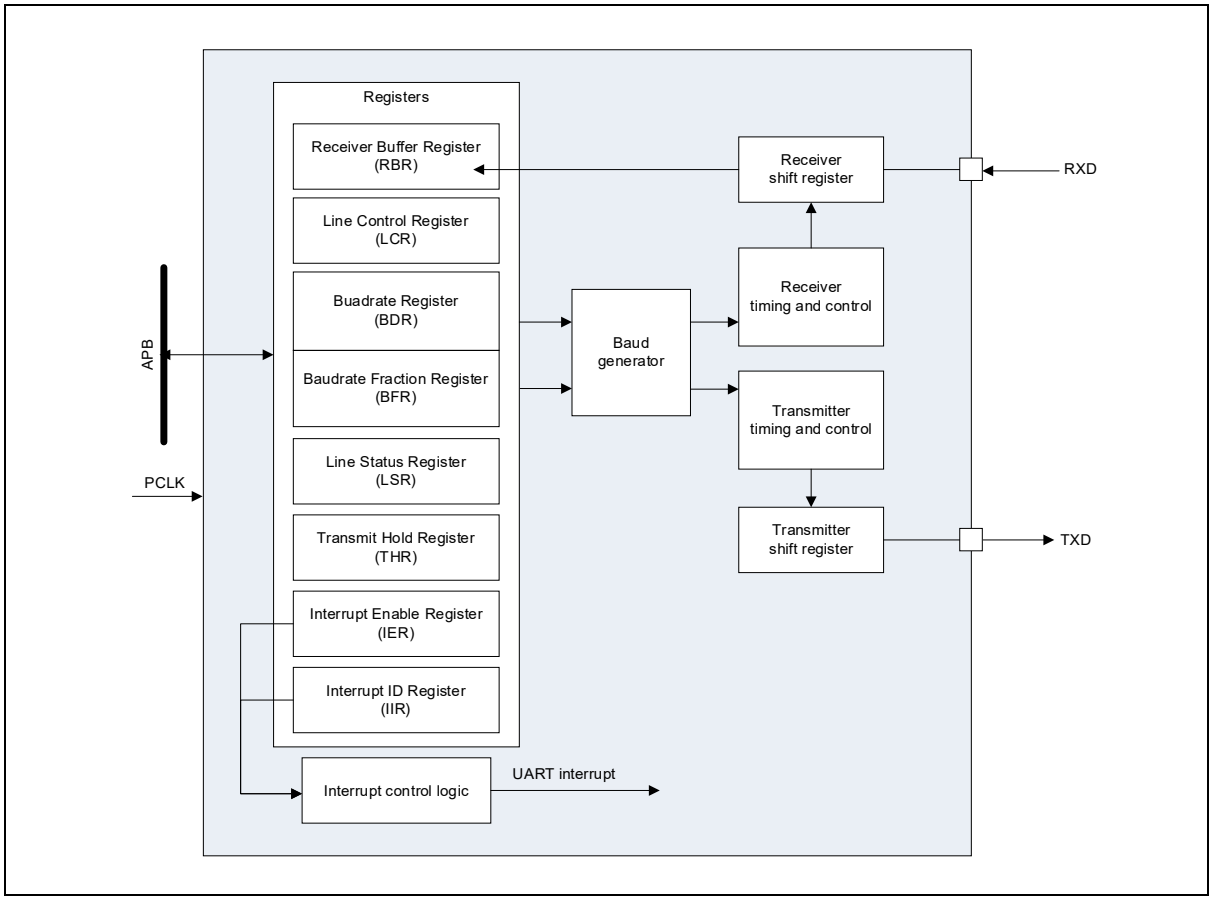
The memory address for the Rx transfers is the target address, whereas the address for Tx transfers is the source address.

The TRANSCNT[11:0] bit in the DMAn_CR register records the number of transfers. The counter decreases by 1 every time a single-ended transfer is completed. When the counter reaches zero, a DMA complete flag is sent to the UART control block. Then, the corresponding interrupt is flagged if the interrupt has been set enabled.

11.3.1 UART Block Diagram

Figure 83 shows a block diagram of the UART.

Figure 83. UART Block Diagram



11.3.2 UART Pins and Signals

Table 92 describes external pins assigned for UART.

Table 92. Pin Assignment of UART: External Pins

Pin Name	Type	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
TXD0	O	UART channel 0 transmit output	O	O	O
RXD0	I	UART channel 0 receive input	O	O	O
TXD1	O	UART channel 1 transmit output	O	O	O
RXD1	I	UART channel 1 receive input	O	O	O
TXD2	O	UART channel 2 transmit output	O	O	N/A
RXD2	I	UART channel 2 receive input	O	O	N/A
TXD3	O	UART channel 3 transmit output	O	O	O
RXD3	I	UART channel 3 receive input	O	O	O
TXD4	O	UART channel 4 transmit output	O	O	N/A
RXD4	I	UART channel 4 receive input	O	O	N/A
TXD5	O	UART channel 5 transmit output	O	O	N/A
RXD5	I	UART channel 5 receive input	O	O	N/A

11.3.2.1 UART Bidirectional Communications

UART bidirectional communication requires at least two pins such as Receive Data In (RXD) and Transmit Data Out (TXD).

The RXD is a serial data input. Oversampling techniques are used for data recovery. They discriminate between valid incoming data and noise.

The TXD is a serial data output. When the UART module is enabled and no data needs to be transmitted, the TXD pin is high state.

11.3.3 UART Character Description

The data length can be set to 5, 6, 7 or 8 bits, by configuring the DLEN[1:0] in the UARTn_LCR register as described below:

- For 5-bit character length: DLEN[1:0] = '00' (refer to Figure 88)
- For 6-bit character length: DLEN[1:0] = '01' (refer to Figure 87)
- For 7-bit character length: DLEN[1:0] = '10' (refer to Figure 86)
- For 8-bit character length: DLEN[1:0] = '11' (refer to Figure 85)

By default, the signal (TXD or RXD) is in low state during the start bit. It is in high state during the stop bit.

Figure 84 shows that the signal value (TXD and RXD) can be inverted, separately for each signal, through UARTn_DCR's data inversion control.

Figure 84. Data Inversion Control Diagram

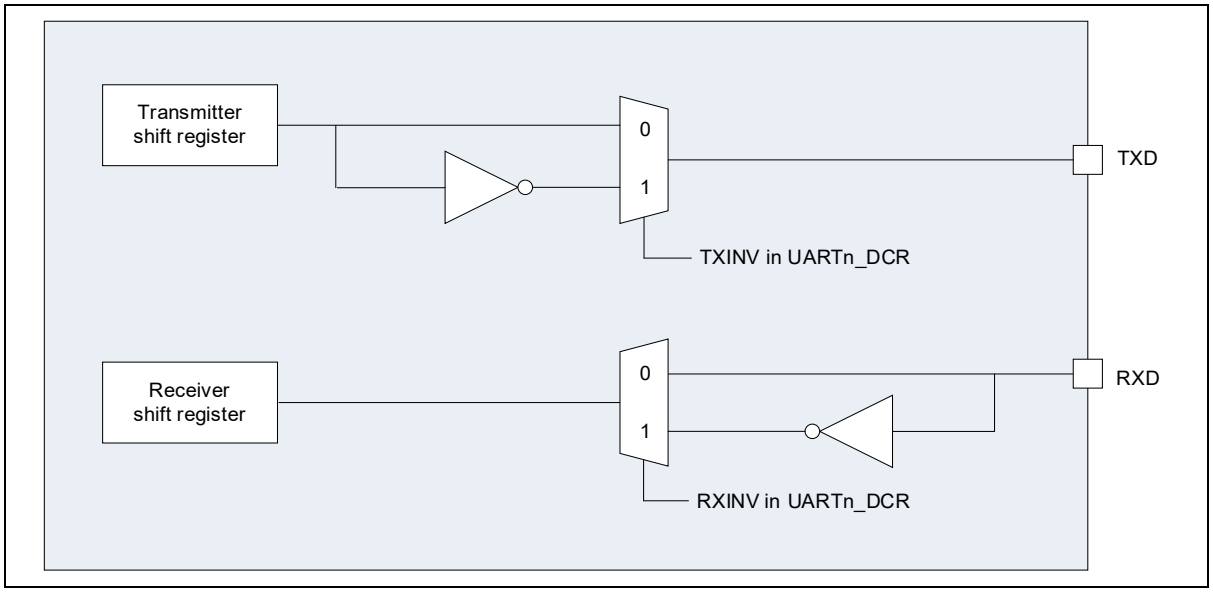


Figure 85 shows a timing diagram of data length programming. A Break condition is recognized on receiving ‘0’s for longer than the time taken to receive the entire data word (i.e., the sum of the start, data, parity, and stop bits). The transmission and reception clocks are generated when the enable bit is set for the UART peripheral and clock, respectively.

Figure 85. Timing Diagram of 8-bit Data Length

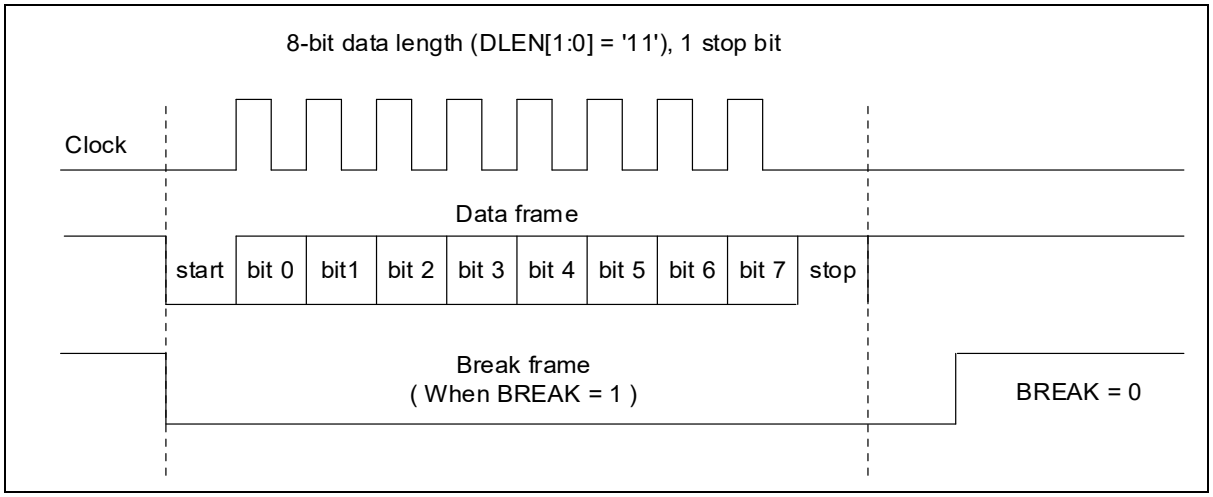


Figure 86. Timing Diagram of 7-bit Data Length

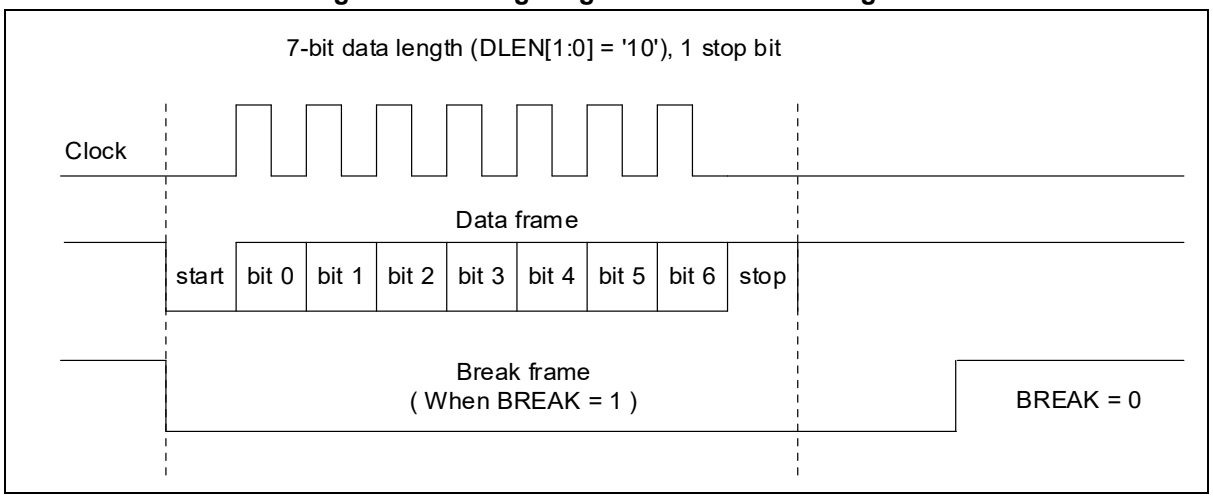


Figure 87. Timing Diagram of 6-bit Data Length

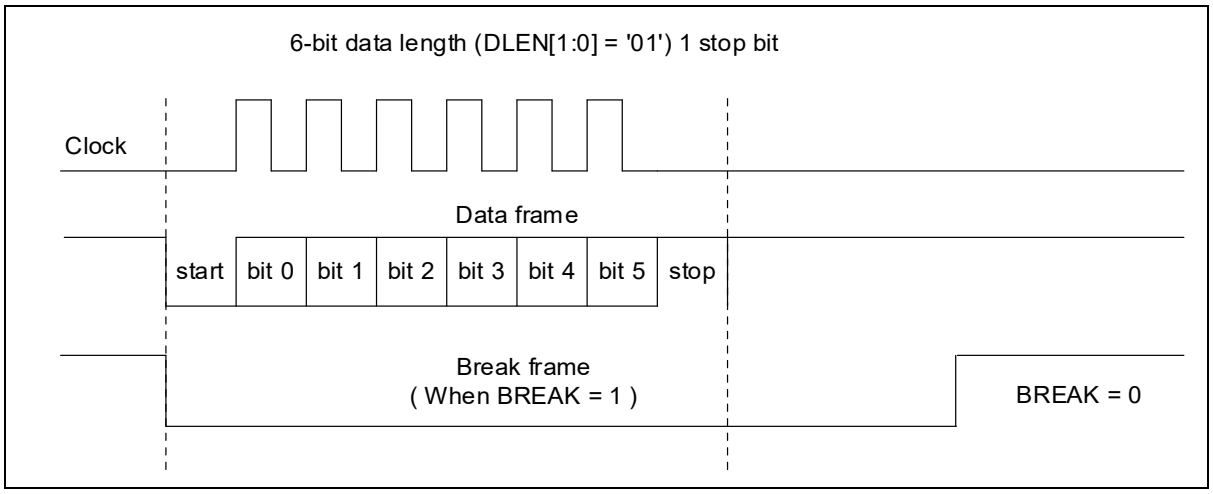
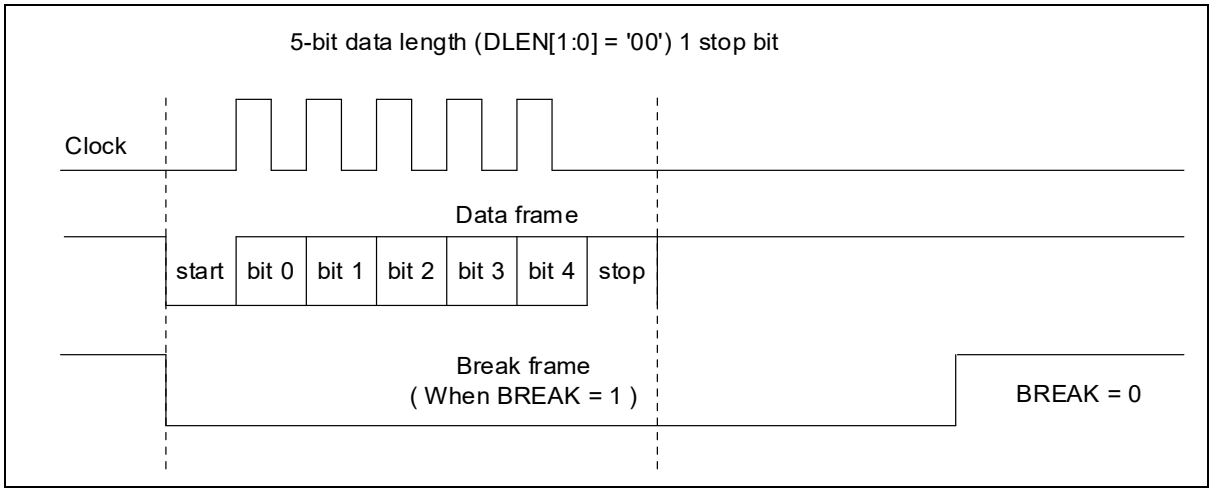


Figure 88. Timing Diagram of 5-bit Data Length



11.3.4 UART Transmitter

11.3.4.1 Data Format

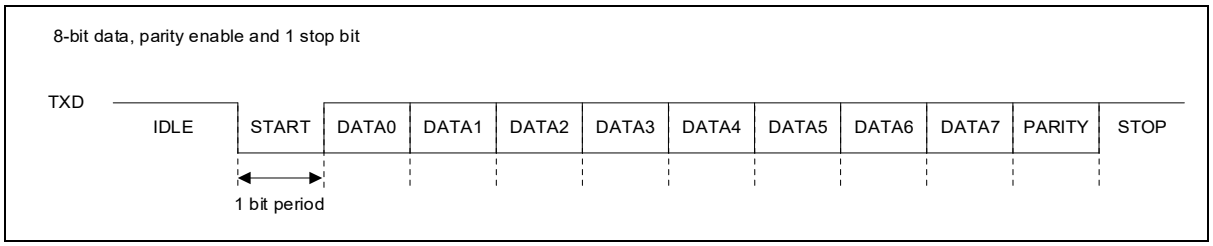
The UART transmitter is in charge of transmitting data. For each data word, the start bit, data bits, optional bit, and stop bit are shifted serially in and out of the corresponding register, one bit at a time at the Least Significant Bit (LSB).

The number of data bits can be determined by setting the DLEN[1:0] bits in the UARTn_LCR register. The UART transmitter sends data words of either 5, 6, 7, or 8 bits, based on set value of the DLEN[1:0] bits in the UARTn_LCR register.

The parity bit type can be determined by setting the PARITY and PEN bits in the UARTn_LCR register. If even parity is selected, the parity bit is determined by the bit sum of all the data bits. For odd parity, the parity bit takes the opposite value to that of even parity. The number of stop bits is defined by configuring the STOPBIT bit in the UARTn_LCR register.

Figure 89 shows an example of a transmitted data frame.

Figure 89. Example of a Transmitted Data Frame



NOTES:

1. The UART peripheral and clock must be enabled to activate the UART transmitter function.
2. The data in the Transmit Shift Register is output on the TXD pin.

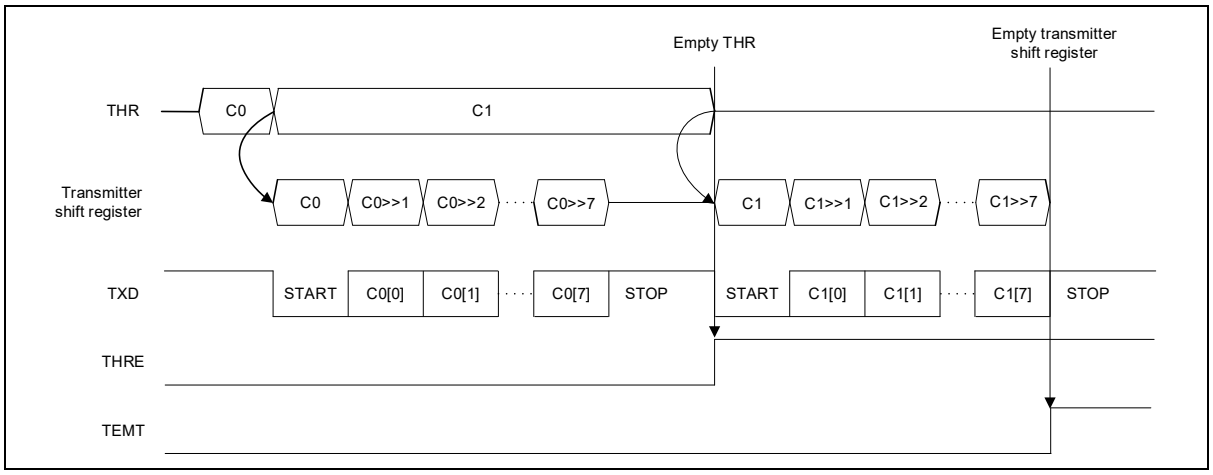
11.3.4.2 Transmit Interrupt

During UART transmission, several types of interrupt flags are used.

When the Transmit data Hold Register (UARTn_THR) is empty, the THRE interrupt flag is set to '1'; when the Transmit Shift Register is empty, the TEMT interrupt flag is set to '1'.

Users can select the best interrupt timing for their applications, by referring Figure 90.

Figure 90. Transmit Interrupt Timing Diagram



11.3.4.3 Character Transmission

During UART transmission, data is shifted out at the Least Significant Bit first on the TXD pin. In this mode, the UART transmission register consists of a buffer (UARTn_THR) between the internal bus and the Transmit Shift Register.(1)

Every character is preceded by a start bit which corresponds to a low logic level for one bit period. The character is terminated by a configurable number of stop bits. The number of stop bits can be configured to 1, 1.5 or 2, depending on the STOPBIT bit status.

NOTE:

1. The UART peripheral and clock enable bits must be set in the SCU_PER2 and SCU_PCER2 registers before writing the data to be transmitted.

The UART peripheral and clock enable bits must not be reset during data transmission. Resetting the UART peripheral and clock enable bits during the transmission corrupts the data on the TXD pin as the baud-rate counter gets frozen. The current data being transmitted are then lost.

11.3.4.4 Configurable Stop Bits

The number of stop bits to be transmitted with every character can be programmed in the STOPBIT bit in the UARTn_LCR register.

- 1 stop bit: Default number of stop bits.
- 1.5 / 2 stop bits: Number of stop bits that users can configure.

A break transmission is possible to transmit long breaks (break of length greater than the time of entire data word – i.e., The sum of the start bit, data bits, parity bit, and stop bits).

11.3.4.5 Character Transmission Procedure

Users can use DMA, Interrupts, and Polling to transmit the UART communication packets following the procedure below:

1. Entering data in the UARTn_THR register starts UART communication.
2. Since data in the UARTn_THR register is transferred to the Transmit Shift Register for the communication, updating the UARTn_THR register does not affect the data in communication.
    A. If the UARTn_THR register is updated, the Transmit Shift Register is automatically updated with the data in the UARTn_THR register after entire data in the Transmit Shift Register is output.
    B. The Transmit Shift Register continues to output its updated data for the communication.
3. DMA, Interrupts, and Polling can be used to update the UARTn_THR register after confirming the communication is completed.
    A. With the DMA Tx enabled,
        i. The UART block generates the DMA Request signal immediately after data in the UARTn_THR register is transferred to the Transmit Shift Register.
        ii. The DMA updates the memory data with the UARTn_THR register. When this operation is repeated as many times as set in the DMA counter, the DMA Done signal is generated, the IID[2:0] bits is set to ‘101’, and the DMA Tx interrupt is generated.
    B. Checking if the TXE interrupt is generated or verifying that the IID[2:0] value is ‘001’,
        i. The TXE interrupt is generated when the communication is completed after the UARTn_THR register transfers data and the Transmit Shift Register outputs data. By checking the generation of the TXE interrupt, users can update the UARTn_THR register to restart the communication.
        ii. If the IID[2:0] bits in the UARTn_IIR register is set to ‘001’ without the generation of the TXE interrupt, it means that data in the UARTn_THR register has been transferred to the Transmit Shift Register.
        iii. If the UARTn_THR register is not updated with new data, the TXE interrupt is generated when the Transmit Shift Register completes outputting its current data.
4. Polling the TEMT and THRE bits in the UARTn_LSR register,

A. The TEMT bit implies that the Transmit Shift Register is empty.

B. The THRE bit implies that the UARTn_THR register is empty.

(The UARTn_THR register can be updated only when TEMT and THRE bits are enabled.)

11.3.4.6 Single Byte Communication

When data is stored in the UARTn_THR (Transmitter Hold Register) register for transmission, it is automatically transferred to the Transmit Shift Register.

Before the Transmit Shift Register transfers data, the Transmitter outputs a start bit. After the start bit, the Transmit Shift Register shifts one bit at a time to the output of the Transmitter.

After data in the Transmit Shift Register is output by the bit size set in the DLEN[1:0] bits in the UARTn_LCR register, the Transmitter outputs a stop bit to complete the communication.

During the data transmission or after the communication is completed, if data in the UARTn_THR register is updated, the Transmit Shift Register that output data will be updated with new data to start next communication.

The Transmitter, which outputs a stop bit, restarts communication outputting a start bit.

11.3.4.7 Break Condition

Setting the BREAK(1) bit in the UARTn_LCR register transmits a long break (Break of length greater than the time of entire data word - i.e., The sum of the start, data, parity, and stop bits). Whereas writing a '0' to the bit disables the break condition.

NOTE:

1. If a '1' is written to the BREAK bit, a break is sent on the Tx line immediately. Therefore, users must confirm the completion of Tx transmission and set the BREAK bit.

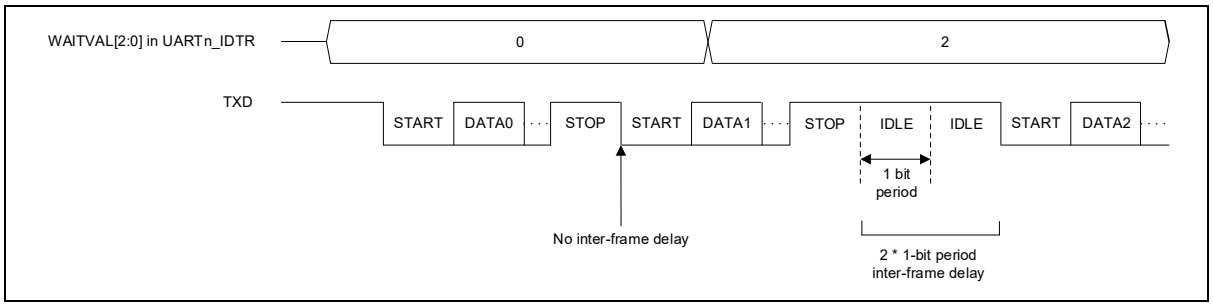
11.3.4.8 Inter-Frame Delay for Data Transmission

The inter-frame delay functionality creates an idle state between two characters on the TXD line. The length of the idle state can be determined by configuring the WAITVAL[2:0] bits in the UARTn_IDTR.

If the WAITVAL[2:0] bits is set to '000', a delay does not occur; otherwise, the transmitter waits for the number of bit periods defined in the WAITVAL[2:0] after transmitting the STOP part of TXD.

Figure 91 shows an example format of the transmit data.

Figure 91. Example of Transfer Data Format



11.3.5 UART Receiver

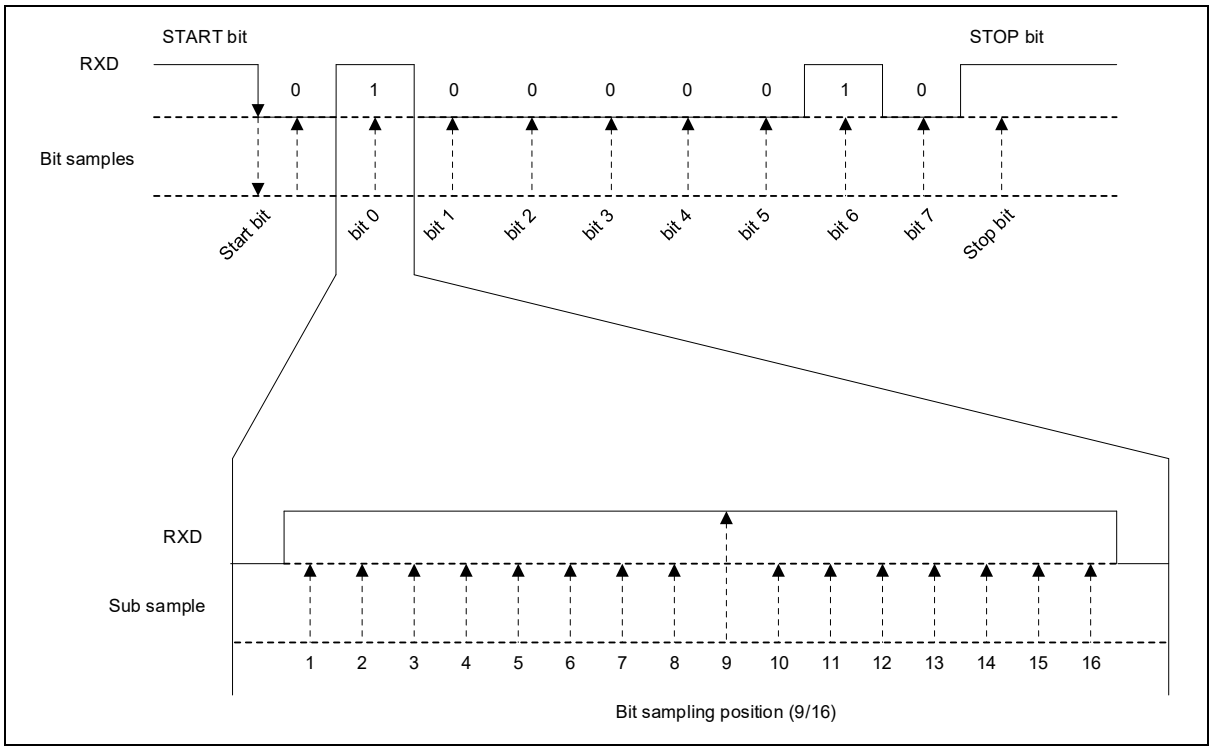
11.3.5.1 Receive Data Sampling Timing

Timing of the UART operation is described below:

Once a falling edge is detected in the receive line, the UART determines that a start bit is being received. The UART oversamples the received data 16 times per bit beginning from the start bit. Among the 16 samples, the value at the 9th clock pulse is determined to represent the value of the bit.

The UART receiver can receive data words of either 5, 6, 7, or 8 bits, depending on the DLEN[1:0] bits in the UARTn_LCR register.

Figure 92. Sampling Timing of UART Receiver



To enhance protection against external glitch noise, it is recommended to enable port debouncing[1] in the PCU module.

NOTE:

1. To use the port debounce function, the debounce clock in the SCU_MCCR4 and SCU_MCCR5 registers must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction.

11.3.5.2 Start Bit Detection

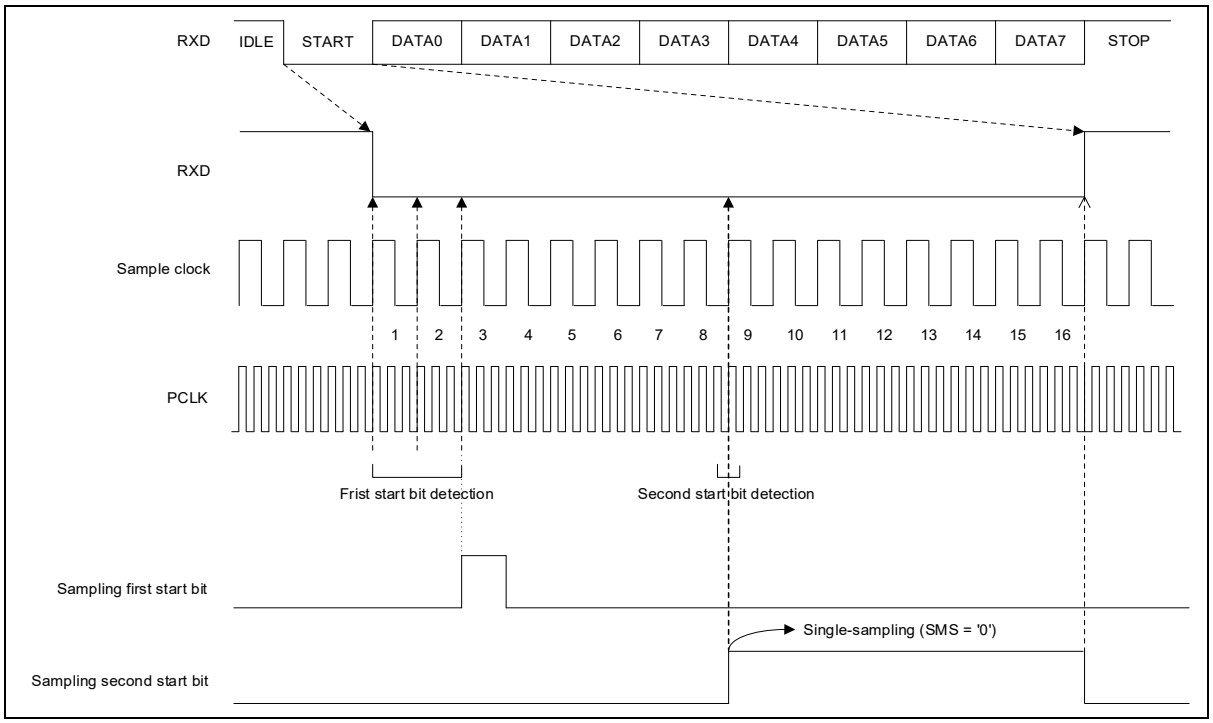
On the falling edge of the start bit, a digital filter rejects noise signal. A pulse shorter than ((3 × (1 / baud_rate)) / 16) s is rejected by this filter. Even when a pulse is longer than ((3 × (1 / baud_rate)) / 16) s, it can be rejected under the multi-sampling strategy during start-bit sampling.

In the UART, the start bit is detected when a specific sequence of samples is recognized as shown below:

- Single sampling sequence: 1 1 1 0 0 0 X X X X X 0 X X X X X X X
- Multi sampling sequence: 1 1 1 0 0 0 X X X X 0 0 0 X X X X X

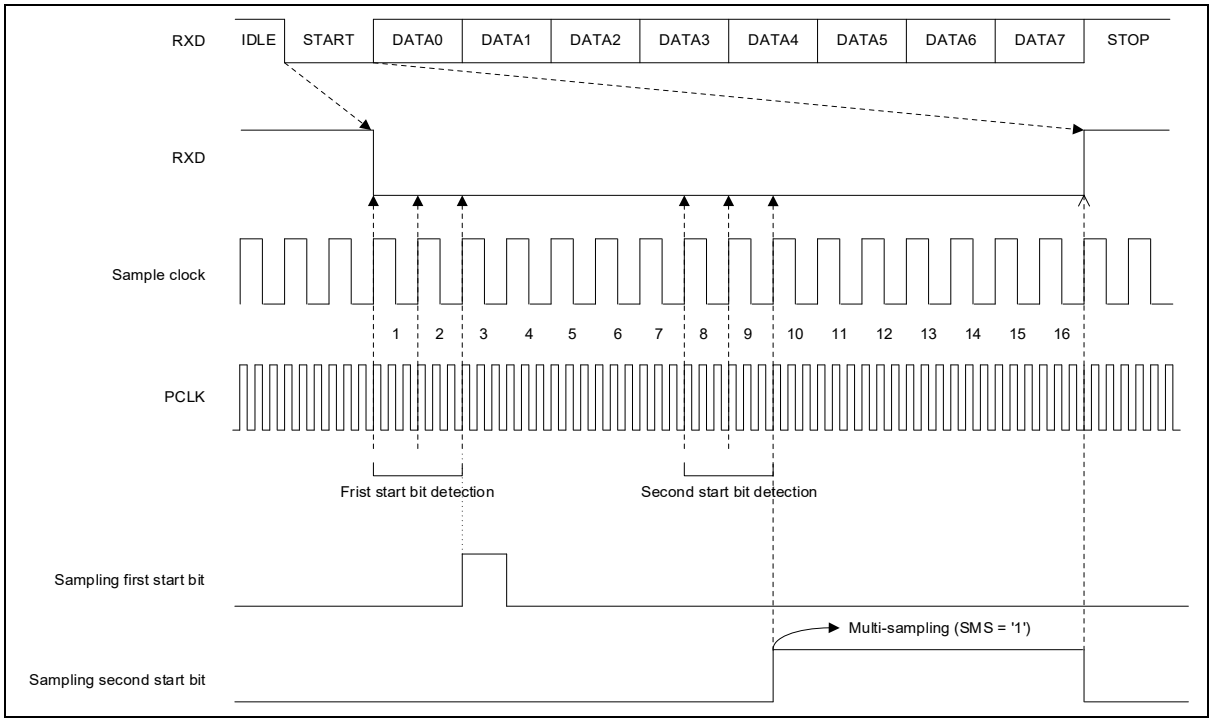
In single-sampling mode, the start bit is confirmed in the following sequence. The first sampling on the 1st, 2nd, and 3rd bits finds the three bits are all '0's, and the second sampling on the 9th bit also finds that the bit is '0'.

Figure 93. Start Bit Detection of Single-Sampling Timing



Similarly, the start bit is confirmed in the following sequence in multi-sampling mode. The first sampling on the 1st, 2nd, and 3rd bits finds that these three bits are all ‘0’s, and the second sampling on the 8th, 9th, and 10th bits finds that more than two bits are ‘0’s.

Figure 94. Start Bit Detection of Multi-Sampling Timing



NOTE:

1. If the sequence is not complete, the start bit detection stops and the receiver returns to the idle state, where it waits for a falling edge on RXD line.

11.3.5.3 Data Bit Sampling

If the DMS in UARTn_IDTR register is set to ‘1’, the values sampled at the 8th, 9th, and 10th clock pulses is determined to represent the value of the receive data bit. The multi-sampling technique is used for data recovery by discriminating between valid incoming data and noise.

Figure 95 shows the Single-sampling of data bits when the DMS is set to ‘0’.

Figure 95. Single-Sampling Timing of Data Bit

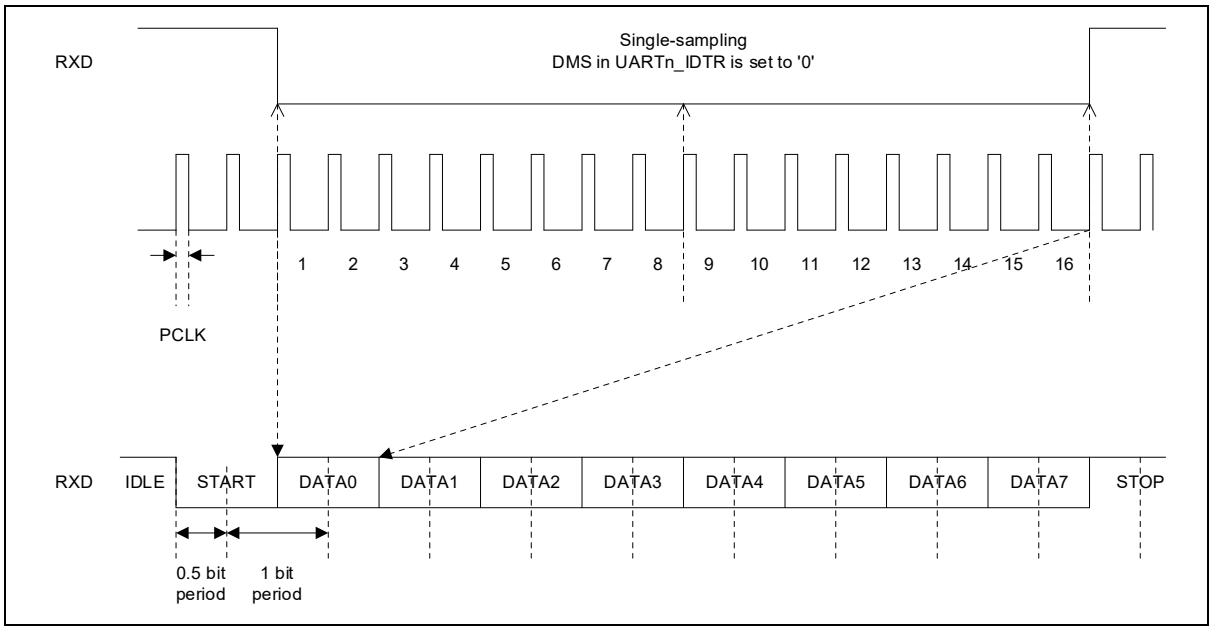
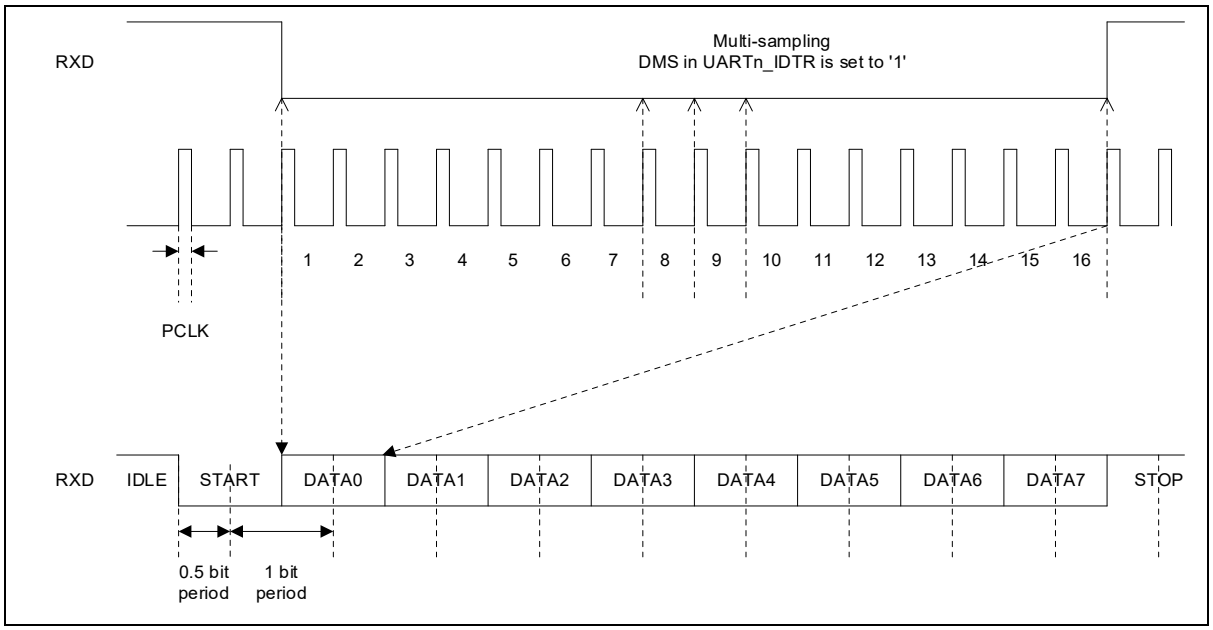


Figure 96 shows the Multi-sampling of data bits when the DMS is set to ‘1’.

Figure 96. Multi-Sampling Timing of Data Bit



11.3.5.4 Data Sampling Strategy

An internal synchronous circuit is used in the receiver block to prevent abnormal noise in the RXD signal line. The start bit and data bits can be either single-sampled or multi-sampled. The SMS bit in the UARTn_IDTR register defines the sampling strategy for the start bit, and the DMS bit in the UARTn_IDTR register defines the sampling strategy for the data bits.

Figure 97. Data Sampling Timing

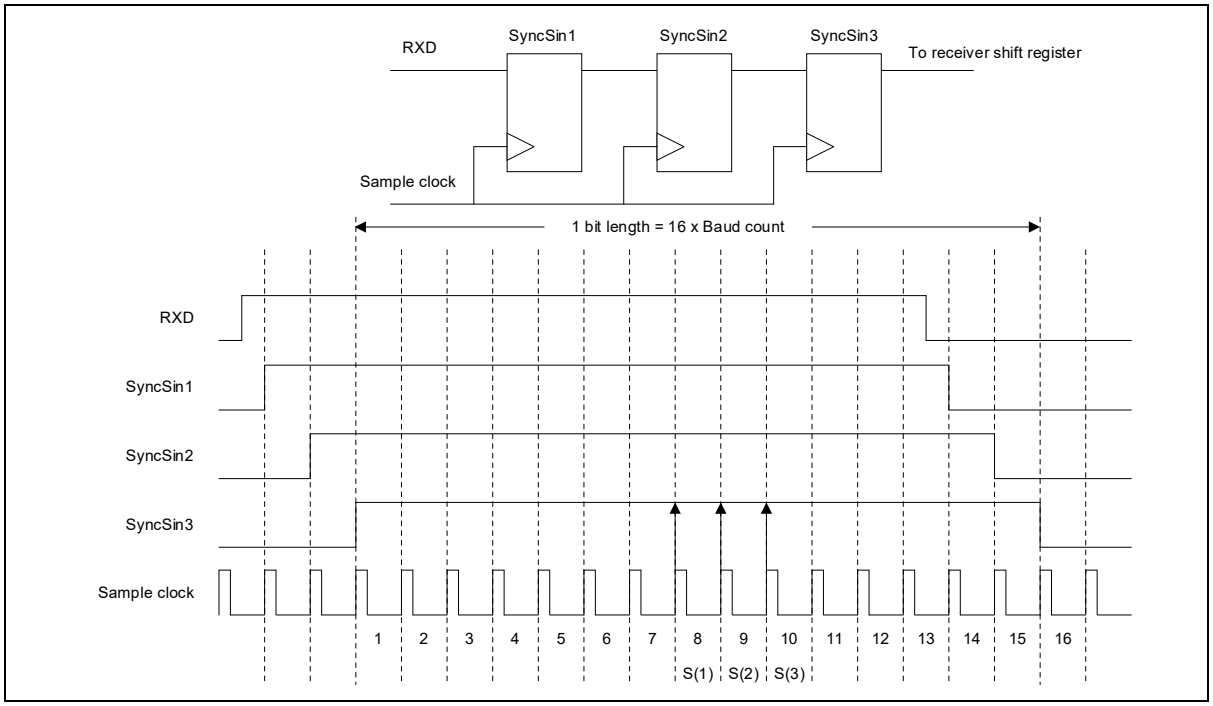


Figure 97 shows a block diagram and a sampling timing diagram of the anti-noise synchronous circuit.

If the DMS bit is set enabled, the value of each bit is sampled at multiple clock pulses. For the length of each data bit, values at S(1), S(2), and S(3) are taken out of 16 samples to determine the bit's value by taking the value indicated by a majority of the samples.

If the DMS bit is set disabled, on the other hand, the single-sampling process takes the value at S(2) only. The DMS bit does not affect the start bit, but the SMS bit works in a similar way to the Start bit.

Table 93. Received Bit Value from Sampled Data

Sampled Value	Received Bit Value
000	0
001	0
010	0
011	1
100	0
101	1
110	1
111	1

11.3.5.5 Character Reception

When the falling edge is detected on the RXD Pin, the Receiver determines it as a start bit and starts the reception operation. The Data frame following the start bit is transferred one bit at a time at the LSB and stored in the Receiver Shift Register.

Receiving the stop bit means that a set of communication is completed. If the UARTn_RBR (Receive data Buffer Register) register is empty, data in the Receiver Shift Register is transferred into this register and kept until it is read out over the Bus.

[Character Reception Procedure]

DMA, Interrupt, and Polling can be used to read data from the UARTn_RBR register after confirming that the communication is completed, as described below:

- With DMA enabled, the DMA Rx Request signal is generated when data in the UARTn_RBR is updated. The DMA stores data of the UARTn_RBR into Memory as many times as set in the DMA counter and then sends the DMA Done signal to the UART. The UART outputs the DMA Rx Complete interrupt upon receiving the DMA Done signal.
- If the interrupt is generated, the IID[2:0] bits in the UARTn_IIR register must be checked. If the IID[2:0] bits is set to ‘010’, this means that the reception operation is completed and the UARTn_RBR register can be read.
- To confirm the reception is completed using polling, the DR in UARTn_LSR register must be checked. If data is received successfully, the DR bit is set to ‘1’ and data in the UARTn_RBR can be read. Reading the UARTn_RBR register clears the DR bit.

11.3.5.6 Break Condition

When a break condition is received, the UART generates break interrupt.

11.3.5.7 Overrun Error

The OE bit in the UARTn_LSR register indicates whether the Overrun Error has been detected. An Overrun Error occurs if the Receiver completes the new Reception operation when the UARTn_RBR data was not read (The DR bit in the UARTn_LSR register is '1'). At this moment, although the Overrun Error occurs, data in the UARTn_RBR register is updated.

Reading the UARTn_LSR register clears the OE bit.

11.3.5.8 Framing Error

When the Frame data transmission is finished, if the sampled value is read as ‘0’ at the point of time expected to receive the stop bit, the FE bit in the UARTn_LSR register is set to ‘1’ (Even when multi-sampling, only the first sampled value is read). This means that the Frame data transmission is not completed on time.

The Frame data size can be determined by configuring the DLEN[1:0] bits in the UARTn_LCR register. If the data size set in the DLEN[1:0] bits and the received data’s bit size are different, problems may occur. Reading the UARTn_LSR register clears the FE bit.

11.3.5.9 Parity Error

The parity bit is appended to the data frame and provides a simple way to check whether the Receiver received correct data frame. If the parity bit is incorrect, the PE bit in the UARTn_LSR register is set to '1'. Reading the UARTn_LSR register clears the PE bit.

11.3.5.10 Configurable Stop Bits during Reception

During Reception operations, the first stop bit is sampled regardless of the STOPBIT bit in the UARTn_LCR register value. Users can change the STOPBIT value while the Receiver is operating.

11.3.6 UART Baud-rate Generation

To establish communication via a UART channel, users must set an appropriate baud-rate. To this end, a programmable baud-rate generator is built in, providing a baud-rate divisor ranging from 1 to 65,535.

Users must write an appropriate divisor value to the 16-bit UARTn_BDR register to obtain the desired baud-rate.

Below is the formula for calculating the baud-rate:

$$\text{UARTn\_BDR} = \frac{(\text{UART clock set by } \text{SCU}_{\text{MCCR7}})}{16 \times \ BaudRate}$$

If the UART clock has been set in the SCU_MCCR7(1) register, the PCLK clock speed must satisfy the formula below:

$$\text{PCLK} \geq (\text{UART clock set by } \text{SCU}_{\text{MCCR7}}) \times 2$$

NOTE:

1. The UART only uses a clock source set by MCCR7. It does not use the PCLK.

Table 94 lists the divisor and error rate for each baud-rate, at the UART clock speed of 70 MHz.

Table 94. Examples of Baud Rate Calculation

UART Clock = 70 MHz		
Baud-rate	**Divisor (UARTn_BDR)**	**Error (%)**
1,200	3,645	0.02 %
2,400	1,822	0.05 %
4,800	911	0.05 %
9,600	455	0.16 %
19,200	277	0.38 %
38,400	113	0.83 %
57,600	75	1.27 %
115,200	37	2.64 %

The 8-bit fraction counter compensates for the fractional part of the actual baud-rate divisor. This

function is required for accurate baud-rate because the divisor latch register is the integer part of the baud-rate divisor.

By adding the fractional part indicated in the UARTn_BFR register to the integer part indicated in the UARTn_BDR register, the correct divisor is obtained to calculate the accurate baud-rate.

NOTE:

1. If the UARTn_BDR register value is equal to or less than two, the UARTn_BFR register value is always zero regardless of the UARTn_BFR register's set value.

For example, when the baud-rate is 9,600 bps:

$$\frac{UART\ clock}{16 \times\ BaudRate} = \frac{70{,}000{,}000}{16 \times\ 9{,}600} = 455.7291, \qquad Divisor = 455, Float = 0.7291$$

$$CNT = Float \times 256\ = 0.7291 \times 256 = 186$$

$$UARTn\_BDR\ =\ 455, UARTn\_BFR\ =\ 186$$

Table 95. Examples Calculation of Baud-rate using UART Fraction Counter

UART Clock = 70 MHz			
Baud-rate	**Divisor (UARTn_BDR)**	**FCNT (UARTn_BFR)**	**Error (%)**
1200	3645	213	0.0 %
2400	1822	234	0.0 %
4800	911	117	0.0 %
9600	455	186	0.0 %
19200	277	221	0.0 %
38400	113	238	0.0 %
57600	75	244	0.0 %
115200	37	250	0.0 %

11.3.7 Tolerance of the UART Receiver to Clock Deviation

During Rx operation, each bit of data is sampled 16 times. To determine the value of the read data, the 8th sampling signal is used for Single-sampling, and the 8th, 9th, and 10th sampling signals are used for Multi-sampling.

Therefore, a slight difference is allowed unless the UART Rx signal is outside the sampling point within a frame (regardless of communication speed). This can be calculated by the formular shown below:

- Single-sampling: 7 / (16 × Total_bit_size)
- Multi-sampling: 6 / (16 × Total_bit_size)
- Total_bit_size: Receive bit size. (Data bit (5 to 8 bits) + Start bit + Stop bit)

Based on the calculation results from the formular above, Table 96 shows an acceptable error that depends on the bit size of transmitted data.

Table 96. Tolerance of the UART Receiver

Sampling	DLEN[1:0] Bits	UART Frame(1)	Tolerance of UART Receive
Single-Sampling	00	| St | 5-bit data | Sp |	7 / (16 × 7) = 6.25%
	01	| St | 6-bit data | Sp |	7 / (16 × 8) = 5.46%
	10	| St | 7-bit data | Sp |	7 / (16 × 9) = 4.86%
	11	| St | 8-bit data | Sp |	7 / (16 × 10) = 4.37%
Multi-Sampling	00	| St | 5-bit data | Sp |	6 / (16 × 7) = 5.35%
	01	| St | 6-bit data | Sp |	6 / (16 × 8) = 4.68%
	10	| St | 7-bit data | Sp |	6 / (16 × 9) = 4.16%
	11	| St | 8-bit data | Sp |	6 / (16 × 10) = 3.75%

NOTE:

1. St: Start bit, Sp: Stop bit.

11.3.8 UART Parity Control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PEN in UARTn_LCR register. Depending on the frame length defined in the DLEN[1:0] in the UARTn_LCR register, the possible UART frame formats are as listed in Table 97.

Table 97. UART Frame Formats

DLEN[1:0] Bits	PEN Bit	UART Frame(1)(2)
00	0	| St | 5-bit data | Sp |
00	1	| St | 5-bit data | P | Sp |
01	0	| St | 6-bit data | Sp |
01	1	| St | 6-bit data | P | Sp |
10	0	| St | 7-bit data | Sp |
10	1	| St | 7-bit data | P | Sp |
11	0	| St | 8-bit data | Sp |
11	1	| St | 8-bit data | P | Sp |

NOTE:

1. St: Start bit, Sp: Stop bit, P: Parity bit
2. In the data register, the parity is always taking the MSB position (5th, 6th, 7th or 8th depending on the DLEN[1:0] bits value).

11.3.8.1 Even Parity

The parity bit is calculated to obtain an even number of ‘1’s inside the frame made of the 5, 6, 7 or 8 LSB bits (Depending on the DLEN[1:0] bit values) and the parity bit when PEN bit is ‘1’.

As an example, if data = ’01011001’, and 4 bits are set, then the parity bit will be ‘0’ if even parity is selected (PARITY=1 and PEN= 1).

11.3.8.2 Odd Parity

The parity bit is calculated to obtain an odd number of ‘1’s inside the frame made of the 5, 6, 7 or 8 LSB bits (Depending on the DLEN[1:0] bit values) and the parity bit when PEN bit is ‘1’.

As an example, if data = ’01011001' and 4 bits are set, then the parity bit will be ‘1’ if odd parity is selected (PARITY=0 and PEN=1).

11.3.8.3 Stick Parity

The STICKP bit in the UARTn_LCR register determines whether to use stick parity.

- If PEN = ‘1’, PARITY = ‘1’, and STICKP = ‘1’, then the parity bit is always ‘0’.
- If PEN = ‘1’, PARITY = ‘0’, and STICKP = ‘1’, then the parity bit is always ‘1’.
- If STICKP = ‘0’, stick parity is disabled.

11.3.8.4 Various Configurations of Parity Bit

A parity bit is generated based on the settings of the PEN, PARITY and STICKP bit in the UARTn_LCR register.

Table 98 describes various configurations to create a parity bit.

Table 98. Various Configurations to Create Parity Bit

STICKP	PARITY	PEN	Parity
X	X	0	No parity
0	0	1	Odd parity
0	1	1	Even parity
1	0	1	Forced ‘1’ to stick parity
1	1	1	Forced ‘0’ to stick parity

11.3.8.5 Parity Checking in Reception

If the parity check fails, the PE flag is set by configuring the PE bit in the UARTn_LSR register, and the corresponding interrupt is generated if the RLSIE bit in the UARTn_IER register is set to ‘1’.

Reading the UARTn_LSR register clears the PE bit to ‘0’.

11.3.8.6 Parity Generation in Transmission

If the PEN bit in the UARTn_LCR register is set to ‘1’, the Transmitter generates a parity bit between the last data bit and the stop bit. The parity bit is set to ‘0’ or ‘1’ to ensure that the total number of ‘1’s in the data and parity bits is either even or odd depending on how the PARITY and STICKP bits are set.

11.3.9 Continuous Communication using DMA and UART

The UART module supports DMA transfers using the DMA device built in the microcontroller. The start of the memory address for data transfer and the length of the data to be transferred are determined by the programming registers in the DMA module.

Completion of the data transfer is related to a notification by the transfer complete flag. Once the entire transmit data is written to the Transmit data Hold Register (THR), the UARTn_IIR register’s DMA Tx complete flag (TXE bit) is set and the interrupt source ID of the complete interrupt is written to the IID[2:0] in the UARTn_IIR register.

Similarly, once the entire receive data is written to the target DMA memory, the UARTn_IIR register’s DMA Rx complete flag is set and the interrupt source ID of the complete interrupt is written to the IID[2:0] bits in the UARTn_IIR register. Therefore, the UART RXD signal already becomes idle when the DMA Rx complete interrupt occurs.

11.3.9.1 Transmission using DMA

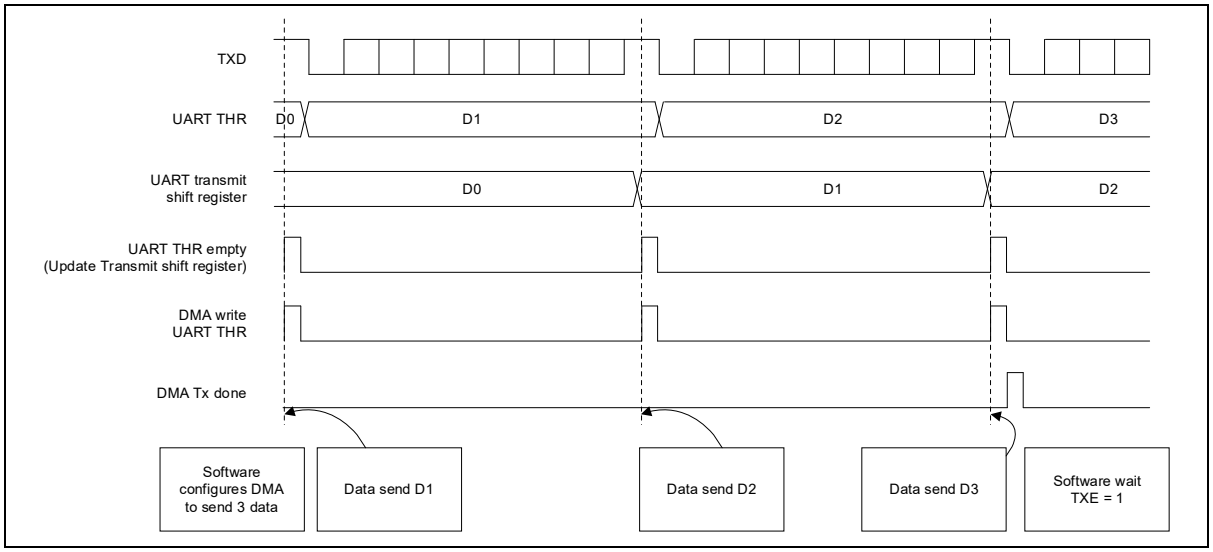
With DMA enabled, when the Transmit Shift Register is updated, the DMA Tx request signal is generated to allow the UARTn_THR register to store new data.

When the UARTn_THR register stores the new data using DMA transfer and the Transmit Shift Register completes output data, the UARTn_THR register transmits the new data to the Transmit Shift Register again. The data in the Transmit Shift Register is output to the TXD Line according to the Transmitter operation.

After the DMA transmits data as many times as set in the DMA Tx count (The set number in Figure 98 is 3), the DMA generates the DMA Tx done signal. Upon receiving this signal, the UART outputs the DMA Tx complete interrupt to inform that the DMA Tx operation is completed.

When the DMA operation is completed, unless new data is written to the UARTn_THR register, the last data in the Transmit Shift Register is output and the Transmitter operation is complete. (TXE interrupt is generated.)

Figure 98. Transmission using DMA



11.3.9.2 Reception using DMA

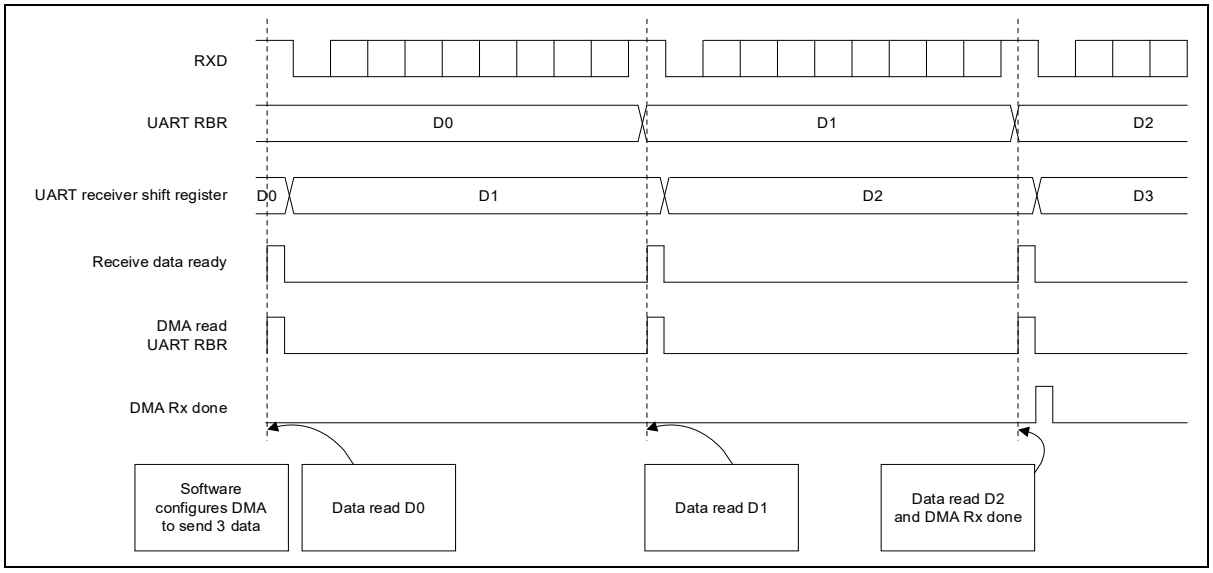
When the Receiver Shift Register is updated according to the reception operation, new data is stored in the UARTn_RBR register. With DMA enabled, when the UARTn_RBR register is updated with new data, the DMA Rx Request signal is generated.

DMA reads the UARTn_RBR register to store the data into memory.

After the DMA stores the data of the UARTn_RBR register into memory as many times as set in the DMA Rx count (the set number in Figure 99 is 3), the DMA generates the DMA Rx done signal.

Upon receiving this signal, the UART outputs the DMA Rx Complete interrupt to inform that the DMA Rx operation is completed.

Figure 99. Reception using DMA



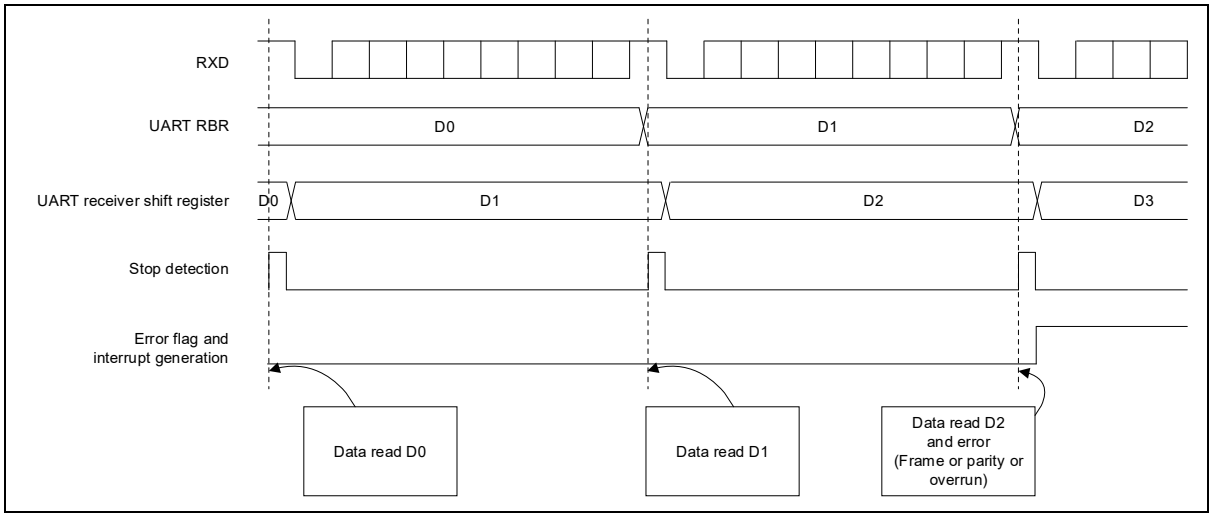
11.3.9.3 Error Flagging and Interrupt Generation in Communication

Frame errors, Parity errors, and Overrun errors may occur after the data receiving is completed, during Rx operation. If any of these errors occurs as a result of the Rx operation, the Receive Line Status Error Interrupt is generated depending on the settings of the RLSIE bit in the UARTn_IER register.

Then, the IID bits in the UARTn_IIR register represents the interrupt source ID, and the UARTn_LSR register shows the type of error. The UARTn_LSR register is updated whenever an error occurs, even if the RLSIE bit in the UARTn_IER register is disabled.

Although an error occurs, data in the Receiver Shift Register is transmitted to UARTn_RBR register and then UART Rx operation is maintained normally.

Figure 100. Error Flagging and Interrupt Generation Timing Diagram



11.4 UART Interrupts

During UART communications, various interrupts can be generated by different events. They can be checked by reading the UARTn_IIR register.

When the microcontroller core accesses the UARTn_IIR register, the UART locks all interrupts and the highest-priority interrupt is read.

Interrupts can be occurred even while the register is being accessed by the microcontroller core; however, the interrupt status remains unchanged until the current access is completed.

Among the pending interrupts, the highest-priority interrupt's source ID is represented to the IID[2:0] bits in the UARTn_IIR register.

The UART module uses a total of seven interrupts with different priorities. These interrupts include the followings:

- Receive line status interrupt
- Receive data ready interrupt
- Transmit data hold register empty interrupt
- Transmit complete interrupt
- DMA receive complete interrupts
- DMA transmit complete interrupts
- Transmit complete and DMA Tx complete interrupts

11.5 UART Registers

The base addresses and register map of the UARTs are described in the following tables:

Table 99. Base Address of UART

Name	Base Address
UART0	0x4000_8000
UART1	0x4000_8100
UART2	0x4000_8200
UART3	0x4000_8300
UART4	0x4000_8400
UART5	0x4000_8500

Table 100. UART Register Map

Name	Offset	Type	Description	Reset Value	Reference
UARTn_RBR	0x0000	RO	UART n Receive Data Buffer Register	0x0000_0000	11.5.1
UARTn_THR	0x0000	WO	UART n Transmit Data Hold Register	0x0000_0000	11.5.2
UARTn_IER	0x0004	RW	UART n Interrupt Enable Register	0x0000_0000	11.5.3
UARTn_IIR	0x0008	RO	UART n Interrupt ID Register	0x0000_0001	11.5.4
UARTn_LCR	0x000C	RW	UART n Line Control Register	0x0000_0000	11.5.5
UARTn_DCR	0x0010	RW	UART n Data Control Register	0x0000_0000	11.5.6
UARTn_LSR	0x0014	RO	UART n Line Status Register	0x0000_0060	11.5.7
UARTn_BDR	0x0020	RW	UART n Baud-rate Divisor Latch Register	0x0000_0000	11.5.8
UARTn_BFR	0x0024	RW	UART n Baud-rate Fraction Counter Register	0x0000_0000	11.5.9
UARTn_IDTR	0x0030	RW	UART n Inter-Frame Delay Time Register	0x0000_0000	11.5.10

NOTE:

1. n = 0 to 5.

11.5.1 UARTn_RBR: UART n Receive Data Buffer Register

The UARTn_RBR register is an 8-bit read-only register. Received data is read from this register, and the maximum length of data is 8 bits. The last received data is retained until a new byte is received.

UART0_RBR=0x4000_8000, UART1_RBR=0x4000_8100, UART2_RBR=0x4000_8200
UART3_RBR=0x4000_8300, UART4_RBR=0x4000_8400, UART5_RBR=0x4000_8500

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
Reserved			RBR[7:0]
-			0x00
-			RO

7 0	RBR[7:0]	Receive buffer register

11.5.2 UARTn_THR: UART n Transmit Data Hold Register

The UARTn_THR register is an 8-bit write-only register. Data is stored in this register to be transmitted. However, data written to the UARTn_THR register cannot be read but sent to the Transmit Shift Register when this register is empty.

UART0_THR=0x4000_8000, UART1_THR=0x4000_8100, UART2_THR=0x4000_8200
UART3_THR=0x4000_8300, UART4_THR=0x4000_8400, UART5_THR=0x4000_8500

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
Reserved			THR[7:0]
-			0xXX
-			WO

7 0	THR[7:0]	Transmit data hold register

11.5.3 UARTn_IER: UART n Interrupt Enable Register

The UARTn_IER register enables six types of UART interrupts. Each interrupt generates an interrupt output signal. If users set all bits in the UARTn_IER register to '0', all UART interrupts become disabled. A particular interrupt can be set by setting the corresponding bit in this register to '1'.

UART0_IER=0x4000_8004, UART1_IER=0x4000_8104, UART2_IER=0x4000_8204
UART3_IER=0x4000_8304, UART4_IER=0x4000_8404, UART5_IER=0x4000_8504

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
Reserved				DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
-				0	0	0	0	0	0
-				RW	RW	RW	RW	RW	RW

5	DTXIEN	Enable or disable the DMA transmit complete interrupt
	0	Disables the DMA transmit complete interrupt.
	1	Enables the DMA transmit complete interrupt.
4	DRXIEN	Enable or disable the DMA receive complete interrupt
	0	Disables the DMA receive complete interrupt.
	1	Enables the DMA receive complete interrupt.
3	TXEIE	Enable or disable the transmit complete interrupt
	0	Disables the transmit complete interrupt.
	1	Enables the transmit complete interrupt.
2	RLSIE	Enable or disable the receive line status interrupt
	0	Disables the receive line status interrupt.
	1	Enables the receive line status interrupt.
1	THREIE	Enable or disable the transmit data hold register empty (THRE) interrupt
	0	Disables the THRE interrupt.
	1	Enables the THRE interrupt.
0	DRIE	Enable or disable the data receive interrupt
	0	Disables the data receive interrupt.
	1	Enables the data receive interrupt.

11.5.4 UARTn_IIR: UART n Interrupt ID Register

The UARTn_IIR register is a read-only register informing the operation of the interrupt that is set to enabled in the UARTn_IER register.

UART0_IIR=0x4000_8008, UART1_IIR=0x4000_8108, UART2_IIR=0x4000_8208
UART3_IIR=0x4000_8308, UART4_IIR=0x4000_8408, UART5_IIR=0x4000_8508

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
Reserved			TXE	IID[2:0]	IPEN
-			0	000	1
-			R0	R0	R0

4	TXE	Transmit complete (refer to Table 101)
		TXE Indicates whether the transmit shift register is empty. TXE = 0 indicates that UARTn_THR and transmit shift register is currently empty and thus new data can be written to the UARTn_THR register.
3 1	IID[2:0]	Interrupt source ID (Refer to Table 101)
		IID[2:0] Displays the interrupt with the highest priority among those currently active.
0	IPEN	Presence of pending interrupts IPEN Indicates there are currently unprocessed interrupts. IPEN=0 indicates that a certain interrupt condition has occurred, and IPEN=1 means that there are no currently unprocessed interrupts.
		0 An interrupt is pending
		1 There are no interrupts pending

Table 101. Interrupt ID and Control

Priority Level	TXE	IID[2:0]			IPEN	Interrupt Source		
	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Interrupt Name	Interrupt Condition	Interrupt Clear
-	0	0	0	0	1	N/A	-	-
1	0	0	1	1	0	Receive line status	Overrun, parity, framing, break error, etc.	Read UARTn_LSR
2	0	0	1	0	0	Receive data present	There is data received	Read the receive register or UARTn_IIR
3	0	0	0	1	0	Transmit hold register empty	Transmit data hold register empty	Write to the transmit data hold register or UARTn_IIR
4	1	X	X	X	X	Transmit register empty	Transmit shift register and transmit data hold register is empty	Write to the transmit data hold register or read UARTn_IIR
5	0	1	1	0	0	DMA Rx complete	DMA Rx complete	Read UARTn_IIR
6	0	1	0	1	0	DMA Tx complete	DMA Tx complete	Read UARTn_IIR
7	1	1	X	X	X	Transmit register empty and DMA complete	The transmit register is empty and DMA Tx has been completed.	Read UARTn_IIR

11.5.5 UARTn_LCR: UART n Line Control Register

The UARTn_LCR register is an 8-bit register.

UART0_LCR=0x4000_800C, UART1_LCR=0x4000_810C, UART2_LCR=0x4000_820C
UART3_LCR=0x4000_830C, UART4_LCR=0x4000_840C, UART5_LCR=0x4000_850C

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
Reserved				BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]
-				0	0	0	0	0	00
-				RW	RW	RW	RW	RW	RW

6	BREAK	When this bit is set to '1', the TXD pin is driven low to alert the receiver. BREAK is a break control bit. If BREAK = '1', the serial output pin (TXD pin) is driven low (Spacing state), whereas writing a 0 to the bit disables the break condition. 0 General communication mode 1 Transmit break mode
5	STICKP	Use stick parity. The setting of this bit is valid only when the PEN bit is set to '1'. STICKP determines whether to use stick parity. If PEN = '1', PARITY = '1', and STICKP = '1', the parity bit is always set low. If PEN = '1', STICKP = '1', and PARITY = '0', the transmit parity bit is always set high. If STICKP = 0, stick parity is disabled. 0 Does not use stick parity. 1 Uses stick parity.
4	PARITY	Parity mode selection PARITY determines which parity mode is used between odd and even parity. If PEN = '1' and PARITY = '0', odd parity is used depending upon the number of data bits; if PEN = '1' and PARITY = '1', even parity is used. 0 Odd-parity mode 1 Even-parity mode
3	PEN	Enable parity PEN determines whether to enable the use of parity. If PEN = '1', the transmitter creates a parity bit between the last data bit and the stop bit and the receiver inspects this parity bit. (The parity bit is set to '0' or '1' to ensure that the total number of 1s in the data and parity bits is either even or odd depending on how the PARITY and STICKP bits are set. 0 Disables parity. 1 Enables parity.
2	STOPBIT	The number of stop bits demanded by the preceding data bits STOPBIT defines the number of stop bits attached to the end of each transmitted / received data word. STOPBIT = '0' includes 1 stop bit at the end of each transmitted data word. STOPBIT = '1' includes 1.5 or 2 stop bits at the end of each transmitted data word depending upon the number of transmit bits defined at DLEN[1:0]; a 5-bit data word includes 1.5 stop bits and a 6-, 7-, or 8-bit data word includes 2 stop bits. The receiver checks only the first stop bit regardless of the number of stop bits defined at DLEN[1:0] and STOPBIT. 0 The number of stop bits is 1. 1 The number of stop bits is 1.5 or 2. If the data word is 5 bits long, 1.5 stop bits are added.

		If the data word is 6, 7, or 8 bits long, 2 stop bits are added.
1 0	DLEN[1:0]	The length of data bits included by each data transfer DLEN[1:0] defines the length of each transmitted / received data bit length.
		00 5-bit data
		01 6-bit data
		10 7-bit data
		11 8-bit data

11.5.6 UARTn_DCR: UART n Data Control Register

The UARTn_DCR register is a 32-bit register that controls Tx or Rx data inversion.

UART0_DCR=0x4000_8010, UART1_DCR=0x4000_8110, UART2_DCR=0x4000_8210
UART3_DCR=0x4000_8310, UART4_DCR=0x4000_8410, UART5_DCR=0x4000_8510

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
Reserved				RXINV	TXINV	Reserved
-				0	0	-
-				RW	RW	-

3	RXINV	Receive data inversion selection
		0 Inputs normal receive data.
		1 Inputs inverted receive data.
2	TXINV	Transmit data inversion selection
		0 Outputs normal transmit data.
		1 Outputs inverted transmit data.

11.5.7 UARTn_LSR: UART n Line Status Register

The UARTn_LSR register is a read-only register that shows the status of data transmission and reception. This register reports the status of data transfers between the transmitter and receiver. Through this register, users can check the status of the UART lines and set interrupts as follows: Status interrupts for bits 1, 2, 3, and 4 can be called if they are set enabled in the RLSIE bit in the UARTn_IER register. Other bits enable their corresponding interrupts to be called if they are set enabled in the UARTn_IER register.

UART0_LSR=0x4000_8014, UART1_LSR=0x4000_8114, UART2_LSR=0x4000_8214
UART3_LSR=0x4000_8314, UART4_LSR=0x4000_8414, UART5_LSR=0x4000_8514

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
Reserved			TEMT	THRE	BI	FE	PE	OE	DR	
-			1	1	0	0	0	0	0	
-			RO	RO	RO	RO	RO	RO	RO	

6	TEMT	Indicates the transmit registers are empty. TEMT indicates whether the transmit registers are empty. When both UARTn_THR and transmit shift register are empty, the bit is set to '1'. Once either UARTn_THR or transmit shift register holds any data, the bit is cleared to '0'.
	0	Data is being transmitted through the transmit registers.
	1	The transmit registers are empty.
5	THRE	Indicates the transmit data hold register is empty. THRE indicates whether the UARTn_THR is empty, which means the UART is ready to receive new data from the microcontroller core for transmission. UARTn_THR empty interrupts can be generated if THREIE is set enabled in UARTn_IER. The THRE bit is set to 1 when new data can be written to UARTn_THR as the previous transmit data has been transferred from UARTn_THR to the transmit shift register. The bit is cleared to '0' when new data is written to UARTn_THR by the microcontroller core (When UARTn_THR is no longer empty).
	0	The transmit data hold register is not empty.
	1	The transmit data hold register is empty.
4	BI	Indicates a break condition has been detected. BI shows the occurrence of a break interrupt. If the incoming receive data remains in low state for longer than the time taken to receive the entire data word (i.e., The sum of the start, data, parity, and stop bits), the bit is set to '1'. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to '0'.
	0	Normal.
	1	A break condition has been detected.
3	FE	Indicates presence of a frame error. FE Indicates whether a communication framing error has been detected. A framing error signifies that a received data word does not have an appropriate stop bit. The FE bit is set to 1, when the stop bit attached to the last data bit, or the parity bit is detected to be '0'. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to '0'.
	0	No frame error present.
	1	A frame error has been detected. The received data does not have appropriate stop bits.

2	PE	Indicates presence of a parity error. PE indicates whether a parity error has been detected. If a received data word does not meet the odd/even parity condition defined in UARTn_LCR, the bit is set to '1'. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to '0'.
		0 No parity error present
		1 A parity error has been detected. The received data does not satisfy parity conditions.
1	OE	Indicates presence of an overrun error. OE indicates whether an overrun error has been detected. When a data word is transferred to UARTn_RBR before the register reads the previous data word, an error occurs, and the OE bit is set to '1'. Once UARTn_LSR is read by the microcontroller core, the overrun error is not detected, and the bit is cleared to '0'.
		0 No overrun error present.
		1 An overrun error has been detected. Additional data has arrived when the UARTn_RBR is already full.
0	DR	Indicates data reception status. DR indicates that data reception has been completed. The bit is set to '1' when a data word has completely been received and transferred to UARTn_RBR. Once the data is read by UARTn_RBR, the bit is cleared to '0'.
		0 There is no data in the receive data buffer register.
		1 Data is received and stored in the receive data buffer register.

11.5.8 UARTn_BDR: UART n Baud-rate Divisor Latch Register

The UARTn_BDR register is a 16-bit register.

UART0_BDR=0x4000_8020, UART1_BDR=0x4000_8120, UART2_BDR=0x4000_8220
UART3_BDR=0x4000_8320, UART4_BDR=0x4000_8420, UART5_BDR=0x4000_8520

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
Reserved		BDR[15:0]	
-		0x0000	
-		RW	

15 0	BDR[15:0](1)	Baud-rate divisor latch (1 to 65535)

NOTE:

1. If the UARTn_BDR value is equal to or less than two, the UARTn_BFR is always zero regardless of the UARTn_BFR set value.

11.5.9 UARTn_BFR: UART n Baud-rate Fraction Counter Register

The UARTn_BFR register is an 8-bit register.

UART0_BFR=0x4000_8024, UART1_BFR=0x4000_8124, UART2_BFR=0x4000_8224
UART3_BFR=0x4000_8324, UART4_BFR=0x4000_8424, UART5_BFR=0x4000_8524

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
Reserved			BFR[7:0]
-			0x00
-			RW

7 0	BFR[7:0](1)	Enable the fraction counter to compensate error from the integer baud-rate divisor	
		0	Disables the fraction counter.
		Other	Enables the fraction counter. In fractional compensation mode, the value of the fraction counter is added to the integer baud-rate divisor.

NOTE:

1. If the UARTn_BDR value is equal to or less than two, the UARTn_BFR is always zero regardless of the UARTn_BFR set value.

11.5.10 UARTn_IDTR: UART n Inter-Frame Delay Time Register

The UARTn_IDTR register is an 8-bit register. A dummy delay is inserted between two consecutive transmit data frames.

UART0_IDTR=0x4000_8030, UART1_IDTR=0x4000_8130, UART2_IDTR=0x4000_8230
UART3_IDTR=0x4000_8330, UART4_IDTR=0x4000_8430, UART5_IDTR=0x4000_8530

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
Reserved			SMS	DMS	Reserved	WAITVAL[2:0]
-			0	0	-	000
-			RW	RW	-	RW

7	SMS[1]	Enable multi-sampling the start bit If the SMS bit is set to 1, the start bit is sampled three times. The value indicated by the majority of the samples is taken. 0 Disables multi-sampling the start bit. At this setting, sampling is conducted only once at clock pulse 8 (Among the 16 pulses in total). 1 Enables multi-sampling the start bit. At this setting, sampling is conducted three times at clock pulses 7, 8, and 9 (Among the 16 pulses in total). The value indicated by a majority of the three samples is taken.
6	DMS	Enable multi-sampling each data bit If DMS bit is set to 1, each data bit is sampled three times. The value indicated by the majority of the samples is taken. 0 Disables multi-sampling each data bit. At this setting, sampling is conducted only once at clock pulse 8 (Among the 16 pulses in total). 1 Enables multi-sampling each data bit. At this setting, sampling is conducted three times at clock pulses 7, 8, and 9 (among the 16 pulses in total). The value indicated by a majority of the three samples is taken.
3 0	WAITVAL[2:0]	Defines the wait time for the next data frame. [unit: 1-bit period] **Wait Time = WAITVAL[2:0] / Baud-rate** WAITVAL[2:0] Based on the value (0 through 7) indicated by the WAITVAL[2:0] bits, a wait time is set between successive data transfers. The formula above is used to compute the wait time.

NOTE:

1. After the microcontroller's power-on, if the UART bit in the SCU_PER2 register is disabled and enabled again, the SMS bit is set to '1'.

11.5.11 UART Register Map Summary

Table 102. UART Register Map Summary

Offset	Register name	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
0x00	UARTn_RBR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RBR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x00	UARTn_THR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	THR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x04	UARTn_IER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
	Reset value																											0	0	0	0	0	0
0x08	UARTn_IIR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TXE	IID[2:0]			IPEN
	Reset value																												0	0	0	0	1
0x0C	UARTn_LCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x10	UARTn_DCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RXINV	TXINV	Res	Res
	Reset value																													0	0		
0x14	UARTn_LSR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TEMT	THRE	BI	FE	PE	OE	DR
	Reset value																										1	1	0	0	0	0	0
0x20	UARTn_BDR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BDR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	UARTn_BFR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BFR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x30	UARTn_IDTR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SMS	DMS	Res	Res	Res	WAITVAL[2:0]		
	Reset value																									0	0				0	0	0

12. Serial Peripheral Interface (SPI)

12.1 SPI Introduction

The A34M420 series has three built-in Serial Peripheral Interface (SPI) modules. The SPI modules are synchronized by clocks, and the specifications of the transfer clocks are adjustable.

The SPI module supports communications between one master and slaves. Slaves can be selected by the Slave Select (SS) signal.

The SPI module performs four-wire synchronous transfers via four signal terminals (SS, SCK, MOSI, and MISO). Its separate Transmit and Receive Buffers enable full-duplex communication, which is capable of reading and writing data simultaneously.

Specifically, the SPI module of the A34M420 series can enhance the port drive strength so that the SPI rate can be supported up to 16 MHz with the enhanced port drive strength.

12.2 SPI Main Features

The SPI of the A34M420 features the followings:

- Selectable between the master and slave operations
- Full-duplex and four-wire synchronous transfers
  - SS: Slave Select
  - SCK: Serial Clock
  - MOSI: Master Output, Slave Input
  - MISO: Master Input, Slave Output
- SPI clock speed and polarity adjustable
- Separate transmit and receive data registers with different data transfer sizes
  - Available transmit/receive data sizes: 8-, 9-, 16-, and 17-bit
- Configurable interrupts triggered by the transmit status and SS signal
- Loop-back mode for internal checkups
- User-programmable start, burst, and stop delay settings
- Up to 16 MHz of SPI clock supported.
- Port drive strength for high-speed communication
- Selectable between MSB first transfer or LSB first transfer
- DMA Tx, Rx transfers

12.3 SPI Implementation

The transmitter and receiver of the Serial Peripheral Interface (SPI) share the same clock but are independent of each other. This enables full-duplex transfers. The transmitter and receiver are equipped with double buffers, thereby supporting back-to-back data transfers either by reading previous receive data from the SPI_RDR register while subsequent data is being received, or by writing subsequent data to the SPI_TDR register while previous data is being transmitted.

To enable the SPI transmission and reception, the MOSI and MISO lines of the master and slave must be connected directly, enabling back-to-back transfers between them. The most significant bits are given priority during these data transfers. Communication is always commenced by the master. Once the master transfers data to the slave through the MOSI pin, the slave responds through the MISO pin.

Table 103 describes the built-in SPI features.

Table 103. SPI Implementation

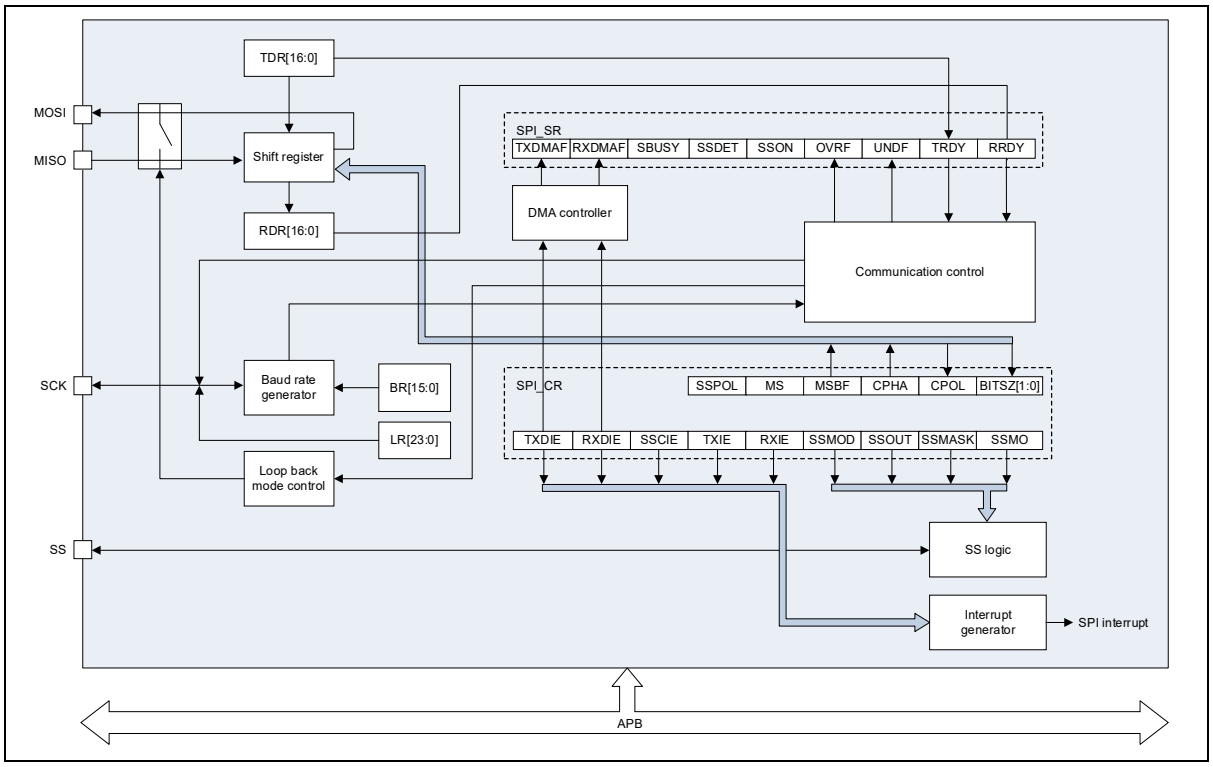
SPI Feature	Description
DMA	DMA transfer for receive and transmit
Data Size	Data size selection for transmit and receive (8-, 9-, 16-, 17-bit)
MSB / LSB Selection	Transfer format selection (MSB or LSB)
Error Flag	Error flags (Overrun, Underrun)
Latency	Programmable latency (start, burst, and stop delay)

12.4 SPI Functional Description

12.4.1 SPI Block Diagram

Figure 101 shows the block diagram for the SPI of the A34M420.

Figure 101. SPI Block Diagram



12.4.2 SPI Pins and Internal Signals

Table 104 describes the pins assigned for the SPI.

Table 104. Pin Assignment of SPI: External Pins

Pin Name	Type	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
SS0	I/O	SPI0 serial port I/O signal for slave selection	O	O	O
SCK0	I/O	SPI0 clock I/O (Master: output/ slave: input)	O	O	O
MOSI0	I/O	SPI0 transmit and receive data (Master: output/ slave: input)	O	O	O
MISO0	I/O	SPI0 transmit and receive data (Master: input/ slave: output)	O	O	O
SS1	I/O	SPI1 serial port I/O signal for slave selection	O	O	N/A
SCK1	I/O	SPI1 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI1	I/O	SPI1 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO1	I/O	SPI1 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS2	I/O	SPI2 serial port I/O signal for slave selection	O	N/A	N/A
SCK2	I/O	SPI2 clock I/O (Master: output/ slave: input)	O	N/A	N/A
MOSI2	I/O	SPI2 transmit and receive data (Master: output/ slave: input)	O	N/A	N/A
MISO2	I/O	SPI2 transmit and receive data (Master: input/ slave: output)	O	N/A	N/A

12.4.3 Communications between Master and Slave

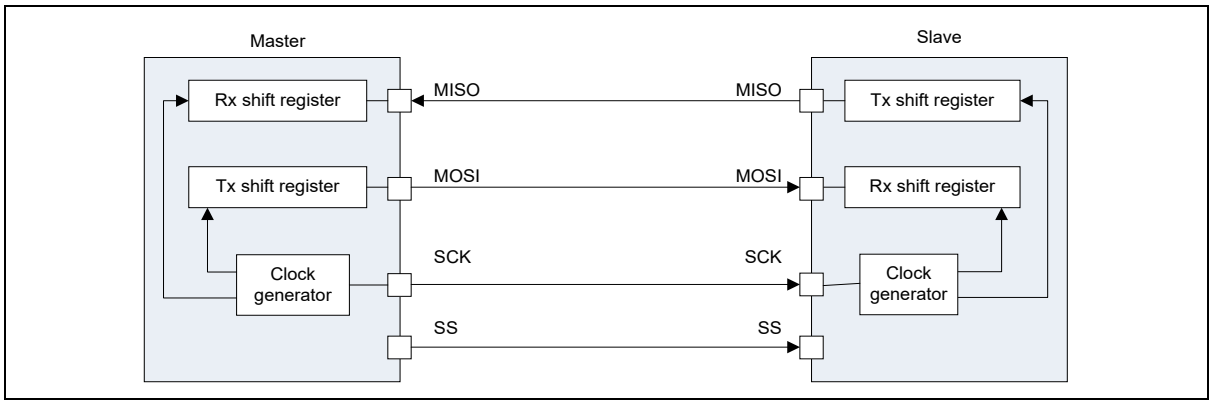
12.4.3.1 Full-Duplex Communication

Figure 102 shows a normal connection for the communication between the SPI-Master and a Slave. For the full-duplex communication, the SPI- Master and slave are connected as shown in Figure 102.

The SPI- Master generates the SCK clock for transmission, and outputs data on the MOSI pin. At the same time, the Slave transmits data through the MISO pin according to the Master's SCK.

The SS pin outputs Low for the Slave Chip Select when the Master starts transmitting.

Figure 102. Block Diagram of SPI Master to Slave Connection: Full-Duplex



12.4.3.2 Simplex Communication

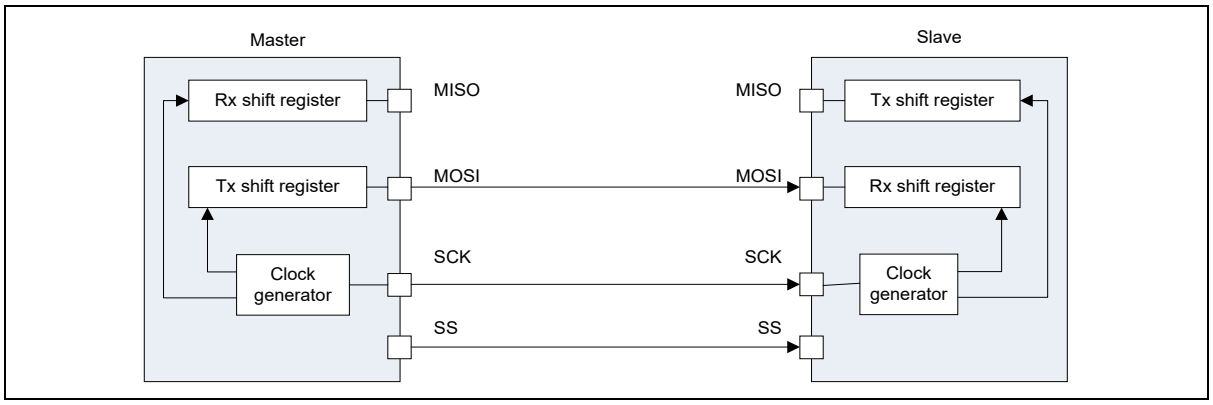
If only the Master's transmission and the Slave's reception are required, the Master and Slave need to be connected as shown in Figure 103.

The SPI Master generates the SCK clock for transmission and outputs data on the MOSI pin. At the same time, the Slave receives data on the MOSI pin according to the Master's SCK.

The SS pin outputs Low for the Slave Chip Select when the Master starts transmitting.

The MISO pin can be used as an alternative function pin such as a GPIO pin.

Figure 103. Block Diagram of SPI Master to Slave Connection: Simplex



12.4.4 Slave Selection Pin

The SS line is used for slave select input that enables the slave to communicate with the master. The SS pin can be driven as a standard IO port of the master device and is configured by setting the SSMOD in SPI_CR register.

If the SSMOD is set to '1', the pin is driven internally according to the settings of the SSOUT in SPI_CR register. If the SSMOD bit is set to '0', the pin performs one of the two functions depending on the settings of the SSMO in SPI_CR register.

When SSMO = '1' and SSMOD = '0', the SS pin is used only in Master mode. In this case, the SS signal is driven to Low when the Master starts communication. This low state is maintained until the SPI becomes disabled.

When SSMO = '0' and SSMOD = '0', this setting is used as Master mode. In Slave mode, the SS pin functions as an input pin. When the SS signal is Low, the device is selected as a Slave; when it is driven High, the Slave selection is released.

For the device in Master mode, therefore, users can either set the SS pin as an output pin to deactivate the pin (SSMOD = '0' , SSMO = '0') or set the pin as an input pin, feed it with a high-level signal (SSMOD = '1', SSOUT = '1'), and, after a period of waiting time, activate the pin (SSMOD = '0', SSMO = '1') or feed the pin with a low-level signal (SSMOD = '1', SSOUT = '0') to generate a falling edge.

12.4.5 Communication Formats

12.4.5.1 Clock Phase and Polarity Controls

Each SPI has four operating modes that synchronizes data transferred through the MOSI and MISO lines with the SCK clock. There are four mode SPI transfer format which the CPHA and CPOL bits in the SPIn_CR register set the phase and polarity of the SPI clock.

The CPHA (Clock phase control) bit is used to select a transfer format among two different types. When the bit is set to '0', data is read at the first clock edge, which is an odd-numbered edge; when it is set to '1', data is read at the second clock edge, which is an even-numbered edge.

The CPOL (Clock polarity control) bit is used to set the default level of the clock signal as active-high or active-low. This does not significantly affect the transfer format. Switching the bit causes the clock signal to be inverted (e.g., Active-high is inverted to active-low; idle-low is inverted to idle-high).

To ensure appropriate communication between the master and slave devices, the two devices must be set in the same operating mode. Therefore, it might be needed to reconfigure the master device to fit the requirements of peripheral slaves.

Figure 104 and Figure 105 illustrate SPI transfer timings when the CPHA is set to ‘0’.

When the CPHA ‘0’, the Master and Slave devices can read data at an odd-numbered (First) clock edge and change data at an even-numbered (Second) clock edge.

The CPOL is used to set the default level of the SCK clock signal. If the CPOL = ‘0’, the default level is set Low; if the CPOL = ‘1’, the default level is set High.

The MSBF bit is used to select among the MSB- or LSB-first transfer modes for output from the MISO line. If the MSBF = ‘1’, data is shifted out bit by bit from the Most Significant Bit down to the Least Significant Bit; if the MSBF = ‘0’, data is shifted out bit by bit from the Least Significant Bit down to the Most Significant Bit.

Once all data has been transferred and the SS pin is set to ‘1’, the SCK clock signal is no longer generated and the MISO and MOSI lines become High or Low, depending on the last data of SPI.

Figure 104. SPI Transfer Timing 1 of 4 (CPHA = ‘0’, CPOL = ‘0’, MSBF = ‘0’)

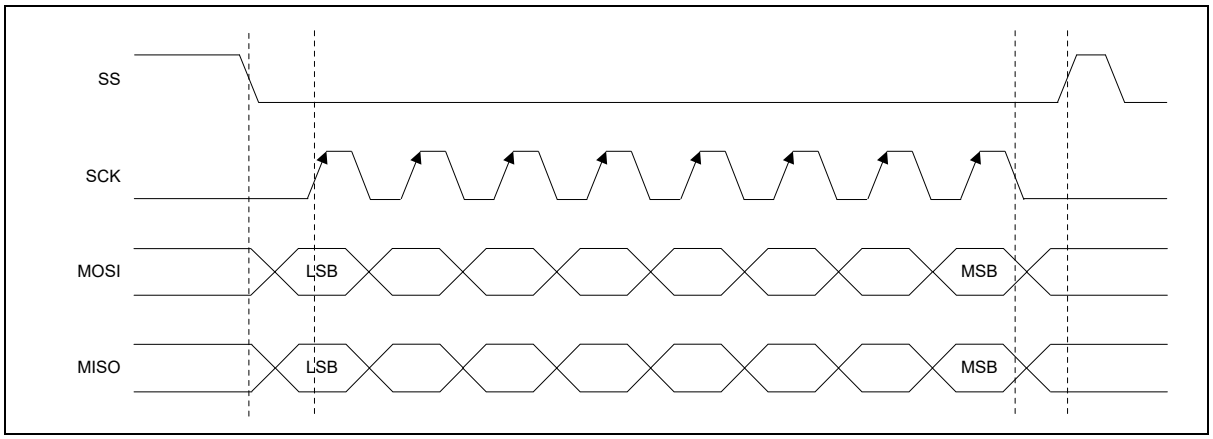


Figure 105. SPI Transfer Timing 2 of 4 (CPHA = ‘0’, CPOL = ‘1’, MSBF = ‘1’)

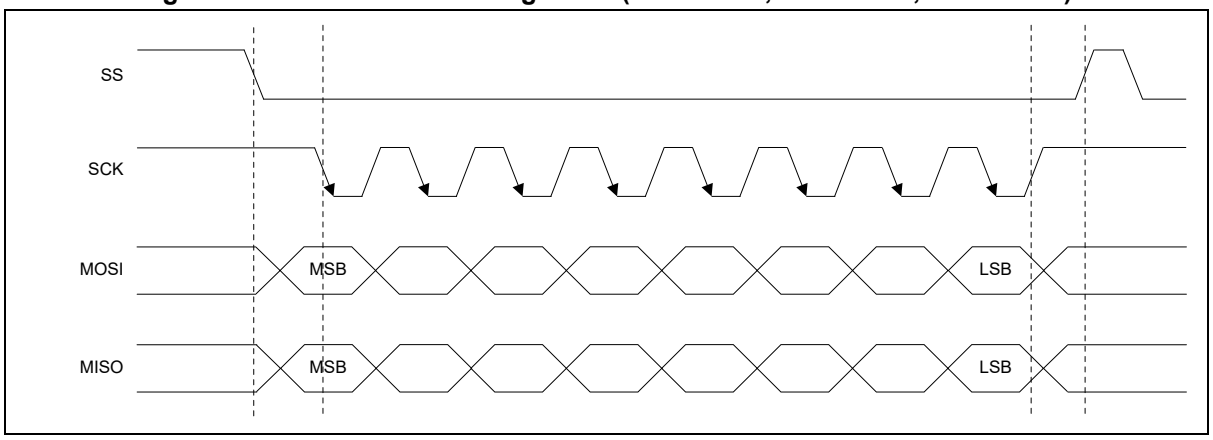


Figure 106 and Figure 107 illustrate SPI transfer timings when the CPHA is set to ‘1’.

When the CPHA = ‘1’, the master and slave devices can read data at an even-numbered (Second) clock edge and change data at an odd-numbered (First) clock edge.

The CPOL is used to set the default level of the SCK clock signal. If the CPOL = ‘0’, the default level is set low; if the CPOL = ‘1’, the default level is set high.

The MSBF bit is used to select among the MSB- or LSB-first transfer modes for output from the MISO line. If the MSBF = ‘1’, data is shifted out bit by bit from the Most Significant Bit down to the Least Significant Bit; if the MSBF = ‘0’, data is shifted out bit by bit from the Least Significant Bit down to the Most Significant Bit.

Once all data has been transferred and the SS pin is set to ‘1’, the SCK clock signal is no longer generated and the MISO and MOSI lines become High or Low, depending on the last data of SPI.

Figure 106. SPI Transfer Timing 3 of 4 (CPHA = ‘1’, CPOL = ‘0’, MSBF = ‘0’)

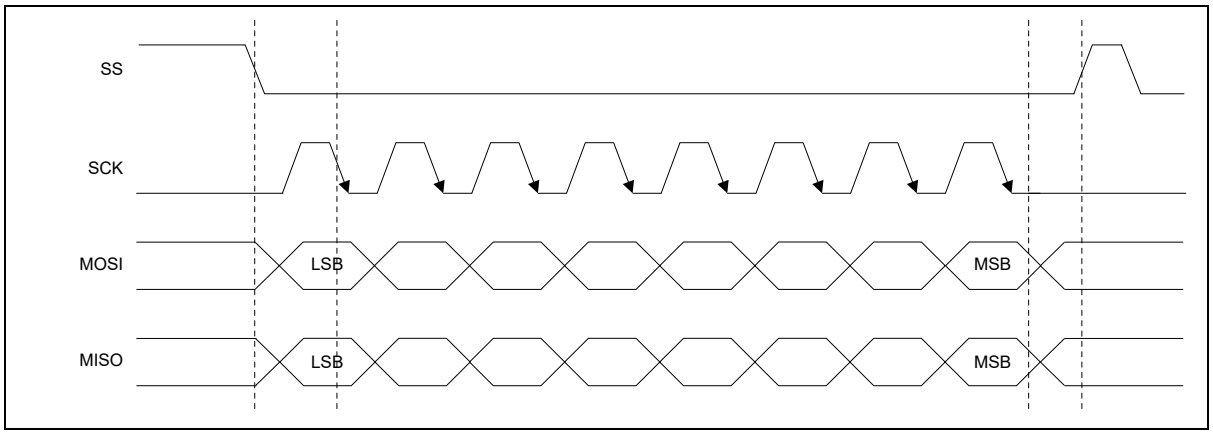
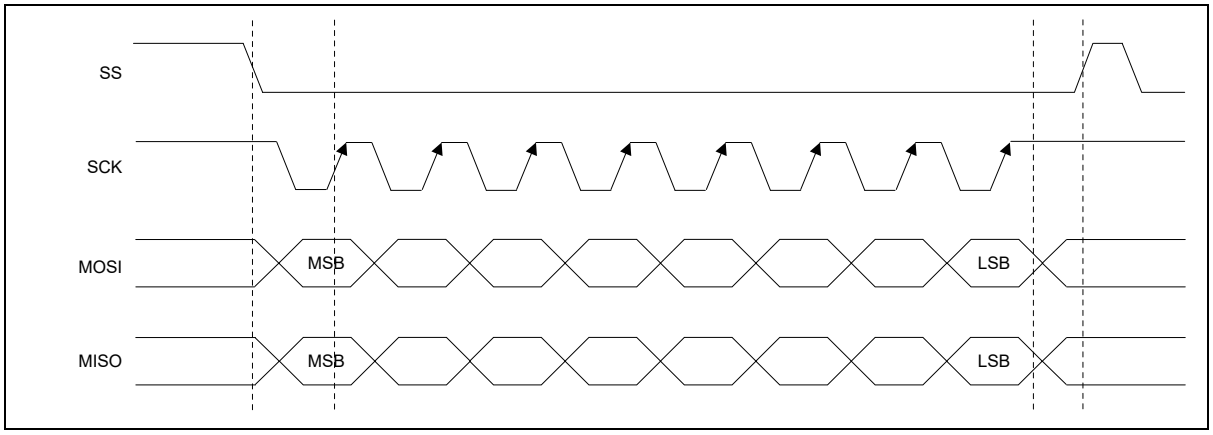


Figure 107. SPI Transfer Timing 4 of 4 (CPHA = ‘1’, CPOL = ‘1’, MSBF = ‘1’)



12.4.5.2 Data Frame Format

A data frame follows the rules below to form a format:

- When the LSB is transferred first, data is transferred starting from bit-0.
- When the MSB is transferred first, data is transferred starting from bit-7.

12.4.6 Configuration of SPI

Typically, users can set the initial setting of the SPI to Master mode or Slave mode by following the procedure below:

1. Set the GPIO pin to fit the alternative function. (MOSI, MISO, SCK, and SS pins)
2. Enable the SPI related bits of the SCU_PER and SCU_PRER registers.
3. Steps 4, 5, …, and 9 (Listed below) describe how to set the SPIn_CR register and can be set in any order.
4. Configure a combination of the CPOL and CPHA bits.
5. Enable the TXBC and RXBC bits of the SPIn_CR register to clear the Transmit Buffer and Receive Buffer, respectively.
6. Set the MS in SPIn_CR register depending on whether the SPI is in Master mode or Slave mode.
7. Set the MSBF in SPIn_CR register to determine whether to transmit the MSB first or the LSB first.
8. Set the BITSZ[1:0] in SPI_CR register to determine the data transfer size.
9. In Master mode, set the SSMO in SPI_CR register to ‘1’ to enable the SS output signal.
10. Write the baud-rate to the SPIn_BR register to determine the SPI communication rate.

When all the above steps are complete, the basic initial setup is concluded. Additional setups such as SPI latency settings and information on how to use DMA functionality are described in the later sections in this chapter.

For more information on the data communication, refer to section 12.4.8

12.4.7 Procedure for Enabling SPI

The SPI_EN register is used to enable or disable the SPI. When the ENABLE bit in the SPIn_EN register is set to ‘1’, the SPI is ready to transmit or receive, based on the SPI settings. The communication and clock start to operate immediately after the SPI is enabled. To process the DMA signal, the DMA-dedicated registers need to be controlled.

When the ENABLE bit is written as ‘1’, the dummy data of the Transmit Buffer will be shifted. To prevent this, data must be written to the SPIn_TDR register before the ENABLE bit is set to be enabled.

12.4.8 Data Transmission and Reception Procedures

12.4.8.1 Sequence Handling

This section describes how to transmit and receive data in master mode or slave mode, in consideration of the interrupt and polling methods.

Transmit methods in master mode are as follows:

[Interrupt Method]

1. When the SPI initial setup is complete, set the ENABLE bit in the SPIn_EN register to '1'.
2. Set the microcontroller core NVIC interrupts.
3. Write the first data to start the transfer to the SPIn_TDR register.
4. Set the TXIE bit in the SPIn_CR register to '1' to enable the transmit interrupt.
5. Process the following in the SPI interrupt service routine.
   A. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
   B. If the transmit buffer is ready, write the data to the SPIn_TDR register.

[Polling Method]

1. When the SPI initial setup is complete, set the ENABLE bit in the SPIn_EN register to '1'.
2. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
   A. If the transmit buffer is ready, write the data to be sent to the SPIn_TDR register.

Receive methods in master mode are as follows:

[Interrupt Method]

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to '1'.
2. Set the microcontroller core NVIC interrupts.
3. Write dummy data to start the transfer to the SPIn_TDR register.
4. Set the RXIE bit in the SPIn_CR register to '1' to enable the receive interrupt.
5. Process the following in the SPI interrupt service routine.
   A. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
   B. If the transmit buffer is ready, write dummy data to the SPIn_TDR register.
   C. Check that the receive buffer is not empty by checking the RRDY flag in the SPIn_SR register.
   D. Read the SPIn_RDR register if the RRDY flag in the SPIn_SR register is '1'.

[Polling Method]

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to '1'.

2. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
   A. If the transmit buffer is ready, write a dummy data to the SPIn_TDR register.
3. Check that the receive buffer is not empty by checking the RRDY flag in the SPIn_SR register.
   A. Read the SPIn_RDR register if the RRDY flag in the SPIn_SR register is '1'.

Transmit and receive methods in slave mode are as follows:

[Interrupt Method]

1. When the SPI initial setup is complete, set the ENABLE in SPIn_EN register to '1'.
2. Set the microcontroller core NVIC interrupts.
3. Write the first data to be sent to the SPIn_TDR register before receiving data.
   (Because the slave device sends data simultaneously when the master device sends data, the slave device must store the data to be sent in advance.)
4. Set the RXIE bit in the SPIn_CR register to '1' to enable the receive interrupt.
5. Process the following in the SPI interrupt service routine.
   A. Check that data is received from the master by checking the RRDY flag in the SPIn_SR register.
   B. If the RRDY flag in the SPIn_SR register is '1', read the received data from the SPIn_RDR register and write the next data to be sent to the SPIn_TDR register.

[Polling Method]

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to '1'.
2. Write the data to be sent to the SPIn_TDR register before receive data.
   (Because the slave device sends data simultaneously when the master device sends data, the slave device must store the data to be sent in advance.)
3. Check that data is received from the master by checking the RRDY flag in the SPIn_SR register.
   A. If the RRDY flag in the SPIn_SR register is '1', read the received data from the SPIn_RDR register and write the data to be sent to the SPIn_TDR register.

12.4.8.2 Procedure for Disabling the SPI

If the SPI is disabled, users need to follow the deactivation procedure described below. It is important that this deactivation procedure is performed before the system enters low-power mode where the clock stops. If the deactivation procedure cannot be performed before the system enters low-power mode, the on-going transaction may be corrupted.

The SPI deactivation procedure is as follows:

1. When the SPI communication is completed, disable the SPI Transmit Interrupt or Receive Interrupt.
2. Set the ENABLE bit in the SPIn_EN register to '0' to disable the SPI.

3. Before restoring the SPI communication, write the next transfer data to the SPIn_TDR register.

12.4.8.3 Data Packing

The SPI transmit and receive data size can be set by setting the BITSZ[1:0] bits in the SPIn_CR register, which can be set by 17-, 16-, 8-,or 7-bit.

12.4.8.4 DMA Handshake

The SPI supports the DMA handshaking operation. To operate a DMA handshake, the DMA registers should first be set. (Refer to chapter 7 for more information.)

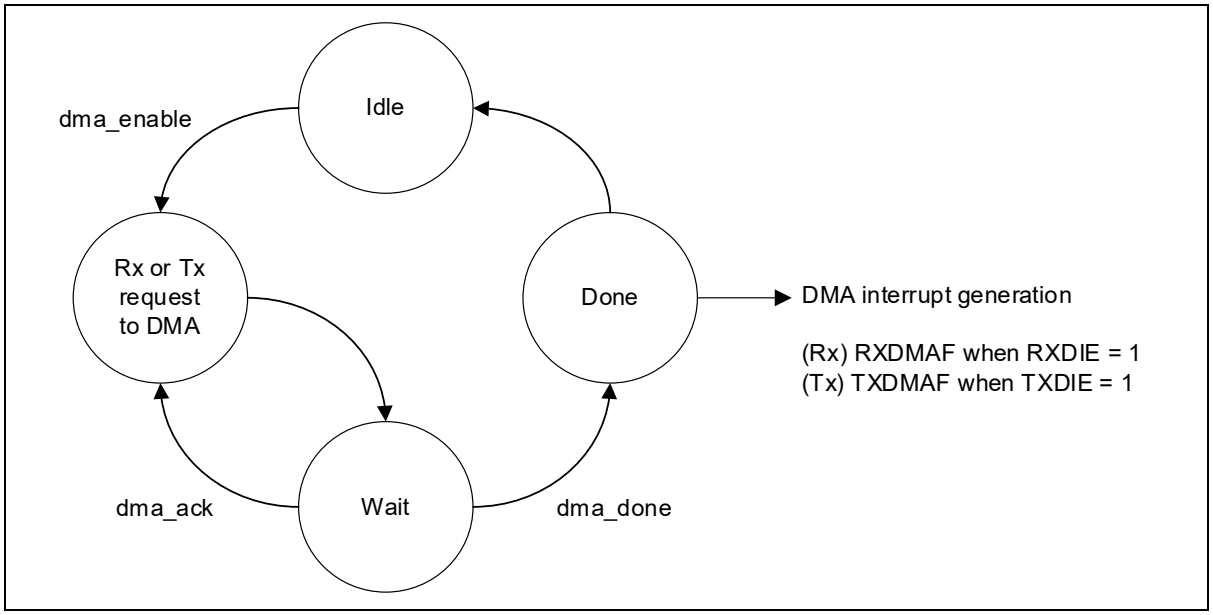
The SPI uses two DMA channels. As the transmitter and receiver are independent of each other, the SPI can operate the two channels at the same time. Once the DMA channel for the receiver is enabled and the Receive Buffer (SPIn_RDR register) is filled, the SPI sends an Rx request to the DMA to empty the buffer and waits for an acknowledge signal from the DMA.

If the Receive Buffer (SPIn_RDR register) is filled again after the acknowledge signal, the SPI sends an Rx request. If the DMA Rx DONE becomes High, the RXDMAF flag in the SPIn_SR register becomes '1' and an interrupt is serviced when the RXDIE bit in the SPIn_CR register is set.

Similarly, if the Transmit Buffer (SPIn_TDR register) is empty after the DMA channel for the transmitter is enabled, the SPI sends a Tx request to the DMA to fill the buffer and waits for an acknowledge signal from the DMA. If the Transmit Buffer is empty again after the acknowledge signal, the SPI sends a Tx request.

If the DMA Tx DONE becomes High, the TXDMAF flag in the SPIn_SR register becomes '1' and an interrupt is serviced when the TXDIE bit in the SPIn_CR register is set. The Slave Transmitter sends dummy data at the first transfer (8 to 17 SCK pulses) in DMA handshake mode.

Figure 108. DMA Handshake State Diagram



12.4.8.5 Communication Diagrams with DMA

The procedure of how to set up the SPI using DMA is shown below:

[DMA TX]

1. Set up the initial settings of DMA and SPI.
   A. Enable the DMAEN bit in the DMAn_SR register.
   B. Write SPI data into memory.
   C. Configure each bit of the DMAn_CR register by referring to the followings.
      i. The TRANSCNT[11:0] bit field determines the number of DMA transfers.
      ii. The PERISEL[4:0] bit field selects the SPIn_Tx.
      iii. The SIZE[1:0] bit field selects the size of a data transfer among byte, half-word, and word.
      iv. The DIR sets the direction from memory to peripheral as TX.
   D. Set the DMAn_PAR register to 0x4000_9000, the address of the SPI_TDR register.
   E. To the DMAn_MAR register, write the initial address of the memory to be written.
2. Enable TXDIE (DMA Tx Done Interrupt Enable) bit in the SPIn_CR register and SPI NVIC settings.
   A. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR register to send the SPI data using DMA.
   B. Once the DMA transfer is started and when value of the TRANSCNT[11:0] bits in the DMAn_CR register reaches ‘0’, the TXDMAF flag in the DMAn_SR register that indicates that the DMA transfer ends is set to ‘1’.

[DMA RX]

1. Set up the initial settings of DMA and SPI.
   A. Enable the DMAEN bit in the DMAn_SR register.
   B. Configure each bit of the DMAn_CR register by referring to the followings.
      i. The TRANSCNT[15:0] bit field determines the number of DMA transfers.
      ii. The PERISEL[4:0] bit field selects the SPIn_Rx.
      iii. The SIZE[1:0] bit field selects the size of a data transfer among byte, half-word, and word.
      iv. The DIR sets the direction from peripheral to memory as RX.
   C. Set the DMAn_PAR register to 0x4000_9000, the address of the SPIn_RDR register.
   D. To the DMAn_MAR register, write the initial address of the memory to be read.
2. Enable RXDIE (DMA Rx Done Interrupt Enable) bit in the SPIn_CR register and SPI NVIC

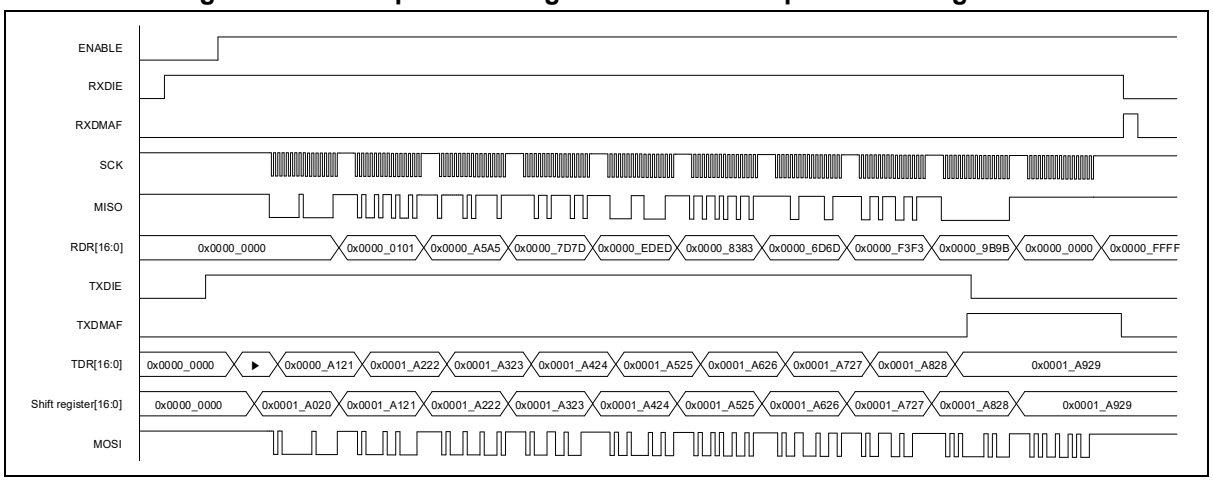
settings.

A. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR register to receive the SPI data using DMA.

B. Once the DMA transfer starts and when the value of the TRANSCNT[11:0] bits in the DMAn_CR register reaches '0', the RXDMAF flag in the DMAn_SR register indicating that the DMA transfer ends is set to '1'.

C. Read SPI data into memory.

Figure 109. Example of Timing of SPI Transfer Operation using DMA



12.4.9 SPI Status Flags

12.4.9.1 Transmit Buffer Empty Flag (TRDY)

The TRDY flag is set to '1' when the Transmit Register is ready for data transmission. If this flag is '0', it means that the SPI is not ready for Tx transfer.

The initial status of this flag is '1', and data transfer is possible.

12.4.9.2 Transmit/Receive Operation Flag (SBUSY)

The SBUSY flag is set to the initial state of '0' when the transmission and reception operations are IDLE. If the transmission or reception is in progress for communication, this flag is set to '1', which indicates busy.

12.4.9.3 Receive Buffer Ready Flag (RRDY)

The RRDY flag is set to '1' when the Receive Register is ready to receive data. If this flag is '0', it means that the Receive Register has no received data.

The initial status of this flag is '0', which requires receiving data.

12.4.9.4 SS Signal Status Flag (SSON)

The SSON flag implies the status of the SS signal. The SSON flag is set to '1' when the SS signal is

active, and the SSON flag is set to ‘0’ when the SS signal is deactivated.

12.4.9.5 Rising or Falling Edge of SS Signal Detect Flag (SSDET)

The SSDET flag is set to ‘1’ when falling edge or rising edge of the SS signal is detected. Writing ‘0’ to this bit field changes the flag value to ‘0’.

12.4.9.6 DMA Transmit Operation Complete Flag (DMA to SPI, TXDMAF)

The TXDMAF flag is set to ‘1’ when DMA transfer operation (Transmission) is completed. During DMA transmitting operation or if DMA is disabled, this flag is set to ’0’.

12.4.9.7 DMA Receive Operation Complete Flag (SPI to DMA, RXDMAF)

The RXDMAF flag is set to ‘1’ when DMA transfer operation (Reception) is completed. During DMA receiving operation or if DMA is disabled, this flag is set to ‘0’.

12.4.10 SPI Error Flags

12.4.10.1 Overrun Flag (OVRF)

The OVRF flag is set to ‘1’ if additional SPI data is received while the SPI_RDR register is not yet read. Reading the SPI_RDR register clears the OVRF flag.

12.4.10.2 Transmit Underrun Error Flag (UDRF)

The UDRF flag is set to ‘1’ when performing the SPI data transmitting operation without data ready to be transmitted to the SPIn_TDR register. Writing data to the SPIn_TDR register clears the UDRF flag.

12.5 SPI Registers

The base addresses and register map of the SPIs are described in the following tables:

Table 105. Base Address of SPI

Name	Base Address
SPI0	0x4000_9000
SPI1	0x4000_9100
SPI2	0x4000_9200

Table 106. SPI Register Map

Name	Offset	Type	Description	Reset Value	Reference
SPIn_TDR	0x0000	WO	SPI Transmit Data Register	-	12.5.1
SPIn_RDR	0x0000	RO	SPI Receive Data Register	0x0000_0000	12.5.2
SPIn_CR	0x0004	RW	SPI Control Register	0x0000_1020	12.5.3
SPIn_SR	0x0008	RC	SPI Status Register	0x0000_0006	12.5.4
SPIn_BR	0x000C	RW	SPI Baud-rate Register	0x0000_FFFF	12.5.5
SPIn_EN	0x0010	RW	SPI Enable Register	0x0000_0000	12.5.6
SPIn_LR	0x0014	RW	SPI Delay Length Register	0x0001_0101	12.5.7

NOTE:

1. n = 0, 1, and 2.

12.5.1 SPIn_TDR: SPI Transmit Data Register

The SPI_TDR register is a 17-bit sized read/write register. It contains serial transmit data.

SPI0_TDR=0x4000_9000, SPI1_TDR=0x4000_9100, SPI2_TDR=0x4000_9200

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
Reserved		TDR[16:0]	
-		0xXXXX	
-		WO	

16 0	TDR[16:0]	Transmit data register

12.5.2 SPIn_RDR: SPI Receive Data Register

The SPI_RDR register is a 17-bit sized read/write register. It contains serial receive data.

SPI0_RDR=0x4000_9000, SPI1_RDR=0x4000_9100, SPI2_RDR=0x4000_9200

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
Reserved		RDR[16:0]	
-		0x0000	
-		WO	

16 0	RDR[16:0]	Receive data register

12.5.3 SPIn_CR: SPI Control Register

The SPI_CR register is a 20-bit sized read/write register and can configure SPI operation mode.

SPI0_CR=0x4000_9004, SPI1_CR=0x4000_9104, SPI2_CR=0x4000_9204

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
Reserved		TXBC	RXBC	DTXIE	DRXIE	SSCIE	TXIE	RXIE	SSMOD	SSOUT	LBE	SSMARK	SS0MO	SS0POL	Reserved	MS	MSBF	CPHA	CPOL	BITSZ[1:0]
-		0	0	0	0	0	0	0	0	1	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. (Auto-clear)	
		0	No action
		1	Clear Tx buffer
19	RXBC	Rx buffer clear bit. (Auto-clear)	
		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	SS edge change interrupt enable bit.	
		0	SS interrupt is disabled.
		1	SS 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 (SPI_CR[12]) SS signal is in Auto output operation mode.
		1	SS output signal is set by SSOUT. (SS Manual output)
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. Receive data when SS signal is active.
		1	SS signal masking is enabled. Receive data at SCK 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.
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(1)	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(1)	SPI clock polarity bit.	
		0	Idle-low clocks selected.
		1	Idle-high clocks selected.
1 0	BITSZ[1:0]	Transmit / Receive data bits select bit.	
		00	8-bit
		01	9-bit
		10	16-bit
		11	17-bit

NOTE:

1. 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

12.5.4 SPIn_SR: SPI Status Register

The SPI_SR register is a 10-bit sized read/write register. It contains the status of SPI interface.

SPI0_SR=0x4000_9008, SPI1_SR=0x4000_9108, SPI2_SR=0x4000_9208

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
Reserved			TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	UDRF	TXIDLE	TRDY	RRDY
-			0	0	0	0	0	0	0	1	1	0
-			RWC0	RWC0	RO	RWC0	RC	RO	RO	RO	RO	RO

9	TXDMAF	DMA Transmit Operation Complete flag. (DMA to SPI)
	0	DMA transmit operation is working or is disabled.
	1	DMA transmit operation is done. This value is cleared by writing ‘1’ to this field.
8	RXDMAF	DMA receive operation complete flag. (SPI to DMA)
	0	DMA receive operation is working or is disabled.
	1	DMA receive operation is done. This value is cleared by writing ‘1’ to this field.
7	SBUSY	Transmit or receive flag
	0	Idle state
	1	Transmit or receive in progress
6	SSDET	The rising or falling edge of SS signal detect flag.
	0	SS edge is not detected.
	1	SS edge is detected. 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. This bit is cleared by reading SPI_RDR.
3	UDRF	Transmit underrun error flag.
	0	Transmit underrun is not occurred.
	1	Transmit underrun is occurred. This bit is cleared by writing SPI_TDR.
2	TXIDLE	Transmit / receive operation flag.
	0	SPI is transmitting data
	1	SPI is in IDLE state.
1	TRDY	Transmit buffer empty flag.
	0	Transmit buffer is busy.
	1	Transmit buffer is ready. This bit is cleared by writing data to SPI_TDR.
0	RRDY	Receive buffer ready flag.
	0	Receive buffer has no data.
	1	Receive buffer has data. This bit is cleared by reading data from SPI_RDR.

12.5.5 SPIn_BR: SPI Baud-rate Register

The SPI_BR register is a 16-bit sized read / write register. Baud-rate can be set by writing the register.

SPI0_BR=0x4000_900C, SPI1_BR=0x4000_910C, SPI2_BR=0x4000_920C

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
Reserved		BR[15:0]	
-		0xFFFF	
-		RW	

1	BR[15:0]	Baud-rate setting bits Baud-rate = PCLK / (BR[15:0] + 1) (BR[15:0] must be set 2 or greater. (BR[15:0] ≥ 2)

NOTE:

1. For SPI speed, it is recommended to set the BR[15:0] value to two or greater so that the SPI input clock is divided by at least 3.
   e.g., PCLK = 24 MHz, BR = 2, SPI freq. = 24 MHz / (2 + 1) = 8 MHz

12.5.6 SPIn_EN: SPI Enable Register

The SPI_EN register is a bit sized read/write register. It contains SPI enable bit.

SPI0_EN=0x4000_9010, SPI1_EN=0x4000_9110, SPI2_EN=0x4000_9210

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
Reserved			ENABLE
-			0
-			RW

0	ENABLE	SPI enable bit	
		0	SPI is disabled. The SPI_SR register is initialized by writing '0' to this bit but other registers aren't initialized.
		1	SPI is enabled. When this bit is written as '1', the dummy data of transmit buffer will be shifted. To prevent this, write data to the SPI_TDR register before this bit is active.

12.5.7 SPIn_LR: SPI Delay Length Register

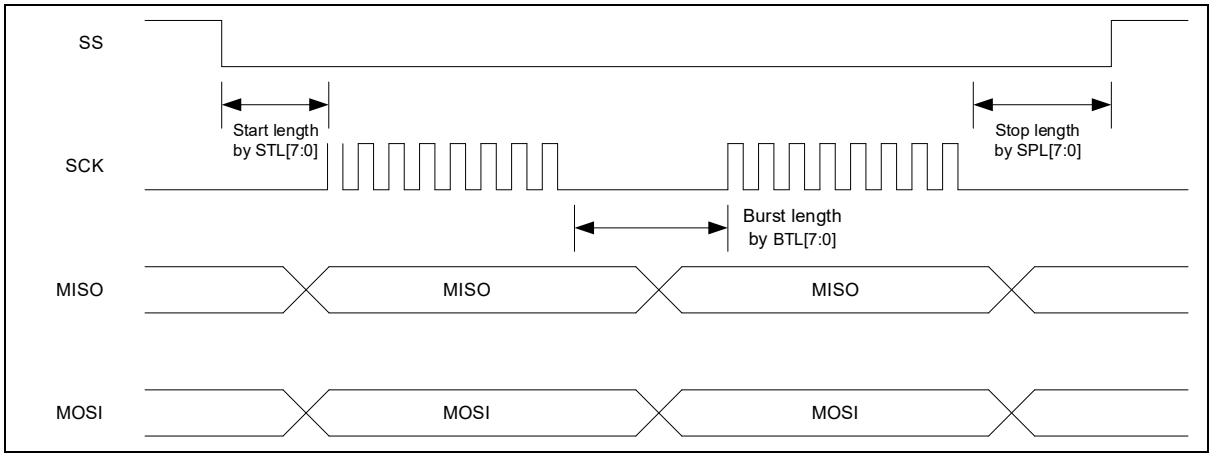
The SPI_LR register contains the length values of Start, Burst, and Stop. The SPI must be disabled in the SPI_EN register before configuring this register.

SPI0_LR=0x4000_9014, SPI1_LR=0x4000_9114, SPI2_LR=0x4000_9214

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
Reserved	SPL[7:0]	BTL[7:0]	STL[7:0]
-	0x01	0x01	0x01
-	RW	RW	RW

23 16	SPL[7:0]	Stop length value 0x01 to 0xFF: 1 to 255 SCKs. (SPL[7:0] ≥ 1)
15 8	BTL[7:0]	Burst length value 0x01 to 0xFF: 1 to 255 SCKs. (BTL[7:0] ≥ 1)
7 0	STL[7:0]	Start length value 0x01 to 0xFF: 1 to 255 SCKs. (STL[7:0] ≥ 1)

Figure 110. SPI Waveforms (STL[7:0], BTL[7:0], and SPL[7:0])(1)



NOTE:

1. Due to the nature of code flash memory, the access time is slower than RAM, causing flash waits. Since the reception of data during SPI high-speed communication performed in code flash memory may be faster than the data access time, so we recommend using a delay between the sending SPI data. By running code in SRAM and using DMA, you can use it without delay in high-speed communication.

$$BTL(burst\ delay) \geq \frac{(Access\ wait + 1) \times SCK \times (xbit + 2)}{HCLK} + \frac{4}{(BR + 1)}$$

12.5.8 SPI Registers Map Summary

Table 107. SPI Register Map Summary

Offset	Register name	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
0x00	SPIn_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[16:0]																
	Reset value																-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x00	SPIn_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPIn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXBC	RXBC	TXDIE	RXDIE	SSCIE	TXIE	RXIE	SSMOD	SSOUT	LBE	SSMARK	SSMO	SSPOL	Res.	Res.	MS	MSBF	CPHA	CPOL	BITSZ[1:0]	
	Reset value												0	0	0	0	0	0	0	0	1	0	0	0	0			1	0	0	0	0	0
0x08	SPIn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	UDRF	TXIDLE	TRDY	RRDY
	Reset value																							0	0	0	0	0	0	0	1	1	0
0x0C	SPIn_BR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	SPIn_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENABLE
	Reset value																																0
0x14	SPIn_LR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPL[7:0]								BTL[7:0]								STL[7:0]							
	Reset value									0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

13. Inter-Integrated Circuit (I2C)

13.1 I2C Introduction

The I2C (Inter-integrated Circuit) interface built in the A34M420 satisfies the standard I2C communication protocol and is used for serial communication with internal and external devices via the I2C protocol. Equipped with two units, it supports both master and slave modes, and is capable of transmitting and receiving data in bytes by using interrupts or polling.

The I2C of the A34M420 operates in Standard mode (100 kHz) or Fast mode (400 kHz) and supports General call.

It helps communicate with various peripherals that have the same bus type. To use the I2C, it is recommended to set the SCL and SDA pins to open-drain and then connect external pull-up resistors to render their output signals ‘HIGH’.

13.2 I2C Main Features

The I2C of the A34M420 features the followings:

- Compliant with I2C protocol
- Two units supported
- Master and slave modes
- Multi-slave mode
- General call
- Transfer rates configurable
  - Maximum transfer rate: 400 kHz
- I2C interrupts
- 7-bit addressing
- Delay time can be set for pin SCL’s high or low waveform
- Hold time can be set for previous data
- Generates and detects STOP, START, and ACK signals
- Noise canceller

13.3 I2C Implementation

I2C module has all features of an I2C peripheral as shown in Table 108.

Table 108. Features of I2C

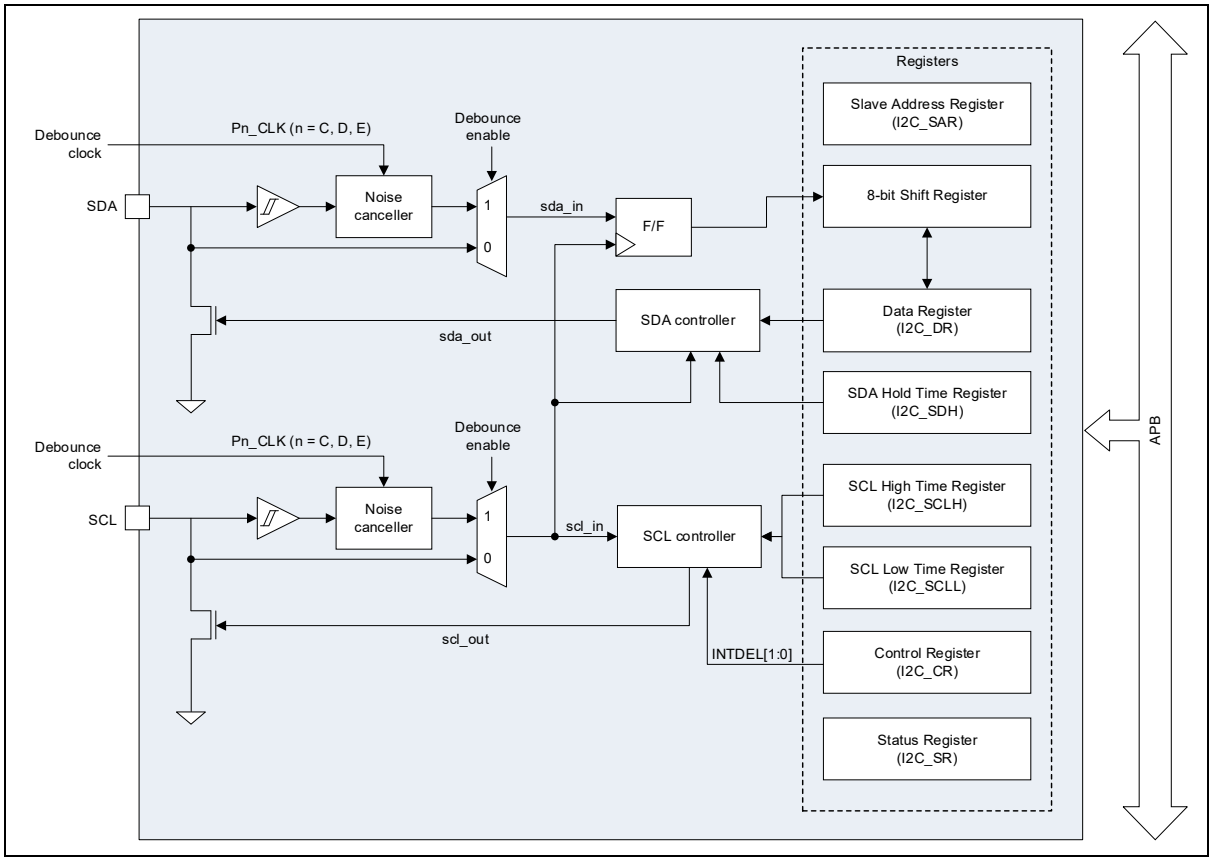
I2C Features	I2C
7-bit addressing mode	O
Standard-mode (up to 100 kbit/s)	O
Fast-mode (up to 400 kbit/s)	O
General call	O

13.4 I2C Functional Description

13.4.1 I2C Block Diagram

Figure 111 shows a block diagram of the I2C module.

Figure 111. I2C Block Diagram



NOTES:

1. When the corresponding port of the I2C is a sub-function for SCL / SDA pin, the SCL / SDA pins are automatically configured in open-drain outputs and the input latch is read in the case of reading the pins. The corresponding internal pull-up resistor could be determined by the port n control register.
2. For more information on the clock source used by the noise canceler, refer to section 4.8.31 or section 4.8.32.

13.4.2 I2C Pins and Signals

Table 109 describes pins assigned for the I2C interface.

Table 109. Pin Assignment of I2C: External Pins

Pin Name	Type	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
SCL0	I/O	I2C channel 0 serial clock bus line (Open-drain)	O	O	O
SDA0	I/O	I2C channel 0 serial data bus line (Open-drain)	O	O	O
SCL1	I/O	I2C channel 0 serial clock bus line (Open-drain)	O	O	N/A
SDA1	I/O	I2C channel 0 serial data bus line (Open-drain)	O	O	N/A

13.4.3 I2C Mode Selection

The I2C module operates in one of the following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

The I2C module supports the four operation modes above and operates as a slave by default. It can operate as a master by configuring the START bit in the I2C_CR register.

START and STOP conditions can be controlled by software, and ACK functionality and General call recognition capability can be controlled by software too.

13.4.4 I2C Initialization

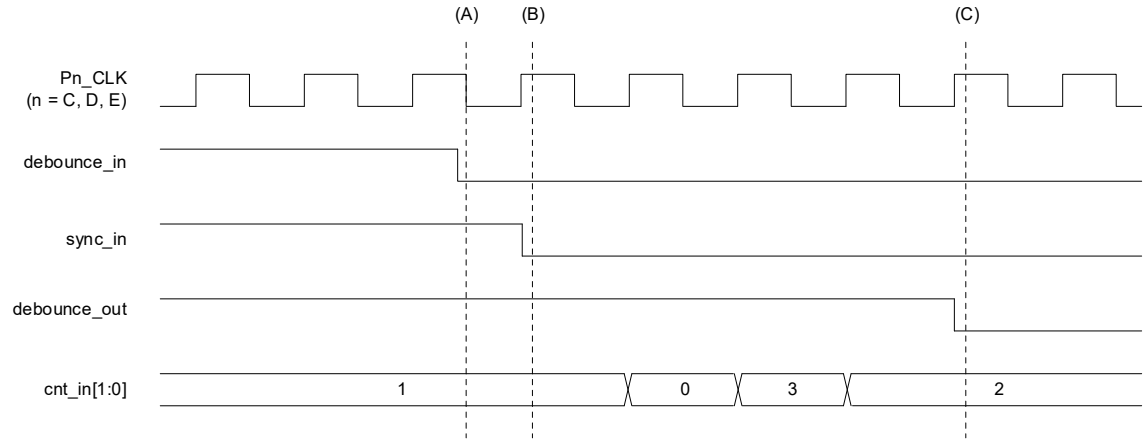
13.4.4.1 Enabling and Disabling the Peripheral

To use the I2C, follow the procedure below:

1. Set the I2C bit in the SCU_PER2 register to ‘1’. This enables the I2C module.
2. Set the I2C bit in the SCU_PCER2 register to ‘1’. This enables the I2C module clock.
3. Configure the I2C_CR register to control the I2C communication.

13.4.4.2 Noise Canceller

Figure 112. Noise Canceller Timing



Pn_CLK: The clock selected by Port n (C, D, E) for using debounce.

debounce_in: Noise canceller input signal

sync_in: The Internal signal that synchronizes the debounce_in signal to the rising of Pn_CLK.

debounce_out: Debounced output signal

cnt_in: Internal 2-bit counter

To apply the noise canceller to I2C, follow the procedure below:

1. Synchronize the debounce_in, input signal to the Noise canceller, with the rising edge of the Pn_CLK to generate the sync_in signal.
2. The sync_in signal checks the internal counter value from the starting point of the (B) section and outputs the filtered value at the point of (C).
3. The time elapsed between the input signal and output signal is calculated by the formular below:

   A. Pn_CLK period × 4 to Pn_CLK × 5 (n = C, D, E)

13.4.4.3 Using Noise Canceller in I2C Module

To enable noise canceller, follow the procedure below.

1. I2C module uses PC, PD, PE. So, select the PGBDCSEL (PC, PD) bits in the SCU_MCCR5 register or the PGCDCSEL (PE, PF, PG) bits in the SCU_MCCR5 register. According to the port to be used and set the clock source for debouncing.
2. Set the PGBDCDIV (PC, PD) bits in the SCU_MCCR5 register or the PGCDCDIV (PE, PF, PG) bits in the SCU_MCCR5 register to divide the clock source for debouncing.
3. Enable debouncing by setting the PDEx (x = 0 to 15) bits in the Px_DER (x = A to G) registers according to the port pin using I2C module.

NOTE:

1. If debounce is used, the I2C may experience a slowdown due to the Clock Stretching that can occur as much as the Debounce Delay. Users can increase the I2C speed by setting the SCLH[15:0] bits in the I2C_SCLH register to the value less than the current setting.

13.4.5 I2C Software Reset

The I2C can be reset by software, that is, users can reset the I2C by configuring the SCU_PER2 register. Writing '0' to the I2C in the SCU_PER2 register resets all related registers to the initialization state.

To use the I2C module again, set the I2Cn in the SCU_PER2 register to '1' again.

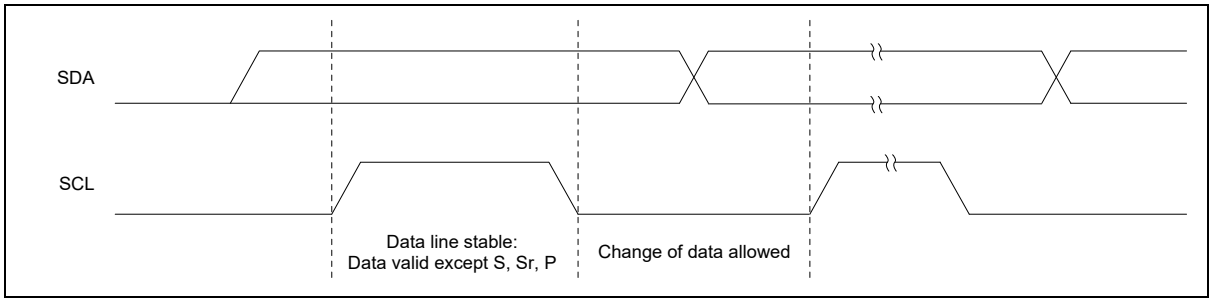
13.4.6 I2C Module Protocol

13.4.6.1 I2C Bit Transfer

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". However, START (S), repeated START (Sr), and STOP (P) occur when the SDA data changes while the SCL signal is high.

Figure 113 shows a timing diagram of the I2C bus bit transfer.

Figure 113. I2C Bus Bit Transfer Timing Diagram



13.4.6.2 START, Repeated START, and STOP

During the operation of the I2C bus, unique situations arise, which are defined as the START (S) and STOP (P) conditions (See Figure 114):

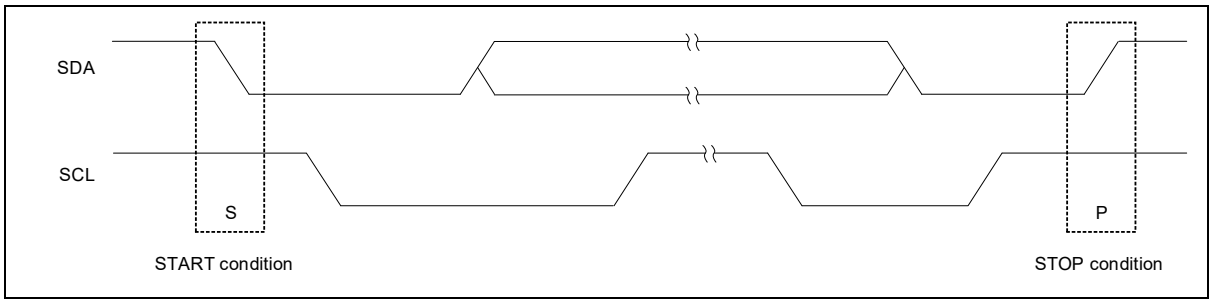
- START (S) condition: An “H” to “L” transition on the SDA line while SCL is “H”
- STOP (P) condition: An “L” to “H” transition on the SDA line while SCL is “H”

These START and STOP conditions are always generated by the master. The bus is considered busy once the START condition occurs and is considered as free again after the STOP condition occurs.

Thus, the bus stays busy between a START and a STOP. If a repeated START (Sr) is generated instead of a STOP, the bus remains busy. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical. For the remainder of this document, therefore, an S will be used to represent both the START and repeated START conditions, unless Sr is particularly relevant.

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

Figure 114. START and STOP Conditions



13.4.6.3 Acknowledge

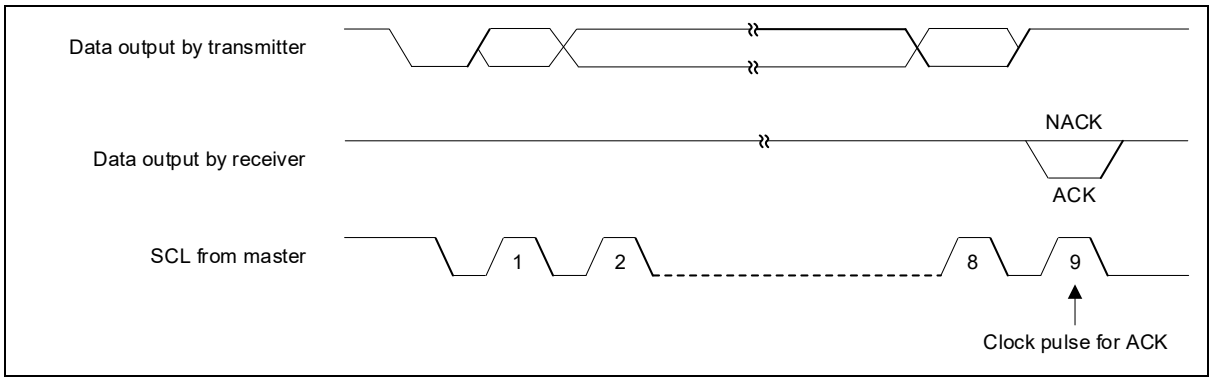
Data transfer with acknowledgement is necessary. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (“H”) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains at stable “L” during the “H” period of this clock pulse. (See Figure 115)

Set-up and hold times must also be considered. When a slave does not 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. Then the master can generate either the STOP condition to abort the transfer, or the repeated START condition to start a new transfer.

If a slave-receiver acknowledges the slave address but can receive no more data bytes later during the transfer, the slave leaves the data line at “H” and the master generates either the STOP or the repeated START condition.

If a master-receiver is involved in a transfer, the slave-transmitter must release the SDA line (“H”) during the acknowledge clock pulse of the end of data. So, the master could generate the STOP or repeated START condition.

Figure 115. I2C Bus Response

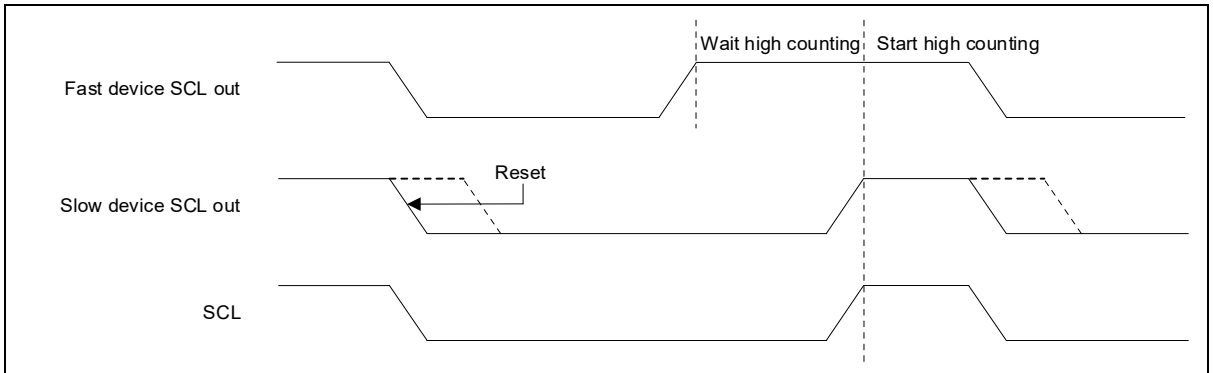


13.4.6.4 Synchronization

All masters generate their own clock on the SCL line to transfer messages through the I2C-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 I2C interfaces to the SCL line. This means that an “H” to “L” transition on the SCL line will cause the devices concerned to start counting off their “L” period and, once a device clock has gone “L”, it will hold the SCL line in that state until the clock “H” state is reached.

Figure 116. Clock Synchronization during Arbitration



NOTE:

1. 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 concerned have counted off their “L” period, the clock line will be released and go “H”. Then, there will 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”.

13.4.6.5 Arbitration

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

Arbitration takes place on the SDA line while the SCL line is in the “H” level in the condition that one master transmits the “H” level but another master is transmitting the “L” level; as a result, the master transmitting the “H” level switches off its DATA output stage because the level on the bus doesn’t correspond to its own level.

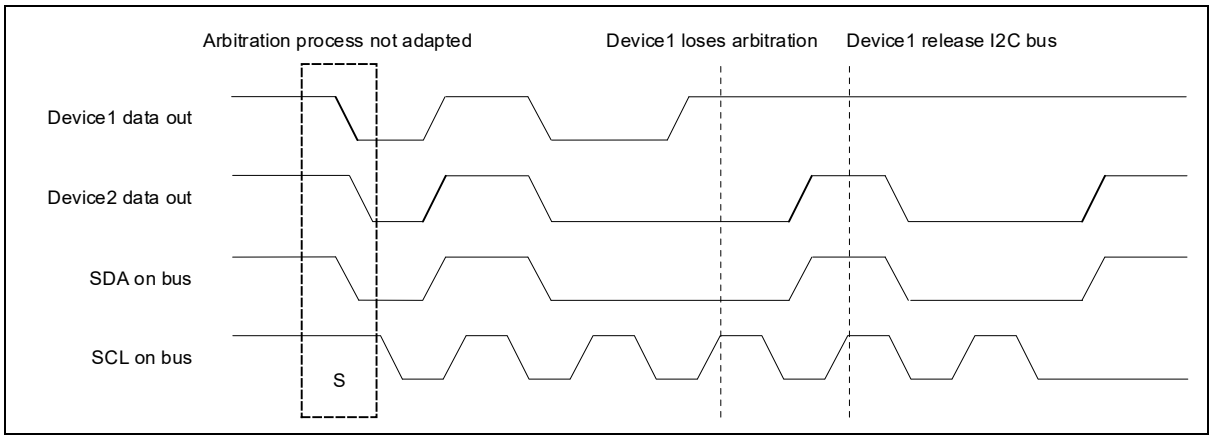
Arbitration can continue for many bits. Its first stage is the comparison of the address bits. If the masters are trying to address the same device, arbitration proceeds with comparing either the data-bits (If they are master-transmitters) or the acknowledge-bits (If they are master-receivers).

Because address and data information on the I2C 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 117 shows the arbitration process for two masters. Of course, more may be involved depending on how many masters are connected to the bus. As soon as a difference is made between the internal data level of the master generating Device1 Data out and the actual level on the SDA line, its data output is switched off, which means that a Device1 release I2C bus. This will not affect the data transfer initiated by the winning master.

Figure 117. Arbitration Process between Two Masters



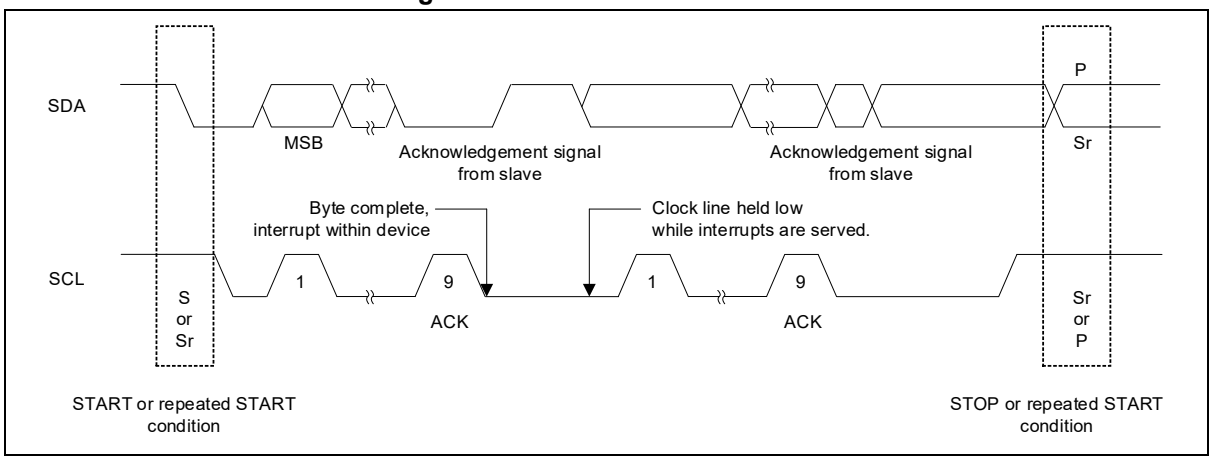
13.4.6.6 Data Transfer

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an acknowledge bit and is transferred MSB first. (Refer to Figure 118.)

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 to “L” to force the master into a wait state. If the slave is ready to transfer another byte of data, releases clock line SCL and master continues data transfer.

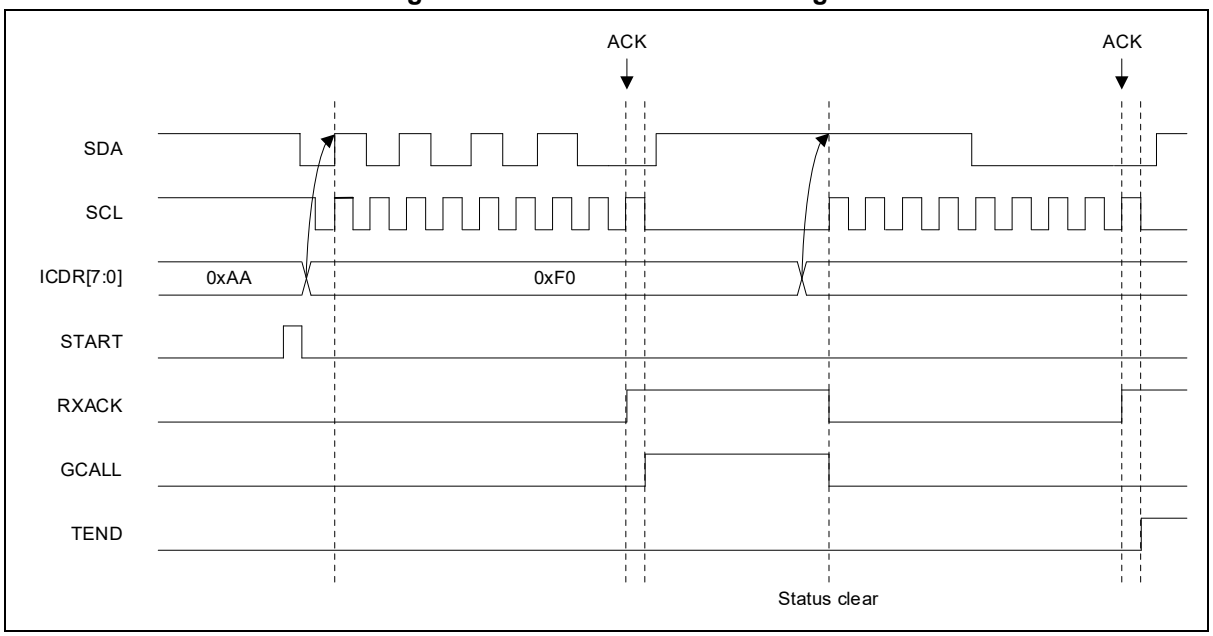
A message starting with such an address can be terminated by an occurrence of the STOP condition, even during the transmission of a byte. In this case, no acknowledgement is generated.

Figure 118. I2C Bus Data Transfer



13.4.6.7 Master Transmit Timing

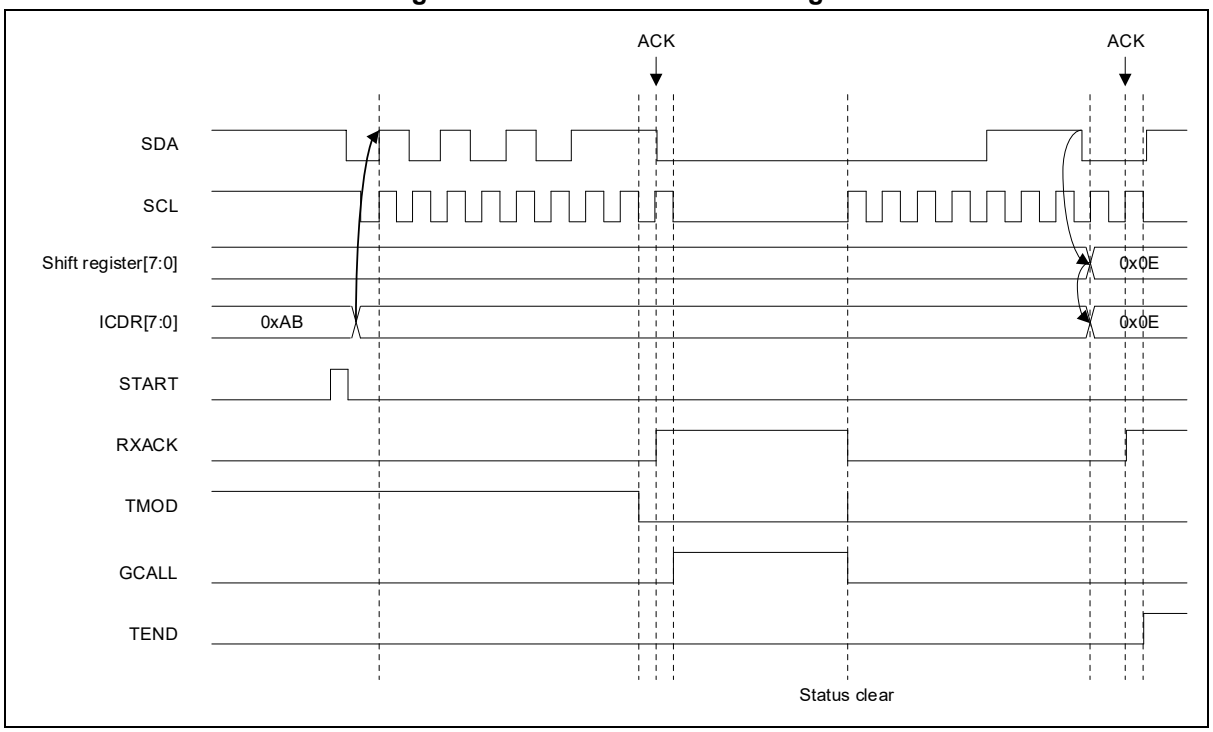
Figure 119. Master Transmit Timing



1. When the value of bit-0 stored in the I2Cn_DR register is ‘0’, if the START bit in the I2Cn_CR register is written as ‘1’, Master operates in Transmit mode.
2. 8-bit data stored in the I2Cn_DR register is transmitted during eight SCL pulses and ACK is received at the rising edge of the 9th pulse.
3. If ACK is normally received, the RXACK flag in the I2Cn_SR register becomes ‘1’ and the GCALL flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse. In Master mode, GCALL flag in the I2Cn_SR register is used as AACK (Address ACK judgment) bit.
4. If AACK is confirmed after address transmission, clear the I2Cn_SR register and transmit the data stored in the I2Cn_DR register again according to the SCL pulse. When data transfer is completed, the TEND flag in the I2Cn_SR register becomes ‘1’.

13.4.6.8 Master Receive Timing

Figure 120. Master Receive Timing

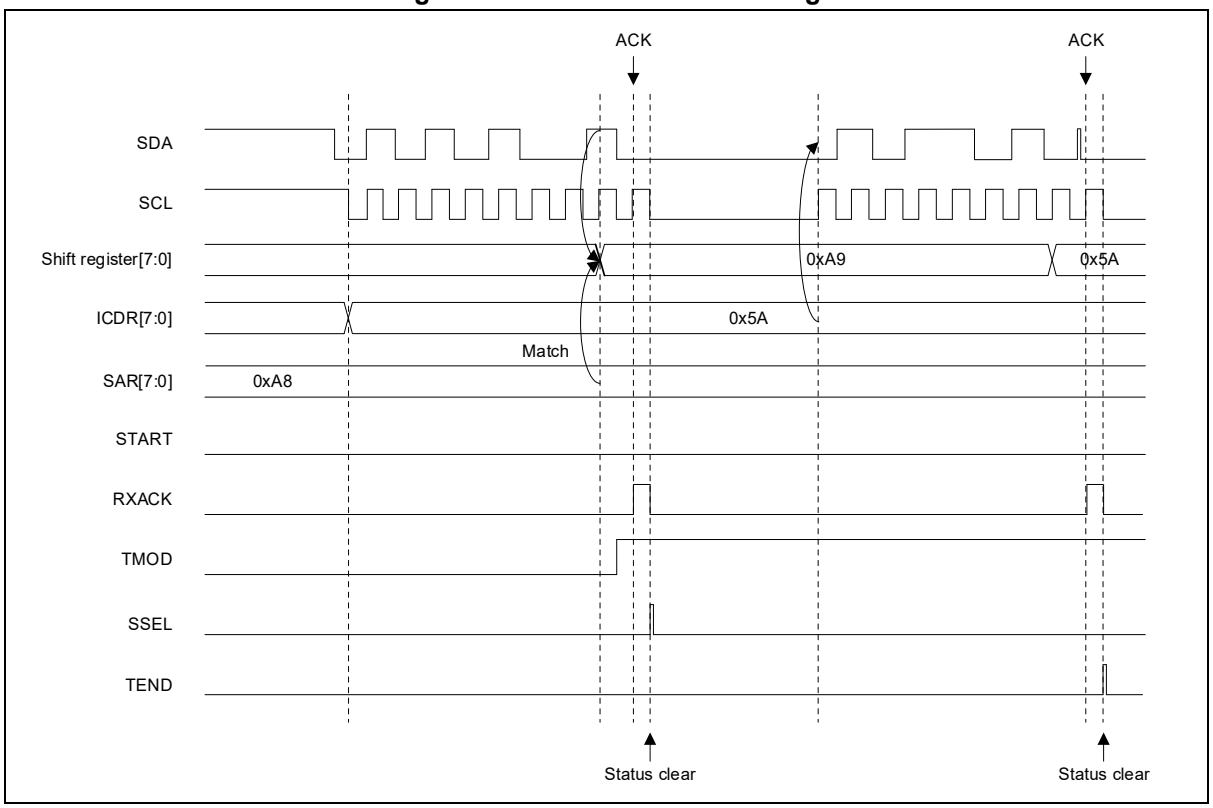


1. When the value of bit-0 stored in the I2Cn_DR register is ‘1’, if START bit in the I2Cn_CR register is written as ‘1’, Master operates in Receive mode.
2. 8-bit data stored in the I2Cn_DR register is transmitted during eight SCL pulses, and TMODE flag in the I2Cn_SR register becomes ‘0’ (Receive mode).
3. If ACK is normally received at the rising edge of the 9th pulse, the RXACK flag in the I2Cn_SR register becomes ‘1’ and the GCALL flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse. In master mode, GCALL flag in the I2Cn_SR register is used as AACK (address ACK judgment) bit.
4. If ACK is confirmed after sending address, clear the I2Cn_SR register and prepare to receive data from slave.
5. The data received through the SDA line is saved in the shift register (SHFTR) at the 8th pulse of SCL and then immediately saved to the I2Cn_DR register without delay.

At the 9th pulse of SCL, the RXACK flag in the I2Cn_SR register becomes ‘1’. When data reception is completed, the TEND flag in the I2Cn_SR register becomes ‘1’.

13.4.6.9 Slave Transmit Timing

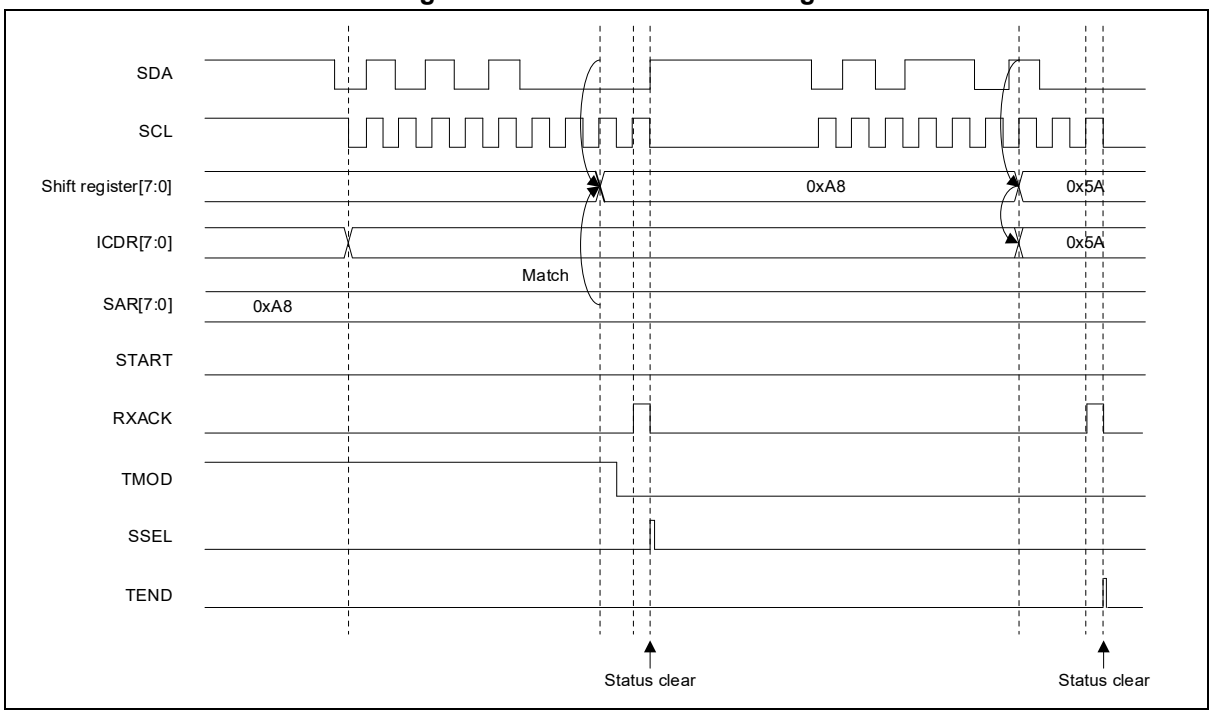
Figure 121. Slave Transmit Timing



1. If the START bit in the I2Cn_CR register is kept as ‘0’, it operates in slave mode.
2. When SDA drops to Low while both SCL and SDA are high, the start condition starts, and the data received through the SDA line is saved in the shift register (SHFTR).
3. If the SHFTR[0] is '1' after the I2Cn_SAR register value and the address (SHFTR[7:1]) value stored in the shift register (SHFTR) match, it is transmit mode, so the TMODE flag in the I2Cn_SR register is '1'.
4. When the address matches, the SSEL flag in the I2Cn_SR register changes to ‘1’ and it is selected as slave, and the data stored in the I2Cn_DR register is transmitted according to the SCL pulse.
5. When all 8-bit data is transmitted, the TEND flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse of SCL.

13.4.6.10 Slave Receive Timing

Figure 122. Slave Receive Timing



1. If the START bit in the I2Cn_CR register is kept as ‘0’, it operates in slave mode.
2. When SDA drops to Low while both SCL and SDA are high, the start condition starts, and the data received through the SDA line is saved in the shift register (SHFTR).
3. If the I2Cn_SAR register value and the Address (SHFTR[7:1]) value stored in the shift register (SHFTR) match and SHFTR[0] is '0' in the GCALL state, it is Receive mode, so the TMODE flag in the I2C_SR register is '0'.
4. When the address matches, the SSEL flag in the I2Cn_SR register is changed to ‘1’ and it is selected as slave and receives 8-bit data from master.
5. When all 8-bit data is received, it is saved to the shift register (SHFTR) and then immediately saved to the I2Cn_DR register without delay.

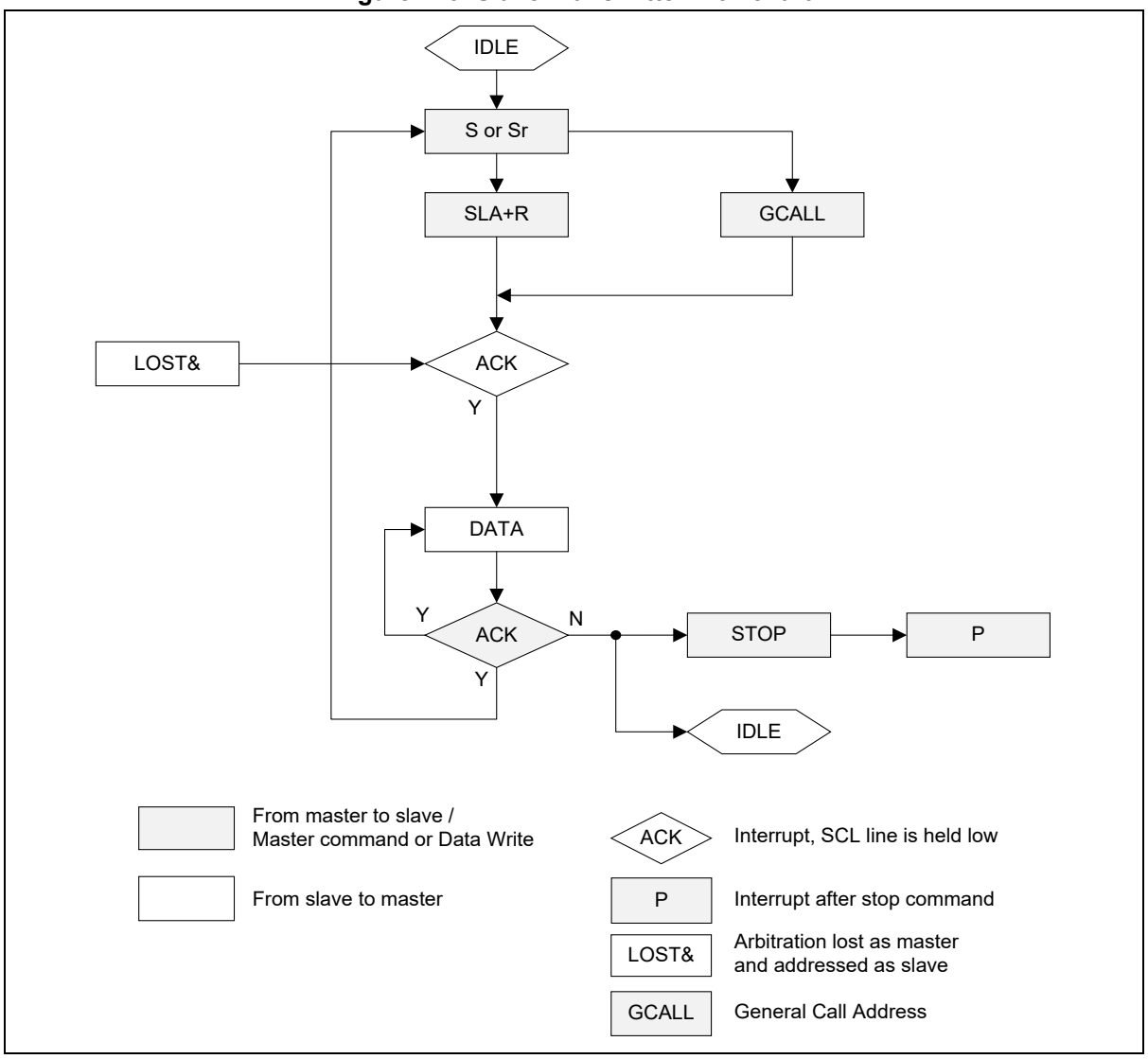
At the 9th pulse of SCL, the RXACK flag in the I2C_SR register becomes ‘1’. When data reception is completed, the TEND flag in the I2Cn_SR register becomes ‘1’.

13.4.7 I2C Slave Mode

13.4.7.1 Slave Transmitter

Figure 123 shows a flowchart of the transmitter in slave mode.

Figure 123. Slave Transmitter Flowchart



To operate the I2C in slave transmitter mode, users must follow the steps below:

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.
2. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
3. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.
4. When the START condition is detected, the I2C receives one byte of data and compares it with value of the SVAD[6:0] in the I2Cn_SAR register. In the case that the GCEN bit in the I2Cn_SAR register is set enabled, the I2C raises the “general call” flag if the received data is

0x00.

5. If the received address does not match the SVAD[6:0] in the I2Cn_SAR register, the I2C becomes idle and waits until another START bit is detected. If the ACKEN bit is set enabled and the received address matches the address in the SVAD[6:0] bit field, the I2C triggers the SSEL interrupt and maintains the SCL line in the “L” level. If the ACKEN bit is set disabled, the I2C becomes idle even though the received address matches the SVAD[6:0] bit field. When the SSEL interrupt occurs, users must load the data in the I2Cn_DR register for transmission and release the SCL line by writing a value 0xFF to the I2Cn_SR register.
6. One byte of data is transmitted.
7. Once the transmission is completed, the I2C triggers the TEND interrupt regardless of an ACK signal from the master, and it maintains the SCL line in the “L” level. The slave performs one of the following operations:

   Case 1. If a NACK signal is detected, the I2C waits for the STOP condition or consecutive START condition.

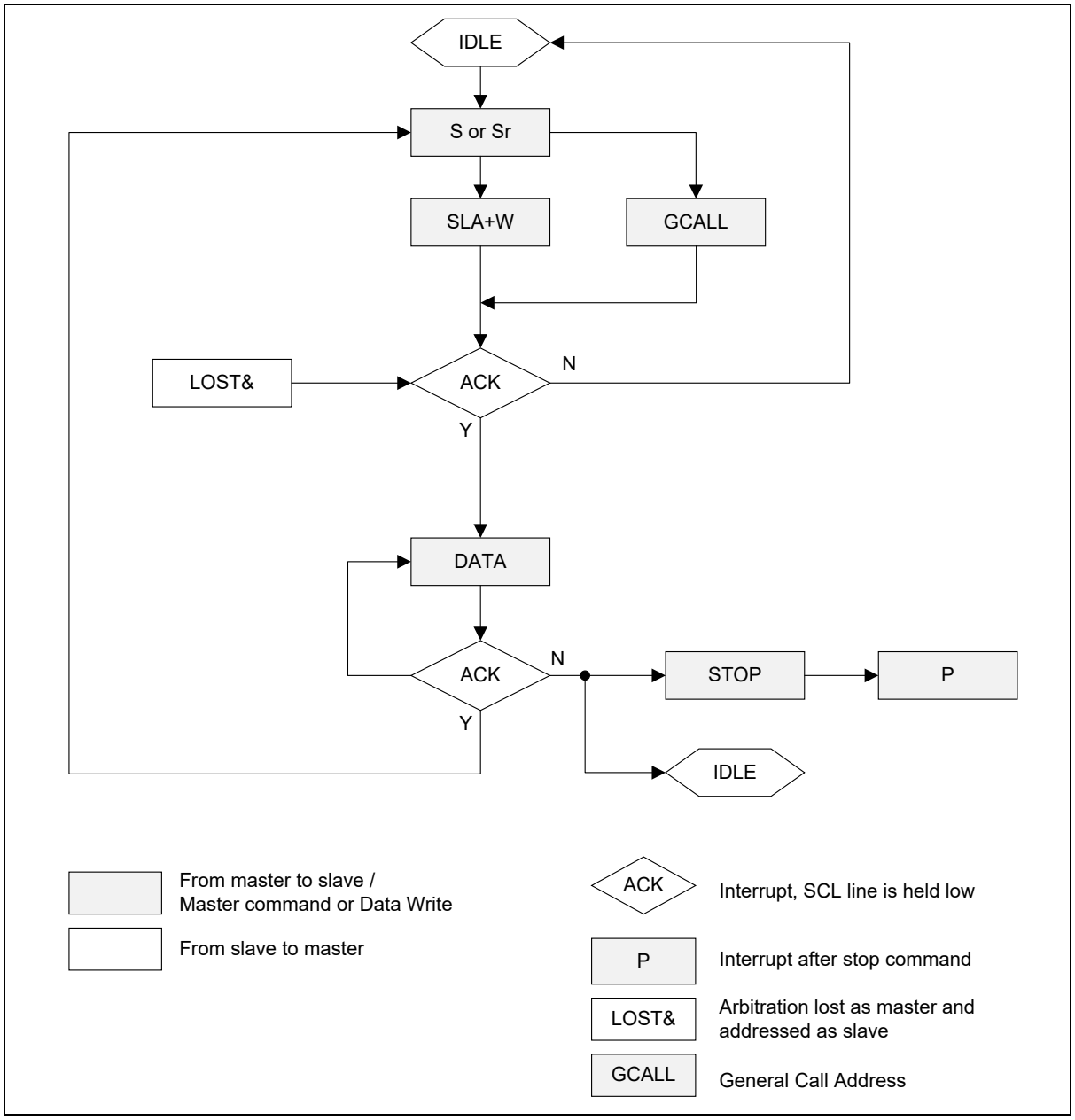
   Case 2. If an ACK signal from the master is detected, the I2C loads the next data in the I2Cn_DR register.

   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.
8. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been completed. To clear the I2C_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

13.4.7.2 Slave Receiver

Figure 124 shows a flowchart of the receiver in slave mode.

Figure 124. Slave Receiver Flowchart



To operate the I2C in slave receiver mode, users must follow the steps below:

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.
2. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.

3. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.
4. When the START condition is detected, the I2C receives one byte of data and compares it with the SVAD[6:0] in the I2Cn_SAR register. In the case that the GCEN bit in the I2Cn_SAR register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.
5. If the received address does not match the SVAD[6:0] in the I2Cn_SAR register, the I2C becomes idle and waits until another START bit is detected. If the ACKEN bit is set enabled and the received address matches the address in the SVAD[6:0] bit field, the I2C triggers the SSEL interrupt and maintains the SCL line in the “L” level. If the ACKEN bit is set disabled, the I2C becomes idle even though the received address matches the SVAD[6:0] bit field. If the SSEL interrupt occurs, the I2C is ready to receive data. Release the SCL line by writing a value 0xFF to the I2Cn_SR register.
6. One byte of data is received.
7. Once the reception is completed, the I2C triggers the TEND interrupt regardless of an ACK signal from the slave, and it maintains the SCL line in the “L” level. The slave performs one of the following operations:

   Case 1. If an ACKEN = ‘0’, a NACK signal is generated and the I2C waits for the STOP condition or consecutive START condition.

   Case 2. If an ACKEN = ‘1’, an ACK signal is generated and the I2C can continue receiving data from the master.

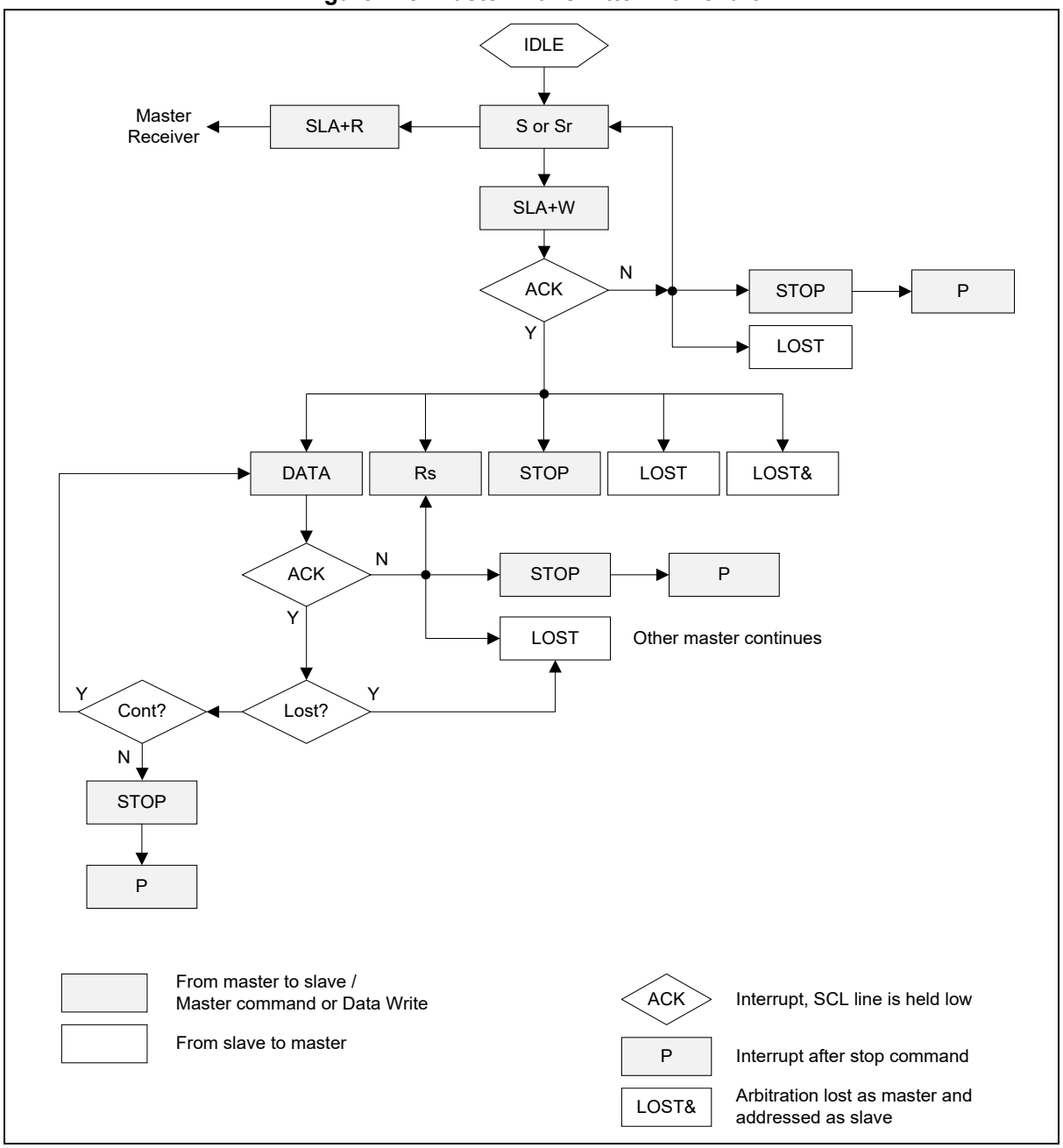
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.
8. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been completed. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

13.4.8 I2C Master Mode

13.4.8.1 Master Transmitter

Figure 125 shows a flowchart of the transmitter in master mode.

Figure 125. Master Transmitter Flowchart



To operate the I2C in master transmitter mode, users must follow the steps below:

1. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the INTEN bit in the I2Cn_CR register. to enable the I2C interrupts.

3. Enter “SLA+W” in the I2Cn_DR register (“SLA” = slave address, “W” = write). “W” must be set to ‘0’ in master transmitter mode. Note that the I2Cn_DR register is used for both address and data.
4. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.
5. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.
6. Write a ‘1’ to the START bit in the I2Cn_CR register to transmit the START condition. And then, configure interrupts and ACK processing conditions. Once the START bit is set enabled, the 8-bit data in the I2Cn_DR register is transmitted at the specified transfer speed.
7. First is the ACK processing sector for the address packet sent from the master. The slave receives a 7-bit address and a 1-bit transmit-receive direction. The master checks for an ACK response in the “H” sector of the ninth clock pulse. If the master has bus priority, the GCALL interrupt occurs, regardless of receiving an ACK. If the I2C loses its bus mastership, the MLOST flag in the I2Cn_SR register is set to ‘1’. And the I2C either waits in idle mode or is operated in slave mode. To operate the I2C in slave mode when it has lost its bus mastership, the ACKEN bit in the I2Cn_CR register must be set to ‘1’ and the SVAD[6:0] in I2Cn_SAR register must be set to the specified slave address. When the I2C operates in slave transmit or receive mode, its SCL must be maintained in the “L” level, so that the I2C can decide whether to continue or stop transmission.

   The following is an example of how to operate the I2C when it does not lose its mastership during the first byte transmission. The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

   Case 1. The master receives an ACK signal from the slave. Thereby, the master sends data to the slave. In this case, the data is written to the I2Cn_DR register for transmission.

   Case 2. Even if an ACK signal has been received, the master can stop transmission. In this case, the STOP in I2Cn_CR register must be set enabled.

   Case 3. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case 1, go to step 8; for Case 2, go to step 10; and for Case 3, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘1’, go to the section on master receiver.
8. One byte of data is transmitted. Bus arbitration is valid during data transmission.
9. Next is the ACK processing sector for the data transmitted by the master. The I2C maintains the SCL in the “L” level. If the I2C’s bus mastership is taken over by another master during data transmission arbitration, the MLOST flag in the I2Cn_SR register is set to ‘1’. In this case, the I2C waits until it becomes idle. Once the data in the I2Cn_DR register is completely transmitted, the I2C triggers the TEND interrupt.

   The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

Case A. The master receives an ACK signal from the slave. Thereby, the master sends data to the slave. In this case, the data is written to the I2Cn_DR register for transmission.

Case B. Even if an ACK signal has been received, the master can stop transmission. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

Case C. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

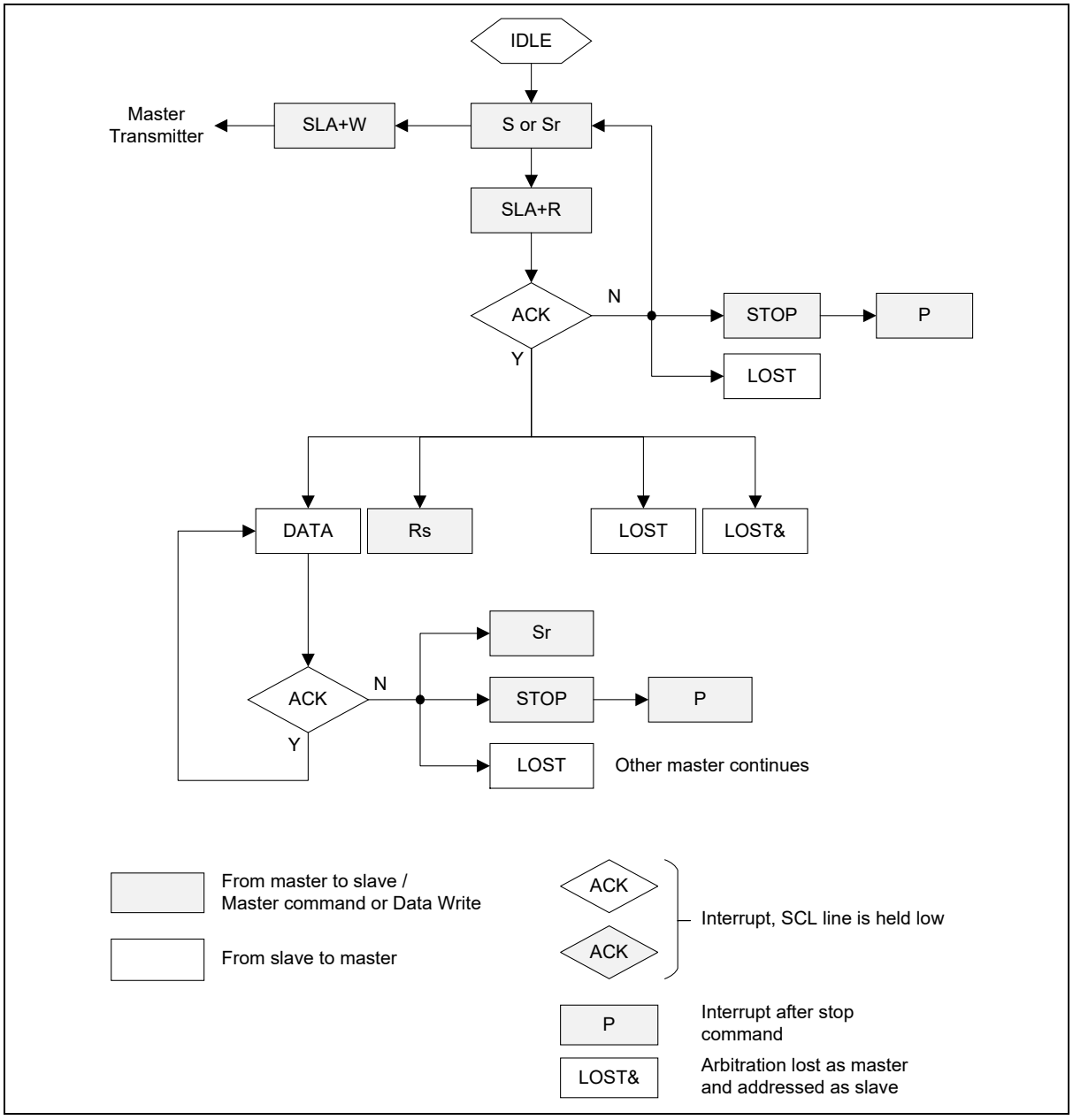
The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case A, go to step 8; for Case B, go to step 10; and for Case C, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘1’, go to the section on master receiver.

10. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been stopped by the STOP bit. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

13.4.8.2 Master Receiver

Figure 126 shows a flowchart of the receiver in master mode.

Figure 126. Master Receiver Flowchart



To operate the I2C in master receiver mode, users must follow the steps below:

1. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.
3. Enter “SLA+R” in the I2C_DR register (“SLA” = slave address, “R” = read). “R” must be set to ‘1’ in master receiver mode. Note that the I2Cn_DR register is used for both address and data.
4. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.

5. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.

6. Write a ‘1’ to the START bit in the I2Cn_CR register to transmit the START condition. And then, configure interrupts and ACK processing conditions. Once the START bit is set enabled, the 8-bit data in the I2Cn_DR register is transmitted at the specified transfer speed.

7. The ACK processing sector for the address packet sent from the master. The slave receives a 7-bit address and a 1-bit transmit-receive direction. The master checks for an ACK response in the “H” sector of the ninth clock pulse. If the master has bus priority, the GCALL interrupt occurs, regardless of receiving an ACK. If the I2C loses its bus mastership, the MLOST flag in the I2Cn_SR register is set to ‘1’. And the I2C either waits in idle mode or is operated in slave mode. To operate the I2C in slave mode when it has lost its bus mastership, the ACKEN bit in the I2Cn_CR register must be set to ‘1’ and the SVAD[6:0] in the I2Cn_SAR register must be set to the specified slave address. When the I2C operates in slave transmit or receive mode, its SCL must be maintained in the “L” level, so that the I2C can decide whether to continue or stop transmission. The following is an example of how to operate the I2C when it does not lose its mastership during the first byte transmission.

   The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

   Case 1. The master receives an ACK signal from the slave. Thereby, the I2C is ready to receive data. In this case, the receive data is read from the I2Cn_DR register.

   Case 2. Even if an ACK signal has been received, the master can stop reception. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

   Case 3. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For case 1, go to step 8; for case 2, go to step 10; and for case 3, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘0’, go to the section on master transmitter.

8. One byte of data is received.

9. The ACK processing sector for the data received from the slave. The I2C maintains the SCL in the “L” level. Once a byte of data is received, the TEND interrupt occurs in the I2C.

   The I2C (master) performs one of the following operations regardless of the RXACK in I2Cn_SR register:

   Case A. The master continues to receive data from the slave. The ACKEN bit in the I2Cn_CR register is set to ‘1’ to transmit an ACK signal so that the slave can transmit the next data.

   Case B. The master receiver stops receiving because an ACK is not generated in the next receive data. To do so, the ACKEN bit in the I2Cn_CR register must be cleared.

   Case C. The master can stop data reception through generating NACK signal. When an ACK signal is not generated, the master stops data transfer. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

Case D. ACK signals are not generated, and the master sends the consecutive START condition. In this case, “SLA+R/W” must be loaded to the I2C_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For case A, go to step 8; for case B and C, go to step 10; and for case D, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘0’, go to the section on master transmitter.

10. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been stopped by the STOP bit. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

13.5 I2C Low-Power Modes

Table 110. Effect of Low-Power Modes on I2C

Mode	Description
SLEEP	No effect
DEEP-SLEEP	The I2C module and clock are disabled. I2C module must be re-initialized after waking from DEEP-SLEEP mode.

13.6 I2C interrupts

Setting the INTEN bit in the I2Cn_CR register to ‘1’ can enable the I2C interrupt. With the INTEN bit enabled, the interrupt is executed as the SCL pulse is output when the master sets the START bit in I2Cn_CR register to ‘1’.

Operation status of the I2C can be monitored by checking the corresponding bits of the I2Cn_SR register, and writing ‘1’ to the bit field clears the corresponding flag. For detailed information of the I2Cn_SR register, refer to section 13.4.7 and section13.4.8.

13.7 I2C Registers

The base addresses of the I2Cs and register map are described in the followings:

Table 111. Base Address of I2C

Name	Base Address
I2C0	0x4000_A000
I2C1	0x4000_A100

Table 112. I2C Register Map

Name	Offset	Type	Description	Reset Value	Reference
I2Cn_DR	0x0000	RW	I2C n Data Register	0x0000_00FF	13.7.1
I2Cn_SR	0x0008	RW	I2C n Status Register	0x0000_0000	13.7.2
I2Cn_SAR	0x000C	RW	I2C n Slave Address Register	0x0000_0000	13.7.3
I2Cn_CR	0x0014	RW	I2C n Control Register	0x0000_0000	13.7.4
I2Cn_SCLL	0x0018	RW	I2C n SCL Low Duration Register	0x0000_FFFF	13.7.5
I2Cn_SCLH	0x001C	RW	I2C n SCL High Duration Register	0x0000_FFFF	13.7.6
I2Cn_SDH	0x0020	RW	I2C n SDA Hold Register	0x0000_7FFF	13.7.8

NOTE:

1. n = 0 and 1.

13.7.1 I2Cn_DR: I2C n Data Register

The I2Cn_DR register is a 32-bit register. It stores received byte-sized data or serial data for transmission.

I2C0_DR=0x4000_A000, I2C1_DR=0x4000_A100

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
Reserved			ICDR[7:0]
-			0xFF
-			RW

7 0	ICDR[7:0]	In transmit mode, the data stored in this register is output from pin SDA in the serial data line. In receive mode, data received at pin SDA is stored in this register; hence, the stored data is loaded by reading this register.

13.7.2 I2Cn_SR: I2C n Status Register

The I2Cn_SR register is a 32-bit read/write accessible register. It displays the status of the I2C bus interface. Writing to the register clears the status bits. Once an I2C interrupt other than the stop interrupt occurs, the SCL line is set low. To release this SCL setting, a value 0xFF must be written to the SR register. This will clear the status of the GCALL, TEND, STOP, SSEL, MLOST, and RXACK bits.

I2C0_SR=0x4000_A008, I2C1_SR=0x4000_A108

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
Reserved			GCALL	TEND	STOP	SSEL	MLOST	BUSY	TMOD	RXACK
-			0	0	0	0	0	0	0	0
-			RWC1	RWC1	RWC1	RWC1	RWC1	RO	RO	RWC1

7	GCALL	This bit signifies different things, depending on whether the I2C is in master or slave mode. (If the I2C module functions as a master, the bit represents whether the ACK address (AACK) has been received from a slave. If the I2C module is a slave, it indicates whether a general call has been detected.)	
		0	Master mode: AACK has not been received.
		1	Master mode: AACK has been received.
		0	Slave mode: A general call has not been detected.
		1	Slave mode: A general call has been detected.
6	TEND	One-byte transmit end flag	
		0	Transmit is in progress.
		1	Transmit has been completed.
5	STOP	Stop flag	
		0	No stop has been detected.
		1	A stop has been detected.

4	SSEL	Slave flag	
		0	The module has not been selected as a slave.
		1	The module has been selected as a slave.
3	MLOST	Mastership loss flag	
		0	Mastership has not been lost.
		1	Mastership has been lost.
2	BUSY	Bus busy flag	
		0	The I2C bus is idle.
		1	The I2C bus is busy.
1	TMOD	Transmit / receive mode flag	
		0	Receive mode
		1	Transmit mode
0	RXACK	Rx ACK flag	
		0	An Rx ACK has not been occurred.
		1	An Rx ACK has been occurred.

13.7.3 I2Cn_SAR: I2C n Slave Address Register

The I2Cn_SAR register is an 8-bit readable and writable register. The first seven bits store the address selected when the I2C module operates as a slave.

I2C0_SAR=0x4000_A00C, I2C1_SAR=0x4000_A10C

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
Reserved			SVAD[6:0]	GCEN
-			0x00	0
-			RW	RW

7 1	SVAD[6:0]	7-bit slave address	
0	GCEN	Enable or disable general call	
		0	Dismisses the general call address.
		1	Receives the general call address.

13.7.4 I2C_CR: I2C n Control Register

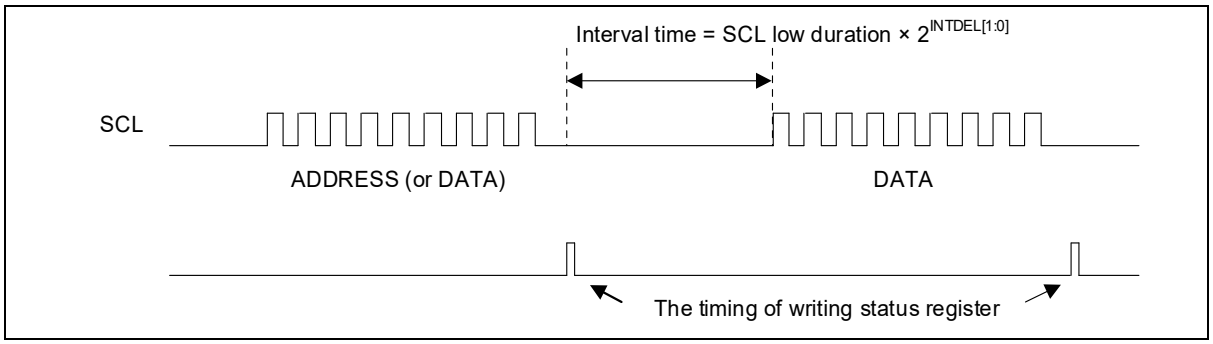
The I2Cn_CR register is a 32-bit read/write accessible register. It sets the operating modes and enablement of the I2C module.

I2C0_CR=0x4000_A014, I2C1_CR=0x4000_A114

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
Reserved		INTDEL[1:0]	IIF	Reserved	INTEN	ACKEN	Reserved	STOP	START
-		00	0	-	0	0	-	0	0
-		RW	RO	-	RW	RW	-	RW	RW

9 8	INTDEL[1:0]	Internal delay between address and data transfers (Or between two data transfers)
	0	1 × SCL low duration
	1	2 × SCL low duration
	2	4 × SCL low duration
	3	8 × SCL low duration
7	IIF	Interrupt flag
	0	No interrupt has occurred, or the flagged interrupt has been cleared.
	1	An interrupt has occurred.
4	INTEN	Enable or disable interrupts
	0	Disables interrupts.
	1	Enables interrupts.
3	ACKEN	Enable ACK in receive mode
	0	No ACK signal is generated.
	1	ACK signal is generated.
1	STOP	Enable or disable Stop. If this bit is set to '1' in transmit mode, the next transmission is stopped even if an ACK signal has been received.
	0	Disables stop.
	1	Enables stop. Transmission is stopped when this bit is set to '1'.
0	START	Transmission starts in master mode.
	0	The I2C waits in slave mode.
	1	The I2C starts transmitting in master mode.

Figure 127. Internal Delay (INTDEL[1:0]) in Master Mode



13.7.5 I2Cn_SCLL: I2C n SCL Low Duration Register

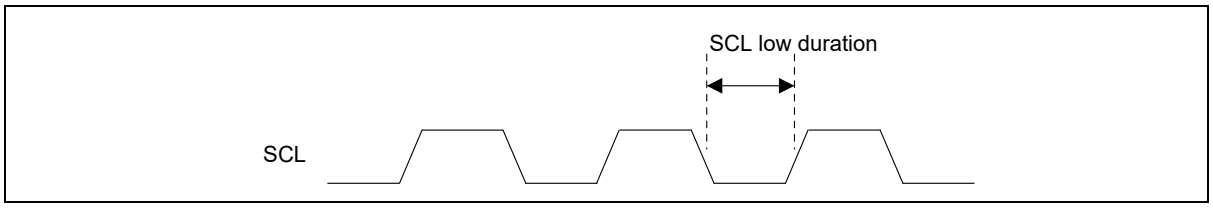
The I2Cn_SCLL register is a 32-bit register. The SCL low duration time can be set in master mode.

I2C0_SCLL=0x4000_A018, I2C1_SCLL=0x4000_A118

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
Reserved		SCLL[15:0]	
-		0xFFFF	
-		RW	

15 0	SCLL[15:0]	SCL low duration value SCL low duration = (PCLK period × SCLL[15:0]) + 2 × PCLK period

Figure 128. SCL Low (SCLL[15:0]) Timing



13.7.6 I2Cn_SCLH: I2C n SCL High Duration Register

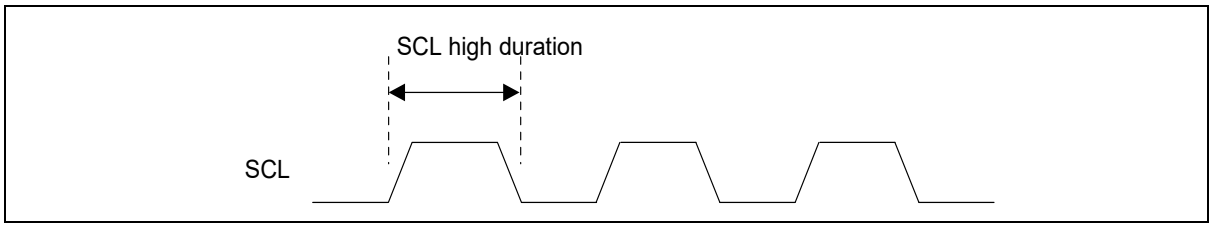
The I2Cn_SCLH register is a 32-bit register. The SCL high duration time can be set in master mode.

I2C0_SCLH=0x4000_A01C, I2C1_SCLH=0x4000_A11C

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
Reserved		SCLH[15:0]	
-		0xFFFF	
-		RW	

15 0	SCLH[15:0]	SCL high duration value SCL high duration = (PCLK period × SCLH[15:0]) + 3 × PCLK period

Figure 129. SCL High (SCLH[15:0]) Timing



13.7.7 I2C_SDH: I2C n SDA Hold Register

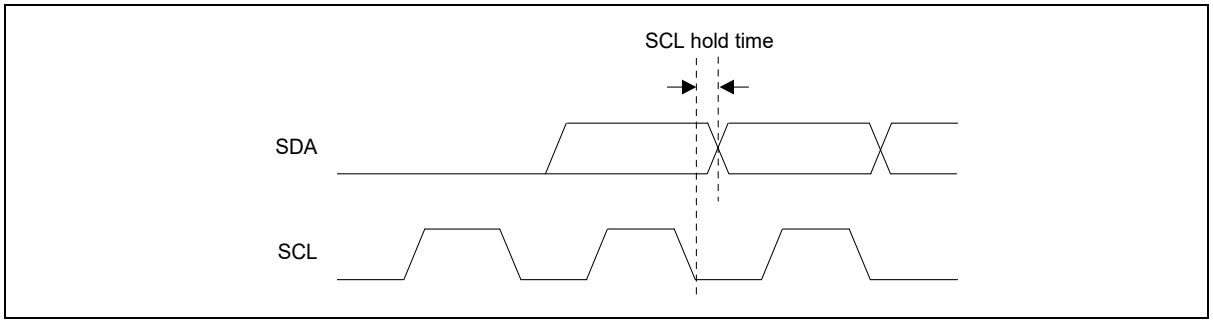
The I2C_SDH register is a 15-bit R/W register. The SCL hold time can be set in master mode.

I2C0_SDH=0x4000_A020, I2C1_SDH=0x4000_A120

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
Reserved		SDH[14:0]	
-		0x7FFF	
-		RW	

14 0	SDH[14:0]	SDA hold time value SDA hold time = (PCLK period × SDH[14:0]) + 4 × PCLK period

Figure 130. SDA hold (SDH[14:0]) timing



13.7.8 I2C Register Map Summary

Table 113. I2C Register Map Summary

Offset	Register name	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
0x00	I2Cn_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ICDR[7:0]							
	Reset value																									1	1	1	1	1	1	1	1
0x08	I2Cn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GCALL	TEND	STOP	SSEL	MLOST	BUSY	TMOD	RXACK
	Reset value																									0	0	0	0	0	0	0	0
0x0C	I2Cn_SAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SVAD[6:0]							GCEN
	Reset value																									0	0	0	0	0	0	0	0
0x14	I2Cn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTDEL[1:0]		IIF	Res.	Res.	INTEN	ACKEN	Res.	STOP	START
	Reset value																							0	0	0			0	0		0	0
0x18	I2Cn_SCLL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SCLL[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x1C	I2Cn_SCLH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SCLH[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x20	I2Cn_SDH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDH[14:0]														
	Reset value																		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

14. Motor Pulse-Width Modulation (MPWM)

14.1 MPWM Introduction

The A34M420 supports two Motor Pulse-Width Modulation (MPWM) units. The MPWM in the A34M420 operates total of three PWM modes as shown below to support various applications:

- Motor PWM mode for motor control
- Normal PWM mode for a variety of PWM applications
- Individual PWM mode for specific applications such as IH cooker

In Motor or Normal PWM mode, the MPWM module of the A34M420 has many different PWM control functionalities such as up to six channels' outputs, dead-time, up to six ADC triggers, a variety of interrupt event generations in many different conditions, protection events, and so on.

In Individual PWM mode, unlike Motor and Normal PWM modes, the MPWM module operates separately for each phase, and controls up to three outputs individually.

14.2 MPWM Main Features

The MPWM module of the A34M420 has the following key features in Motor and Normal PWM modes:

- Outputs from up to six channels
- 16-bit counter that repeats operation
- Up count / up-down count mode
- Symmetrical / asymmetrical mode is supported.
- Dead-time for half-bridge circuit is supported.
- Six ADC trigger outputs
- Interval interrupt mode
- Processing of the protection and overvoltage events

The key features in Individual PWM mode are listed below:

- 16-bit counter that operates independently at each phase of U, V and W
- Dead-time that can be set independently at each phase of U, V and W
- Interrupt that is supported independently at each phase of U, V and W. (Protection / overvoltage interrupts are assigned one interrupt vector per MPWM module.)
- Protection event and overvoltage event that operate independently at each phase of U, V and W, and follow-up processing of the events are supported. (Input pins for the V, W protection / overvoltage events are common in the MPWM0 and MPWM1.)
- Capture and sub-capture functionalities that measure intervals between rising / falling / both edges are supported.

14.3 MPWM Functional Description

14.3.1 MPWM Block Diagram

Figure 131 shows a block diagram of MPWM.

Figure 131. MPWM Block Diagram

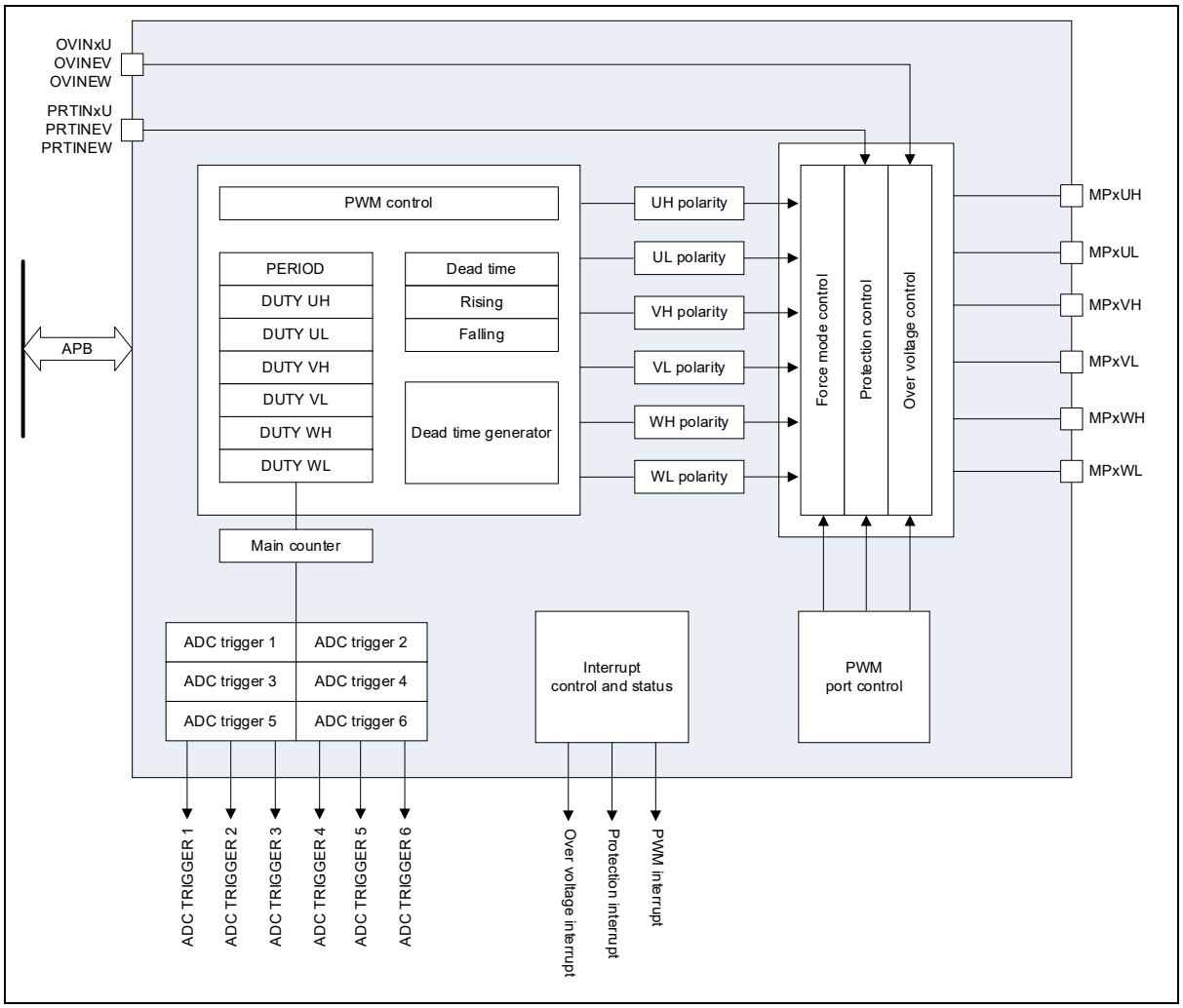


Figure 132 shows a block diagram of the MPWM in Individual PWM mode.

Figure 132. MPWM Block Diagram (Individual PMW Mode)

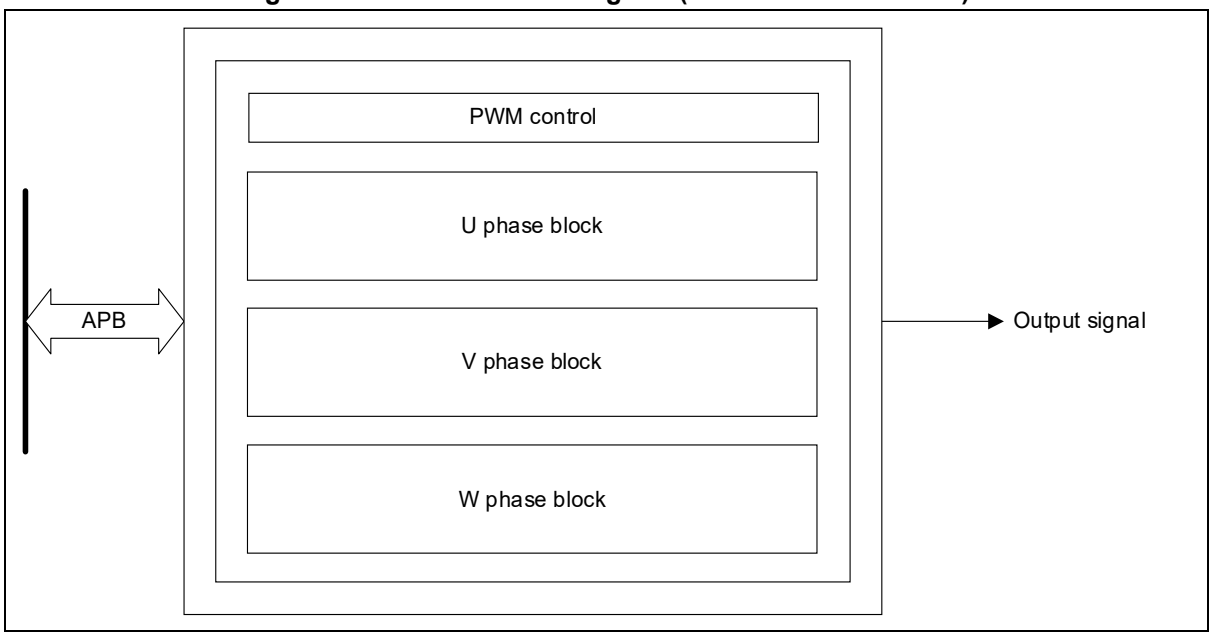
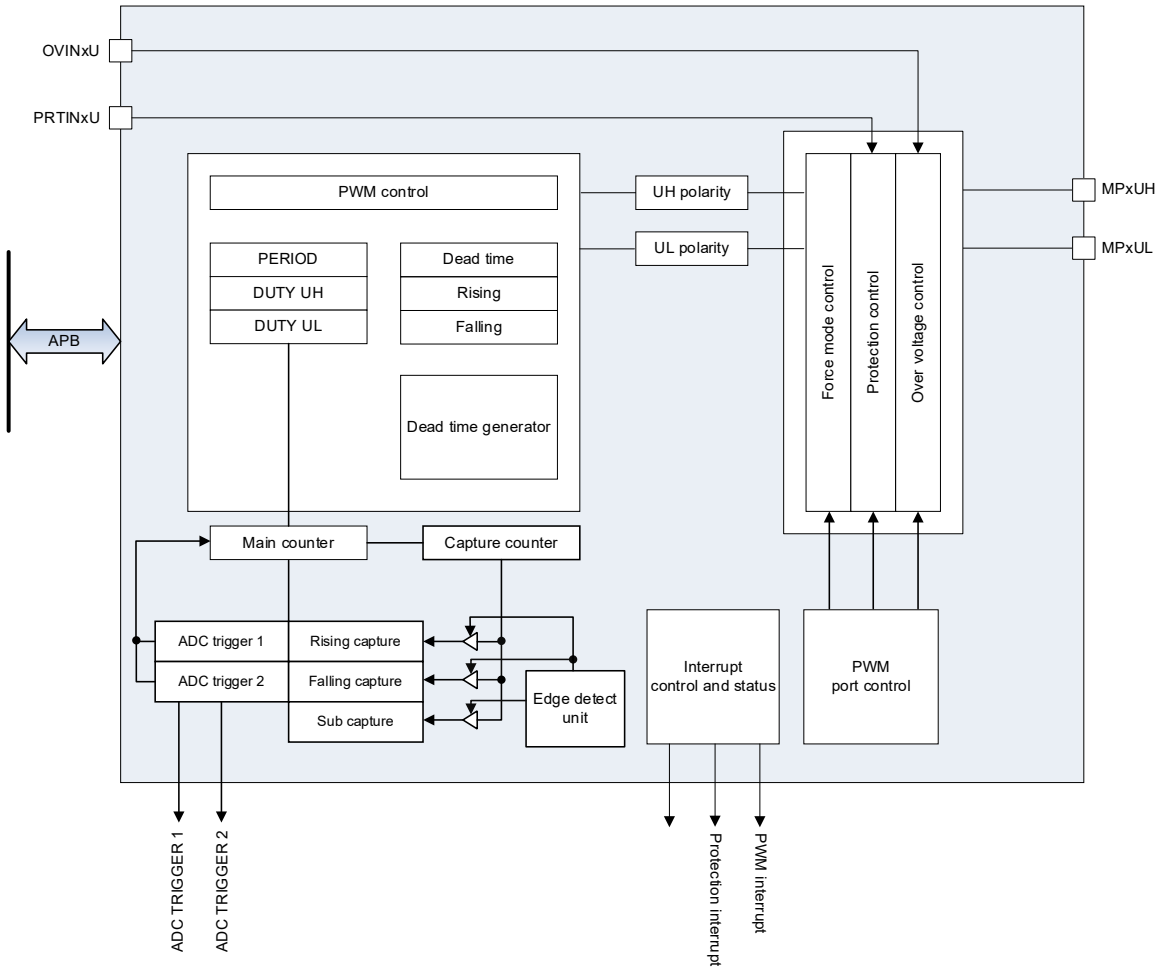


Figure 133 shows a block diagram of the MPWM at phase U.

Figure 133. MPWM Block Diagram (phase U)



NOTE:

The figure represents the phase U block only. The phase V and W blocks are identical to the phase U block. Each phase block can be configured independently; however, the protection and overvoltage interrupts are used in common.

14.3.2 Pins and Internal Signals

Table 114 describes external pins connected to the MPWM module.

Table 114. Pin Assignment of MPWM: External Pins

Pin Name	Type	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
MP0UH	O	MPWM0 high side output port of phase U	O	O	O
MP0UL	O	MPWM0 low side output port of phase U	O	O	O
MP0VH	O	MPWM0 high side output port of phase V	O	O	O
MP0VL	O	MPWM0 low side output port of phase V	O	O	O
MP0WH	O	MPWM0 high side output port of phase W	O	O	O
MP0WL	O	MPWM0 low side output port of phase W	O	O	O
MP1UH	O	MPWM1 high side output port of phase U	O	O	O
MP1UL	O	MPWM1 low side output port of phase U	O	O	O
MP1VH	O	MPWM1 high side output port of phase V	O	O	O
MP1VL	O	MPWM1 low side output port of phase V	O	O	O
MP1WH	O	MPWM1 high side output port of phase W	O	O	O
MP1WL	O	MPWM1 low side output port of phase W	O	O	O
PRTIN0U	I	Protection input pin dedicated to MPWM0 phase U in individual PWM mode	O	O	O
PRTIN1U	I	Protection input pin dedicated to MPWM1 phase U in individual PWM mode	O	O	O
OVIN0U	I	Overvoltage input pin dedicated to MPWM0 phase U in individual PWM mode	O	O	O
OVIN1U	I	Overvoltage input pin dedicated to MPWM1 phase U in individual PWM mode	O	O	O
PRTINEV	I	Protection input pin dedicated to MPWM0 and MPWM1 phase V in individual PWM mode	O	O	O
PRTINEW	I	Protection input pin dedicated to MPWM0 and MPWM1 phase W in individual PWM mode	O	O	O
OVINEV	I	Overvoltage input pin dedicated to MPWM0 and MPWM1 phase V in individual PWM mode	O	O	O
OVINEW	I	Overvoltage input pin dedicated to MPWM0 and MPWM1 phase W in individual PWM mode	O	O	O
CAPEU	I	Input pin in MPWM0 and MPWM1 phase U dedicated to capturing in individual PWM mode	O	O	O
CAPEV	I	Input pin in MPWM0 and MPWM1 phase V dedicated to capturing in individual PWM mode	O	O	O
CAPEW	I	Input pin in MPWM0 and MPWM1 phase W dedicated to capturing in individual PWM mode	O	O	O
SCAPEU	I	Input pin in MPWM0 and MPWM1 phase U dedicated to sub-capturing in individual PWM mode	O	O	O
SCAPEV	I	Input pin in MPWM0 and MPWM1 phase V dedicated to sub-capturing in individual PWM mode	O	O	O
SCAPEW	I	Input pin in MPWM0 and MPWM1 phase W dedicated to sub-capturing in individual PWM mode	O	O	O

MPWM0 and MPWM1 have pins assigned for the protection operation as described in Table 115. For more information about the protection operation, refer to 14.3.14 Protection and overvoltage.

Table 115. Time-base Register of MPWMn

Type		MPWM0 Protection	MPWM1 Protection
Motor and normal PWM modes		PRTIN0U OVIN0U	PRTIN1U OVIN1U
Individual PWM mode	U	PRTIN0U OVIN0U	PRTIN1U OVIN1U
	V	PRTINEV	PRTINEV
	W	PRTINEW	PRTINEW

NOTE:

1. In individual PWM mode, V and W of PRTINE / OVINE pins operate simultaneously with MPWM0 and MPWM1.

14.3.3 Time-base Unit

The MPWM module of the A34M420 has a 16-bit counter that is a main component of the MPWM module. This 16-bit counter supports up counter and up-down counter modes, and the input clock for the counter can be controlled by the SCU miscellaneous clock block.

Table 116 describes registers related to the MPWM operation time.

Table 116. Time-base Register of MPWMn

Name	PWM Mode	Description	Access Type
MPWMn_PRD	Motor / normal	PWM period setting register	Read-write
MPWMn_PRDU	Individual	U phase PWM period setting register	Read-write
MPWMn_PRDV	Individual	V phase PWM period setting register	Read-write
MPWMn_PRDW	Individual	W phase PWM period setting register	Read-write
MPWMn_CR1	Motor / normal	Period interrupt interval control register	Read-write
MPWMn_CR3	Individual	Period interrupt interval control register for each U, V, and W phases	Read-write
MPWMn_CNT	Motor / normal	Counter register	Read only
MPWMn_CNTU	Individual	U phase Counter register	Read only
MPWMn_CNTV	Individual	V phase Counter register	Read only
MPWMn_CNTW	Individual	W phase Counter register	Read only

Values of the MPWMn_PRD, MPWMn_PRDU, MPWMn_PRDV, and MPWMn_PRDW registers are applied to the MPWM logic at different times, depending on the value set in the UAO bit in the MPWMn_MR register. For more information about the application timing, refer to section 14.3.7.

14.3.4 Counter Modes

In Normal PWM mode, the up counter and up-down counter are supported, depending on the set value of the UPDOWN bit in the MPWMn_MR register. However, in Motor and Individual PWM modes, only up-down counter is supported.

Counter modes available in each PWM mode are listed in Table 117. For more description of each PWM mode, refer to section 14.3.7.

Table 117. Counter Modes of MPWM

PWM Mode	Counter Mode
Motor PWM	Up-down counter
Normal PMW	Up counter
	Up-down counter
Individual PWM	Up-down counter

If the counter operates in up counter mode, the counter is incremented by one starting from 0x0001 to PERIOD[15:0] value. When the counter increases and reaches PERIOD[15:0] value, then the counter is reset to 0x0001 at the next clock and is incremented by one starting from 0x0001 again. This operation repeats until the counter stops.

Period T in up counter mode can be calculated by the formular below:

$$T_{Period} = PERIOD[15:0] / MPWM\ clock$$

If the counter operates in up-down counter mode, the counter is incremented by one starting from one to PERIOD[15:0] value, the same as in up counter mode. However, when the counter increases and reaches PERIOD[15:0] value, the counter operates as a down counter from the next clock, decreasing one. On reaching zero, the counter operates as an up counter from the next clock, increasing by one.

Period T in up-down counter mode can be calculated by the formular below:

$$T_{Period} = 2 \times PERIOD[15:0] / MPWM\ clock$$

14.3.5 Symmetric and Asymmetric Mode

In Up-down counter mode, the PWM can output symmetrically around the period matching event or asymmetrically, by configuring the MCHMOD[1:0] bits in the MPWMn_MR register.

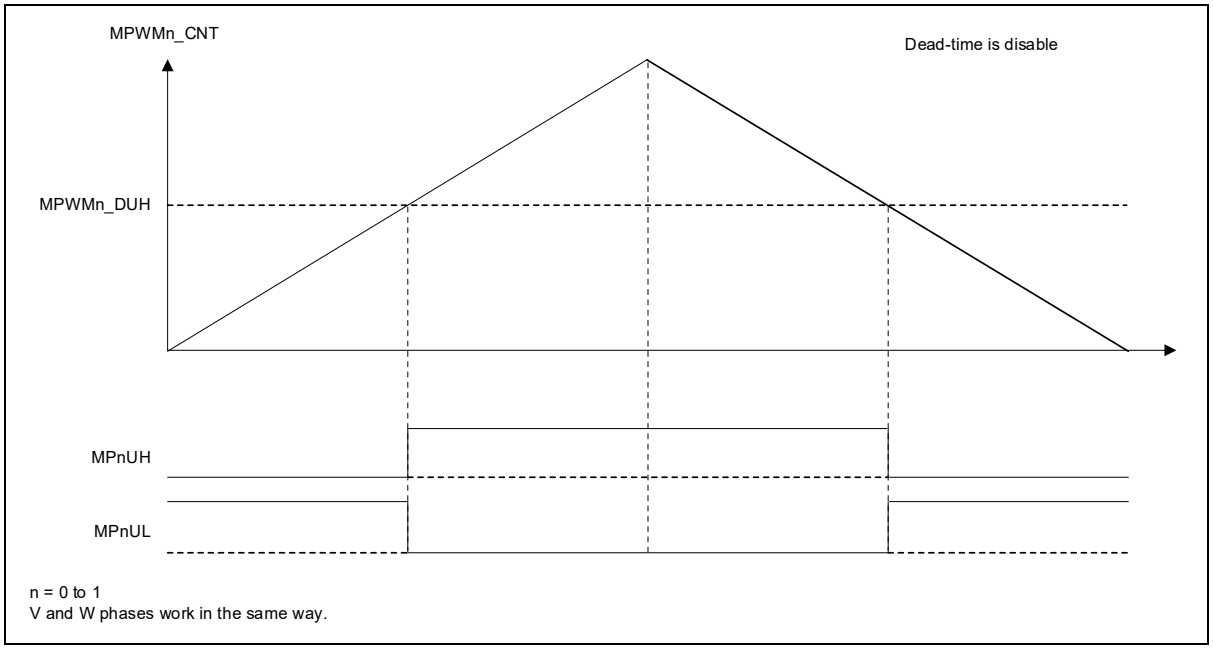
However, in Normal PWM mode, the PWM is fixed to output symmetrically, regardless of the value set in the MCHMOD[1:0] bits in the MPWMn_MR register.

In symmetric mode, there are two types of modes as described below:

- 2-channel symmetric mode: The high and low sides of the output can be controlled independently.
- 1-channel symmetric mode: The high and low sides of the output can be controlled in complementary symmetry.

Figure 134 shows a timing diagram of the PWM outputs in 1-channel symmetric mode.

Figure 134. 1-channel Symmetric Mode Timing Diagram

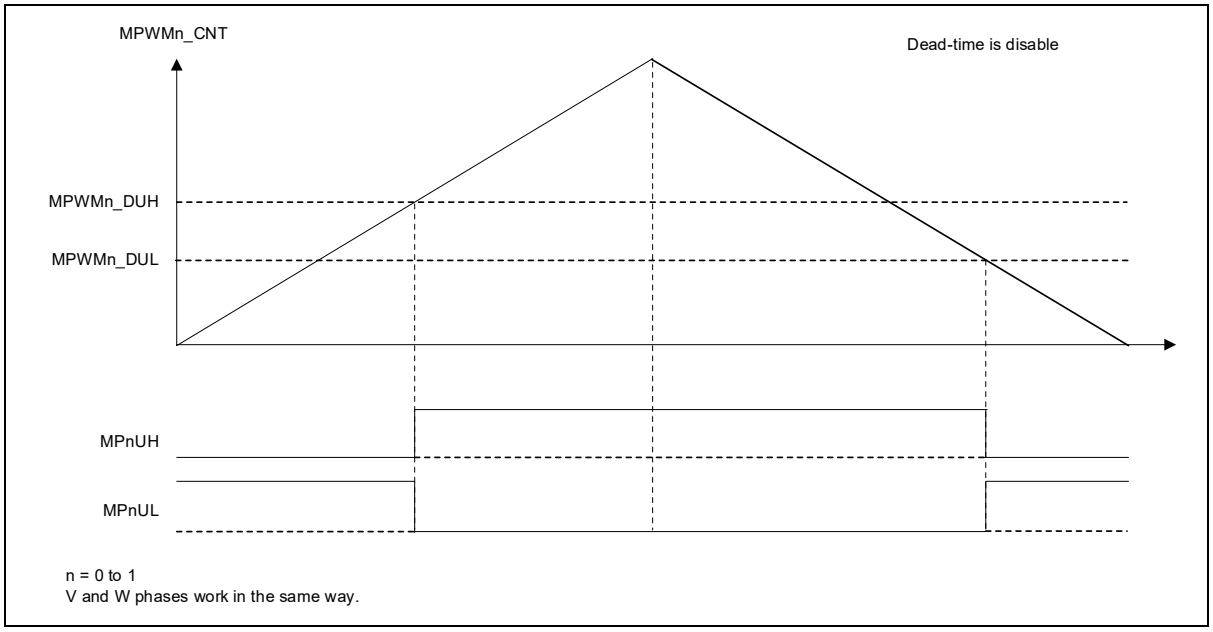


In asymmetric mode, there is only one type of mode as described below:

- 1-channel asymmetric mode: The high and low sides of the output can be controlled in complementary symmetry.

Figure 135 shows a timing diagram of the PWM outputs in 1-channel asymmetric mode.

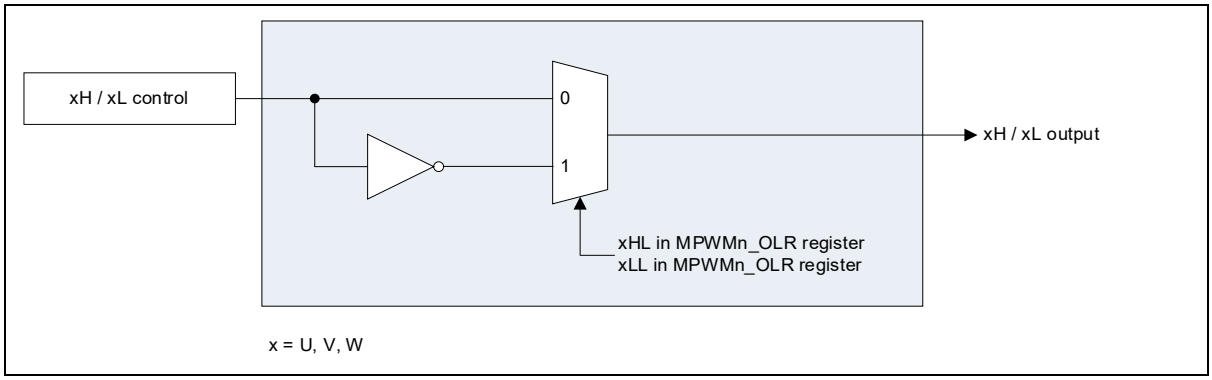
Figure 135. 1-channel Asymmetric Mode Timing Diagram



14.3.6 Output Control

The MPWM module includes an output control block that controls the outputs as shown in Figure 136.

Figure 136. Output Control Block



With the configuration of the output control block, users can set the output levels of both the operating counter and the non-operating counter by using the MPWMn_OLR register. The users simply set the MPWMn_OLR register considering the external circuit's switch polarity and desired PWM type.

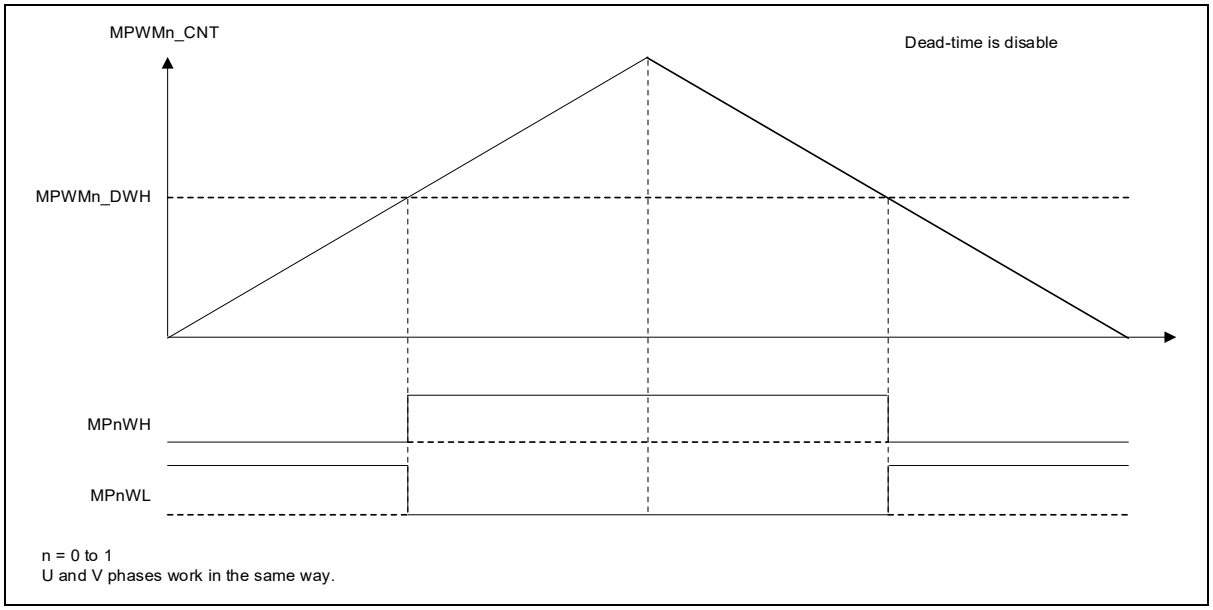
Each bit of the MPWMn_OLR register, such as WHL, VHL, UHL, WLL, VLL, and ULL, determines the output level of the PWM when the counter starts, as described in Table 118.

Table 118. Output Level (MPWM_OLR = 0x00)

PWM Output	Level	Normal PWM Mode (MOTORB[1:0] = 1)		Motor PWM Mode (MOTORB[1:0] = 0)
		Up Count Mode (UPDOWN = 0)	Up-down Count Mode (UPDOWN = 1)	
MPnWH	Basic output level	LOW	HIGH	LOW
	Inverted output level	HIGH	LOW	HIGH
MPnWL	Basic output level	LOW	LOW	HIGH
	Inverted output level	HIGH	HIGH	LOW
MPnVH	Basic output level	LOW	HIGH	LOW
	Inverted output level	HIGH	LOW	HIGH
MPnVL	Basic output level	LOW	LOW	HIGH
	Inverted output level	HIGH	HIGH	LOW
MPnUH	Basic output level	LOW	HIGH	LOW
	Inverted output level	HIGH	LOW	HIGH
MPnUL	Basic output level	LOW	LOW	HIGH
	Inverted output level	HIGH	HIGH	LOW

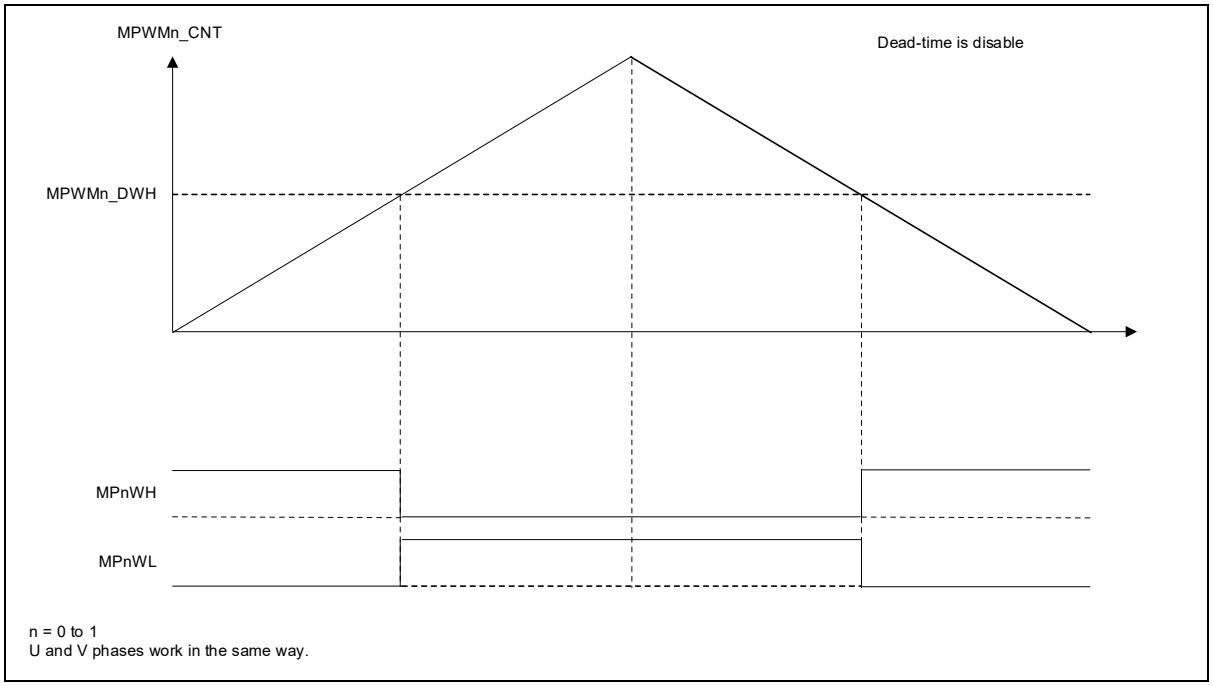
For example, if the WHL and WLL bits of the MPWMn_OLR register are set to ‘0’ in Motor PWM mode (1-channel symmetric mode), a period output of the MPnWH and MPnWL appears as shown in Figure 137.

Figure 137. MPnWH / MPnWL Output at Motor PWM Mode (WHL = ‘0’, WLL = ‘0’)



If the WHL and WLL bits of the MPWMn_OLR register are set to '1' in Motor PWM mode (1-channel symmetric mode), the output of a period of the MPnWH and MPnWL appears as shown in Figure 138.

Figure 138. MPnWH / MPnWL Output at Motor PWM Mode (WHL = '1', WLL = '1')



Each bit of the MPWMn_OLR register, such as DOLWH, DOLVH, DOLUH, DOLWL, DOLVL, and DOLUL, determines the output level of the PWM when the counter is disabled. Please remember that the output level can be a polar reversal according to the values of the bits WHL, VHL, UHL, WLL, VLL, and ULL.

14.3.7 PWM Modes

PWM modes can be controlled by configuring the MOTORB[1:0] bits in the MPWMn_MR register. Three PWM modes such as Motor, Normal, and Individual PWM modes are available for the PWM operations as described in Table 119.

Table 119. PWM Modes of MPWM

PWM Mode	Counter Mode	Symmetric/Asymmetric Mode
Motor PWM	Only Up-down counter	2-channel symmetric mode
		1-channel asymmetric mode
		1-channel symmetric mode
Normal PWM	Up counter	-
	Up-down counter	(Only 1-channel symmetric mode)
Individual PWM	Only Up-down counter	2-channel symmetric mode
		1-channel asymmetric mode
		1-channel symmetric mode

14.3.7.1 Motor PWM / Up-down Counter / 2-channel Symmetric Mode

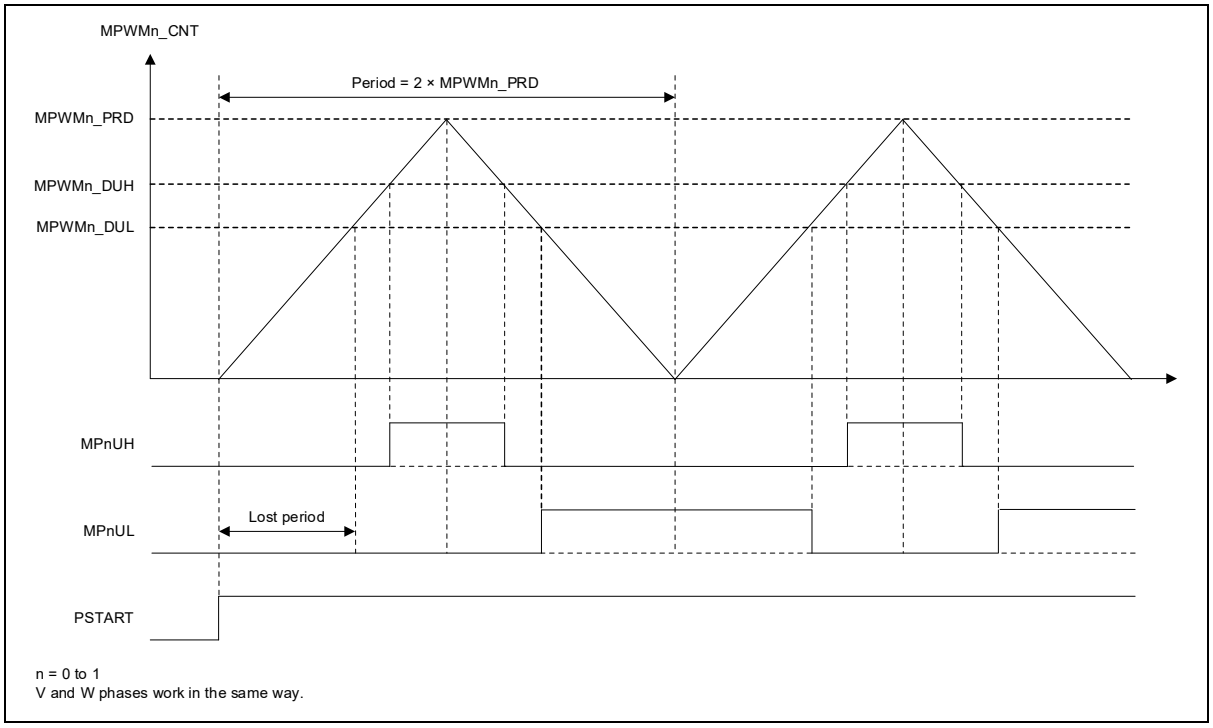
In this mode, the PWM module can generate up to six outputs of the same period.

Once the counter starts operating, it repeats incrementing and decrementing in accordance with the period. The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value.

Therefore, the high and low sides operate independently and symmetrically around the period matching event. Note that the low side pulse is not output during the first period since the PWM has started.

Figure 139 shows PWM waveform in Motor PWM/up-down counter/2-channel symmetric mode.

Figure 139. PWM Waveform in Motor PWM/Up-down Counter/2-channel Symmetric Mode



14.3.7.2 Motor PWM / Up-down Counter / 1-channel Asymmetric Mode

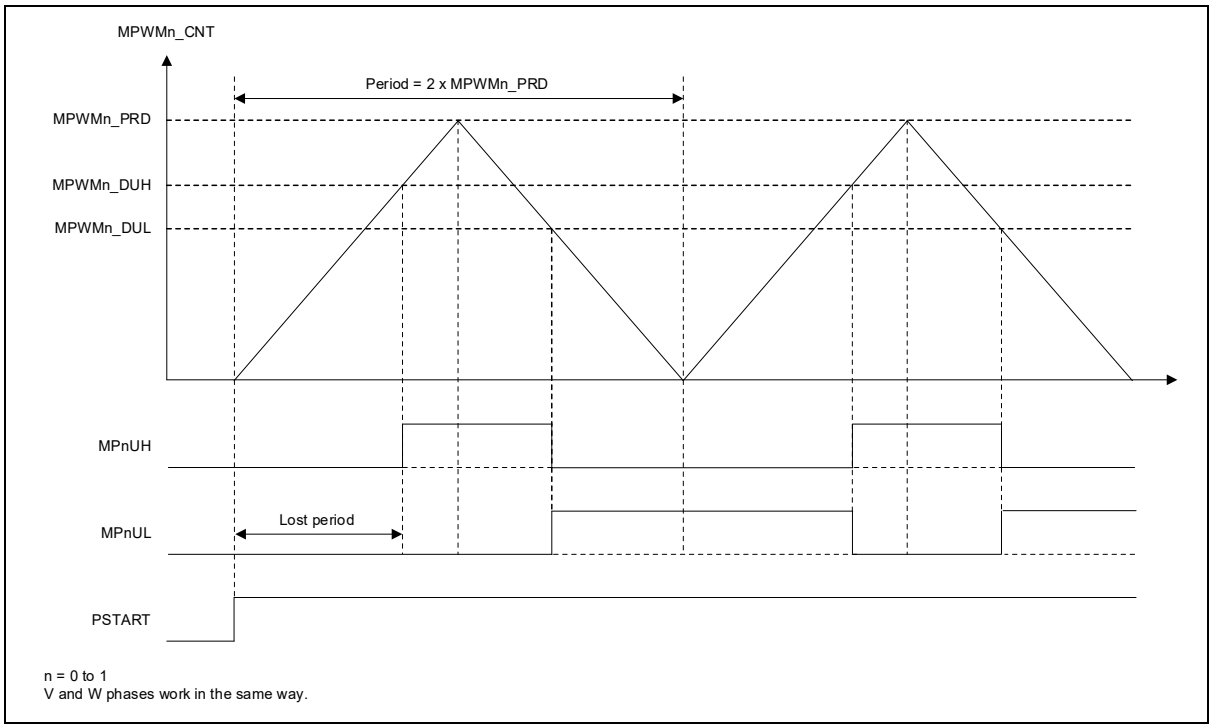
In this mode, the PWM module can generate up to three pairs of PWM outputs that are complementary symmetries of the same period. A pair consists of two channels.

If the counter operates as an up counter, the PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while if the counter operates as a down counter, the PWM outputs inversion of the output when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value.

The low side output appears in complementary symmetry of the high side output. Using the characteristics of operating asymmetrically, the PWM phases can be controlled.

Figure 140 shows PWM waveform in Motor PWM/up-down counter/1-channel asymmetric mode.

Figure 140. PWM Waveform in Motor PWM/Up-down Counter/1-channel Asymmetric Mode



14.3.7.3 Motor PWM / Up-down Counter / 1-channel Symmetric Mode

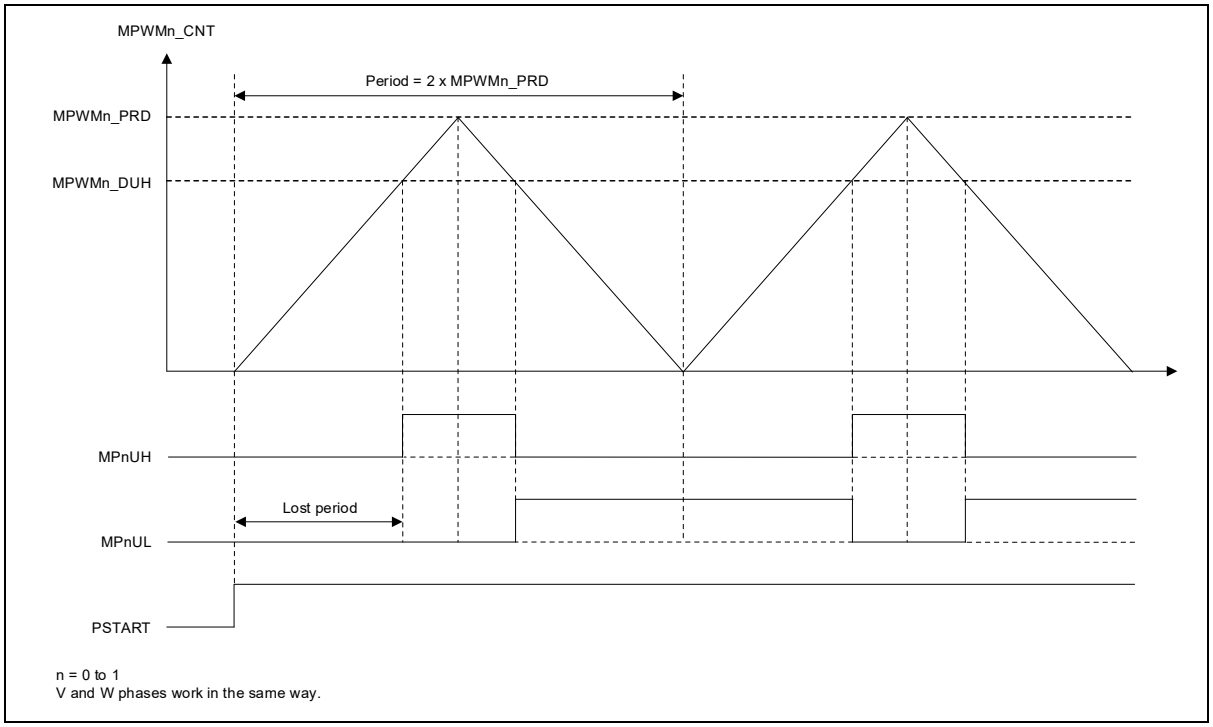
In this mode, the PWM module can generate up to three pairs of PWM outputs that are complementary symmetries of the same period. A pair consists of two channels.

The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWM_DxH register matches the counter value; while the low side output appears in complementary symmetry of the high side output.

In this mode, the DUTY_xL[15:0] value in the MPWM_DxL register does not affect the PWM outputs.

Figure 141 shows PWM waveform in Motor PWM/up-down counter/1-channel symmetric mode.

Figure 141. PWM Waveform in Motor PWM/Up-down Counter/1-channel Symmetric Mode



14.3.7.4 Normal PWM / Up Counter Mode

In this mode, the PWM module can generate up to six outputs of the same period.

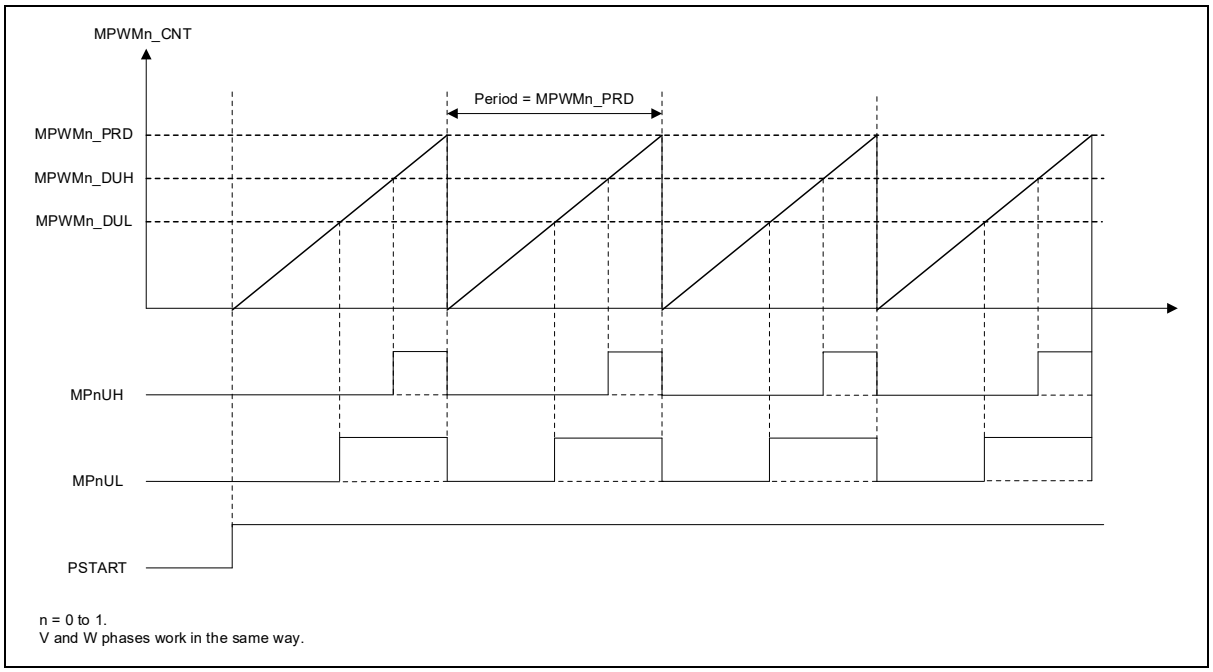
Once the counter starts operating, it repeats incrementing from '0' to 'PERIOD[15:0]' in accordance with the period. The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value.

When the counter reaches PERIOD[15:0], both of the high and low sides are reversed at the next clock, and this will be repeated. Like this, the high and low sides of the outputs operate independently.

Note that the high and low side pulses are not output during the first period since the PWM has started.

Figure 142 shows PWM waveform in Normal PWM/up counter mode.

Figure 142. PWM Waveforms in Normal PWM/Up Counter Mode



14.3.7.5 Normal PWM / Up-down Counter Mode

In this mode, the PWM module can generate up to six outputs of the same period.

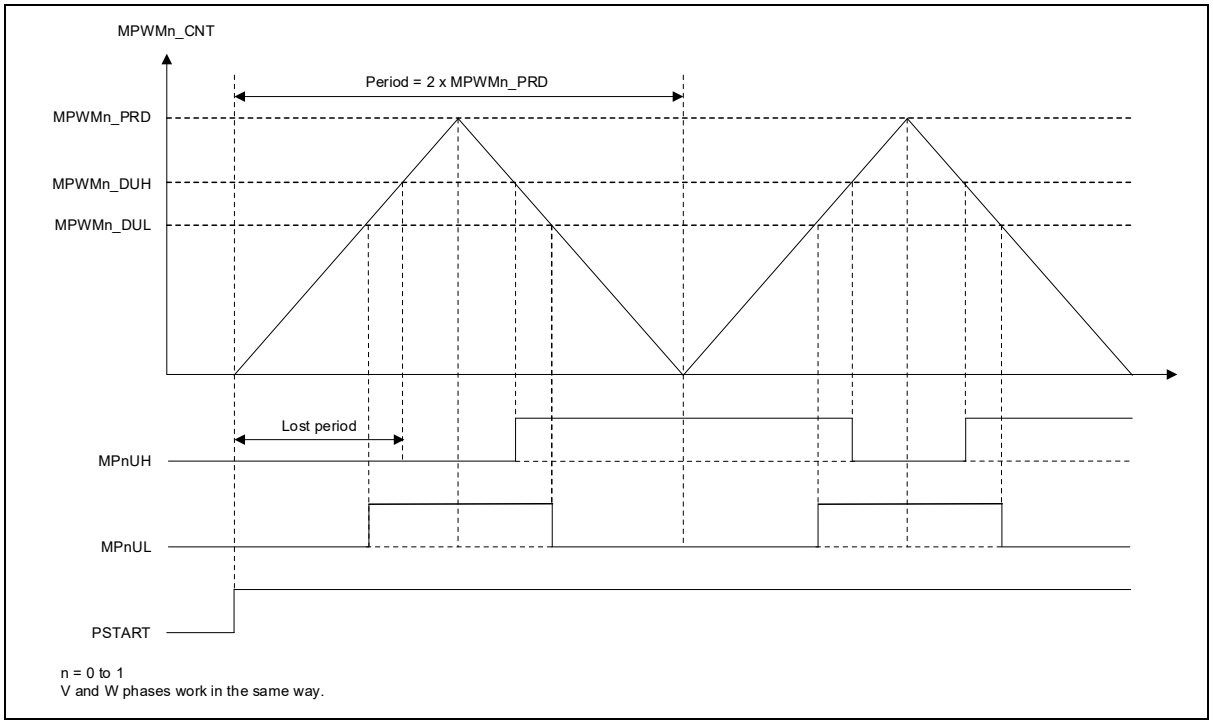
Once the counter starts operating, it repeats incrementing and decrementing in accordance with the period. The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value. Therefore, the high and low sides of the outputs operate independently.

In this mode, the PWM module operates in a similar way as in the Motor PWM/Up-down counter/2-channel symmetric mode, but the output polarity is reversed.

Note that the high side pulse is not output during the first period since the PWM has started.

Figure 143 shows PWM waveform in Normal PWM/up-down counter mode.

Figure 143. PWM Waveform in Normal PWM/Up-down Counter Mode



14.3.7.6 Individual PWM / Up-down Counter / 2-channel Symmetric Mode

In this mode, the PWM module can generate up to three pairs of PWM outputs of which period can be set individually. A pair consists of two channels, and the duty values of the two channels can be set individually in this mode.

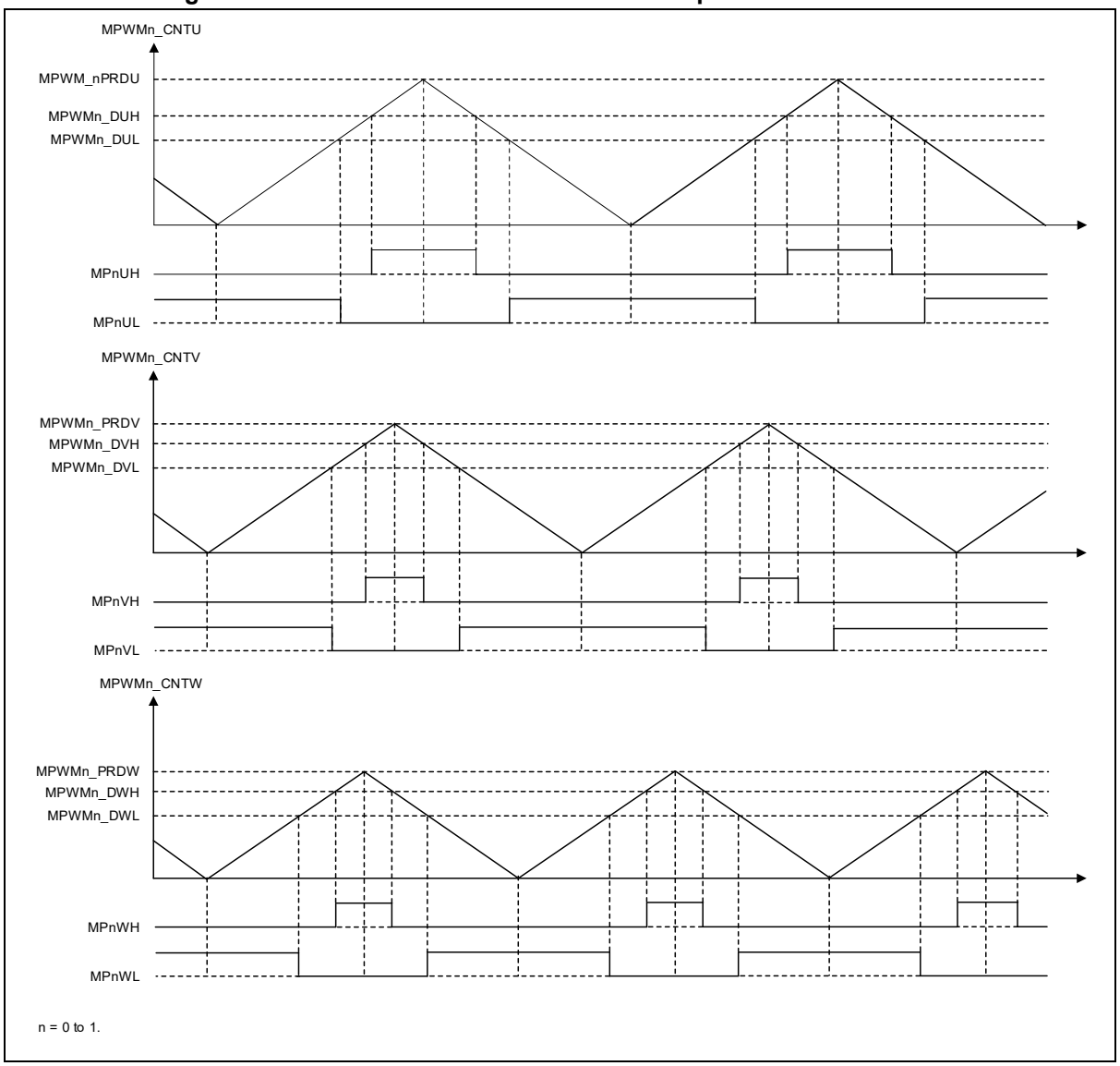
A total of three counters can start independently, and each counter repeats incrementing and decrementing in accordance with individual period.

The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value. Therefore, the high and low sides of the outputs operate independently and symmetrically around the period matching event.

Note that the low side pulse is not output during the first period since the PWM has started.

Figure 144 shows PWM waveform in Individual PWM/up-down counter/2-channel symmetric mode.

Figure 144. PWM Waveform in Normal PWM/Up-down Counter Mode



14.3.7.7 Individual PWM / Up-down Counter / 1-channel Asymmetric Mode

In this mode, the PWM module can generate up to three pairs of PWM outputs of which period can be set individually. A pair consists of two channels, and each channel operates in complementary symmetry.

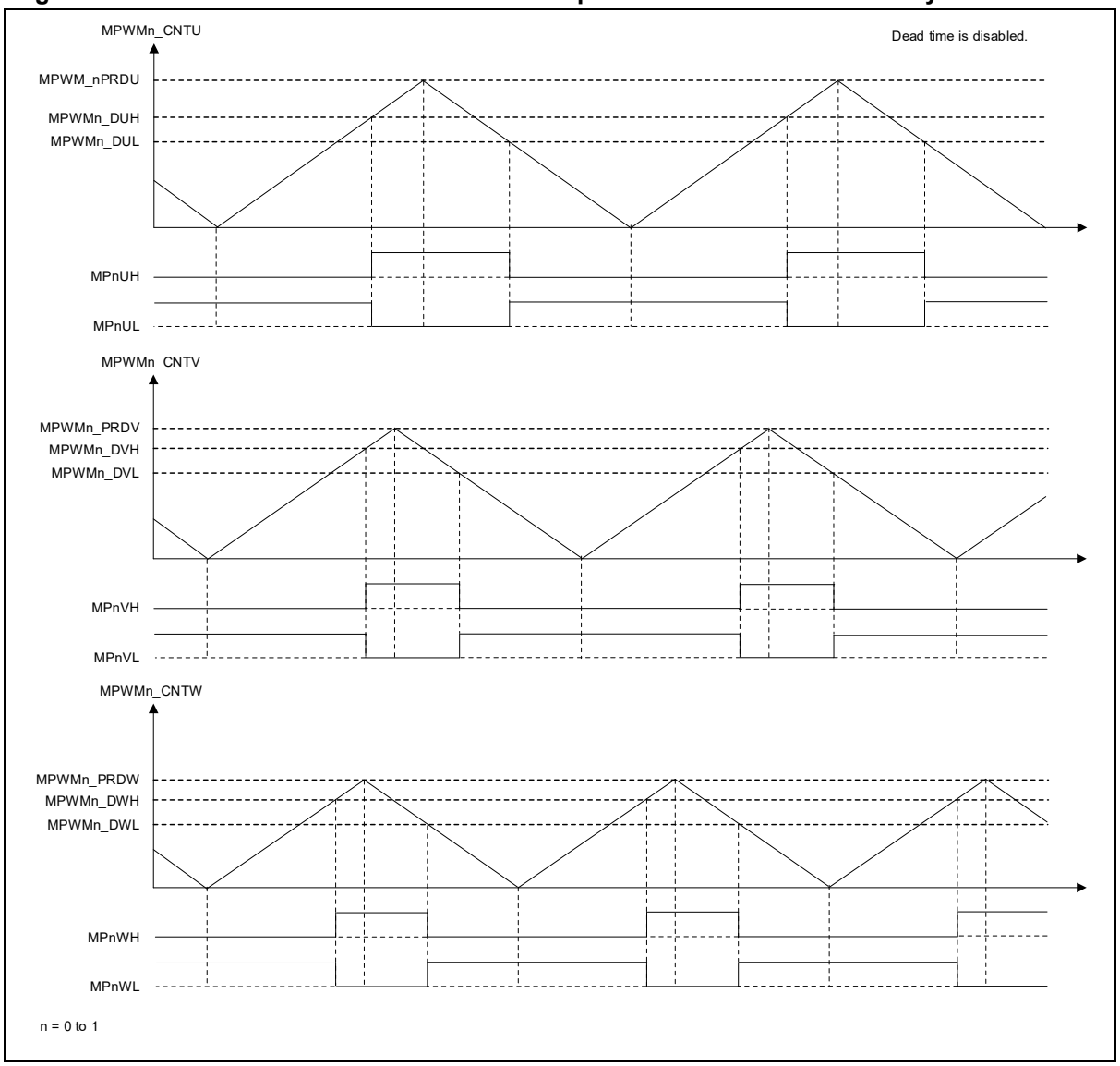
Each of the three counters can start independently, and each counter repeats incrementing and decrementing in accordance with individual period.

If the counter operates as an up counter, the PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while if the counter operates as a down counter, the PWM outputs inversion of the output when the DUTY_xL[15:0] value in the MPWMn_DxL register matches the counter value.

The low side output appears in complementary symmetry of the high side output. Using the characteristics of operating asymmetrically, the PWM phases can be controlled.

Figure 145 shows PWM waveform in individual PWM/up-down counter/1-channel asymmetric mode.

Figure 145. PWM Waveform in Individual PWM/Up-down Counter/1-channel Asymmetric Mode



14.3.7.8 Individual PWM / Up-down Counter / 1-channel Symmetric Mode

In this mode, the PWM module can generate up to three pairs of PWM outputs of which period can be set individually. A pair consists of two channels, and each channel operates in complementary symmetry.

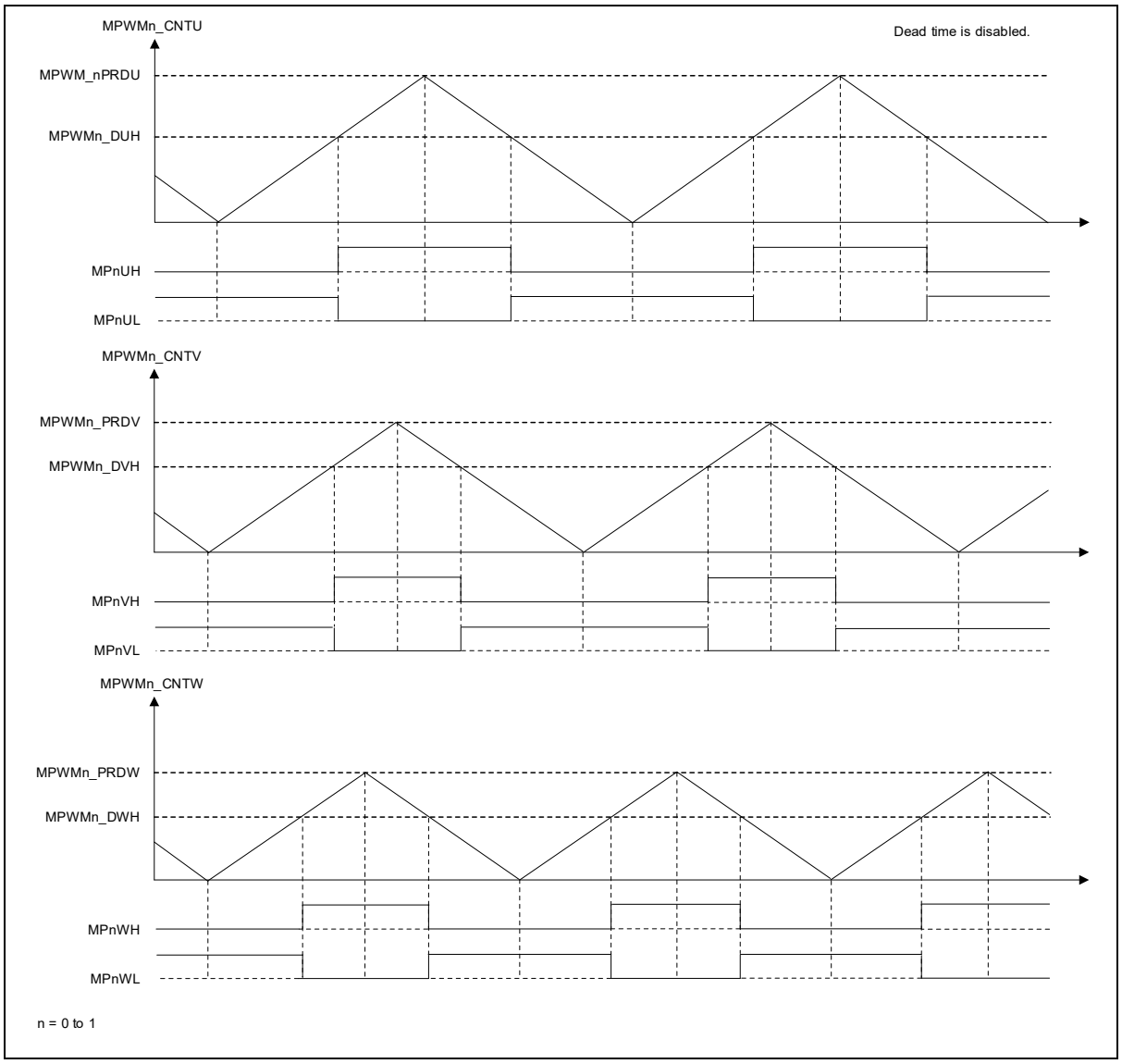
Each of the three counters can start independently, and each counter repeats incrementing and decrementing in accordance with individual period.

The PWM outputs with the high side reversed when the DUTY_xH[15:0] value in the MPWMn_DxH register matches the counter value; while the low side output appears in complementary symmetry of the high side output.

In this mode, the DUTY_xL[15:0] value in the MPWMn_DxL register do not affect the PWM outputs.

Figure 146 shows PWM waveform in individual PWM/up-down counter/1-channel symmetric mode.

Figure 146. PWM Waveform in Individual PWM/Up-down Counter/1-channel Symmetric Mode



14.3.8 Complimentary Outputs and Dead-time Insertion

14.3.8.1 Complimentary Outputs

Circuits such as half-bridge can control devices by activating the high side output and low side output alternately, which is called complementary symmetric output.

When dead-time is off (dead-time is described in the following sections), sum of the duty cycle of the high side output and the duty cycle of the low side output must be 100% (period time) in the complementary symmetric output method.

$$\text{Duty cycle of high side} + \text{Duty cycle of low side} = 100\%$$

14.3.8.2 Dead-Time Insertion

Usually, circuits operating in the complementary symmetric output method require latency which is called “dead-time”. Dead-time prevents arm short circuits that can be caused by the switch on / off delay characteristics associated with the output.

Table 120 lists the mode where dead-time can be supported.

Table 120. Dead-time Support of MPWMn

PWM Mode	Counter Mode	Symmetric / Asymmetric mode	Dead-time Support
Motor PWM	Only up-down counter	2-channel symmetric mode	N/A
		1-channel asymmetric mode	O
		1-channel symmetric mode	O
Normal PWM	Up counter	-	N/A
	Up-down counter	(Only 1-channel symmetric mode)	N/A
Individual PWM	Only up-down counter	2-channel symmetric mode	N/A
		1-channel asymmetric mode	O
		1-channel symmetric mode	O

In Motor PWM mode, dead-time can be enabled by setting the xDTEN bit in the MPWMn_DTRx register to ‘1’.

With dead-time enabled, when the high side output or low side output becomes active, dead-time is inserted. The dead-time is determined by configuring the DTCLK[1:0] and DT[7:0] bits of the MPWMn_DTRx register, and can be obtained by calculating the formular shown below:

$$\text{Dead time} = \frac{\text{DT}[7:0]}{\text{MPWM clock} \div 2^{\text{DTCLK}[1:0]+1}}$$

In Individual PWM mode, dead-time can be enabled by setting the xDTEN bit in the MPWMn_DTRx register to '1' for the phases U, V, and W respectively. The operation method and operation time are the same as in Motor PWM mode.

Figure 147 shows a timing diagram of the dead-time operation in 1-channel symmetric mode set to high active.

Figure 147. Dead-time Operation Timing Diagram (1-channel Symmetric Mode, DTMDSEL=0)

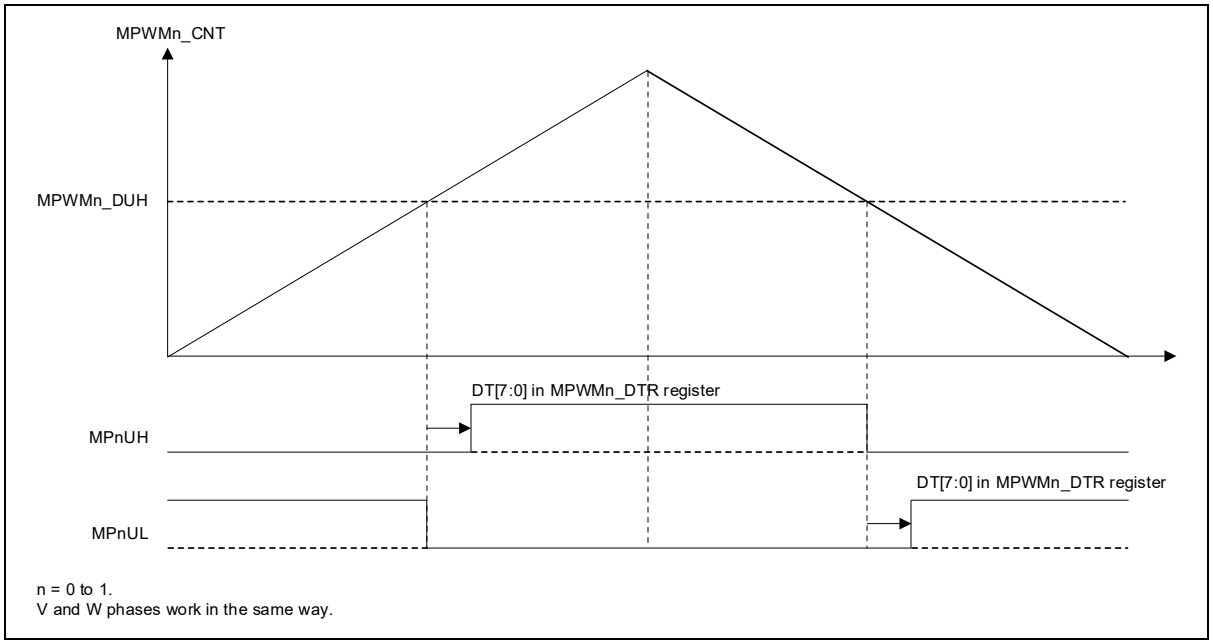
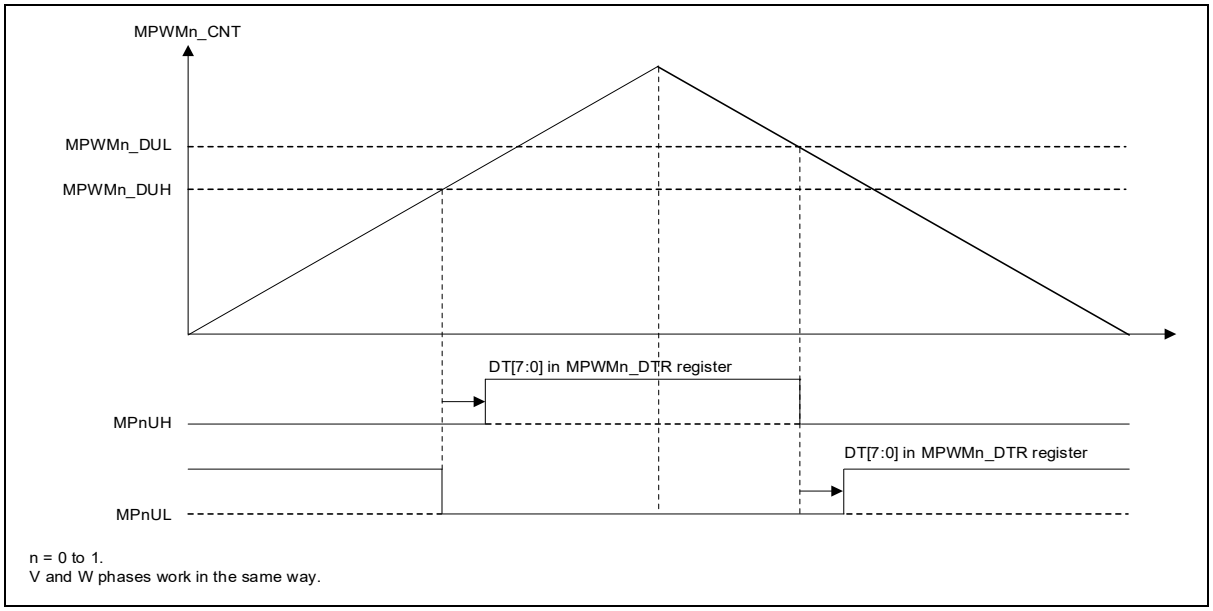


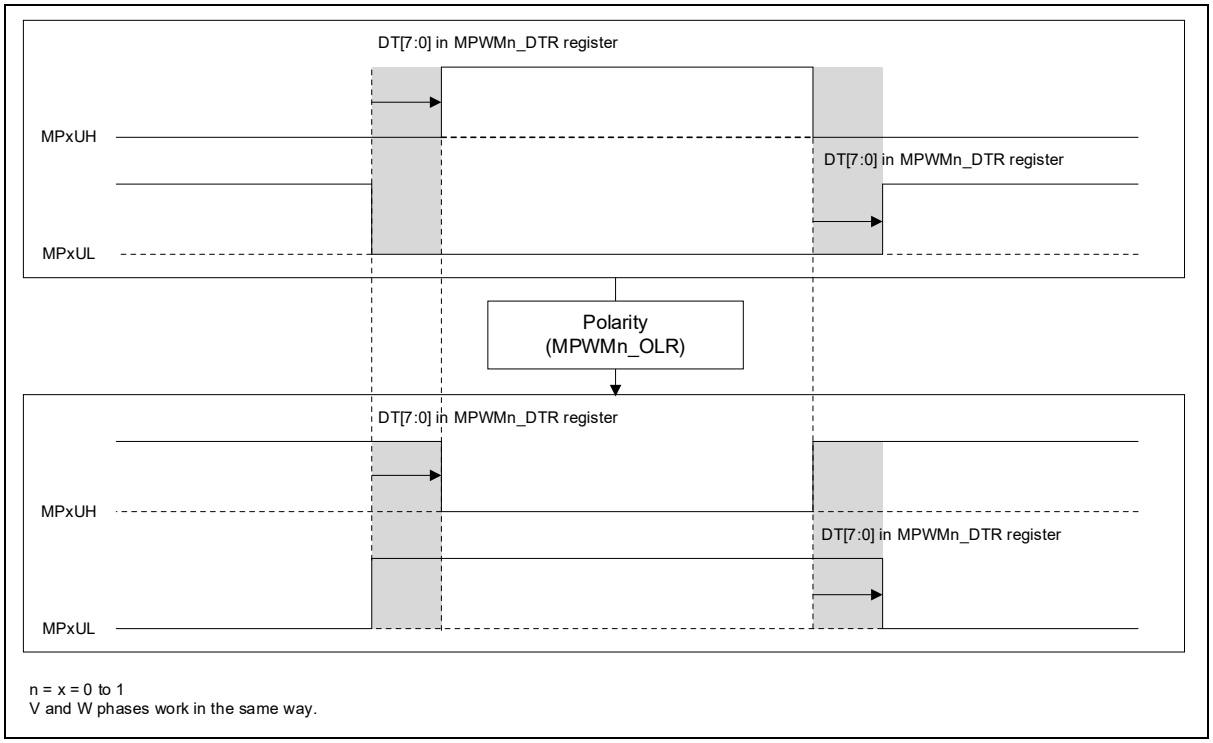
Figure 148 shows a timing diagram of the dead-time operation in 1-channel asymmetric mode set to high active.

Figure 148. Dead-time Operation Timing Diagram (1-channel Asymmetric Mode, DTMDSEL=0)



If the PWM output polarity function is used in Motor PWM mode, the high side and low side may become high simultaneously in the dead-time section as shown in Figure 149, causing problems with use of high active devices.

Figure 149. Dead-time of Polarity Settings



To solve this problem, users can move the dead-time position by following the description in Table 114, using the DTMDSEL bit in the MPWMn_DTR register.

Table 121. DTMDSEL Bit Operation

DTMDSEL	Description
0	Dead-time is inserted at the leading edge of PWMxH and the trailing edge of PWMxL.
1	Dead-time is inserted at the trailing edge of PWMxH and the leading edge of PWMxL.

NOTE: x = U, V, and W.

Figure 150 and Figure 151 show timing diagrams of the dead-time operations in which the DTMDSEL bit is set to ‘1’ in 1-channel symmetric mode and 1-channel asymmetric mode, respectively.

Figure 150. Dead-time Operation Timing Diagram (1-channel Symmetric Mode, DTMDSEL=1)

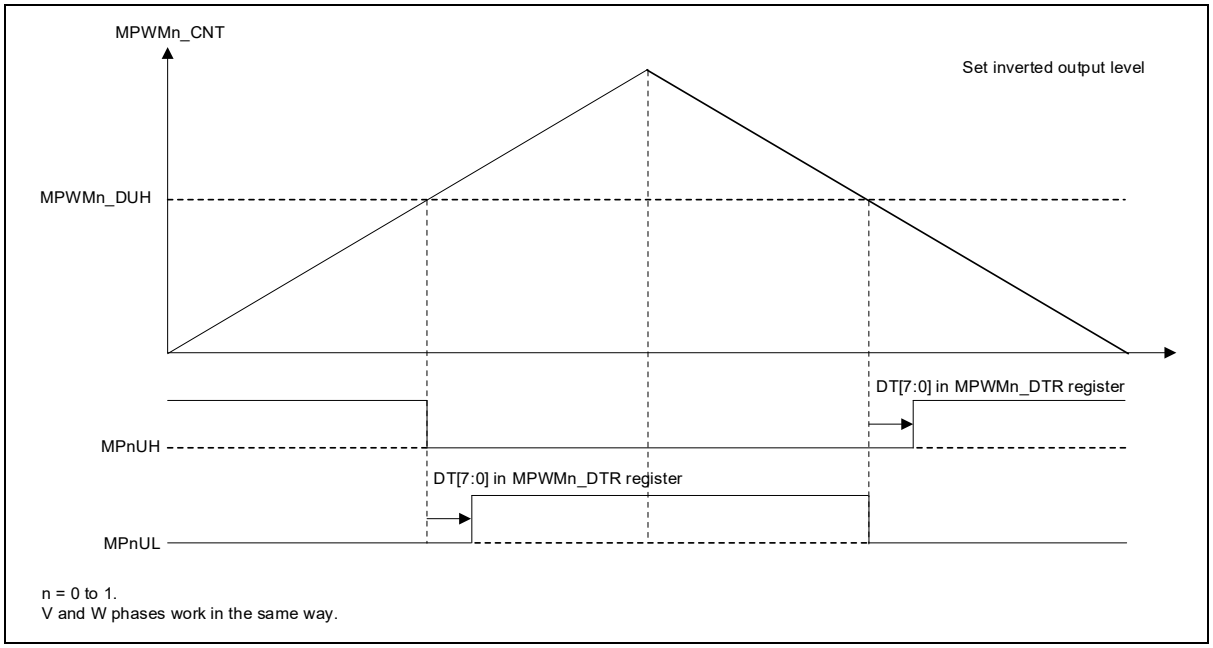
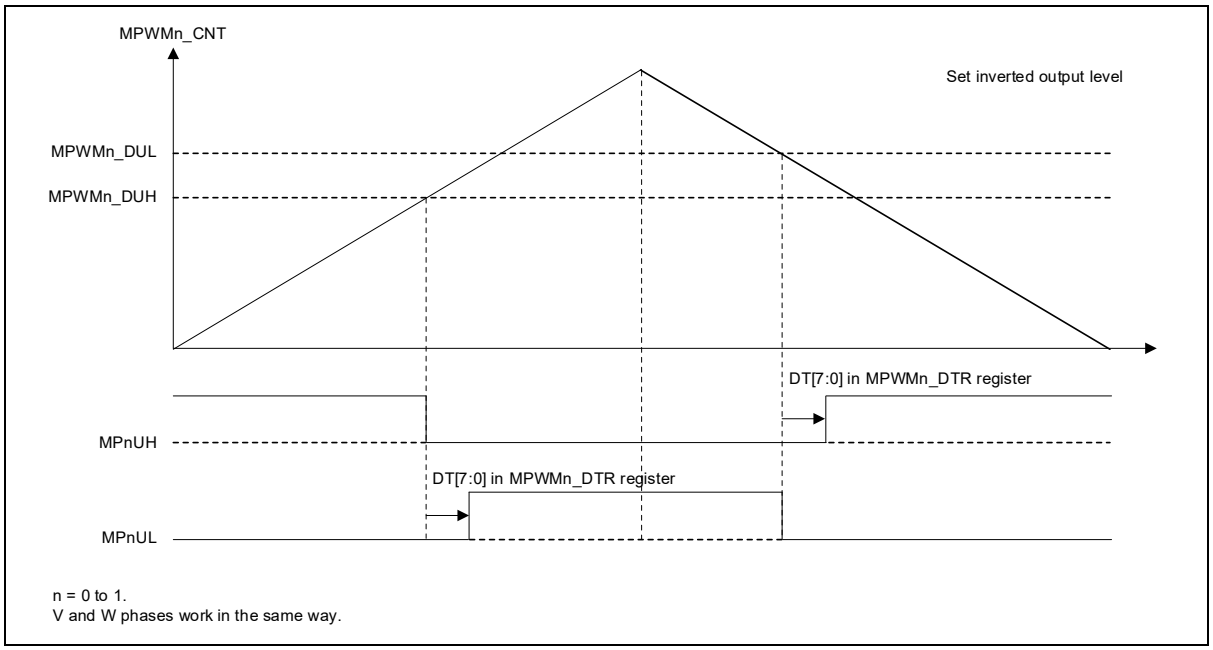


Figure 151. Dead-time Operation Timing Diagram (1-channel Asymmetric Mode, DTMDSEL=1)

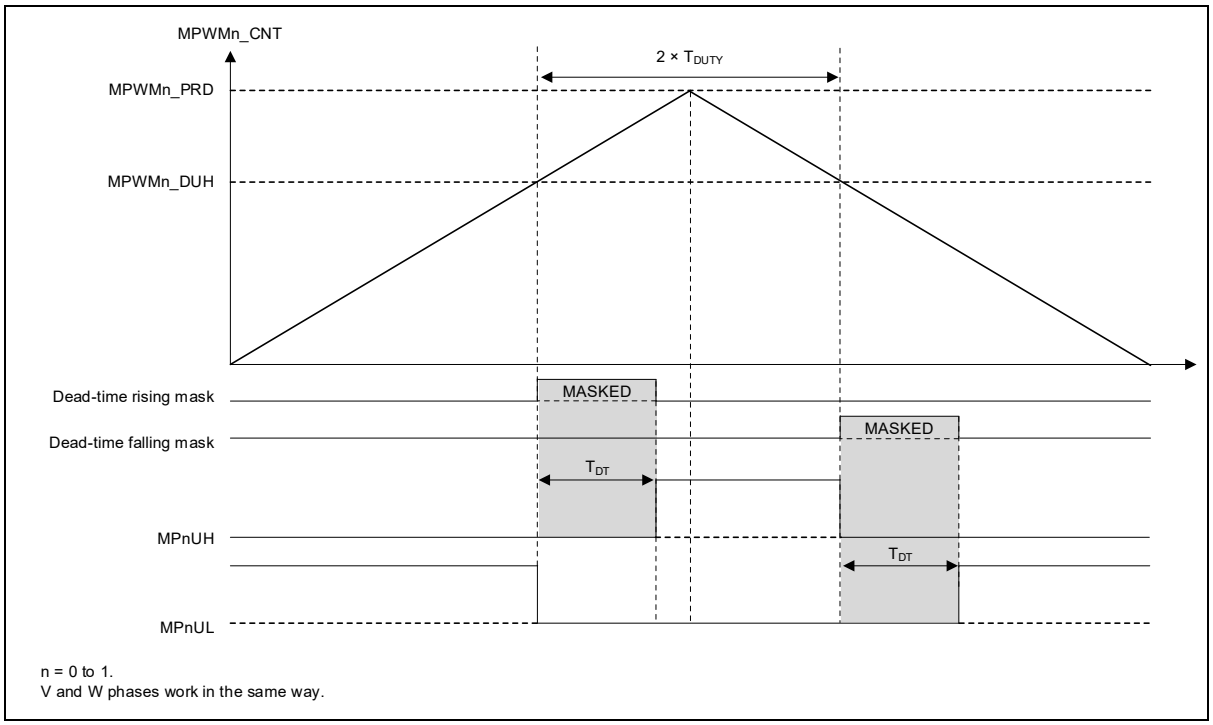


14.3.8.3 Special Case about Dead-time

Figure 152 shows a timing diagram of the dead-time operations in the typical case where the T_{DUTY} is larger than the T_{DT}.

Dead-time mask is enabled at the Duty Matches and disabled when the dead-time counter reaches the dead-time set by the DTCLK[1:0] and DT[7:0] bits of the MPWM_DTR register.

Figure 152. Typical Dead-time Operation (T_{DUTY} > T_{DT})



Figures below (Figure 153 to Figure 158) show timing diagrams of dead-time operations in special cases.

Figure 153. High Side Minimum Pulse Timing (T_{DUTY} < T_{DT} < 2 × T_{DUTY})

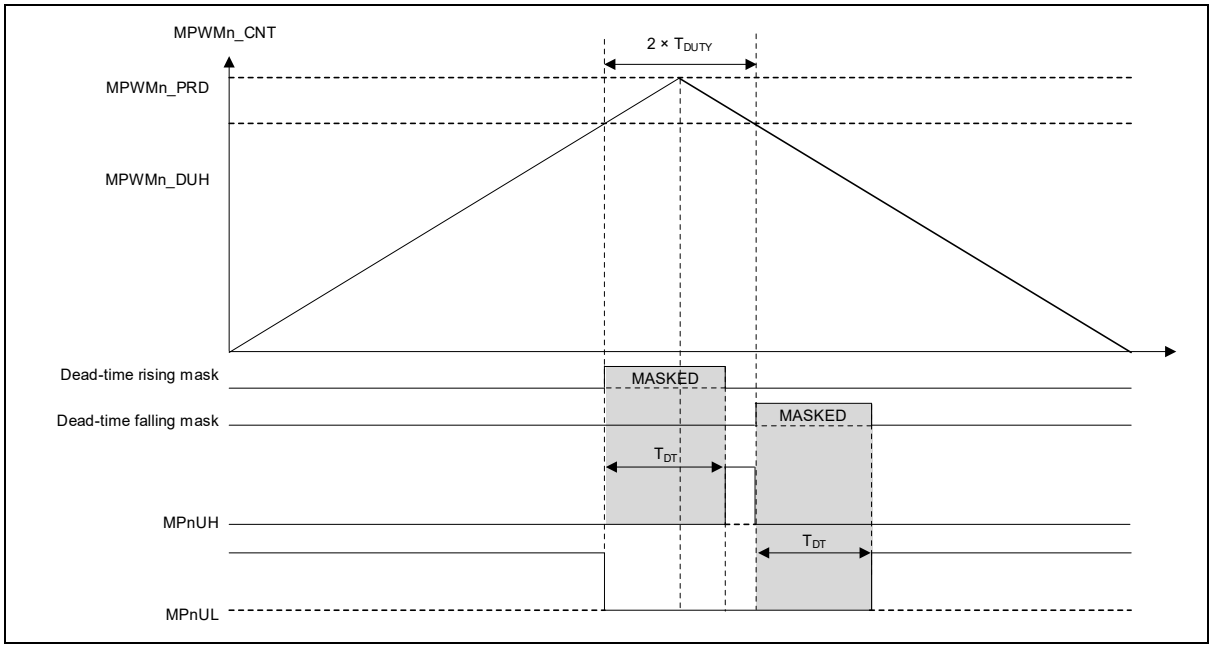


Figure 154. High Side Zero Pulse Timing (T_{DT} > 2 × T_{DUTY})

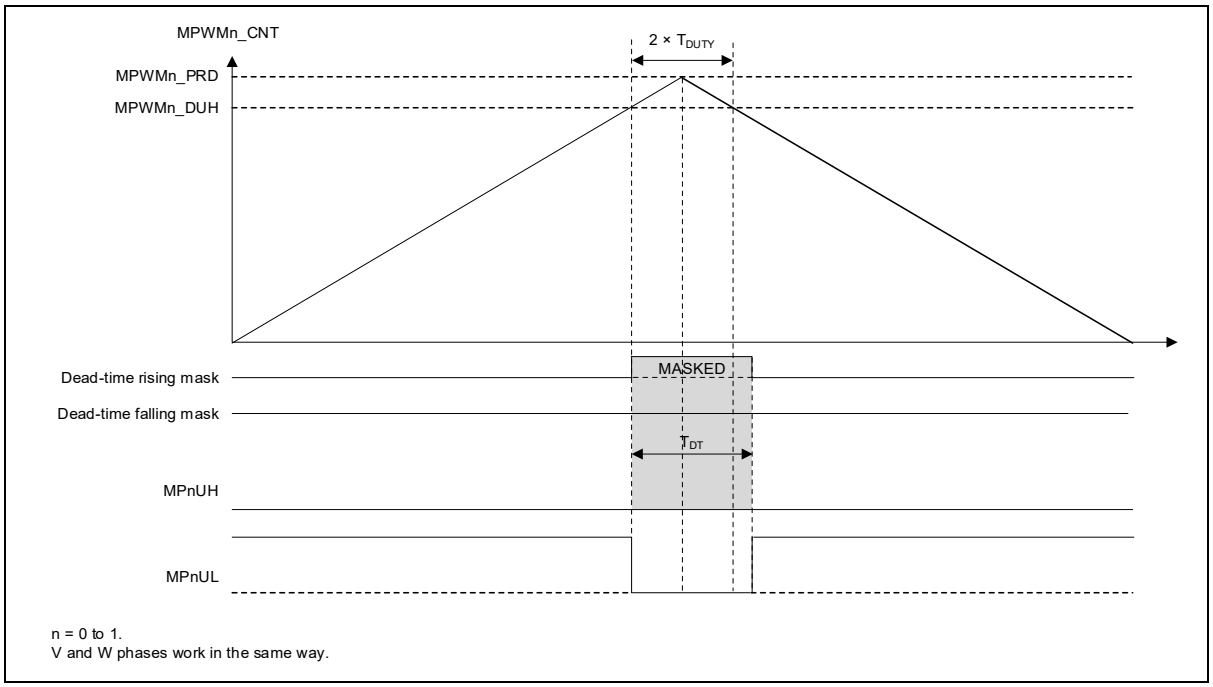


Figure 155. Low Side Minimum Pulse Timing (T_{DT} < period - T_{DUTY})

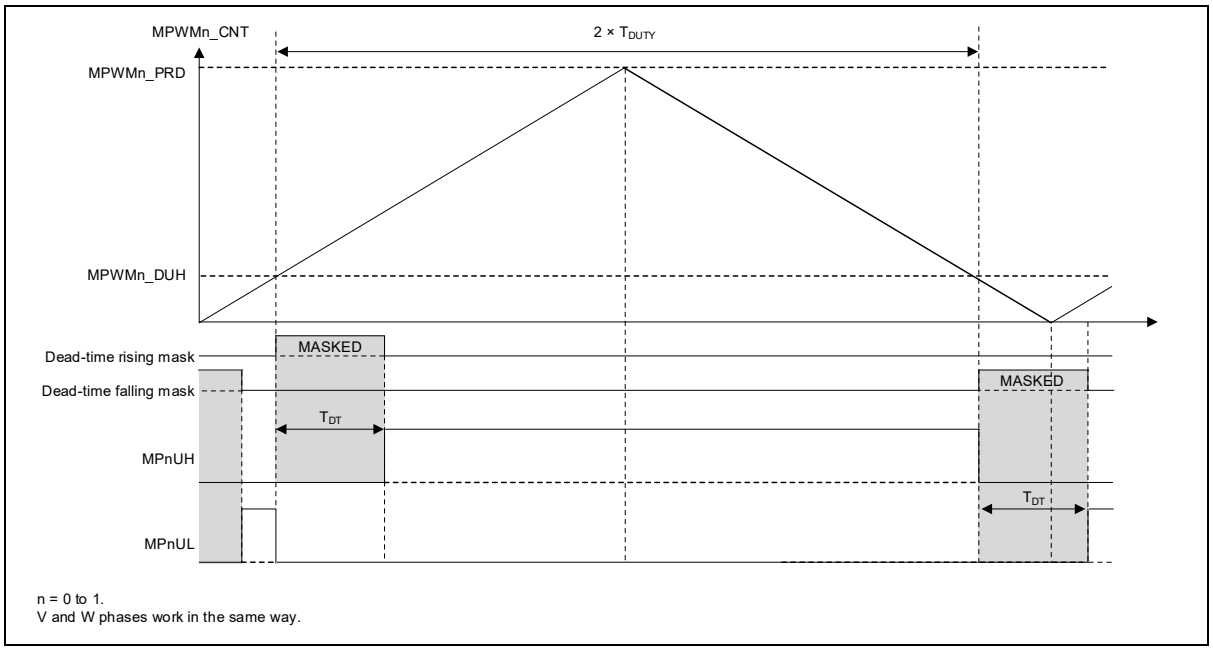


Figure 156. Low Side Zero Pulse Timing (T_{DT} > period - T_{DUTY})

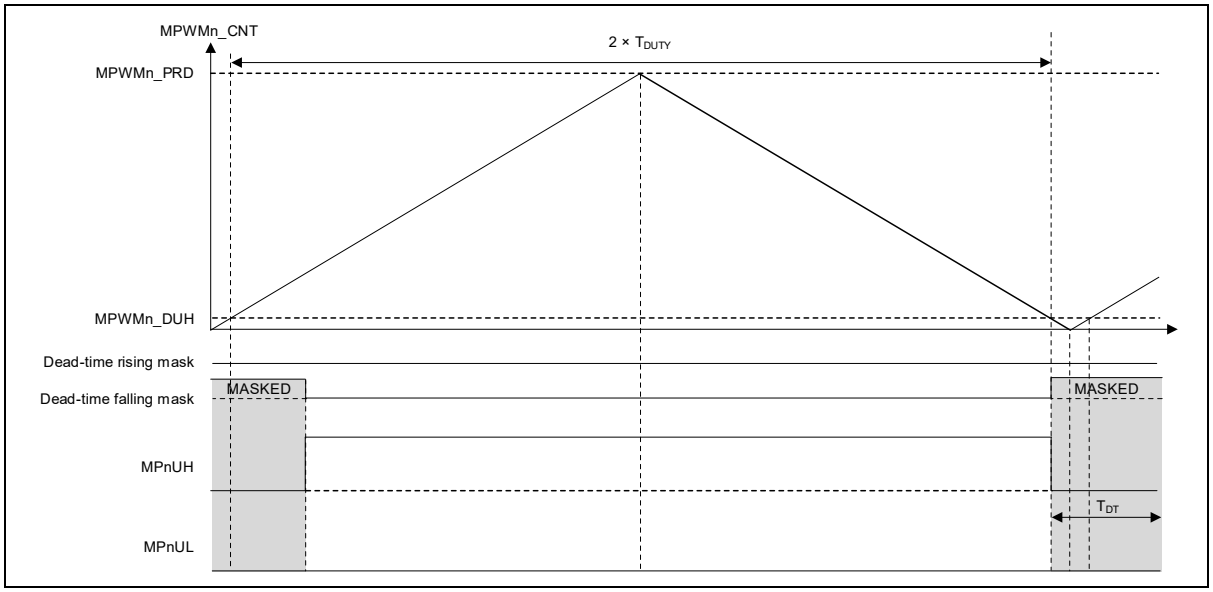


Figure 157. High Side Always-On (T_{DUTY} = period, when dead-time is disabled.)

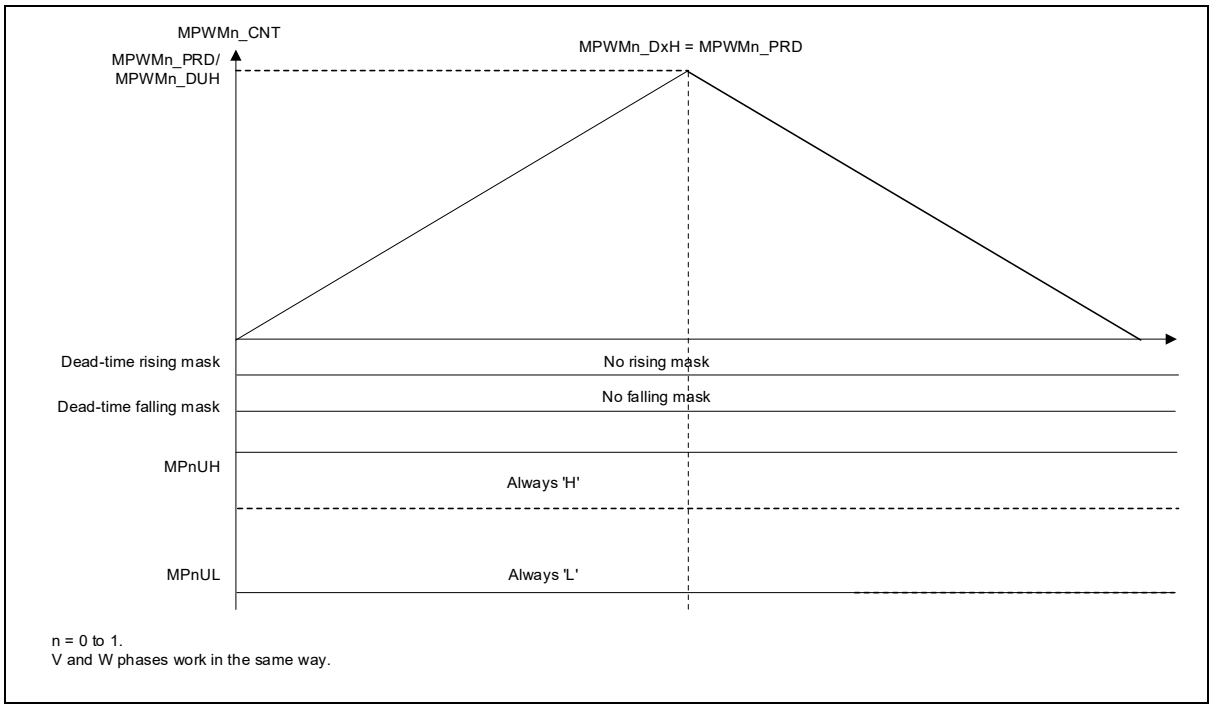
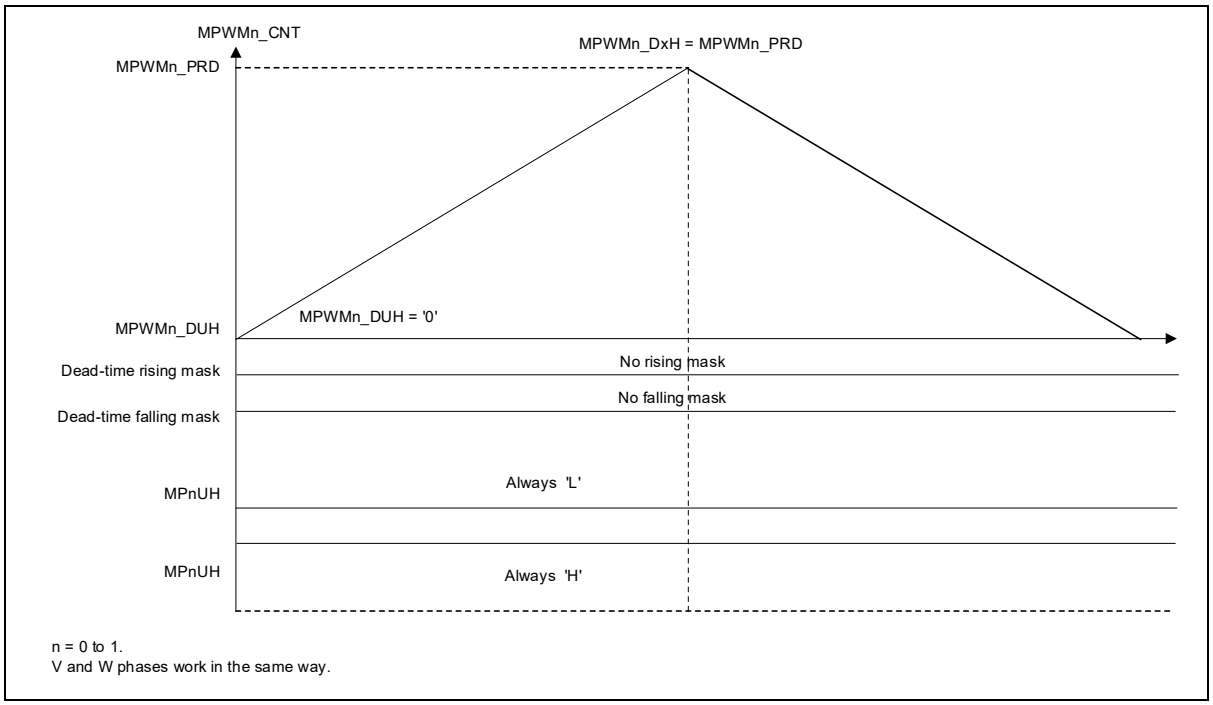


Figure 158. Low Side Always-On (T_{DUTY} = 0, when dead-time is disabled.)



14.3.9 Duty and Period Update Timing

Users can determine the time when the duty and period are applied to internal logic by configuring the UAO, TUP, and BUP bits of the MPWM_MR register.

The UAO bit allows the users to select an update method between two shown below:

- If UAO = 0, the updates are performed at preset time.
- If UAO = 1, the updates for the duty and period are performed immediately when requested. With these settings, the duty and period registers are updated after two PWM clock cycles.

The TUP and BUP bits are valid when the UAO bit is set to '0'. The duty and period values are applied to the internal logic according to the TUP and BUP bits when the period matching event or bottom event occurs.

14.3.10 IRQ Interval

In Motor and Normal PWM modes, users can adjust intervals between interrupts by configuring the IRQN[2:0] bits in the MPWMn_CR1 register. Number set in the IRQN[2:0] bits can be applied to the bottom, period matching, and duty matching (or ATRm) interrupts. For detailed information of these interrupts, refer to section 14.3.18. After adjusting the intervals between interrupts, the current interrupt count can be confirmed in the IRQCNT[2:0] bits in the MPWMn_SR register.

In Individual PWM mode, users can adjust the interrupt interval for the U, V, or W phase by configuring the UIRQN[2:0], VIRQN[2:0], or WIRQN[2:0] bits of the MPWMn_CR3 register, respectively. In addition, as in Motor PWM mode and Normal PWM mode, the current interrupt count of the U, V, or W phase can be confirmed at the WIRQCNT[2:0] bits in the MPWMn_SR register, VIRQCNT[2:0] bits in the MPWMn_SR register, or UIRQCNT[2:0] bits in the MPWMn_SR register.

Please remember that the IRQCNT[2:0] and UIRQCNT[2:0] are represented in the same bit position.

14.3.11 Direction Bit

In Motor and Normal PWM modes, users can see if the current counter is up counter or down counter by checking the DOWN bit in the MPWMn_SR register; while in Individual PWM mode, users can see if the current counter of the phase U, V, or W is up counter or down counter by checking the UDOWN, VDOWN, or WDOWN bit in the MPWMn_SR register.

Note that the DOWN and UDOWN are represented in the same bit position.

14.3.12 ADC Synchronization

A certain application may require that the ADC module synchronizes to the PWM before measuring analog signals, and after the measurement, is controlled based on the measured data. To this end, the MPWM module of A34M420 causes a conversion trigger toward the ADC module.

Table 122 describes the trigger sources that can be generated in the MPWM module.

Table 122. Trigger Source by MPWMn

Tigger Source (ADC Module)	Motor/Normal PWM mode	Individual PWM Mode
0	ATR1	ATR1
1	ATR2	ATR2
2	ATR3	ATR3
3	ATR4	ATR4
4	ATR5	ATR5
5	ATR6	ATR6
6	PERIOD	PERIODU
7	BOTTOM	BOTTOMU
8	-	PERIODV
9	-	BOTTOMV
10	-	PERIODW
11	-	BOTTOMW

In Motor and Normal PWM modes, the period matching and bottom trigger occur repeatedly in a fixed period, according to the set in the MPWMn_PRD register. However, the triggers due to the values of the MPWMn_ATRx registers (x = 1 to 6) occur in variable periods depending on the ATCNT[15:0] bits.

The ATR (ADC trigger counter register) register supports the following functionalities:

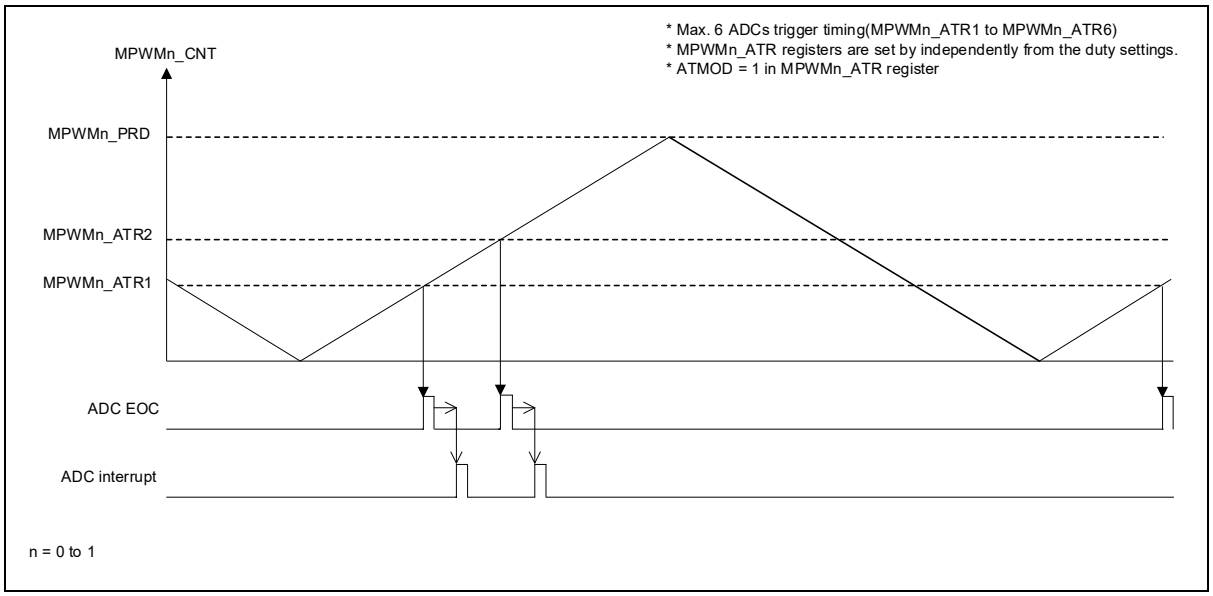
- Selection of ADC trigger source counter
- Selection of count match intervals
- Selection of internal application and update timing

In Individual PWM mode, configuring the ATSRC[1:0] bits in the MPWMn_ATRx register allows to select a trigger source counter of U, V, or W phase, and each trigger source counter can be set in the MPWMn_ATRx register independently. (x = 1 to 6). In Motor and Normal PWM modes, the ATSRC[1:0] bits must be set to default value '00'.

In addition, users can select a count match interval by configuring each ATR register. Configuring the ATMOD[1:0] bits in the MPWMn_ATRx register causes an ADC trigger, if the corresponding counter value matches the ATCNT[15:0] value while the counter is operating as an up counter, a down counter, or an up-down counter. Note that the counter causes the ADC triggers a single time per period when it operates as an up counter or a down counter; while the counter causes the ADC trigger two times per period when it operates as an up-down counter.

The MPWMn_ATRx register (x = 1 to 6) is applied to the internal logic at a specific point of time, similar to the period and duty control registers. A certain application may require changing the time when the ADC trigger occurs, in conjunction with PWM duty. To this end, the UAO, TUP, BUP bits of the MPWMn_MR register configure the update timing so that the values set in the MPWMn_ATRx register are applied to the internal logic.

Figure 159. ADC Conversion Timing Diagram by MPWM



14.3.13 Forced Output Mode

A certain application requires to set the PWM outputs to high or low, and immediately stop the external switch operation. The MPWM of the A34M420 provides a function to force each output under external condition or for user purposes.

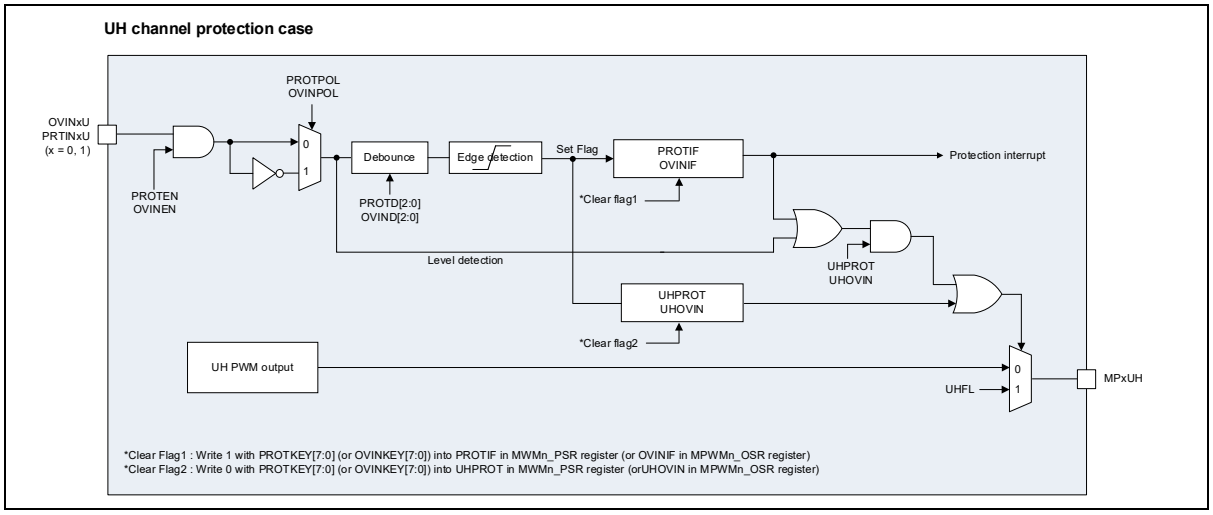
The MPWMn_FOLR register allows to set the polarity of the six forced outputs. Usually, it is set to the off pole of the externally connected switch.

The forced outputs can be caused during the protection operation, and for more information, refer to section 14.3.14.

14.3.14 Protection and Overvoltage

Figure 160 shows a block diagram of the protection and overvoltage. When the protection or overvoltage event occurs, the outputs can be forcibly set to the preset high or low instead of the PWM outputs.

Figure 160. Protection and Overvoltage Block Diagram



The MPWM_PCR and MPWMn_OCR registers support the following protection operations:

- Protection input activation
- Protection input polarity selection
- Protection input debounce setup
- Protection interrupt
- During the protection operation, setup of whether to force output for each output.

Regarding the protection operation, the MPWMn_FOLR register allows users to enable or disable forced outputs using software. When writing to these registers using software, the FOLKEY [7:0] bits in the MPWMn_FOLR register must be written with ‘0xCA’.

Refer to section 14.3.2 or protection pin information, and section 0 for protection related interrupts.

14.3.15 Individual PWM Mode

The MOTORB[1:0] bits in the MPWMn_MR register select the MPWM mode, and specifically, when the MOTORB is set to '3', the current mode is Individual PWM mode. Each Individual PWM mode for the U, V, and W phases consists of high and low of the corresponding phase, such as UH and UL, VH and VL, and WH and W, and operates with own counter.

This characteristic can be used for applications such as IH cooker that control frequencies separately for each pair of PWM outputs.

For Individual PWM mode, A34M420 provides a separate register group (MPWMn_CR3 to MPWMn_SCAPx) and describes it in section 14.7.

14.3.16 Input Capture Function

Individual PWM mode provides the capability to capture inputs on the external input pins. These input capture pins are listed in Table 123.

Table 123. Input Capture Pins of MPWM

Pin Name	Description
CAPEU	Input pin in MPWMn phase U dedicated to capturing in individual PWM mode
CAPEV	Input pin in MPWMn phase V dedicated to capturing in individual PWM mode
CAPEW	Input pin in MPWMn phase W dedicated to capturing in individual PWM mode

The input capture function can be controlled by configuring the MPWMn_CAPCNTx, MPWMn_RCAPx, and MPWMn_FCAPx registers.

14.3.17 Input Sub-capture Function

In addition to the input capture function, Individual PWM mode provides the input sub-capture capability for the additional external input pins. These input sub-capture pins are listed in Table 124.

Table 124. Sub-capture Input Pin of MPWM

Pin Name	Description
SCAPEU	Input pin in MPWMn phase U dedicated to sub-capturing in individual PWM mode
SCAPEV	Input pin in MPWMn phase V dedicated to sub-capturing in individual PWM mode
SCAPEW	Input pin in MPWMn phase W dedicated to sub-capturing in individual PWM mode

The input sub-capture function can be controlled by configuring the MPWMn_SCAPx register.

14.3.18 Interrupt Generation

Table 125 describes the interrupts that the MPWM module can cause in Individual PWM mode.

Table 125.Interrupt Vector of MPWMn

MPWM Unit	Interrupt Priority	Vector Address	Interrupt Source	Motor PWM Mode / Normal PWM Mode	Individual PWM Mode
MPWM0	45	0x0000_00F4	MPWM0PROT	O	O
	46	0x0000_00F8	MPWM0OVV	O	O
	47	0x0000_00FC	MPWM0 (U)	O	O
	48	0x0000_0100	MPWM0 (V)	-	O
	49	0x0000_0104	MPWM0 (W)	-	O
MPWM1	50	0x0000_0108	MPWM1PROT	O	O
	51	0x0000_010C	MPWM1OVV	O	O
	52	0x0000_0110	MPWM1 (U)	O	O
	53	0x0000_0114	MPWM1 (V)	-	O
	54	0x0000_0118	MPWM1 (W)	-	O

Table 126 describes the interrupts that the MPWM module can cause in Motor PWM mode or Normal PWM mode.

Table 126. Interrupt Source and Event of Motor PWM and Normal PWM Modes

Interrupt Source	Interrupt Event Description
MPWM0PROT	Protection event by protection input pins (PRTIN0U / PRTINEV / PRTINEW)
MPWM0OVV	Overvoltage event by overvoltage input pins (OVIN0U / OVINEV / OVINEW)
MPWM1PROT	Protection event by protection input pins (PRTIN1U / PRTINEV / PRTINEW)
MPWM1OVV	Overvoltage event by overvoltage input pins (OVIN1U / OVINEV / OVINEW)
MPWMn	Bottom matching
	Period matching
	Duty or ATRx matching

Table 127 describes the interrupts that the MPWM module can cause in Individual PWM mode.

Table 127. Interrupt Source and Event of Individual PWM Mode

Interrupt Source	Interrupt Event Description
MPWM0PROT	Protection event by protection input pins (PRTIN0U / PRTINEV / PRTINEW)
MPWM0OVV	Overvoltage event by overvoltage input pins (OVIN0U / OVINEV / OVINEW)
MPWM1PROT	Protection event by protection input pins (PRTIN1U / PRTINEV / PRTINEW)
MPWM1OVV	Overvoltage event by overvoltage input pins (OVIN1U / OVINEV / OVINEW)
MPWM0W	MPWM0 W phase bottom
	MPWM0 W phase period matching
	MPWM0 Duty WH or ATR6 matching
	MPWM0 Duty WL or ATR3 matching
MPWM1W	MPWM1 W phase bottom of MPWM1
	MPWM1 W phase period matching
	MPWM1 Duty WH or ATR6 matching
	MPWM1 Duty WL or ATR3 matching
MPWM0V	MPWM0 V phase bottom
	MPWM0 V phase period matching
	MPWM0 Duty VH or ATR5 matching
	MPWM0 Duty VL or ATR2 matching
MPWM1V	MPWM1 V phase bottom
	MPWM1 V phase period matching
	MPWM1 Duty VH or ATR5 matching
	MPWM1 Duty VL or ATR2 matching
MPWM0U	MPWM0 U phase bottom
	MPWM0 U phase period matching
	MPWM0 Duty UH or ATR4 matching
	MPWM0 Duty UL or ATR1 matching
MPWM1U	MPWM1 U phase bottom
	MPWM1 U phase period matching
	MPWM1 Duty UH or ATR4 matching
	MPWM1 Duty UL or ATR1 matching

14.4 MPWM Low-Power Modes

Table 128. Effect of Low-Power Modes on MPWM

Mode	Description
SLEEP	Support
DEEP-SLEEP	Not support

14.5 MPWM Interrupts

Table 129. Interrupt on MPWM

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
Reserved (Individual PWM mode: W period)	PRDWIF	PRDWIE	Write “1” in PRDWIF
Reserved (Individual PWM mode: W bottom)	BOTWIF	BOTWIE	Write “1” in BOTWIF
Reserved (Individual PWM mode: V period)	PRDVIF	PRDVIE	Write “1” in PRDVIF
Reserved (Individual PWM mode: V bottom)	BOTVIF	BOTVIE	Write “1” in BOTVIF
Reserved (Individual PWM mode: U period)	PRDUIF	PRDIE	Write “1” in PRDUIF
Reserved (Individual PWM mode: U bottom)	BOTUIF	BOTIE	Write “1” in BOTUIF
WH duty or ATR6 matching	DWHIF	WHIE	Write “1” in DWHIF
VH duty or ATR5 matching	DVHIF	VHIE	Write “1” in DVHIF
UH duty or ATR4 matching	DUHIF	UHIE	Write “1” in DUHIF
WL duty or ATR3 matching	DWLIF	WLIE	Write “1” in DWLIF
VL duty or ATR2 matching	DVLIF	VLIE	Write “1” in DVLIF
UL duty or ATR1 matching	DULIF	ULIE	Write “1” in DULIF
PRTIN0U / PRTIN1U / PRTINEV / PRTINEW input	PROTIF	PROTIE	Write “1” in PROTIF
OVIN0U / OVIN1U / OVINEV / OVINEW input	OVINIF	OVINIE	Write “1” in OVINIF

14.6 MPWM Registers: Motor PWM and Normal PWM Modes

The base addresses of the MPWMs and register map are described in the followings:

Table 130. Base Address of MPWM

Name	Base Address
MPWM0	0x4000_4000
MPWM1	0x4000_5000

Table 131. MPWM Register Map

Name	Offset	Type	Description	Reset Value	Reference
MPWMn_MR	0x0000	RW	MPWM n Mode Register	0x0000_0000	14.6.1
MPWMn_OLR	0x0004	RW	MPWM n Output Level Register	0x0000_0000	14.6.2
MPWMn_FOLR	0x0008	RW	MPWM n Forced Output Register	0x0000_0000	14.6.3
MPWMn_PRD	0x000C	RW	MPWM n PWM Period Register	0x0000_0002	14.6.4
MPWMn_DUH	0x0010	RW	MPWM n Duty UH Register	0x0000_0001	14.6.5
MPWMn_DVH	0x0014	RW	MPWM n Duty VH Register	0x0000_0001	14.6.6
MPWMn_DWH	0x0018	RW	MPWM n Duty WH Register	0x0000_0001	14.6.7
MPWMn_DUL	0x001C	RW	MPWM n Duty UL Register	0x0000_0001	14.6.8
MPWMn_DVL	0x0020	RW	MPWM n Duty VL Register	0x0000_0001	14.6.9
MPWMn_DWL	0x0024	RW	MPWM n Duty WL Register	0x0000_0001	14.6.10
MPWMn_CR1	0x0028	RW	MPWM n Control Register 1	0x0000_0000	14.6.11
MPWMn_CR2	0x002C	RW	MPWM n Control Register 2	0x0000_0000	14.6.12
MPWMn_SR	0x0030	RW	MPWM n Status Register	0x0000_0000	14.6.13
MPWMn_IER	0x0034	RW	MPWM n Interrupt Enable Register	0x0000_0000	14.6.14
MPWMn_CNT	0x0038	R	MPWM n Counter Register	0x0000_0000	14.6.15
MPWMn_DTR	0x003C	RW	MPWM n Dead-time Register	0x0000_0000	14.6.16
MPWMn_PCR	0x0040	RW	MPWM n Protection Register	0x0000_0000	14.6.17
MPWMn_PSR	0x0044	RW	MPWM n Protection Status Register	0x0000_0000	14.6.18
MPWMn_OCR	0x0048	RW	MPWM n OverVoltage Detection Register	0x0000_0000	14.6.19
MPWMn_OSR	0x004C	RW	MPWM n OverVoltage Detection Status Register	0x0000_0000	14.6.20
MPWMn_ATR1	0x0058	RW	MPWM n ADC Trigger 1 Register	0x0000_0000	14.6.21
MPWMn_ATR2	0x005C	RW	MPWM n ADC Trigger 2 Register	0x0000_0000	14.6.21
MPWMn_ATR3	0x0060	RW	MPWM n ADC Trigger 3 Register	0x0000_0000	14.6.21
MPWMn_ATR4	0x0064	RW	MPWM n ADC Trigger 4 Register	0x0000_0000	14.6.21
MPWMn_ATR5	0x0068	RW	MPWM n ADC Trigger 5 Register	0x0000_0000	14.6.21

Table 120. MPWM Register Map (continued)

Name	Offset	Type	Description	Reset Value	Reference
MPWMn_ATR6	0x006C	RW	MPWM n ADC Trigger 6 Register	0x0000_0000	14.6.21
MPWMn_CR3	0x0080	RW	MPWM n Control Register 3	0x0000_0000	14.7.1
MPWMn_CR4	0x0084	RW	MPWM n Control Register 4	0x0000_0000	14.7.2
MPWMn_PRDU	0x0090	RW	MPWM n Phase U Period register	0x0000_0002	14.7.3
MPWMn_PRDV	0x0094	RW	MPWM n Phase V Period register	0x0000_0002	14.7.4
MPWMn_PRDW	0x0098	RW	MPWM n Phase W Period register	0x0000_0002	14.7.5
MPWMn_CNTU	0x00A0	RO	MPWM n Phase U Counter register	0x0000_0000	14.7.6
MPWMn_CNTV	0x00A4	RO	MPWM n Phase V Counter register	0x0000_0000	14.7.1
MPWMn_CNTW	0x00A8	RO	MPWM n Phase W Counter Register	0x0000_0000	14.7.2
MPWMn_DTRU	0x00B0	RW	MPWM n Phase U Dead-time Register	0x0000_0000	14.7.3
MPWMn_DTRV	0x00B4	RW	MPWM n Phase V Dead-time Register	0x0000_0000	14.7.4
MPWMn_DTRW	0x00B8	RW	MPWM n Phase W Dead-time Register	0x0000_0000	14.7.5
MPWMn_CAPCNTU	0x00C0	RW	MPWM n Phase U Capture Counter Register	0x0000_0001	14.7.6
MPWMn_CAPCNTV	0x00C4	RW	MPWM n Phase V Capture Counter Register	0x0000_0001	14.7.6
MPWMn_CAPCNTW	0x00C8	RW	MPWM n Phase W Capture Counter Register	0x0000_0001	14.7.6
MPWMn_RCAPU	0x00D0	RW	MPWM n Phase U Rising Capture Register	0x0000_0000	14.7.7
MPWMn_RCAPV	0x00D4	RW	MPWM n Phase V Rising Capture Register	0x0000_0000	14.7.7
MPWMn_RCAPW	0x00D8	RW	MPWM n Phase W Rising Capture Register	0x0000_0000	14.7.7
MPWMn_FCAPU	0x00E0	RW	MPWM n Phase U Falling Capture Register	0x0000_0000	14.7.8
MPWMn_FCAPV	0x00E4	RW	MPWM n Phase V Falling Capture Register	0x0000_0000	14.7.8
MPWMn_FCAPW	0x00E8	RW	MPWM n Phase W Falling Capture Register	0x0000_0000	14.7.8
MPWMn_SCAPU	0x00F0	RW	MPWM n Phase U Sub-capture Register	0x0000_0000	14.7.9
MPWMn_SCAPV	0x00F4	RW	MPWM n Phase V Sub-capture Register	0x0000_0000	14.7.9
MPWMn_SCAPW	0x00F8	RW	MPWM n Phase W Sub-capture Register	0x0000_0000	14.7.9

14.6.1 MPWMn_MR: MPWM n Mode Register

The MPWMn_MR register is used to set the MPWM driving method.

After initial PWM period and duty setting is completed, the UAO bit must be set once to update internal operating registers with these set values. This helps transfer the setup data from the user interface register to internal operating registers. The settings of the UAO bit must remain unchanged for at least two PWM clock cycles; otherwise, the update command may get lost, resulting in retaining old data in the internal registers.

The MCHMOD[1:0] value of this register is valid only when the MOTORB[1:0] value is '0'; otherwise, the MCHMOD[1:0] value is internally ignored and these bits field remains '00'.

The UPDOWN value is valid if the MOTORB[1:0] bits has been set to 0x1; otherwise, the UPDOWN value is internally ignored and remains at '1'. The PWM counter always operates as an up-down counter in Motor PWM mode and Individual PWM mode.

MPWM0_MR=0x4000_4000, MPWM1_MR=0x4000_5000

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
Reserved		MOTORB[1:0]	Reserved	UAO	Reserved	TUP	BUP	Reserved	MCHMOD[1:0]	UPDOWN
-		00	-	0	-	0	0	-	00	0
-		RW	-	RW	-	RW	RW	-	RW	RW

15 14	MOTORB[1:0]	MPWM mode selection bit	
		00	Motor PWM mode
		01	Normal PWM mode
		11	Individual PWM mode
7	UAO	Update timing setting bit	
		0	Updates are performed at designated timings.
		1	The duty and period are updated immediately when requested.
5	TUP	Duty and period update enable bit (when same period)	
		0	Periods and duties are not updated at every period match.
		1	Periods and duties are updated at every period match.
4	BUP	Duty and period update enable bit (when same bottom)	
		0	Periods and duties are not updated at every bottom match.
		1	Periods and duties are updated at every bottom match.
2 1	MCHMOD[1:0]	Channel symmetry/ asymmetry mode selection bit (In Normal PWM mode, it operates without setting)	
		00	2-channel symmetric mode The H-side duty determines the H channel signal's timing of switching between the high and low levels. The L-side duty determines the L channel signal's timing of switching between the high and low levels.
		01	1-channel asymmetric mode The H-side duty determines the H channel signal's timing of switching to the high level. The L-side duty determines the L channel signal's timing of switching to the low level. The H and L channels are inverted.

		10	1-channel symmetric mode The H-side duty determines the H channel signal's timing of switching between the high and low levels. The H and L channels are inverted.
		11	Reserved
0	UPDOWN	PWM Up / Down counter mode selection	
		0	PWM Up counter mode (Used when MOTORB[1:0] = 1)
		1	PWM Up-down counter mode (Used when MOTORB[1:0] = 0, 1, or 3)

NOTE:

1. In Individual PWM mode, MPWM0 and MPWM1 cannot be used simultaneously.

14.6.2 MPWMn_OLR: MPWM n Output Level Register

The MPWMn_OLR register controls the level of each PWM output port.

MPWM0_OLR=0x4000_4004, MPWM1_OLR=0x4000_5004

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
Reserved			DOLWH	DOLVH	DOLUH	DOLWL	DOLVL	DOLUL	Reserved	WHL	VHL	UHL	WLL	VLL	ULL
-			0	0	0	0	0	0	-	0	0	0	0	0	0
-			RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW

13	DOLWH(1)	Output level selection bit when disabling WH PWM output	
		0	Low level
		1	High level
12	DOLVH(1)	Output level selection bit when disabling VH PWM output	
		0	Low level
		1	High level
11	DOLUH(1)	Output level selection bit when disabling UH PWM output	
		0	Low level
		1	High level
10	DOLWL(1)	Output level selection bit when disabling WL PWM output	
		0	Low level
		1	High level
9	DOLVL(1)	Output level selection bit when disabling VL PWM output	
		0	Low level
		1	High level
8	DOLUL(1)	Output level selection bit when disabling UL PWM output	
		0	Low level
		1	High level
5	WHL	Output selection bit when starting WH PWM	
		0	Basic output level (low level)
		1	Inverted output level (high level)
4	VHL	Output selection bit when starting VH PWM	
		0	Basic output level (low level)
		1	Inverted output level (high level)
3	UHL	Output selection bit when starting UH PWM	
		0	Basic output level (low level)
		1	Inverted output level (high level)

2	WLL	Output selection bit when starting WL PWM
		0 Basic output level (low level)
		1 Inverted output level (high level)
1	VLL	Output selection bit when starting VL PWM
		0 Basic output level (low level)
		1 Inverted output level (high level)
0	ULL	Output selection bit when starting UL PWM
		0 Basic output level (low level)
		1 Inverted output level (high level)

NOTE:

2. Refer to Table 118 for the basic output levels in each operating mode. DOLxH, DOLxL refers to the output level when the PWM output has been stopped.

14.6.3 MPWMn_FOLR: MPWM n Forced Output Register

A specific PWM output level can be forcibly generated by an abnormality event triggered by an external condition or the user's intentional manipulation. Once a forcing condition is met, each PWM channel generates an output signal in the level set in the MPWMn_FOLR register.

MPWM0_FOLR=0x4000_4008, MPWM1_FOLR=0x4000_5008

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
Reserved				WHFL	VHFL	UHFL	WLFL	VLFL	ULFL
-				0	0	0	0	0	0
-				RW	RW	RW	RW	RW	RW

5	WHFL	WH output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"
4	VHFL	VH output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"
3	UHFL	UH output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"
2	WLFL	WL output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"
1	VLFL	VL output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"
0	ULFL	UL output forced level selection
		0 Output forced level: "L"
		1 Output forced level: "H"

14.6.4 MPWMn_PRD: MPWM n Period Register

The MPWMn_PRD register sets the PWM period.

MPWM0_PRD=0x4000_400C, MPWM1_PRD=0x4000_500C

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
Reserved		PERIOD[15:0]	
-		0x0002	
-		RW	

15 0	PERIOD[15:0]	16-bit PWM period of phase U, V, and W The bits value must be larger than 0x0010.

14.6.5 MPWMn_DUH: MPWM n Duty UH Register

The MPWMn_DUH register sets the PWM duty of phase U. The PWM duty of U-phase is determined by the MPWMn_DUH and MPWM_DUL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DUH=0x4000_4010, MPWM1_DUH=0x4000_5010

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
Reserved		DUTY_UH[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_UH[15:0]	16-bit PWM duty for UH output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.6 MPWMn_DVH: MPWM n Duty VH Register

The MPWMn_DVH register sets the PWM duty of phase V. The PWM duty of V-phase is determined by the MPWMn_DVH and MPWMn_DVL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DVH=0x4000_4014, MPWM1_DVH=0x4000_5014

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
Reserved		DUTY_VH[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_VH[15:0]	16-bit PWM duty for VH output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.7 MPWMn_DWH: MPWM n Duty WH Register

The MPWMn_DWH register sets the PWM duty of phase W. The PWM duty of W-phase is determined by the MPWMn_DWH and MPWMn_DWL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DWH=0x4000_4018, MPWM1_DWH=0x4000_5018

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
Reserved		DUTY_WH[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_WH[15:0]	16-bit PWM duty for WH output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.8 MPWMn_DUL: MPWM n Duty UL Register

The MPWMn_DUL register sets the PWM duty of the phase U.

MPWM0_DUL=0x4000_401C, MPWM1_DUL=0x4000_501C

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
Reserved		DUTY_UL[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_UL[15:0]	16-bit PWM duty for UL output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.9 MPWMn_DVL: MPWM n Duty VL Register

The MPWMn_DVL register sets the PWM duty of phase V.

MPWM0_DVL=0x4000_4020, MPWM1_DVL=0x4000_5020

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
Reserved		DUTY_VL[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_VL[15:0]	16-bit PWM duty for VL output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.10 MPWMn_DWL: MPWM n Duty WL Register

The MPWMn_DWL register sets the PWM duty of phase W.

MPWM0_DWL=0x4000_4024, MPWM1_DWL=0x4000_5024

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
Reserved		DUTY_WL[15:0]	
-		0x0001	
-		RW	

15 0	DUTY_VL[15:0]	16-bit PWM duty for VL output (Duty can be set to zero.)

NOTE:

1. Setting a full duty generates a “low” output, while setting a zero duty generates a “high” output.

14.6.11 MPWMn_CR1: MPWM n Control Register 1

The MPWMn_CR1 register sets the MPWM interrupt period and enables the MPWM.

MPWM0_CR1=0x4000_4028, MPWM1_CR1=0x4000_5028

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
Reserved			IRQN[2:0]	Reserved	PWMEN
-			000	-	0
-			RW	-	RW

10 8	IRQN[2:0](1)	Interrupt request (IRQ) interval number PRDIRQ, BOTIRQ, and ATRx are made at the specified intervals.(1)	
		7 0	Number of interrupt interval Interval will be IRQN[2:0] + 1.
0	PWMEN	Enable the PWM If this bit is set to ‘0’, the PWM block remains in the reset status but the user interface is accessible. This bit must be set to ‘1’ to operate the PWM block.	
		0	Disables
		1	Enables

NOTE:

1. By default, PRDIRQ and BOTIRQ are made at every period. However, interrupt intervals can be set from 1 to 8 periods. If IRQN[2:0] = 0, IRQ is made at every period. Otherwise, they are made at every “IRQN[2:0] + 1” periods.

14.6.12 MPWMn_CR2: MPWM n Control Register 2

The MPWMn_CR2 register controls the MPWM operations such as start, stop, and halt.

MPWM0_CR2=0x4000_402C, MPWM1_CR2=0x4000_502C

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
Reserved			HALT	Reserved		PSTART
-			0	-		0
-			RW	-		RW

7	HALT	Setting PWM output halting which stops the PWM counter but does not reset it. When HALT = '1', the PWM output retains its state when a halt is made.	
		0	Disable PWM output halt
		1	Enable PWM output halt
0	PSTART(1)	PWM counter start or stop / clear	
		1	Starts the PWM counter (Re-initialized at the second PWM clock period).
		0	Stops and clears the PWM counter.

NOTE:

1. Before setting the PSTART bit, the PWMEN bit must be set to '1' to start the PWM counter.

14.6.13 MPWMn_SR: MPWM n Status Register

The MPWMn_SR register shows MPWM status.

MPWM0_SR=0x4000_4030, MPWM1_SR=0x4000_5030

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
Reserved	WDOWN	WIRQCNT[2:0]	VDOWN	VIRQCNT[2:0]	DOWN / UDOWN	UIRQCNT[2:0] IRQCNT[2:0]	PRDWIF	BOTWIF	PRDVIF	BOTVIF	PRDUIF	BOTUIF	DWHIF / ATR6F	DVHIF / ATR5F	DUHIF / ATR4F	DWLIF / ATR3F	DVLIF / ATR2F	DULIF / ATR1F
-	0	000	0	000	0	000	0	0	0	0	0	0	0	0	0	0	0	0
-	RO	RO	RO	RO	RO	RO	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

23	WDOWN	Current mode of PWM's phase W counter - Motor mode: Not used. - Individual mode: This bit displays which mode the phase W counter is in.
		0 Current count mode is up counter.
		1 Current count mode is down counter.
22 20	WIRQCNT[2:0]	Phase W channel's period match interrupt count value (interval PRDIRQ mode) - Motor mode: Not used. - Individual mode: This bit displays which mode the phase W counter is in.
19	VDOWN	Current mode of PWM's phase V counter - Motor mode: Not used. - Individual mode: This bit displays which mode the phase V counter is in.
		0 Current count mode is up counter.
		1 Current count mode is down counter.
18 16	VIRQCNT[2:0]	Phase V channel's period match interrupt count value (interval PRDIRQ mode) - Motor mode: Not used. - Individual mode: This bit displays which mode the phase V counter is in.
15	DOWN UDOWN	Current mode of PWM's (Motor mode) or phase U's (Individual mode) counter - Motor mode: This bit displays which mode the PWM counter is in. - Individual mode: This bit displays which mode the phase U counter is in.
		0 Current count mode is up counter.
		1 Current count mode is down counter.
14 12	IRQCNT[2:0] UIRQCNT[2:0]	All or phase U channel's period match interrupt count value (interval PRDIRQ mode) - Motor mode: This bit displays the phase U, V, and W channels' interrupt count value. - Individual mode: The bit displays the phase U channel's IRQ interrupt count value.
11	PRDWIF	PWM period match interrupt flag - Motor mode: Period interrupt flag - Individual mode: Phase W channel's period interrupt flag
		0 The interrupt has not occurred.
		1 The interrupt has occurred. (Writing "1" clears the flag.)
10	BOTWIF	PWM bottom match interrupt flag - Motor mode: Bottom interrupt flag - Individual mode: Phase W channel's bottom interrupt flag
		0 The interrupt has not occurred.

		1 The interrupt has occurred. (Writing “1” clears the flag.)
9	PRDVIF	PWM period match interrupt flag - Motor mode: Period interrupt flag - Individual mode: Phase V channel’s period interrupt flag
		0 The interrupt has not occurred.
		1 The interrupt has occurred. (Writing “1” clears the flag.)
8	BOTVIF	PWM bottom match interrupt flag - Motor mode: Bottom interrupt flag - Individual mode: Phase V channel’s bottom interrupt flag
		0 The interrupt has not occurred.
		1 The interrupt has occurred. (Writing “1” clears the flag.)
7	PRDUIF	PWM period match interrupt flag - Motor mode: Period interrupt flag - Individual mode: Phase U channel’s period interrupt flag
		0 The interrupt has not occurred.
		1 The interrupt has occurred. (Writing “1” clears the flag.)
6	BOTUIF	PWM bottom match interrupt flag - Motor mode: Bottom interrupt flag - Individual mode: Phase U channel’s bottom interrupt flag
		0 The interrupt has not occurred.
		1 The interrupt has occurred. (Writing “1” clears the flag.)
5	DWHIF ATR6F(1)	PWM WH duty interrupt flag The duty interrupt is enabled when ATR6 is set disabled. When ATR6IE is disabled - WH duty interrupt flag When ATR6IE is enabled - ATR6 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)
4	DVHIF ATR5F(1)	PWM VH duty interrupt flag The duty interrupt is enabled when ATR5 is set disabled. When ATR5IE is disabled - VH duty interrupt flag When ATR5IE is enabled - ATR5 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)
3	DUHIF ATR4F(1)	PWM UH duty interrupt flag The duty interrupt is enabled when ATR4 is set disabled. When ATR4IE is disabled - UH duty interrupt flag When ATR4IE is enabled - ATR4 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)
2	DWLIF ATR3F(1)	PWM WL duty interrupt flag The duty interrupt is enabled when ATR3 is set disabled. When ATR3IE is disabled - WL duty interrupt flag When ATR3IE is enabled - ATR3 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)
1	DVLIF ATR2F(1)	PWM VL duty interrupt flag The duty interrupt is enabled when ATR2 is set disabled. When ATR2IE is disabled - VL duty interrupt flag When ATR2IE is enabled - ATR2 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)
0	DULIF ATR1F(1)	PWM UL duty interrupt flag The duty interrupt is enabled when ATR1 is set disabled. When ATR1IE is disabled - UL duty interrupt flag When ATR1IE is enabled - ATR1 interrupt flag
		0 Interrupt is not generated.
		1 Interrupt is generated. (Writing “1” clears the flag.)

NOTE:

1. Each of the MPWMn_SR [5:0] status bits are shared by a duty match interrupt event or an ADC trigger match interrupt event. If the ADC trigger mode is set disabled, these bits are flagged by duty match interrupts; otherwise, they are flagged by ADC trigger counter match interrupts. The ADC trigger mode is selected in the ATMOD[1:0] bit field of the

MPWMn_ATRn register.

14.6.14 MPWMn_IER: MPWM n Interrupt Enable Register

The MPWMn_IER register controls the interrupts that are used for PWM control.

MPWM0_IER=0x4000_4034, MPWM1_IER=0x4000_5034

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
Reserved			PRDWIE	BOTWIE	PRDVIE	BOTVIE	PRDIE / PRDUIE	BOTIE / BOTUIE	WHIE	VHIE	UHIE	WLIE	VLIE	ULIE
-			0	0	0	0	0	0	0	0	0	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

11	PRDWIE	Enable the phase W counter period interrupt - Motor mode: Not used. - Individual mode: Phase W channel's period interrupt
	0	Disable
	1	Enable
10	BOTWIE	Enable the phase W counter bottom interrupt - Motor mode: Not used. - Individual mode: Phase W channel's bottom interrupt
	0	Disable
	1	Enable
9	PRDVIE	Enable the phase V counter period interrupt - Motor mode: Not used. - Individual mode: Phase V channel's period interrupt
	0	Disable
	1	Enable
8	BOTVIE	Enable the phase V counter bottom interrupt - Motor mode: Phase U, V, and W channels' period interrupt - Individual mode: Phase U channel's period interrupt
	0	Disable
	1	Enable
7	PRDIE PRDUIE	Enable the phase U counter bottom interrupt - Motor mode: Phase U, V, and W channels' period interrupt - Individual mode: Phase U channel's bottom interrupt
	0	Disable
	1	Enable
6	BOTIE BOTUIE	Enable the phase U counter bottom interrupt - Motor mode: Phase U, V, and W channels' bottom interrupt - Individual mode: Phase U channel's bottom interrupt
	0	Disable
	1	Enable
5	WHIE(1)(2) ATR6IE(3)	WH Duty or ATR6 match interrupt
	0	Disable
	1	Enable
4	VHIE(1)(2) ATR5IE(3)	VH Duty or ATR5 match interrupt
	0	Disable
	1	Enable
3	UHIE(1)(2) ATR4IE(3)	UH Duty or ATR4 match interrupt
	0	Disable

		1	Enable
2	WLIE[1][2] ATR3IE[3]	WL Duty or ATR3 match interrupt	
		0	Disable
		1	Enable
1	VLIE[1][2] ATR2IE[3]	VL Duty or ATR2 match interrupt	
		0	Disable
		1	Enable
0	ULIE[1][2] ATR1IE[3]	UL Duty or ATR1 match interrupt	
		0	Disable
		1	Enable

NOTES:

1. Each control bit of the MPWMn_IER[5:0] is shared by a duty match interrupt event or an ADC trigger match interrupt event.
2. If the ADC trigger mode is set disabled, these bits are flagged by interrupt duty match conditions; otherwise, they are flagged by ADC interrupt trigger counter match conditions.
3. The ADC trigger mode is selected in the ATMOD[1:0] bits of the MPWMn_ATRn register
4. For example, if ULI, UHI, BOTU, or PRDU is flagged, the MPWM_U interrupt occurs.
5. Additionally, ATR1 and ATR4 are used as MPWM_U interrupts; ATR2 and ATR5 are used as MPWM_V interrupts; and ATR3 and ATR6 are used as MPWM_W interrupts.

14.6.15 MPWMn_CNT: MPWM n Counter Register

The MPWM_CNT register is used to read the count value that generates a PWM waveform.

MPWM0_CNT=0x4000_4038, MPWM1_CNT=0x4000_5038

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
Reserved		CNT[15:0]	
-		0x0000	
-		R0	

15 0	CNT[15:0]	PWM counter value Main counter value in PWM

14.6.16 MPWMn_DTR: MPWM n Dead-time Register

The MPWMn_DTR register is used to prevent short circuits that may be caused on a circuit or in the program.

MPWM0_DTR=0x4000_403C, MPWM1_DTR=0x4000_503C

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
Reserved		DTEN	PSHRT	DTMDSEL	Reserved	DTCLK[1:0]	DT[7:0]
-		0	0	0	-	00	0x00
-		RW	RW	RW	-	RW	RW

15	DTEN	Dead-time function You must disable this function in two-channel symmetric mode because this mode does not support the function.
	0	Disables dead-time function.
	1	Enables dead-time function.
14	PSHRT	Short-circuit protection This function is supported in two-channel symmetric mode. In one-channel mode, both the H and L sides are not simultaneously active. When either side is active, the other side always inactive.
	0	Enables short-circuit protection. (Automatically ensures that both the H and L sides are not simultaneously active)
	1	Disables short-circuit protection
13	DTMDSEL	Dead-time mode selection When the POL function is used, a phase's H and L signals can overlap in the high level. This bit is typically set to prevent such a phenomenon.
	0	Inserts dead-time at the leading edge of PWMxH and the trailing edge of PWMxL
	1	Inserts dead-time at the trailing edge of PWMxH and the leading edge of PWMxL
9 8	DTCLK[1:0]	Dead-time pre-scaler
	00	Dead-time counter uses PWM CLK / 2.
	01	Dead-time counter uses PWM CLK / 4.
	10	Dead-time counter uses PWM CLK / 8.
	11	Dead-time counter uses PWM CLK / 16.
7 0	DT[7:0]	Rising or falling edge dead-time value Set as the delay time from the normal polarity to the falling edge level output.
	0x0	Disable the dead-time duration.
	0x01 to 0xFF	Apply the dead-time duration.

NOTE:

1. Values written to this register are applied to the rising or falling edge dead-times of all three phases U, V, and W. To set a particular phase's dead-time, you must configure DTRU, DTRV, and DTRW registers in individual mode. This short-circuit protection is applied to the PWM's internal signals and not to its external signals. When the internal signals from the H- and L-sides are all at the high level, short-circuit protection induces both sides to generate low-level outputs.

14.6.17 MPWMn_PCR: MPWM n Protection Control Register

The MPWMn_PCR register is used to control the protection operation.

MPWM0_PCR=0x4000_4040, MPWM1_PCR=0x4000_5040

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
WPROTEN	WPROTPOL	Reserved	WPROTD[2:0]	VPROTEN	VPROTPOL	Reserved	VPROTD[2:0]	PROTEN	PROTPOL	Reserved	PROTD[2:0]	PROTIE	Reserved	WHPROTM	VHPROTM	UHPROTM	WLPROTM	VLPROTM	ULPROTM
0	0	-	000	0	0	-	000	0	0	-	000	0	-	0	0	0	0	0	0
RW	RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW

31	WPROTPOL	Enable phase W protection input. - Motor mode: Not used. - Individual mode: Phase W protection input
	0	Disable
	1	Enable
30	WPROTPOL	Phase W protection input polarity selection - Motor mode: Not used. - Individual mode: Selects the polarity of phase W protection input.
	0	L-active
	1	H-active
26 24	WPROTD[2:0]	Phase W protection input debounce clock Debounce clock = $MPWM_{CLK}$ / WPROTD[2:0] - Motor mode: Not used. - Individual mode: Enables phase W protection input pin debounce.
	0	No debounce.
	1	$MPWM_{CLK}$ × 1
	2	$MPWM_{CLK}$ × 2
	3	$MPWM_{CLK}$ × 3
	4	$MPWM_{CLK}$ × 4
	5	$MPWM_{CLK}$ × 5
	6	$MPWM_{CLK}$ × 6
	7	$MPWM_{CLK}$ × 7
23	VPROTEN	Enable phase V protection input. - Motor mode: Not used. - Individual mode: Phase V protection input
	0	Disable
	1	Enable
22	VPROTPOL	Phase V protection input polarity selection Motor mode: Not used. Individual mode: Selects the polarity of phase V protection input.
	0	L-active
	1	H-active
18 16	VPROTD[2:0]	Phase V protection input debounce clock Debounce clock = $MPWM_{CLK}$ / VPROTD[2:0] - Motor mode: Not used. - Individual mode: Enables phase V protection input pin debounce.
	0	No debounce.
	1	$MPWM_{CLK}$ × 1
	2	$MPWM_{CLK}$ × 2
	3	$MPWM_{CLK}$ × 3
	4	$MPWM_{CLK}$ × 4
	5	$MPWM_{CLK}$ × 5

		6	$MPWM_{CLK}$ × 6
		7	$MPWM_{CLK}$ × 7
15	PROTEN UPROTEN	Enable PWM or phase U protection input. - Motor mode: PWM protection input - Individual mode: Phase U protection input	
		0	Disable
		1	Enable
14	PROTPOL	PWM or phase U protection input polarity selection - Motor mode: Selects the polarity of PWM protection input. - Individual mode: Selects the polarity of phase U protection input.	
		0	L-active
		1	H-active
10 8	PROTD[2:0]	PWM or phase U protection input debounce clock Debounce clock = MPWMCLK / PROTD[2:0] - Motor mode: PWM protection input pin debounce - Individual mode: Enables phase U protection input pin debounce.	
		0	No debounce.
		1	$MPWM_{CLK}$ × 1
		2	$MPWM_{CLK}$ × 2
		3	$MPWM_{CLK}$ × 3
		4	$MPWM_{CLK}$ × 4
		5	$MPWM_{CLK}$ × 5
		6	$MPWM_{CLK}$ × 6
		7	$MPWM_{CLK}$ × 7
7	PROTIE	Enable the protection interrupt.	
		0	Disables the protection interrupt.
		1	Enables the protection interrupt.
5	WHPROTM	Protection output at the H side of phase W.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
4	VHPROTM	Protection output at the H side of phase V.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
3	UHPROTM(1)	Protection output at the H side of phase U.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
2	WLPROTM	Protection output at the L side of phase W.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
1	VLPROTM	Protection output at the L side of phase V.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
0	ULPROTM	Protection output at the L side of phase U.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.

NOTE:

1. To enable protection output at the H side of phase U in individual mode, users must first enable phase U protection input. And there is only one protection interrupt vector in Individual mode as well. Users can control the vector by checking the interrupt flag in the interrupt register.

14.6.18 MPWMn_PSR: MPWM n Protection Status Register

Using the MPWMn_PSR register, users can check the status of interrupts that are generated during the protection operation.

MPWM0_PSR=0x4000_4044, MPWM1_PSR=0x4000_5044

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
Reserved		PROTKEY[7:0]	PROTIF	Reserved	WHPROTF	VHPROTF	UHPROTF	WLPROTF	VLPROTF	ULPROTF
-		-	0	-	0	0	0	0	0	0
-		WO	RWC1	-	RW	RW	RW	RW	RW	RW

15 8	PROTKEY[7:0]	Protection clear access key (PSR key: 0xCA) To clear a protection flag, you must write to the flag and this access key. Writing to a flag bit without writing to PROTKEY[7:0] is prohibited.	
		0xCA	Enables protection clear access.
		Others	Disables protection clear access.
7	PROTIF	Protection interrupt status	
		0	The protection interrupt has not occurred.
		1	The protection interrupt has occurred. (Writing “1” clears the flag.)
5	WHPROT	Phase W H-side protection flag	
		0	No Protection occurred. (Writing “0” clears the flag)
		1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of WHFL set in the MPWMn_FOLR register.)
4	VHPROT	Phase V H-side protection flag	
		0	No Protection occurred. (Writing “0” clears the flag)
		1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of VHFL set in the MPWMn_FOLR register.)
3	UHPROT	Phase U H-side protection flag	
		0	No Protection occurred. (Writing “0” clears the flag)
		1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of UHFL set in the MPWMn_FOLR register.)
2	WLPROT	Phase W L-side protection flag	
		0	No Protection occurred. (Writing “0” clears the flag)
		1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of WLFL set in the MPWMn_FOLR register.)
1	VLPROT	Phase V L-side protection flag	
		0	No Protection occurred. (Writing “0” clears the flag)
		1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of VLFL set in the MPWMn_FOLR register.)
0	ULPROT	Phase U L-side protection flag	
		0	No Protection occurred.

		(Writing “0” clears the flag)
	1	Protection has occurred. (Writing “1” forcibly generates an output signal at the level of ULFL set in the MPWMn_FOLR register.)

NOTE:

1. If an arbitrary signal is applied to an external protection pin when the PCR register’s corresponding PROTEN bit is enabled, the PWM output will be generated at the level set in the MPWMn_FOLR reguster. Additionally, the user can write to the PSR register to manually inhibit or forcibly generate a specific output.

14.6.19 MPWMn_OCR: MPWM OverVoltage Detection Register

The MPWMn_OCR register controls the protection operation when over-current is caused.

MPWM0_OCR=0x4000_4048, MPWM1_OCR=0x4000_5048

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
WOVINEN	WOVINPOL	Reserved	WOVIND[2:0]	VOVINEN	VOVINPOL	Reserved	VOVIND[2:0]	OVINEN	OVINPOL	Reserved	OVIND[2:0]	OVINIE	Reserved	WHOVINM	VHOVINM	UHOVINM	WLOVINM	VLOVINM	ULOVINM
0	0	-	000	0	0	-	000	0	0	-	000	0	-	0	0	0	0	0	0
RW	RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW

1	WOVINEN	Enable phase W overvoltage detection input. - Motor mode: Not used. - Individual mode: Phase W overvoltage detection input.
	0	Disable
	1	Enable
30	WOVINPOL	Phase W overvoltage detection input polarity selection - Motor mode: Not used. - Individual mode: Selects the polarity of phase W overvoltage detection input.
	0	L-active
	1	H-active
26 24	WOVIND[2:0]	Phase W overvoltage detection input debounce clock Debounce clock = $MPWM_{CLK}$ / WOVIND[2:0] - Motor mode: Not used. - Individual mode: Phase W protection input pin debounce.
	0	No debounce.
	1	$MPWM_{CLK}$ × 1
	2	$MPWM_{CLK}$ × 2
	3	$MPWM_{CLK}$ × 3
	4	$MPWM_{CLK}$ × 4
	5	$MPWM_{CLK}$ × 5
	6	$MPWM_{CLK}$ × 6
	7	$MPWM_{CLK}$ × 7
23	VOVINEN	Enable phase V overvoltage detection input. - Motor mode: Not used. - Individual mode: Phase V overvoltage detection input.
	0	Disable
	1	Enable
22	VOVINEN	Phase V overvoltage detection input polarity selection - Motor mode: Not used. - Individual mode: Selects the polarity of phase V overvoltage detection input.
	0	L-active
	1	H-active
18 16	VOVIND[2:0]	Phase V overvoltage detection input debounce clock Debounce clock = $MPWM_{CLK}$ / VOVIND[2:0] - Motor mode: Not used. - Individual mode: Phase V protection input pin debounce.
	0	No debounce.
	1	$MPWM_{CLK}$ × 1
	2	$MPWM_{CLK}$ × 2
	3	$MPWM_{CLK}$ × 3

		4	$MPWM_{CLK}$ × 4
		5	$MPWM_{CLK}$ × 5
		6	$MPWM_{CLK}$ × 6
		7	$MPWM_{CLK}$ × 7
15	OVINEN UOVINEN	Enable PWM or phase U overvoltage detection input. - Motor mode: PWM overvoltage detection input - Individual mode: Phase U overvoltage detection input.	
		0	Disable
		1	Enable
14	OVINPOL	PWM or phase U overvoltage detection input polarity selection - Motor mode: Selects the polarity of PWM overvoltage detection input. - Individual mode: Selects the polarity of phase U overvoltage detection input.	
		0	L-active
		1	H-active
10 8	OVIND[2:0]	PWM or phase U overvoltage detection input debounce clock Debounce clock = $MPWM_{CLK}$ / OVIND[2:0] - Motor mode: PMW overvoltage detection input debounce - Individual mode: Phase U protection input pin debounce.	
		0	No debounce.
		1	$MPWM_{CLK}$ × 1
		2	$MPWM_{CLK}$ × 2
		3	$MPWM_{CLK}$ × 3
		4	$MPWM_{CLK}$ × 4
		5	$MPWM_{CLK}$ × 5
		6	$MPWM_{CLK}$ × 6
		7	$MPWM_{CLK}$ × 7
7	OVINIE	Enable the overvoltage detection interrupt.	
		0	Disables the protection interrupt.
		1	Enables the protection interrupt.
5	WHOVINM	Enable the overvoltage detection output at the H side of phase W.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
4	VHOVINM	Enable the overvoltage detection output at the H side of phase V.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
3	UHOVINM	Enable the overvoltage detection output at the H side of phase U.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
2	WLOVINM	Enable the overvoltage detection output at the L side of phase W.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
1	VLOVINM	Enable the overvoltage detection output at the L side of phase V.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.
0	ULOVINM	Enable the overvoltage detection output at the L side of phase U.	
		0	Disables protection output.
		1	Enables the protection output at the level set in FOLR.

NOTE:

1. To enable protection output at the H side of phase U in individual mode, users must first enable phase U protection input. And there is only one protection interrupt vector in Individual mode as well. Users can control the vector by checking the interrupt flag in the interrupt register.

14.6.20 MPWMn_OSR: MPWM OverVoltage Detection Status Register

Using the MPWMn_OSR register, users can check the status of interrupts that are generated during the over-current protection operation.

MPWM0_OSR=0x4000_404C, MPWM1_OSR=0x4000_504C

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
Reserved		OVINKEY[7:0]	OVINIF	Reserved	WHOVINF	VHOVINF	UHOVINF	WLOVINF	VLOVINF	ULOVINF
-		-	0	-	0	0	0	0	0	0
-		WO	RWC1	-	RW	RW	RW	RW	RW	RW

15 8	OVINKEY[7:0]	Overvoltage detection clear access key (OSR key: 0xAC) To clear a overvoltage detection flag, you must write to the flag and this access key. Writing to a flag bit without writing to OVINKEY[7:0] is prohibited.
	0xAC	Enables overvoltage detection clear access.
	Others	Disables overvoltage detection clear access.
7	OVINIF	Overvoltage detection interrupt status
	0	The overvoltage detection interrupt has not occurred.
	1	The overvoltage detection interrupt has occurred. (Writing “1” clears the flag.)
5	WHOVIN	Phase W H-side overvoltage detection flag
	0	Overvoltage has not occurred. (Writing “0” clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of WHFL set in the MPWMn_FOLR register.)
4	VHOVIN	Phase V H-side overvoltage detection flag
	0	Overvoltage has not occurred. (Writing “0” clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of VHFL set in the MPWMn_FOLR register.)
3	UHOVIN	Phase U H-side overvoltage detection flag
	0	Overvoltage has not occurred. (Writing “0”: clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of UHFL set in the MPWMn_FOLR register.)
2	WLOVIN	Phase W L-side overvoltage detection flag
	0	Overvoltage has not occurred. (Writing “0” clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of WLFL set in the MPWMn_FOLR register.)
1	VLOVIN	Phase V L-side overvoltage detection flag
	0	Overvoltage has not occurred. (Writing “0” clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of VLFL set in the MPWMn_FOLR register.)
0	ULOVIN	Phase U L-side overvoltage detection flag
	0	Overvoltage has not occurred.

		(Writing “0” clears the flag)
	1	Overvoltage has occurred. (Writing “1” forcibly generates an output signal at the level of ULFL set in the MPWMn_FOLR register.)

NOTE:

1. If an arbitrary signal is applied to an external protection pin when the OCR register’s corresponding OVINEN bit is enabled, the PWM output will be generated at the level set in the MPWMn_FOLR register. Additionally, the user can write to the MPWMn_OSR register to manually inhibit or forcibly generate a specific output.

14.6.21 MPWMn_ATRx: MPWM n ADC Trigger Counter Register

The MPWMn_ATRx register is 32 bits wide.

MPWM0_ATR1=0x4000_4058, MPWM0_ATR2=0x4000_405C, MPWM0_ATR3=0x4000_4060
MPWM0_ATR4=0x4000_4064, MPWM0_ATR5=0x4000_4068, MPWM0_ATR6=0x4000_406C
MPWM1_ATR1=0x4000_5058, MPWM1_ATR2=0x4000_505C, MPWM1_ATR3=0x4000_5060
MPWM1_ATR4=0x4000_5064, MPWM1_ATR5=0x4000_5068, MPWM1_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
Reserved	ATSRC[1:0]	Reserved	ATUDT	Reserved	ATMOD[1:0]	ATCNT[15:0]	
-	00	-	0	-	0	0x0000	
-	RW	-	RW	-	RW	RW	

23 22	ATSRC[1:0]	ADC trigger source counter
	00	Sets phase U counter as the compare source (Default).
	01	Sets phase V counter as the compare source.
	10	Sets phase W counter as the compare source.
	11	Disables.
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 is set enabled, written trigger register values are sent to the trigger compare block after two PWM clock cycles (through synchronization logic)
17 16	ATMOD[1:0]	ADC trigger mode register
	00	ADC trigger Disable
	01	Trigger out when up count match
	10	Trigger out when down count match
	11	Trigger out when up-down count match
15 0	ATCNT[15:0]	ADC trigger counter - ADC trigger counter value must be smaller than the PWM period. - Setting the counter value to ‘0’ prevents interrupts from being triggered.

NOTE:

1. In Individual mode, ATR1 and ATR4 are used as MPWM_U interrupts; ATR2 and ATR5 are used as MPWM_V interrupts; and ATR3 and ATR6 are used as MPWM_W interrupts.

14.6.22 MPWM Register Map Summary: Normal PWM Mode

Table 132. MPWM Register Map Summary: Normal PWM Mode

Offset	Register name	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
0x00	MPWMn_MR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MOTORB[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	UAO	Res.	TUP	BUP	Res.	MCHMOD[1:0]		UPDOWN
	Reset value																	0	0							0		0	0		0	0	0
0x04	MPWMn_OLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOLWH	DOLVH	DOLUH	DOLWL	DOLVL	DOLUL	Res.	Res.	WHL	VHL	UHL	WLL	VLL	ULL
	Reset value																			0	0	0	0	0	0			0	0	0	0	0	0
0x08	MPWMn_FOLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WHFL	VHFL	UHFL	WLFL	VLFL	ULFL
	Reset value																											0	0	0	0	0	0
0x0C	MPWMn_PRD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0x10	MPWMn_DUH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_UH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x14	MPWMn_DVH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_VH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x18	MPWMn_DWH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_WH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	MPWMn_DUL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_UL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x20	MPWMn_DVL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_VL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x24	MPWMn_DWL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_WL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x28	MPWMn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IRQN[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWMEN
	Reset value																						0	0	0								0

Table 121. MPWM Register Map Summary: Normal PWM Mode (continued)

Offset	Register name	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
0x2C	MPWMn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HALT	Res.	Res.	Res.	Res.	Res.	Res.	PSTART
	Reset value																									0							0
0x30	MPWMn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDOWN	WIRQCNT[2:0]			VDOWN	VIRQCNT[2:0]			UDOWN/DOWN	UIRQCNT[2:0] /IRQCNT[2:0]			PRDWIF	BOTWIF	PRDVIF	BOTVIF	PRDUIF	BOTUIF	DWHIF/ATR6F	DVHIF/ATR5F	DUHIF/ATR4F	DWLIF/ATR3F	DVLIF/ATR2F	DULIF/ATR1F
	Reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	MPWMn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRDWIE	BOTWIE	PRDVIE	BOTVIE	PRDIE/PRDUIE	BOTIE/BOTUIE	WHIE	VHIE	UHIE	WLIE	VLIE	ULIE
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x38	MPWMn_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	MPWMn_DTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTEN	PSHRT	DTMDSEL	Res.	Res.	Res.	DTCLK[1:0]		DT[7:0]							
	Reset value																	0	0	0				0	0	0	0	0	0	0	0	0	0
0x40	MPWMn_PCR	WPROTEN	WPROTPOL	Res.	Res.	Res.	WPROTD[2:0]			VPROTEN	VPROTPOL	Res.	Res.	Res.	VPROTD[2:0]			PROTEN	PROTPOL	Res.	Res.	Res.	PROTD[2:0]			PROTIE	Res.	WHPROTM	VHPROTM	UHPROTM	WLPROTM	VLPROTM	ULPROTM
	Reset value	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0		0	0	0	0	0	0
0x44	MPWMn_PSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PROTKEY[7:0]								PROTIF	Res.	WHPROT	VHPROT	UHPROT	WLPROT	VLPROT	ULPROT
	Reset value																	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0
0x48	MPWMn_OCR	WOVINEN	WOVINPOL	Res.	Res.	Res.	WOVIND[2:0]			VOVINEN	VOVINPOL	Res.	Res.	Res.	VOVIND[2:0]			OVINEN	OVINPOL	Res.	Res.	Res.	OVIND[2:0]			OVINIE	Res.	WHOVINM	VHOVINM	UHOVINM	WLOVINM	VLOVINM	ULOVINM
	Reset value	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0		0	0	0	0	0	0
0x4C	MPWMn_OSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OVINKEY[7:0]								OVINIF	Res.	WHOVIN	VHOVIN	UHOVIN	WLOVIN	VLOVIN	ULOVIN
	Reset value																	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0
0x58	MPWMn_ATR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 121. MPWM Register Map Summary: Normal PWM Mode (continued)

Offset	Register name	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
0x5C	MPWMn_ATR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	MPWMn_ATR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	MPWMn_ATR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x68	MPWMn_ATR5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x6C	MPWMn_ATR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATSRC[1:0]		Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value									0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.7 MPWM Registers: Individual PWM Mode

14.7.1 MPWMn_CR3: MPWM n Control Register

To configure the MPWMn_CR3 register, users must set the MOTORB[1:0] bits in the MPWMn_MR register to '11' (Individual PWM mode). This register replaces the MPWMn_CR1 and MPWMn_CR2 registers to set each phase individually for use in Individual PWM mode.

If one phase is used in Individual PWM mode, U-phase should be used. And when using 2 phases, use U and V phases.

MPWM0_CR3=0x4000_4080, MPWM1_CR3=0x4000_5080

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
Reserved	WIRQN[2:0]	WHALT	Reserved	WSTART	WEN	Reserved	VIRQN[2:0]	VHALT	Reserved	VSTART	VEN	Reserved	UIRQN[2:0]	UHALT	Reserved	USTART	UEN
-	000	0	-	0	0	-	000	0	-	0	0	-	000	0		0	0
-	RW	RW	-	RW	RW	-	RW	RW	-	RW	RW	-	RW	RW		RW	RW

22 20	WIRQN[2:0](1)	Phase W interrupt interval setting (PRDIRQ, BOTIRQ, and ATRx are made at the specified intervals) - Motor mode: Not used. - Individual mode: The IRQ intervals only for phase W.
19	WHALT	Halt the phase W counter in the current state (stops the counter clock only) - Motor mode: Not used. - Individual mode: The bit halts the phase W counter (with the capture counter).
	0	No effect.
	1	Read: Halted phase W counter Write: Halts the phase W counter (along with the capture counter)
17	WSTART(2)	Start PWM operation of phase W counter. - Motor mode: Not used. - Individual mode: The bit starts the phase W counter. Writing to this bit has the counter recount from the beginning.
	0	No effect.
	1	Read: Started phase W counter. Write: Starts the phase W counter.
16	WEN	Enable the phase W counter. - Motor mode: Not used. - Individual mode: The bit enables the phase W counter
	0	No effect.
	1	Read: Enabled phase W. Write: Enables phase W.
14 12	VIRQN[2:0](3)	Phase V interrupt interval setting (PRDIRQ, BOTIRQ, and ATRx are made at the specified intervals) - Motor mode: Not used. - Individual mode: The IRQ intervals only for phase V.
11	VHALT	Halt the phase V counter in the current state (stops the counter clock only) - Motor mode: Not used. - Individual mode: The bit halts the phase V counter (with the capture counter).
	0	No effect.
	1	Read: Halted phase V counter

		Write: Halts the phase V counter (along with the capture counter)
9	VSTART	Starts PWM operation of phase V counter. - Motor mode: Not used. - Individual mode: The bit starts the phase V counter. Writing to this bit has the counter recount from the beginning.
	0	No effect.
	1	Read: Started phase V counter. Write: Starts the phase V counter.
8	VEN	Enable the phase V counter. - Motor mode: Not used. - Individual mode: The bit enables the phase V counter
	0	No effect.
	1	Read: Enabled phase V. Write: Enables phase V.
6 4	UIRQN[2:0][4]	Phase U interrupt interval setting (PRDIRQ, BOTIRQ, and ATRx are made at the specified intervals) - Motor mode: The IRQ intervals for phase U, V and W counters. - Individual mode: The IRQ intervals only for phase U.
3	UHALT	Whether to halt the phase U counter in the current state (stops the counter clock only) - Motor mode: The bit halts the phase U, V and W counters. - Individual mode: The bit halts the phase U counter (with the capture counter).
	0	No effect.
	1	Read: Halted phase U counter Write: Halts the phase U counter (along with the capture counter)
1	USTART	Starts PWM operation of phase U counter. - Motor mode: The bit starts the phase U, V and W counters. - Individual mode: The bit starts the phase U counter. Writing to this bit has the counter recount from the beginning.
	0	No effect.
	1	Read: Started phase U counter. Write: Starts the phase U counter.
0	UEN	Enable PWM phase U counter. - Motor mode: The bit enables the phase U, V and W counters. - Individual mode: The bit enables the phase U counter.
	0	No effect.
	1	Read: Enabled phase U. Write: Enables phase U.

NOTES:

1. WIRQN[2:0] bit field can be written to only when the WEN and UEN bit has been set to 1.
2. WSTART bit can be written to only when the WEN and VEN bit has been set to 1.
3. VIRQN[2:0] bit field can be written to only when the VEN bit has been set to '1'.
4. UIRQN[2:0] bit field can be written to only when the UEN bit has been set to '1'.
5. By default, PRDIRQ and BOTIRQ are made at every period. However, interrupt intervals can be set from 1 to 8 periods. If IRQN[2:0] = 0, IRQ is made at every period. Otherwise, they are made at every "IRQN[2:0] + 1" periods.

14.7.2 MPWMn_CR4: MPWM n Control Register 4

To configure the MPWMn_CR4 register, users must set the MOTORB[1:0] in MPWMn_MR register to '11' (Individual PWM mode). This register replaces the CR1 and CR2 registers to set each phase individually for use in Individual PWM mode.

MPWM0_CR4=0x4000_4084, MPWM1_CR4=0x4000_5084

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
Reserved		WCONTI	Reserved	WSTOP	WDIS	Reserved	VCONTI	Reserved	VSTOP	VDIS	Reserved	UCONTI	Reserved	USTOP	UDIS
-		0	-	0	0	-	0	-	0	0	-	0		0	0
-		RW	-	RW	RW	-	RW	-	RW	RW	-	RW		RW	RW

19	WCONTI	Resume the phase W counter. - Motor mode: Not used. - Individual mode: The bit resumes the phase W counter. (together with capture counter)
	0	No effect.
	1	Read: Halted phase W. Write: Resumes the phase W counter.
17	WSTOP	Stop and reset the phase W counter. - Motor mode: Not used. - Individual mode: The bit stops and resets the phase W counter.
	0	No effect.
	1	Read: Started phase W counter Write: Stops and resets phase W counter.
16	WDIS	Disable and reset the phase W counter. - Motor mode: Not used. - Individual mode: The bit disables and resets the phase W counter.
	0	Write: No effect.
	1	Read: Enable phase W counter Write: Disable and resets phase W counter
11	VCONTI	Resume the phase V counter. - Motor mode: Not used. - Individual mode: The bit resumes the phase V counter. (together with capture counter)
	0	No effect.
	1	Read: Halted phase V. Write: Resumes the phase V counter.
9	VSTOP	Stop and reset the phase V counter. - Motor mode: Not used. - Individual mode: The bit stops and resets the phase V counter.
	0	No effect.
	1	Read: Started phase V counter Write: Stops and resets phase V counter.
8	VDIS	Disable and reset the phase V counter. - Motor mode: Not used. - Individual mode: The bit disables and resets the phase V counter.
	0	Write: No effect.
	1	Read: Enable phase V counter. Write: Disable and resets phase V counter.
3	UCONTI	Resume the phase U counter. - Motor mode: The bit resumes the phase U, V and W counter. - Individual mode: The bit resumes the phase U counter. (together with capture counter)

		0	No effect.
		1	Read: Halted phase U, V and W counters (or phase U). Write: Resumes the phase U, V and W counters (or phase U counter).
1	USTOP	Stop and reset the phase U counter. - Motor mode: The bit stops and resets the phase U, V and W counters. - Individual mode: The bit stops and resets the phase U counter.	
		0	No effect.
		1	Read: Started phase U, V and W counters (or phase U counters) Write: Stops and resets phase U, V and W counters (or phase U counter).
0	UDIS	Disable and reset the phase U counters. - Motor mode: The bit disables and resets the phase U, V and W counters. - Individual mode: The bit disables and resets the phase U counter.	
		0	Write: No effect.
		1	Read: Enable phase U counter. Write: Disable and resets phase U counter.

NOTE:

1. Before setting the PSTART bit, the PWMEN bit must be set to ‘1’ to start the PWM counter.

14.7.3 MPWMn_PRDU: MPWM n Phase U Period Register

MPWM0_PRDU=0x4000_4090, MPWM1_PRDU=0x4000_5090

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
Reserved		PERIOD_U[15:0]	
-		0x0002	
-		RW	

15 0	PERIOD_U[15:0]	16-bit PWM period of U phase The bit value must be larger than 0x0010. - Motor mode: Not used. - Individual mode: The bit sets the period for phase U only.

14.7.4 MPWMn_PRDV: MPWM n Phase V Period Register

MPWM0_PRDV=0x4000_4094, MPWM1_PRDV=0x4000_5094

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
Reserved		PERIOD_V[15:0]	
-		0x0002	
-		RW	

15 0	PERIOD_V[15:0]	16-bit PWM period of V phase The bit value must be larger than 0x0010. - Motor mode: Not used. - Individual mode: The bit sets the period for phase V only.

14.7.5 MPWMn_PRDW: MPWM n Phase W Period Register

MPWM0_PRDW=0x4000_4098, MPWM1_PRDW=0x4000_5098

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
Reserved		PERIOD_W[15:0]	
-		0x0002	
-		RW	

15 0	PERIOD_W[15:0]	16-bit PWM period of W phase The bit value must be larger than 0x0010. - Motor mode: Not used. - Individual mode: The bit sets the period for phase W only.

14.7.6 MPWMn_CNTU: MPWM n Phase U Counter Register

MPWM0_CNTU=0x4000_40A0, MPWM1_CNTU=0x4000_50A0

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
Reserved		CNT[15:0]	
-		0x0000	
-		R0	

15 0	CNT[15:0]	16-bit PWM counter value of U phase - Motor mode: Not used. - Individual mode: The value of the phase U counter.

14.7.1 MPWMn_CNTV: MPWM n Phase V Counter Register

MPWM0_CNTV=0x4000_40A4, MPWM1_CNTV=0x4000_50A4

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
Reserved		CNT[15:0]	
-		0x0000	
-		R0	

15 0	CNT[15:0]	16-bit PWM couter value of V phase - Motor mode: Not used. - Individual mode: The value of the phase V counter.

14.7.2 MPWMn_CNTW: MPWM n phase W counter register

MPWM0_CNTW=0x4000_40A8, MPWM1_CNTW=0x4000_50A8

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
Reserved		CNT[15:0]	
-		0x0000	
-		R0	

15 0	CNT[15:0]	16-bit PWM couter value of W phase - Motor mode: Not used. - Individual mode: The value of the phase W counter.

14.7.3 MPWMn_DTRU: MPWM n Phase U Dead-Time Register

Settings in the MPWMn_DTRU register are applied to phases U, V, and W simultaneously in Motor PWM mode, and only to phase U in Individual PWM mode.

MPWM0_DTRU=0x4000_40B0, MPWM1_DTRU=0x4000_50B0

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
DTMDSEL	Reserved	UDTEN	UPSHRT	Reserved	UDTCLK[1:0]	Reserved	ULDT[7:0]
0	-	0	0		00	-	0x00
RW	-	RW	RW		RW	-	RW

31	DTMDSEL	Dead-time mode selection When the POL function is used, a phase's H and L signals can overlap at the high level. This bit is typically set to prevent such a phenomenon. - Motor mode: Not used. - Individual mode: Select the dead-time for phase U.	
		0	Inserts dead-time at the leading edge of PWMxH and the trailing edge of PWMxL.
		1	Inserts dead-time at the trailing edge of PWMxH and the leading edge of PWMxL.
23	UDTEN	Enable the dead-time function. User must disable this function in two-channel symmetric mode because this mode does not support the function. - Motor mode: Not used. - Individual mode: Enable the dead-time function for phase U.	
		0	Disables the dead-time function.
		1	Enables the dead-time function.
22	UPSHRT	Short-circuit protection This function is supported in two-channel symmetric mode. In one-channel mode, both the H- and L-sides are not simultaneously active. When either side is active, the other side always inactive. - Motor mode: Not used. - Individual mode: Enable short-circuit protection for phase U.	
		0	Enables short-circuit protection. (When the H- and L-side outputs are active simultaneously, they are all turned off.)
		1	Disables short-circuit protection.
17 16	UDTCLK[1:0]	Dead-time prescaler - Motor mode: Not used. - Individual mode: Set the dead-time counter for phase U.	
		00	Sets PWM CLK / 2 as the dead-time counter clock.
		01	Sets PWM CLK / 4 as the dead-time counter clock.
		10	Sets PWM CLK / 8 as the dead-time counter clock.
		11	Sets PWM CLK / 16 as the dead-time counter clock.
7 0	ULDT[7:0]	Falling dead-time value (set as the delay time from the normal polarity to the falling level output.) - Motor mode: Not used. - Individual mode: Set the falling dead-time value for phase U.	
		0x00	Disable the dead-time
		0x01 to 0xFF	Apply the falling dead-time value

NOTE:

1. This short-circuit protection is applied to the PWM's internal signals and not to its external signals. When the internal

signals from the H- and L-sides are all in the high level, short-circuit protection induces both sides to generate low-level outputs. In Individual mode, setting only either one of UHDT and ULDT to zero can cause an abnormal operation. Therefore, both bit fields must all be set either to zero or to values other than zero.

14.7.4 MPWMn_DTRV: MPWM n Phase V Dead-Time Register

Settings in the MPWMn_DTRV register are not used in Motor PWM mode and applied only to phase V in Individual PWM mode.

MPWM0_DTRV=0x4000_40B4, MPWM1_DTRV=0x4000_50B4

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
DTMDSEL	Reserved	VDTEN	VPSHRT	Reserved	VDTCLK[1:0]	Reserved	VLDT[7:0]
0	-	0	0		00	-	0x00
RW	-	RW	RW		RW	-	RW

31	DTMDSEL	Dead-time mode selection When the POL function is used, a phase's H- and L-signals can overlap in the high level. This bit is typically set to prevent such a phenomenon. - Motor mode: Not used. - Individual mode: Selects the dead-time for phase V.	
		0	Insert dead-time at the leading edge of PWMxH and the trailing edge of PWMxL.
		1	Insert dead-time at the trailing edge of PWMxH and the leading edge of PWMxL.
23	VDTEN	Enable the dead-time function. You must disable this function in two-channel symmetric mode because this mode does not support the function. - Motor mode: Not used. - Individual mode: Enable the dead-time function for phase V.	
		0	Disable the dead-time function.
		1	Enable the dead-time function.
22	VPSHRT	Short-circuit protection This function is supported in two-channel symmetric mode. In one-channel mode, both the H and L sides are not simultaneously active. When either side is active, the other side is always inactive. - Motor mode: Not used. - Individual mode: Enable short-circuit protection for phase V.	
		0	Enable short-circuit protection. (When the H and L side outputs are active simultaneously, they are all turned off.)
		1	Disable short-circuit protection.
17 16	VDTCLK[1:0]	Dead-time prescaler - Motor mode: Not used. - Individual mode: The bit sets the dead-time counter for phase V.	
		00	Sets PWM CLK / 2 as the dead-time counter clock.
		01	Sets PWM CLK / 4 as the dead-time counter clock.
		10	Sets PWM CLK / 8 as the dead-time counter clock.
		11	Sets PWM CLK / 16 as the dead-time counter clock.
7 0	VLDT[7:0]	Falling dead-time value (set as the delay time from the normal polarity to the falling level output) - Motor mode: Not used. - Individual mode: Set the falling dead-time value for phase V.	
		0x00	Disable the dead-time

0x01 to 0xFF	Apply the falling dead-time value

NOTE:

1. This short-circuit protection is applied to the PWM's internal signals and not to its external signals. When the internal signals from the H- and L-sides are all at the high level, short-circuit protection induces both sides to generate low-level outputs. In Individual mode, setting only either one of VHDT and VLDT to zero can cause an abnormal operation. Therefore, both bit fields must all be set either to zero or to values other than zero.

14.7.5 MPWMn_DTRW: MPWM n Phase W Dead-Time Register

Settings in the MPWMn_DTRW register are not used in Motor PWM mode and applied only to phase W in Individual PWM mode.

MPWM0_DTRW=0x4000_40B8, MPWM1_DTRW=0x4000_50B8

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
WTMDSEL	Reserved	WDTEN	WPSHRT	Reserved	WDTCLK[1:0]	Reserved	WLDT[7:0]
0	-	0	0		00	-	0x00
RW	-	RW	RW		RW	-	RW

31	DTMDSEL	Dead-time mode selection When the POL function is used, a phase's H- and L-signals can overlap in the high level. This bit is typically set to prevent such a phenomenon. - Motor mode: Not used. - Individual mode: The bit selects the dead-time for phase W.
	0	Insert dead-time at the leading edge of PWMxH and the trailing edge of PWMxL.
	1	Insert dead-time at the trailing edge of PWMxH and the leading edge of PWMxL.
23	WDTEN	Enable the dead-time function You must disable this function in two-channel symmetric mode because this mode does not support the function. - Motor mode: Not used. - Individual mode: Enables the dead-time function for phase W.
	0	Disable the dead-time function.
	1	Enable the dead-time function.
22	WPSHRT	Short-circuit protection This function is supported in two-channel symmetric mode. In one-channel mode, both the H- and L-sides are not simultaneously active. When either side is active, the other side is always inactive. - Motor mode: Not used. - Individual mode: Enables short-circuit protection for phase W.
	0	Enable short-circuit protection. (When the H- and L-side outputs are active simultaneously, they are all turned off.)
	1	Disable short-circuit protection.
17 16	WDTCLK[1:0]	Dead-time prescaler - Motor mode: Not used. - Individual mode: Set the dead-time counter for phase W.
	00	Sets PWM CLK / 2 as the dead-time counter clock.
	01	Sets PWM CLK / 4 as the dead-time counter clock.
	10	Sets PWM CLK / 8 as the dead-time counter clock.
	11	Sets PWM CLK / 16 as the dead-time counter clock.
7	WLDT[7:0]	Falling dead-time value (set as the delay time from the normal

0		polarity to the falling level output) - Motor mode: Not used. - Individual mode: Set the falling dead-time value for phase W.
	0x00	Disable the dead-time
	0x01 to 0xFF	Apply the falling dead-time value

NOTE:

1. This short-circuit protection is applied to the PWM's internal signals and not to its external signals. When the internal signals from the H- and L-sides are all at the high level, short-circuit protection induces both sides to generate low-level outputs. In Individual mode, setting only either one of WHDT and WLDT to zero can cause an abnormal operation. Therefore, both bit fields must all be set either to zero or to values other than zero.

14.7.6 MPWMn_CAPCNTx: MPWM n Phase U/V/W Capture Counter Register

The MPWMn_CAPCNTx register is a 17-bit capture counter value made with the main counter value. In Individual PWM mode, phases U, V, and W have different capture counters. The register can only be used in Individual PWM mode and not in Motor PWM mode.

MPWM0_CAPCNTU=0x4000_40C0, MPWM0_CAPCNTV=0x4000_40C4, MPWM0_CAPCNTW=0x4000_40C8, MPWM1_CAPCNTU=0x4000_50C0, MPWM1_CAPCNTV=0x4000_50C4, MPWM1_CAPCNTW=0x4000_50C8

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
CNTCLEAR	Reserved	CAPEN	Reserved		CAPCNTx[16:0]	
0	-	0	-		0x00001	
WO	-	RW	-		RO	

31	CNTCLEAR	Clear the capture counter (auto-clear) - Motor mode: Not used. - Individual mode: Clear the capture counter value.
		0 No effect.
		1 Clear the capture counter value.
27	CAPEN	Enable the capture function. - Motor mode: Not used. - Individual mode: Enable the capture function.
		0 Disable the capture function.
		1 Enable the capture function.
16 0	CAPCNTx[16:0] (x = U, V and W)	Enable the capture function. - Motor mode: Not used. - Individual mode: the capture counter value for the corresponding phase.

NOTE:

1. The capture counter starts by enabling the main counter start bit. When the main counter is counting up, the counter value is the same as the main counter, but when the main counter is counting down, the one bit above the uppermost bit of the period value becomes '1', indicating counting down.

14.7.7 MPWMn_RCAPx: MPWM n Phase U/V/W Capture Rising Value Register

The MPWMn_RCAPx register is copied from the capture counter value on the rising edge of the external signal. It can be used in Individual PWM mode but not in Motor PWM mode.

MPWM0_RCAPU=0x4000_40D0, MPWM0_RCAPV=0x4000_40D4, MPWM0_RCAPW=0x4000_40D8
MPWM1_RCAPU=0x4000_50D0, MPWM1_RCAPV=0x4000_50D4, MPWM1_RCAPW=0x4000_50D8

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
RCAPFLAG	Reserved															RCAPx[16:0]															
0	-															0x00000															
WO	-															RO															

31	RCAPFLAG	Capture counter flag (auto-clear) - Motor mode: Not used. - Individual mode: Writing to the bit clears the capture value.
		0 No capture value has been imported.
		1 Capture value has been received (writing '1' to the bit clears RCAPx and the flag).
16 0	RCAPx[16:0] (x = U, V and W)	Capture counter value when the external signal is rising. - Motor mode: Not used. - Individual mode: capture counter value for the corresponding phase.

14.7.8 MPWMn_FCAPx: MPWM n Phase U/V/W Capture Falling Value Register

The MPWMn_FCAPx register is copied from the capture counter value on the falling edge of the external signal is falling. It can be used in Individual PWM mode but not in Motor PWM mode.

MPWM0_FCAPU=0x4000_40E0, MPWM0_FCAPV=0x4000_40E4, MPWM0_FCAPW=0x4000_40E8
MPWM1_FCAPU=0x4000_50E0, MPWM1_FCAPV=0x4000_50E4, MPWM1_FCAPW=0x4000_50E8

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
FCAPFLAG	Reserved															FCAPx[16:0]															
0	-															0x00000															
WO	-															RO															

31	FCAPFLAG	Capture flag (auto-clear) - Motor mode: Not used. - Individual mode: writing to the bit clears the capture value.
		0 No capture value has been imported.
		1 The capture value has been received (writing '1' to the bit clears FCAPx and the flag).
16 0	FCAPx[16:0] (x = U, V and W)	Capture value when the external signal is falling. - Motor mode: Not used. - Individual mode: the capture value for the corresponding phase.

14.7.9 MPWMn_SCAPx: MPWM n Phase U/V/W Sub Capture Value Register

The MPWMn_SCAPx register is copied from the capture counter value on the rising or falling edge of the external signal. It can be used in Individual PWM mode but not in Motor PWM mode.

MPWM0_SCAPU=0x4000_40F0, MPWM0_SCAPV=0x4000_40F4, MPWM0_SCAPW=0x4000_40F8
MPWM1_SCAPU=0x4000_50F0, MPWM1_SCAPV=0x4000_50F4, MPWM1_SCAPW=0x4000_50F8

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
SCAPFLAG	Reserved		EDGESEL	Reserved												SCAPx[16:0]															
0	-		0	-												0x00000															
RW	-		RW	-												RO															

31	SCAPFLAG	Sub capture flag (auto-clear) - Motor mode: Not used. - Individual mode: Writing to the bit clears the capture value.
		0 No capture value has been imported.
		1 The capture value has been received (writing a '1' to the bit clears FCAPx and the flag).
28	EDGESEL	Select the edge of the external signal to capture the counter value. - Motor mode: Not used. - Individual mode: select the edge type for capturing counter.
		0 Captured at the rising edge.
		1 Captured at the falling edge.
16 0	SCAPx[16:0] (x = U, V and W)	Capture counter value when the external signal is rising or falling. - Motor mode: Not used. - Individual mode: the capture value by external signal.

14.7.10 MPWM Register Map Summary: Individual PWM Mode

Table 133. MPWM Register Map Summary: Individual PWM Mode

Offset	Register name	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
0x80	MPWMn_CR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WIRQN[2:0]			WHALT	Res.	WSTART	WEN	Res.	VIRQN[2:0]			VHALT	Res.	VSTART	VEN	Res.	UIRQN[2:0]			UHALT	Res.	USTART	UEN
	Reset value										0	0	0	0		0	0		0	0	0	0		0	0		0	0	0	0		0	0
0x84	MPWMn_CR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WCONTI	Res.	WSTOP	WDIS	Res.	Res.	Res.	Res.	VCONTI	Res.	VSTOP	VDIS	Res.	Res.	Res.	Res.	UCONTI	Res.	USTOP	UDIS
	Reset value													0		0	0					0		0	0					0		0	0
0x90	MPWMn_PRDU	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD_U[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0x94	MPWMn_PRDV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD_V[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0x98	MPWMn_PRDW	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD_W[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0xA0	MPWMn_CNTU	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA4	MPWMn_CNTV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA8	MPWMn_CNTW	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xB0	MPWMn_DTRU	DTMDSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDTEN	UPSHRT	Res.	Res.	Res.	Res.	UDTCLK[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ULDT[7:0]							
	Reset value	0								0	0					0	0									0	0	0	0	0	0	0	0
0xB4	MPWMn_DTRV	DTMDSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VDTEN	VPSHRT	Res.	Res.	Res.	Res.	VDTCLK[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VLDT[7:0]							
	Reset value	0								0	0					0	0									0	0	0	0	0	0	0	0

Table 122. MPWM Register Map Summary: Individual PWM Mode (continued)

Offset	Register name	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
0xB8	MPWMn_DTRW	DTMDSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDTEN	WPSHRT	Res.	Res.	Res.	Res.	WDTCLK[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WLDT[7:0]							
	Reset value	0								0	0					0	0									0	0	0	0	0	0	0	0
0xC0	MPWMn_CAPCNTU	CNTCLEAR	Res.	Res.	Res.	CAPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPCNTU[15:0]															
	Reset value	0				0												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0xC4	MPWMn_CAPCNTV	CNTCLEAR	Res.	Res.	Res.	CAPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPCNTV[15:0]															
	Reset value	0				0												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0xC8	MPWMn_CAPCNTW	CNTCLEAR	Res.	Res.	Res.	CAPEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPCNTW[15:0]															
	Reset value	0				0												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0xD0	MPWMn_RCAPU	RCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCAPU[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xD4	MPWMn_RCAPV	RCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCAPV[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xD8	MPWMn_RCAPW	RCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCAPW[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xE0	MPWMn_FCAPU	FCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FCAPU[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xE4	MPWMn_FCAPV	FCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FCAPV[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xE8	MPWMn_FCAPW	FCAPFLAG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FCAPW[15:0]															
	Reset value	0																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 122. MPWM Register Map Summary: Individual PWM Mode (continued)

Offset	Register name	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
0xF0	MPWMn_SCAPU	SCAPFLAG	Res	Res	EDGESEL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SCAPU[15:0]															
	Reset value	0			0													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xF4	MPWMn_SCAPV	SCAPFLAG	Res	Res	EDGESEL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SCAPU[15:0]															
	Reset value	0			0													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xF8	MPWMn_SCAPW	SCAPFLAG	Res	Res	EDGESEL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SCAPU[15:0]															
	Reset value	0			0													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15. Analog-to-Digital Converter (ADC)

15.1 ADC Introduction

The ADC module in the A34M420 series consists of three independent 12-bit resolution Analog-to-Digital Converters (ADCs) with configurable up to 24 analog input channels.

This ADC module can process A/D conversion of many different channels in single mode, sequential mode, multiple mode and burst mode. It can sample the voltage inputs of multiple channels at critical times, which is required for various applications and can be achieved using the triggering functionality in association with TIMER and MPWM.

Once the A/D conversion is complete, the result of the ADC is stored in a left-aligned 16-bit data register.

The ADC is mapped to the APB and capable of high-speed operation based on the internal clock source. In addition, it uses built-in DMA to reduce microcontroller core load while enabling fast data sampling.

The analog input monitoring and comparison capabilities of the ADC allow the applications to determine whether the input voltage is above or below the predefined level.

15.2 ADC Main Features

The key features of the ADC module are listed below:

- Composed of three independent ADC units
- 12-bit analog to digital resolution
- 1.5 Msps maximum conversion rate with full resolution
- Three independently implemented ADCs
- 2.7 V minimum to 5.5 V operating voltage range
- Data can be managed by DMA for high-performance conversion.
- Analog Input channels
  - ADC0 is connected to 16 external channels, two internal channels for V_{CORE}, V_{REFINT}
  - ADC1 is connected to 16 external channels, two internal channels for V_{CORE}, V_{REFINT}
  - ADC2 is connected to 16 external channels, two internal channels for V_{CORE}, V_{REFINT}
- Internally dedicated channels
  - V_{REFINT}, V_{CORE} internal reference voltage outputs connected to ADC0, 1 and 2.
- Flexible sampling time control
- Four Conversion modes
  - Single mode
  - Sequential mode
  - Multiple mode
  - Burst mode
- Up to eight sequential conversions can be supported.
- Internal reference outputs (V_{REFINT} and V_{CORE}) for voltage monitoring

- Software trigger sources per channel
  - Internal software trigger source
  - TIMERn (n = 0, 1, 2, …, 9) trigger sources
  - MPWMn (n = 0, 1) trigger sources
- Interrupt generation at the end of conversion in the last sequence, the trigger conversion, the DMA done and the comparison interrupt.
- ADC input range
  - $\mathrm{AGND} \leq \mathrm{ANx\,(x = 0\ to\ 23)} \leq \mathrm{AVDD}\ \{(\mathrm{VDD} - 0.3) \leq \mathrm{AVDD} \leq (\mathrm{VDD} + 0.3)\}$

NOTE:

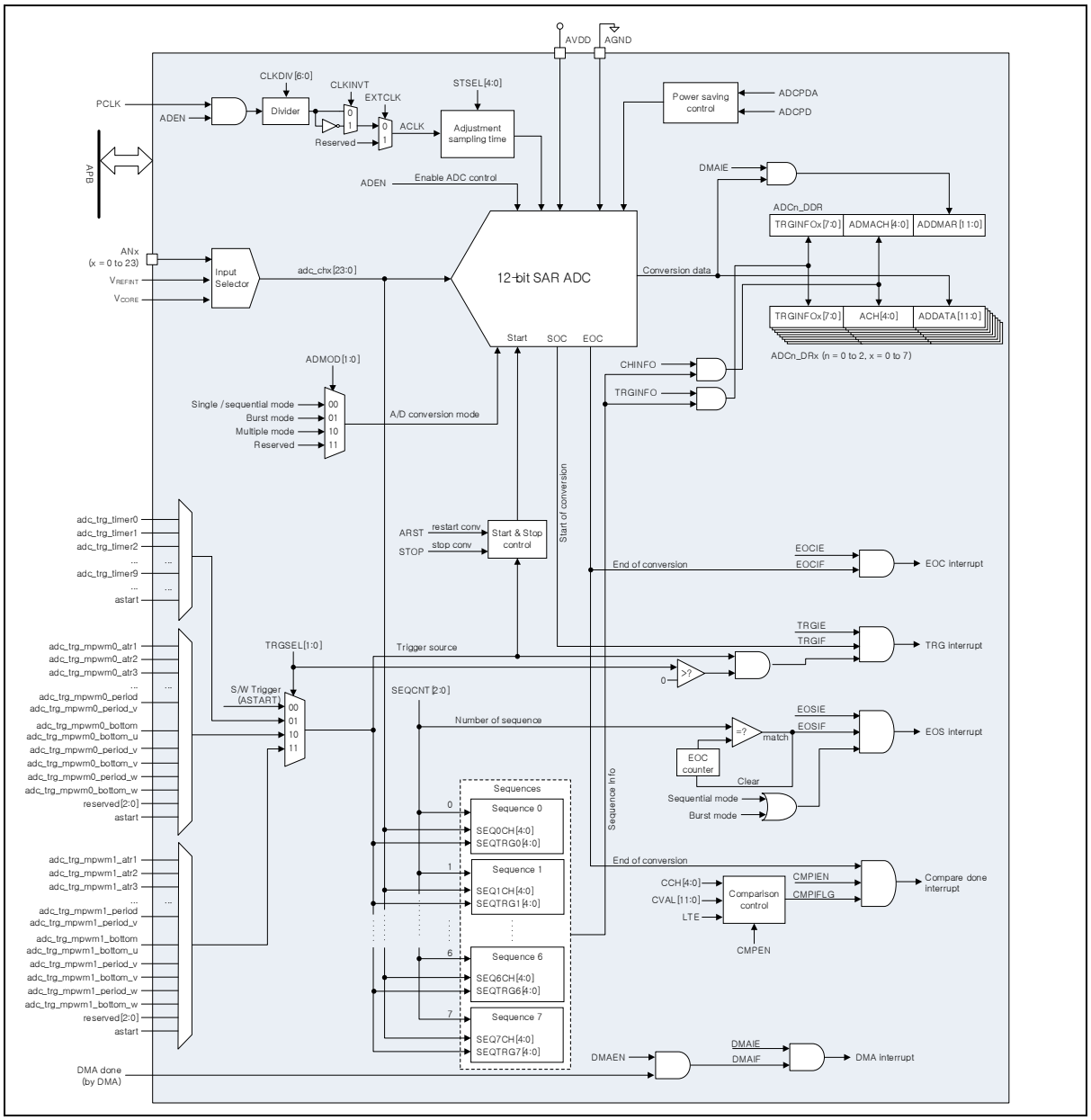
1. ADC performance such as INL or DNL may be degraded if other analog modules use the AVDD reference voltage. Therefore, in applications where the precision of the ADC is important, it is recommended that the software be configured to use only the ADC analog module.

15.3 ADC Functional Description

15.3.1 ADC Block Diagram

Figure 161 shows a block diagram of the 12-bit ADC.

Figure 161. 12-bit ADC Block Diagram



15.3.2 ADC Pins and Signals

Table 134 describes pins assigned for the ADC.

Table 134. Pin Assignment of ADC External Pins

Pin Name	Signal Type[1]	Description	Supported Packages		
			A34M420YL (120-LQFP)	A34M420VL (100-LQFP)	A34M420RL (64-LQFP)
AVDD	P	Analog power (reference voltage)	O	O	O
AGND	P	Analog GND	O	O	O
AN0	A	ADC external analog input 0	O	O	O
AN1	A	ADC external analog input 1	O	O	O
AN2	A	ADC external analog input 2	O	O	O
AN3	A	ADC external analog input 3	O	O	O
AN4	A	ADC external analog input 4	O	O	O
AN5	A	ADC external analog input 5	O	O	O
AN6	A	ADC external analog input 6	O	O	O
AN7	A	ADC external analog input 7	O	O	O
AN8	A	ADC external analog input 8	O	O	O
AN9	A	ADC external analog input 9	O	O	O
AN10	A	ADC external analog input 10	O	O	O
AN11	A	ADC external analog input 11	O	O	O
AN12	A	ADC external analog input 12	O	O	O
AN13	A	ADC external analog input 13	O	O	O
AN14	A	ADC external analog input 14	O	O	O
AN15	A	ADC external analog input 15	O	O	O
AN16	A	ADC external analog input 16	O	O	N/A
AN17	A	ADC external analog input 17	O	O	N/A
AN18	A	ADC external analog input 18	O	O	N/A
AN19	A	ADC external analog input 19	O	O	N/A
AN20	A	ADC external analog input 20	O	O	N/A
AN21	A	ADC external analog input 21	O	O	N/A
AN22	A	ADC external analog input 22	O	O	N/A
AN23	A	ADC external analog input 23	O	O	N/A

NOTE:

1. P: Power, O: Output, A: Analog Input.

Table 135 describes the input and output signals of the ADC.

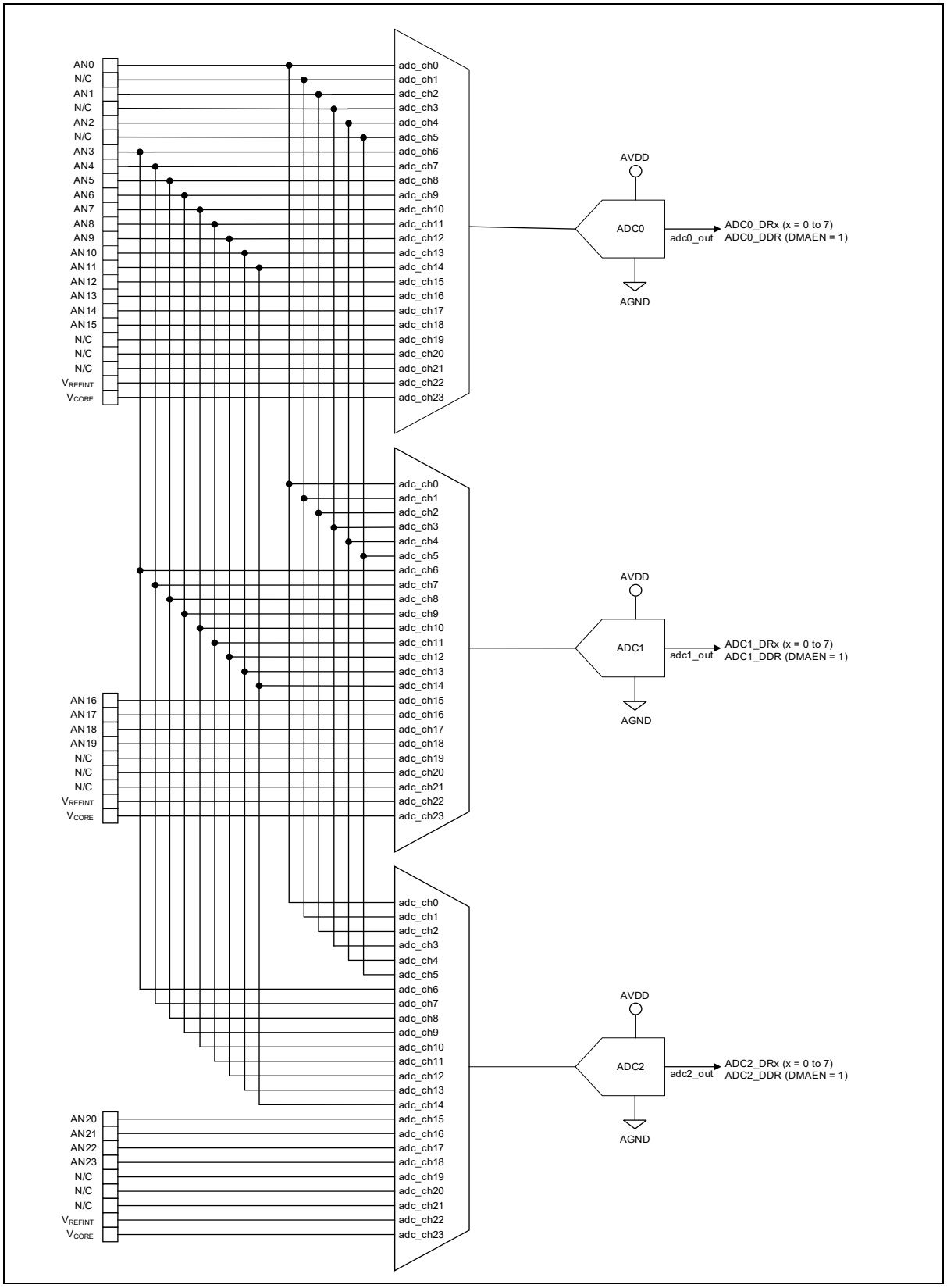
Table 135. ADC Input/Output Signals

Signal Name	Signal Type	Description
ACLK	clock	ADC operating clock
adc_trg_timerx (x = 0 to 9)	Inputs	The internal trigger source of TIMERn (n = 0 to 9)
adc_trg_mpwmx (x = 0 to 11)	Inputs	The internal trigger source of MPWMn (n = 0 to 1)
adc_trg_astart	Inputs	The internal clock of ADC
VREFINT	Input	Output voltage from internal reference voltage 1.0 V
VCORE	Input	Output voltage from internal reference voltage 1.5 V
adc_ch[23:0]	Analog Input	ADC external input signal
adcx_out (x = 0 to 2)	Digital Output	AD Conversion data by ADCn (n = 0 to 2)

15.3.3 ADC Connectivity

15.3.3.1 ADC Internal Channel Wiring

Figure 162. 12-bit ADC Internal Channel Wiring



15.3.3.2 ADC Input Signals and External Input Channels

Table 136. ADC Input Signals and Input Pins(1)

ADC Input Signal	SEQxCH (x = 0 to 7)	ADC0	ADC1	ADC2
adc_ch0	0	AN0	AN0	AN0
adc_ch1	1	-	-	-
adc_ch2	2	AN1	AN1	AN1
adc_ch3	3	-	-	N/C
adc_ch4	4	AN2	AN2	AN2
adc_ch5	5	-	-	-
adc_ch6	6	AN3	AN3	AN3
adc_ch7	7	AN4	AN4	AN4
adc_ch8	8	AN5	AN5	AN5
adc_ch9	9	AN6	AN6	AN6
adc_ch10	10	AN7	AN7	AN7
adc_ch11	11	AN8	AN8	AN8
adc_ch12	12	AN9	AN9	AN9
adc_ch13	13	AN10	AN10	AN10
adc_ch14	14	AN11	AN11	AN11
adc_ch15	15	AN12	AN16	AN20
adc_ch16	16	AN13	AN17	AN21
adc_ch17	17	AN14	AN18	AN22
adc_ch18	18	AN15	AN19	AN23
adc_ch19	19	-	-	-
adc_ch20	20	-	-	-
adc_ch21	21	-	-	-
adc_ch22	22	V_{REFINT}	V_{REFINT}	V_{REFINT}
adc_ch23	23	V_{CORE}	V_{CORE}	V_{CORE}

NOTES:

1. The SEQxCH (x = 0 to 7) are configured in the ADC_SCSR1 and ADC_SCSR2 registers (n = 0 to 2).
2. Single mode conversion uses the SEQ0CH in ADC_SCSR1 register and the ADC_DR0 register only (n = 0 to 2).

15.3.4 ADC Clocks (ACLK)

15.3.4.1 ADC Module Control

Before enabling the ADC clock, users must select an ADC unit to be enabled by setting a bit of the ADC[2:0] in the SCU_PER2 register to '1' which corresponds to the ADC unit. Then, enable the clock of the selected ADC unit by setting a bit of the ADC[2:0] in the SCU_PCER2 register to '1' which corresponds to the clock of the selected ADC unit.

To initialize the registers of each ADC unit, set each corresponding bit of the ADC[2:0] in the SCU_PER2 register and the ADC[2:0] bits in the SCU_PCER2 register to '0' and then set back to '1'.

Each bit of the ADC[2:0] bits in the SCU_PER2 register controls the corresponding ADC unit. To enable or disable all ADC units at a time, set the ADC[2:0] in the SCU_PCER2 register to the number shown below:

- If the ADC[2:0] bits in the SCU_PER2 register = '000', all ADC units are disabled.
- If the ADC[2:0] bits in the SCU_PER2 register = '111', all ADC units are enabled.

To provide the ADC unit with the ADC clock, configure the ADC[2:0] bits in the SCU_PCER2 register referring the settings shown below:

- If the ADC[2:0] bits in the SCU_PCER2 register = '000', all ADC clocks are disabled.
- If the ADC[2:0] bits in the SCU_PCER2 register = '111', all ADC clocks are enabled.

15.3.4.2 ADC Clock Sources

The PCLK operates as the ADC operation clock (ACLK) based on the APB domain. Internal clock source can be selected to operate as the ACLK by setting the EXTCLK bit in the ADCn_CCR register. Each clock source for the three ADC units can be configured independently.

- If the EXTCLK = '0', internal clock source (PCLK) is selected.
- If the EXTCLK = '1', Reserved. (Not effected)

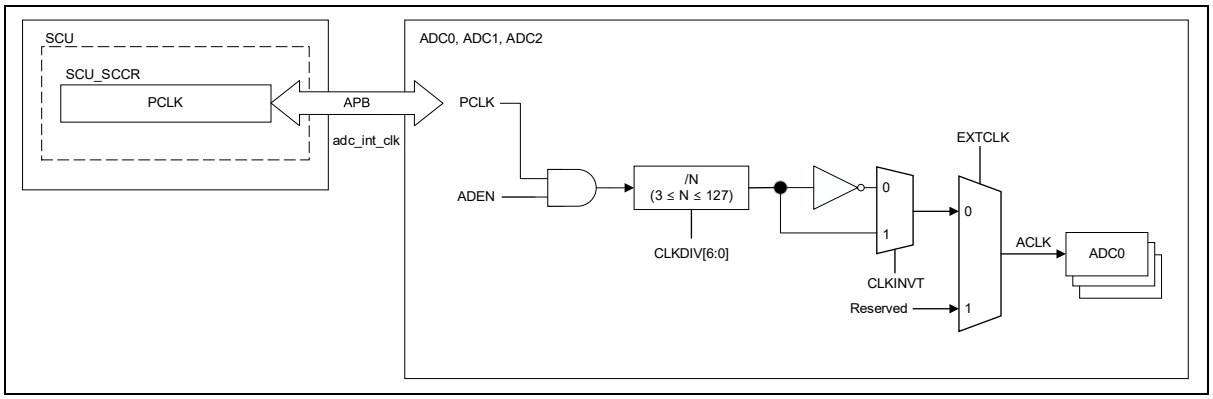
Internal Clock Source

The programmable divider coefficient can be selected to range from 3 to 127 by configuring the CLKDIV[6:0] bits in the ADCn_CCR register, and the PCLK is divided by the coefficient to be used as the ADC operation clock.

NOTE:

1. Main system clock must be divided at least 3 times to be used as an ADC operation clock.
2. The CLKDIV[6:0] bit field in the ADCn_CCR register is used only when the PCLK, the ADC's internal system clock is used.

Figure 163. Block diagram of ADC clock



15.3.5 ADC On-off Control (ADEN)

Before starting the A/D conversion, users must enable the ADC module using the ADEN bit in the ADCn_MR register.

- Setting the ADEN = '1' enables the ADC unit. When the ADC unit is enabled, the ADC functionality and the ASTART bit control are possible.
- Setting the ADEN = '0' disables the ADC unit. When the ADC unit is disabled, the A/D conversion cannot be performed.

15.3.5.1 Software Procedure to Enable the ADC

1. Clear the EOCIF bit in the ADCn_SR register by writing '1'.
2. Setting the ADEN bit in the ADCn_MR register to '1' enables the ADC module.

15.3.5.2 Software Procedure to Disable the ADC

1. Confirm that the ASTART bit and ASTOP bit in the ADC_CR register are set to '0', respectively. This ensures that no conversion is ongoing. If required, set the ASTOP bit to '1' to stop the conversion and then wait until the ASTOP bit becomes '0'[(1)].
2. Disable the ADC module by setting the ADEN bit in the ADCn_MR register to '0'.

NOTE:

1. This ASTOP bit is auto cleared at the next ADC clock cycle.

15.3.6 Constraints when Writing the ADC Control Bits

15.3.6.1 When Writing ADC Control Bit

While accessing the SCU_PER2, SCU_PCER2, and SCU_MCCR4 registers to enable the ADC operation clock source and ADC module, the ADC module must be disabled. For this, set the ADEN bit in the ADCn_MR register to '0'.

During the A/D conversion operations, if users write to each ADC control bit of the SCU_PER2, SCU_PCER2, and SCU_MCCR4 registers, the ADC clock bus and ADC registers are not controllable and become an Unknown state.

Software can write to the control bits configuring the A/D conversion, only when the ADC module is enabled (ADEN = '1') and the A/D conversion is not performed (e.g. ASTART = '0').

- Set the ADEN bit in the ADCn_MR register to '0'.
- Clear all bits in the ADCn_CR register.

15.3.6.2 When Initializing the ADC

The initial state of the End-of-Conversion (EOC) of the ADC is unknown when the microcontroller is first powered on. At this point of time, leakage current may flow if the microcontroller enters SLEEP mode or DEEP-SLEEP mode. To avoid this situation, set the ASTART to '1' and perform two dummy transfers of the ADCn_CR register to clear the EOC.

Users must consider this situation when initializing the ADC, since it cannot be processed by hardware.

15.3.7 Channel Selection

15.3.7.1 ADC Analog Input Channel

Each ADC has up to 24 multiplexed channels.

When the analog inputs come from GPIO pads:

1. For the analog inputs on the ADC channel, set mode of the channel you want to use to ADC using the Pn_MR1 or Pn_MR2 registers (n = A to F).
2. Set the pin direction to 'input' using the Pn_CR register (n = A to F).
3. Disable the internal pull-up or pull-down resistance functionality using the Pn_PCR register (n = A to F).

The ADC is connected to the analog inputs as described below:

- Internal reference voltage 1.0 V (V_{REFINT}) is connected to the ADC0_AN22, ADC1_AN22, and ADC2_AN22.
- Internal reference voltage 1.5 V (V_{CORE}) is connected to the ADC0_AN23, ADC1_AN23, and ADC2_AN23.

NOTE:

1. Since the internal analog inputs (V_{REFINT}, V_{CORE}) use the analog source enabled when the system internal power is applied, additional bit programming for the analog source settings is not necessary.

15.3.7.2 ADC Conversion Channel Selection

To convert the input signal received from the analog channel, users must select a channel to sample by setting the SEQxCH bits in the ADCn_SCSR1 or ADCn_SCSR2 registers. (n = 0 to 2, x = 0 to 7)

For the A/D conversion operation in single mode, use the SEQ0CH bits in the ADCn_SCSR1 register to set the channel.

Table 137. ADC Input Signal and Input Pin(1)

ADC Input Signal	ADC_SCSR1				ADC_SCSR2			
	SEQ0CH[4:0](2)	SEQ1CH[4:0]	SEQ2CH[4:0]	SEQ3CH[4:0]	SEQ4CH[4:0]	SEQ5CH[4:0]	SEQ6CH[4:0]	SEQ7CH[4:0]
adc_ch0	0	0	0	0	0	0	0	0
adc_ch1	1	1	1	1	1	1	1	1
adc_ch2	2	2	2	2	2	2	2	2
adc_ch3	3	3	3	3	3	3	3	3
adc_ch4	4	4	4	4	4	4	4	4
adc_ch5	5	5	5	5	5	5	5	5
adc_ch6	6	6	6	6	6	6	6	6
adc_ch7	7	7	7	7	7	7	7	7
adc_ch8	8	8	8	8	8	8	8	8
adc_ch9	9	9	9	9	9	9	9	9
adc_ch10	10	10	10	10	10	10	10	10
adc_ch11	11	11	11	11	11	11	11	11
adc_ch12	12	12	12	12	12	12	12	12
adc_ch13	13	13	13	13	13	13	13	13
adc_ch14	14	14	14	14	14	14	14	14
adc_ch15	15	15	15	15	15	15	15	15
adc_ch16	16	16	16	16	16	16	16	16
adc_ch17	17	17	17	17	17	17	17	17
adc_ch18	18	18	18	18	18	18	18	18
adc_ch19	19	19	19	19	19	19	19	19
adc_ch20	20	20	20	20	20	20	20	20
adc_ch21	21	21	21	21	21	21	21	21
adc_ch22	22	22	22	22	22	22	22	22
adc_ch23	23	23	23	23	23	23	23	23

NOTE:

1. For the multi-channel sampling, up to 8 channels can be input and the number of conversion is set in the SEQCNT[2:0] in ADCn_MR register.
2. For the single-channel sampling, the channel must be selected by configuring the SEQ0CH[4:0] in ADCn_SCSR1 register. Once the A/D conversion starts with the ASTART in ADCn_CR register set to '1', information of the channel currently converting is stored in the CACH[4:0] in ADCn_CSCR register and the most recently sampled channel value is retained.

15.3.7.3 Register Access when Changing A/D Conversion Channel

The ADCn_SCSR1 or ADCn_SCSR2 registers cannot be modified or changed during the A/D conversion. To change the channel you want to sample, write '1' to the ASTOP bits in the ADCn_CR register to stop the A/D conversion.

Users can select a number of conversion in a sequence by configuring the SEQCNT[2:0] bits in the ADCn_MR register. Updates are available before the Sequence A/D conversion, or immediately after the End-of-Sequence (EOS) before a new sequence is executed.

Assuming that the SEQCNT[2:0] bits is set to '3' and the sequence is configured in the order of adc_ch0, adc_ch1, adc_ch2, and adc_ch3 by using the SEQ0CH, SEQ1CH, SEQ2CH, and SEQ3CH, and the CSEQN[2:0] bits in the ADCn_CSCR register is set to 0x2, then the A/D conversion is performed in the order of adc_ch2 and adc_ch3, and the Sequence End Interrupt Flag, the EOSIF flag in the ADCn_SR register is set to '1'.

15.3.8 Programmable Sampling Time

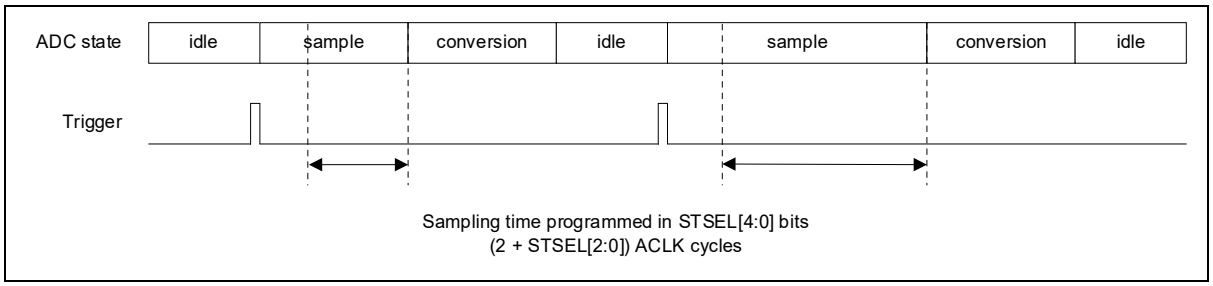
The STSEL[4:0] bits in the ADCn_MR register determine the time window where the next trigger is recognized. This set value is applied right after the occurrence of the current trigger. It is applied equally to all ADC channels and must be set before starting the ADC sampling.

The ADC sampling time is calculated by (2 + STSEL[4:0]) ACLK cycles. The minimum sampling time is two ACLK cycles, and the sampling channel is always active when the STSEL[4:0] bits in the ADCn_MR register are set to '11111'.

- If STSEL[4:0] = '00000': 2 ACLK cycles
- If STSEL[4:0] = '00001': 3 ACLK cycles
- If STSEL[4:0] = '00010': 4 ACLK cycles

  …
- If STSEL[4:0] = '11101': 31 ACLK cycles
- If STSEL[4:0] = '11110': 32 ACLK cycles
- If STSEL[4:0] = '11111': sampling channel is always active

Figure 164. Example of ADC Conversion Timing



15.3.9 Single Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] = 0)

Single mode conversion means that the A/D conversion is performed in a single channel. The single mode performs the sampling and the A/D conversion of analog inputs on the channel selected in the SEQ0CH[4:0] bits in the ADCn_SCSR1 register.

- If ADMOD[1:0] = '00' and SEQCNT[2:0] = '000', A/D conversion in single mode.

There are Four Types of Trigger Sources in Single Mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn (n = 0 to 9) trigger event.
- If TRGSEL[1:0] = '10', MPWM0 trigger event.
- If TRGSEL[1:0] = '11', MPWM1 trigger event.

The SEQCNT[2:0] bits in the ADCn_MR Register in Single Mode:

- If SEQCNT[2:0] = '000', Single channel sampling.

Analog Input Channel in Single Mode:

- If SEQ0CH[4:0] = x (x = 0 to 23), adc_chx channel for A/D conversion.

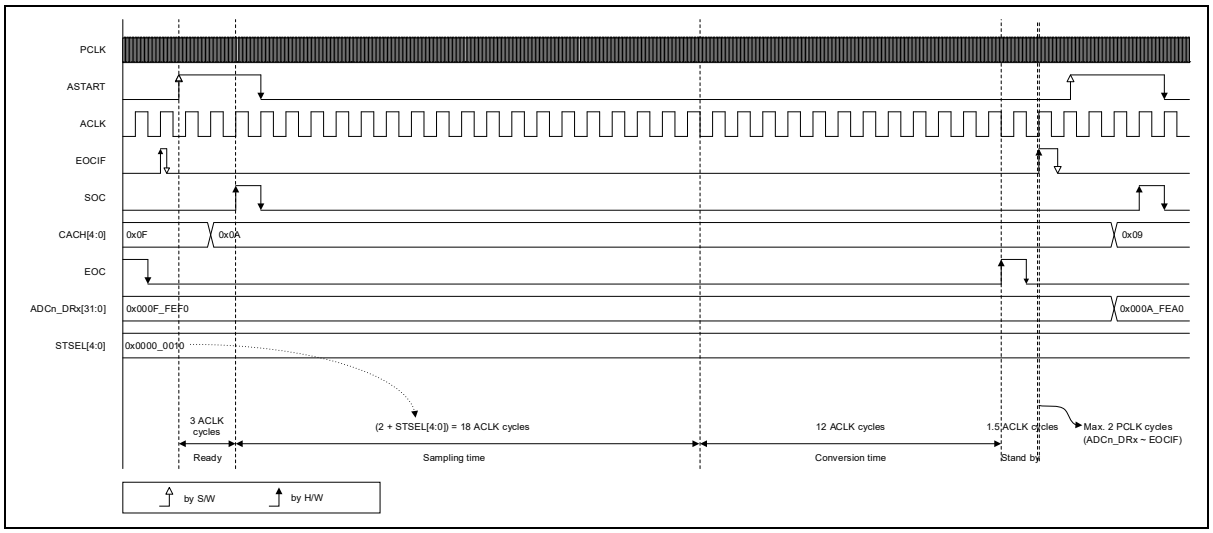
Trigger Event Source Settings:

- If SEQTRG0[3:0] = '0000' to '1111', ADC trigger source.

After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

Figure 165. ADC Single Mode Timing (when ADMOD[1:0] = '00')



Users can obtain the A/D conversion result of the analog input by polling the EOCIF flag in the ADCn_SR register using software or read-accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) at the point of time when the EOCIF flag in the ADCn_SR register interrupt occurs.

15.3.10 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] > 0)

Sequential mode is used when the A/D conversion of the analog inputs are sequentially performed by each trigger source. The A/D conversion of the analog inputs is executed as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register.

- If ADMOD[1:0] = '00' and SEQCNT[2:0] > '000', A/D conversion in sequential mode.

There are four types of trigger sources in sequential mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn (n = 0 to 9) trigger event.
- If TRGSEL[1:0] = '10', MPWM0 trigger event.
- If TRGSEL[1:0] = '11', MPWM1 trigger event.

The SEQCNT[2:0] bits in the ADCn_MR register in sequential mode:

Since the A/D conversion is executed for the multiple channels, the SEQCNT[2:0] bits in the ADCn_MR register must be set to a number other than '0'.

- If SEQCNT[2:0] = '001' to '111', Multi-channel sampling in sequential mode

Analog input channel in sequential mode:

The number of sequence channels that can be configured is set in the SEQCNT[2:0] bit field in the ADCn_MR register.

- If SEQ0CH[4:0] = x (x = 0 to 23), 1st sequence's channel is adc_chx.
- If SEQ1CH[4:0] = x (x = 0 to 23), 2nd sequence's channel is adc_chx.
- If SEQ2CH[4:0] = x (x = 0 to 23), 3rd sequence's channel is adc_chx.
- If SEQ3CH[4:0] = x (x = 0 to 23), 4th sequence's channel is adc_chx.
- If SEQ4CH[4:0] = x (x = 0 to 23), 5th sequence's channel is adc_chx.
- If SEQ5CH[4:0] = x (x = 0 to 23), 6th sequence's channel is adc_chx.
- If SEQ6CH[4:0] = x (x = 0 to 23), 7th sequence's channel is adc_chx.
- If SEQ7CH[4:0] = x (x = 0 to 23), 8th sequence's channel is adc_chx.

Trigger event source settings:

The trigger sources in each sequence that can be configured is set in the SEQTRGx[3:0] bit field in the ADCn_TRG register. In sequence mode, the 1st sequence has the highest priority, and the 8th sequence has the lowest priority.

- If SEQTRG0[3:0] = x, the 1st channel's trigger source is x (x = 0 to 15).
- If SEQTRG1[3:0] = x, the 2nd channel's trigger source is x (x = 0 to 15).
- If SEQTRG2[3:0] = x, the 3rd channel's trigger source is x (x = 0 to 15).
- If SEQTRG3[3:0] = x, the 4th channel's trigger source is x (x = 0 to 15).

- If SEQTRG4[3:0] = x, 5th channel's trigger source is x (x = 0 to 15).
- If SEQTRG5[3:0] = x, 6th channel's trigger source is x (x = 0 to 15).
- If SEQTRG6[3:0] = x, 7th channel's trigger source is x (x = 0 to 15).
- If SEQTRG7[3:0] = x, 8th channel's trigger source is x (x = 0 to 15).

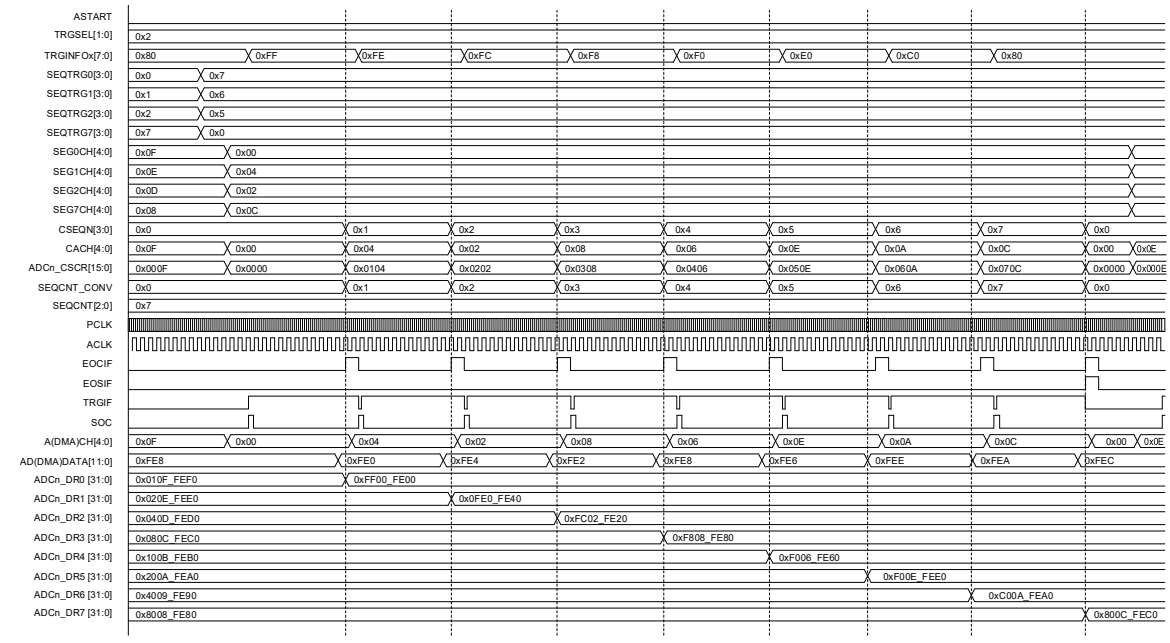
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

The EOCIF flag in the ADCn_SR register is generated as many as the number of sequences set by the SEQCNT[2:0] bits in the ADCn_MR register, and the End-of-Sequence (EOS) flag that indicates the end of the sequence is generated at the time EOC occurs during the last sequence. Users can obtain the digitized data of the analog inputs by polling the EOSIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOSIF interrupt occurs.

Operation in sequential mode is almost identical to those in burst mode. The difference is SOC (Start-of-Conversion) trigger sources. In sequential mode, each sequence's SOC is triggered by the corresponding SEQTRGx[3:0] (x = 0 to 7) trigger, followed by as many conversions as defined by the SEQCNT[2:0] bits in the ADCn_MR register.

Figure 166. ADC Sequential Mode Timing (when ADMOD[1:0] = '00' and SEQCNT[2:0] > 0)



NOTES:

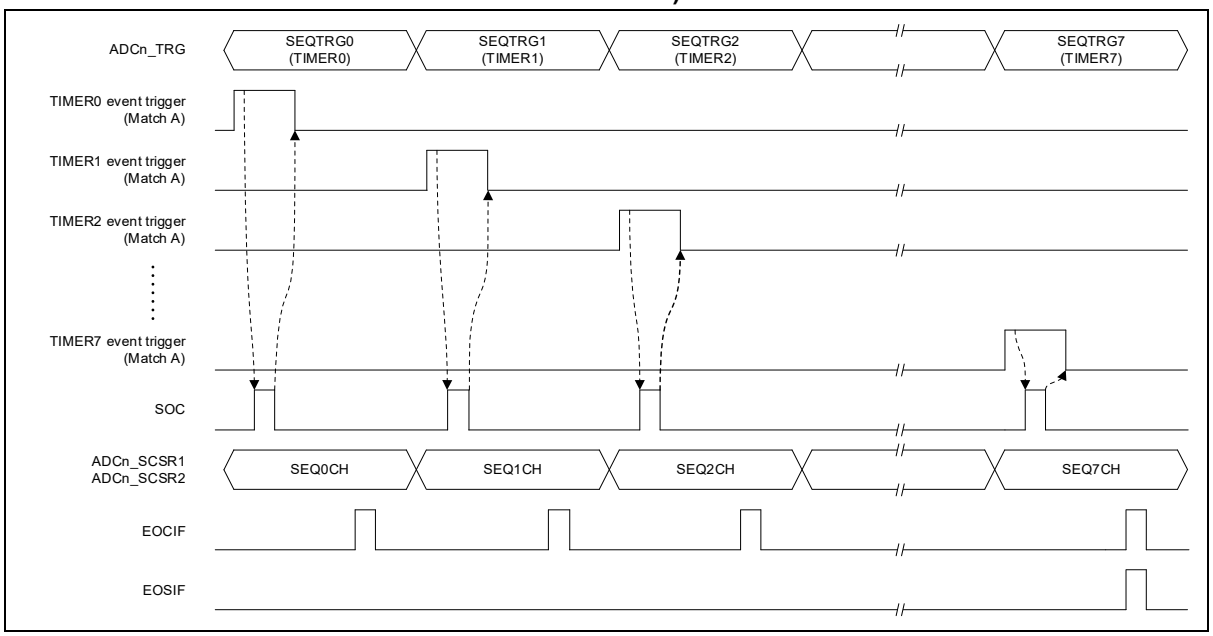
1. PCLK = 140 MHz, ACLK = 10 MHz (CLKDIV = 14).
2. n = 0 to 2, x = 0 to 7.

Figure 167 shows a timing diagram that uses eight sequences in Sequential mode, where each sequence is set to have TIMER trigger source. Based on the sequence' priority, the A/D conversion is

sequentially executed for each trigger event.

When the trigger event of each channel occurs, the internal SOC signal is set to ‘H’ and becomes ‘L’ when the trigger event ends. Then the EOC flag is generated when the A/D conversion of the corresponding analog input signal is complete. The SOC and EOC occur as many times as set in the SEQCNT[2:0] bits, and the sequence end flag EOS is generated when the EOC occurs during the last sequence.

Figure 167. ADC Trigger Timing in Sequential Mode (SEQCNT[2:0] = ‘111’, 8 sequential conversion)



15.3.11 Burst Mode Conversion (ADMOD[1:0] = '01', SEQCNT > 0)

Burst mode is used when the A/D conversion of the analog inputs is performed by one trigger source in the order of channel sequence configuration.

- If ADMOD[1:0] = '01', A/D conversion in burst mode.

There are four types of trigger sources in burst mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn trigger event (n = 0 to 9).
- If TRGSEL[1:0] = '10', MPWM0 trigger event.
- If TRGSEL[1:0] = '11', MPWM1 trigger event.

The SEQCNT[2:0] in ADCn_MR Register in Burst Mode:

Since the A/D conversion is executed for the multi-channels, the SEQCNT[2:0] bits in the ADCn_MR register must be set to a number other than '0' in burst mode.

- If SEQCNT[2:0] = '001' to '111', Multi-channel sampling in burst mode.

Analog Input Channel in Burst Mode:

The number of sequence channels that can be configured is set in the SEQCNT[2:0] bits in the ADCn_MR register. In burst mode, when the trigger event occurs, the A/D conversions are performed in the order starting from SEQ0CH as shown below:

- If SEQ0CH[4:0] = x, 1st sequence's channel is adc_chx (x = 0 to 23).
- If SEQ1CH[4:0] = x, 2nd sequence's channel is adc_chx (x = 0 to 23).
- If SEQ2CH[4:0] = x, 3rd sequence's channel is adc_chx (x = 0 to 23).
- If SEQ3CH[4:0] = x, 4th sequence's channel is adc_chx (x = 0 to 23).
- If SEQ4CH[4:0] = x, 5th sequence's channel is adc_chx (x = 0 to 23).
- If SEQ5CH[4:0] = x, 6th sequence's channel is adc_chx (x = 0 to 23).
- If SEQ6CH[4:0] = x, 7th sequence's channel is adc_chx (x = 0 to 23).
- If SEQ7CH[4:0] = x, 8th sequence's channel is adc_chx (x = 0 to 23).

Trigger Event Source Settings:

In burst mode, the A/D conversion on the SEQxCH[4:0] multi-channel (x = 0 to 7) is triggered by the trigger source event assigned in the BSTTRG[3:0] bits in the ADCn_TRG register.

If the TRGSEL[1:0] bits in the ADCn_MR register is set for the TIMER or MPWM event triggers, SOC (Start of Conversion) signals are triggered by the BSTTRG[3:0] bits in the ADCn_TRG set trigger.

If the BSTTRG[3:0] bits in the ADCn_TRG register is set for the TIMER 3, for example, the ADC conversion gets started by the TIMER 3 trigger. Once an event is triggered by the BSTTRG[3:0] bits set trigger source, the ADC shifts the ADC channels as many times as set in the SEQCNT[2:0] bits in the ADCn_MR.

- If BSTTRG[3:0] = x, Trigger source of burst mode sequence is x (x = 0 to 15).

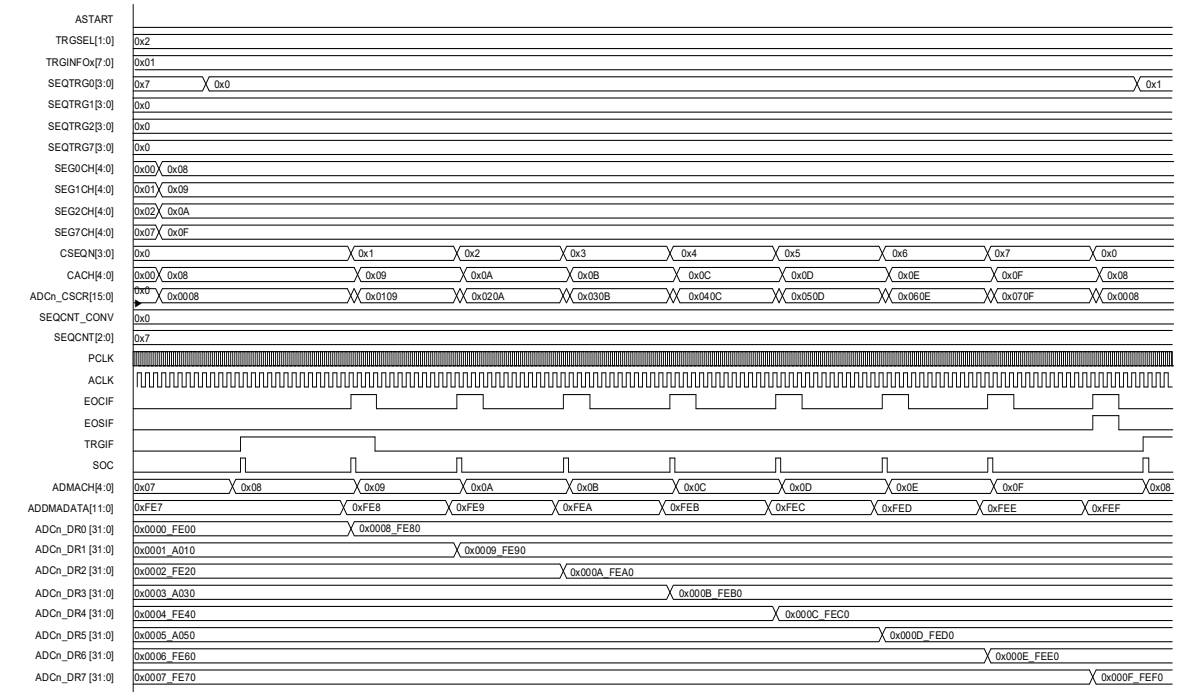
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

The EOCIF flag in the ADCn_SR register is generated as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register, and the End-of-Sequence (EOS) flag that indicates the end of the sequence is generated at the time EOC occurs during the last sequence.

Users can obtain the digitized data of the analog inputs by polling the EOSIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOSIF flag in the ADCn_SR register interrupt occurs.

Figure 168. ADC Burst Mode Timing (when ADMOD[1:0] = '01')



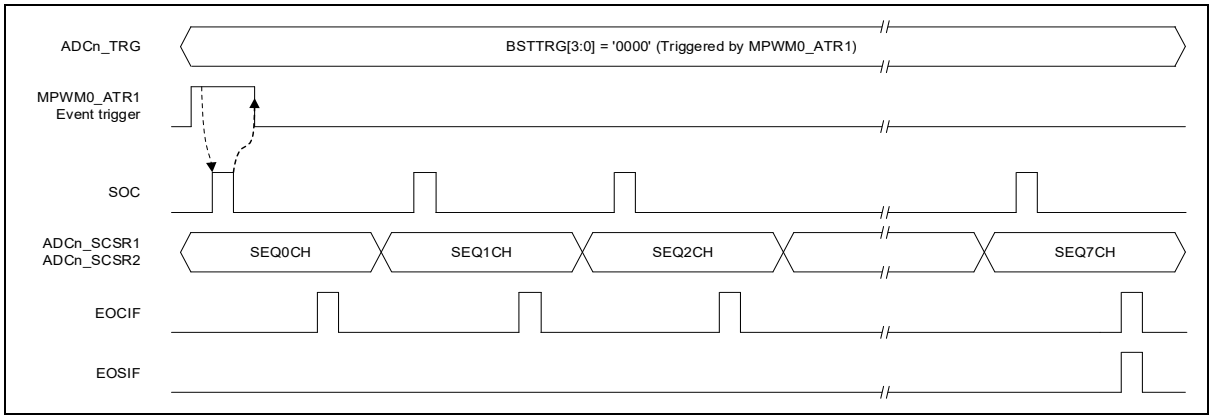
NOTE:

1. PCLK = 140 MHz, ACLK = 10 MHz (CLKDIV = 14)

Figure 169 shows a timing diagram that burst mode continues the A/D conversion for the analog inputs on the 8 channels since the MPWM0_ATR1 trigger event occurs when the BSTTRG[3:0] bits in the ADCn_TRG register is set to MPWM0_ATR1.

When the MPWM0_ATR1 trigger event occurs, the internal SOC signal is set to ‘H’ and becomes ‘L’ if the trigger event ends. Then the EOCIF flag in the ADCn_SR register is generated when the A/D conversion is complete. The SOC and EOCIF flags in the ADCn_SR register occur as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register, and the sequence end flag EOSIF flag in the ADCn_SR register is generated when the EOCIF occurs during the last sequence.

Figure 169. ADC Trigger Timing in Burst Mode (SEQCNT[2:0] = ‘111’, 8 sequential conversion)



15.3.12 Multiple Mode Conversion (ADMOD[1:0] = ‘10’, SEQCNT[2:0] > 0)

In Multiple mode, the A/D conversions are performed in the order in which trigger events occur for each analog input. Since the A/D conversion starts for the analog inputs in the corresponding sequence at the point when the trigger event occurs, the EOS signal indicating end of the sequence is ignored.

- If ADMOD[1:0] = ‘10’, A/D conversion in multiple mode.

There are four types of SOC trigger sources in multiple mode:

- If TRGSEL[1:0] = ‘00’, software trigger event (ACLK).
- If TRGSEL[1:0] = ’01’, TIMERn (n = 0 to 9) trigger event.
- If TRGSEL[1:0] = ‘10’, MPWM0 trigger event.
- If TRGSEL[1:0] = ‘11’, MPWM1 trigger event.

The SEQCNT[2:0] in ADCn_MR register in multiple mode:

Since the A/D conversion is executed for the multiple channels, the SEQCNT[2:0] bits in the ADCn_MR register must be set to a number other than ‘0’.

- If SEQCNT[2:0] = ‘001’ to ‘111’, Multi-channel sampling in multiple mode

Trigger event source settings:

The number of trigger sources in each sequence can be configured is set in the SEQCNT[2:0] bits in the ADCn_MR register.

Users can set the desired trigger sources in the SEQTRGx[3:0] bits in the ADCn_TRG register. Each of these sources triggers an SOC when the triggering conditions are met, regardless of the sequence set in the SEQTRGx[3:0] bits in the ADCn_TRG register (n= 0 to 2, x = 0 to 7).

For example, if the SEQCNT[2:0] bits in the ADCn_MR register is set to '11' and four trigger sources are selected from among the TIMER and MPWM trigger sources, setting start bit of trigger source event lets the conversions be triggered by these trigger sources in the order of reception, unlike sequential or burst modes.

- If SEQTRG0[3:0] = x, 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x, 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x, 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x, 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x, 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x, 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x, 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x, 8th channel's trigger source is x.

NOTES:

1. x = 0 to 15.
2. Unused trigger source of the SEQTRGx[3:0] bits must be set to 14 for avoid unwanted trigger operation.

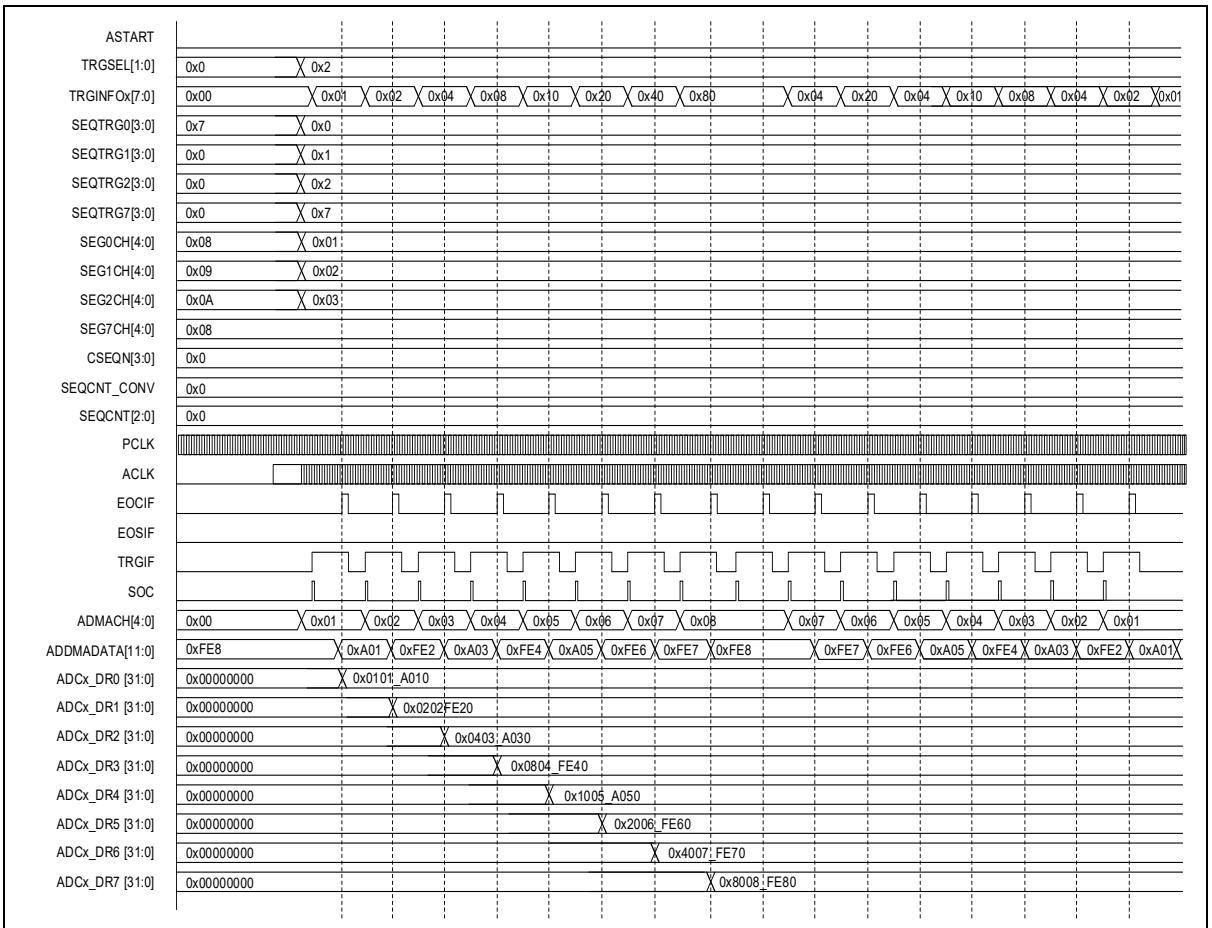
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

The EOCIF flag in the ADCn_SR register is generated as many as the number of sequences set in the SEQCNT[2:0] bits , and the End of Sequence Interrupt Flag EOSIF flag in the ADCn_SR register is not occurred because the A/D conversions are performed at every trigger source event, regardless of the sequence order.

Therefore, users can obtain the digitized data of the analog inputs by polling the EOCIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOCIF flag in the ADCn_SR register interrupt occurs.

Figure 170. ADC Multiple Mode Timing (when ADMOD[1:0] = '10' and MR.SEQCNT[2:0] > 0)



NOTES:

1. PCLK = 140 MHz, ACLK = 10 MHz (CLKDIV = 14)
2. n = 0 to 2, x = 0 to 7.
3. When ADC operates in Multiple Mode, the channel information of current A/D conversion should be referenced with ACH[4:0] in ADCn_DRx register or ADMACH[4:0] in ADCn_DDR register after enabling DMAEN = '1'

15.3.13 Starting Conversions (ASTART, Start Trigger, ARST)

15.3.13.1 Start A/D Conversion (ASTART)

The A/D conversion starts when the ASTART bit in the ADCn_CR register is set to '1' by software as described below:

- If ASTART = 0, ADC conversion does not start. (software trigger)
- If ASTART = 1, ADC conversion starts. This bit is cleared at the next ADC clock cycle. (software trigger)

15.3.13.2 Reset A/D Conversion (ARST)

If the ARST bit in the ADCn_MR register is set to '0' and the software trigger is used as a trigger source, users must set the ASTART bit in the ADCn_CR register to '1' using software to restart the A/D conversion at the end of the A/D conversion sequence.

If the A/D conversions are performed per sequence by the trigger source in Sequential mode, set the ARST bit in the ADCn_MR register to '1' to restart the A/D conversion when the EOSIF flag in the ADCn_SR register is generated at the end of the A/D conversion.

- If ARST = 0, A/D conversion stops at EOS event. (To restart the A/D conversion, the ASTART must be set to '1').
- If ARST = 1, A/D conversion restarts when proper trigger source occurs after End-of-Sequence (EOS) event.

15.3.13.3 Start A/D Conversion by Trigger Source

If the ADC timer trigger source is used, software allows the A/D conversion to start when the timer trigger source occurs after timer operating. (Refer to section 15.3.16)

- The TEN = 1 in the TIMERn_CR2 (n = 0 to 9).
- The PWMEN = 1 in the MPWMn_CR (n = 0 to 1).
- The ASTART = 1 in the ADCn_MR (n = 0 to 2).

When the A/D conversion is complete for one channel, the EOCIF flag is generated and the ASTART bit in the ADC_CR register is cleared by hardware automatically.

In Sequential, Multiple, and Burst mode, the EOCIF flags are generated as many channels as set in a sequence, and the EOSIF flag is generated when sampling of the last channel is complete except Multiple mode.

15.3.14 ADC Timing

The A/D conversion time consists of the followings:

t_{ADC} = 3 ACLK cycles [delay for synchronizing until SOC event]

+ (2 + STSEL[2:0]) ACLK cycles [sampling time]

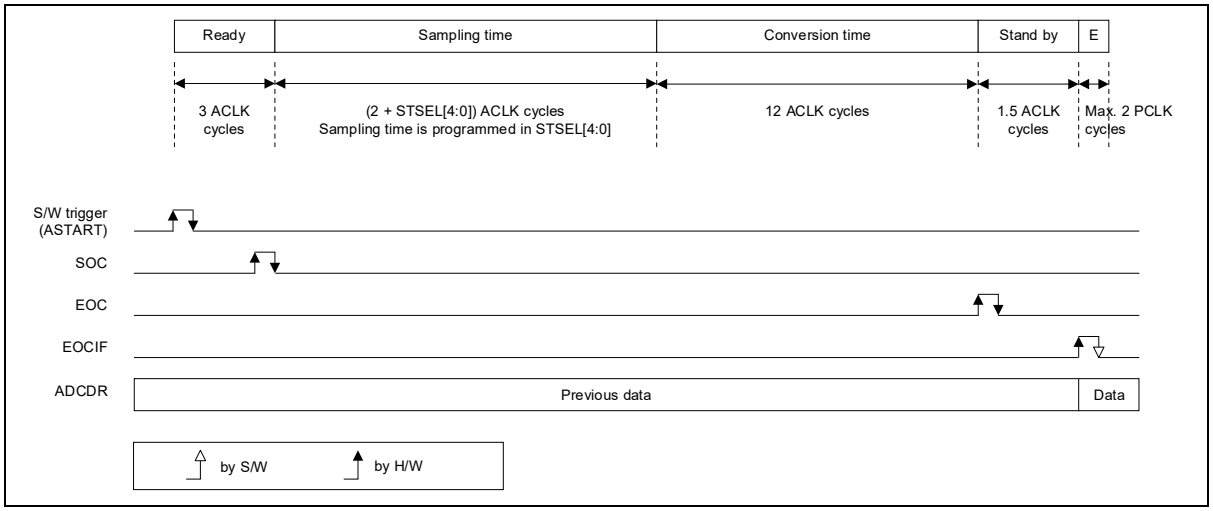
+ 12 ACLK cycles [conversion time]

+ 1.5 ACLK cycles [EOC flagged time]

+ Max. 2 PCLK cycles [EOCIF flagged time]

= (2 + STSEL[4:0]) ACLK cycles [sampling time] + 16.5 ACLK cycles

Figure 171. Analog-to-Digital Conversion Time



The total conversion time is calculated as follow example:

- With f_{ACLK} = 20 MHz and a STSEL[4:0] = 2, sampling time is 4 ACLK cycles
- Then, t_{ADC} = { (2 + 2) + 16.5 } ACLK cycles = 20.5 ACLK cycles = 1.025 µs

ADC sampling timing adjustment is required for high-speed A/D conversion of analog input.

For high-speed analog input A/D conversion, it is necessary to adjust the A/D conversion timing for one sample by setting the ADC operation clock and external resistors R_{AIN} connected to the analog input and STSEL[4:0] bits in the ADCn_MR register value. Table 138 shows the recommended values for R_{AIN} and STSEL[4:0] bits in the ADCn_MR register for A/D conversion when the ADC's reference voltage is 5V and ACLK 25 MHz.

Table 138. ADC Frequency for the Values of R_{AIN}.

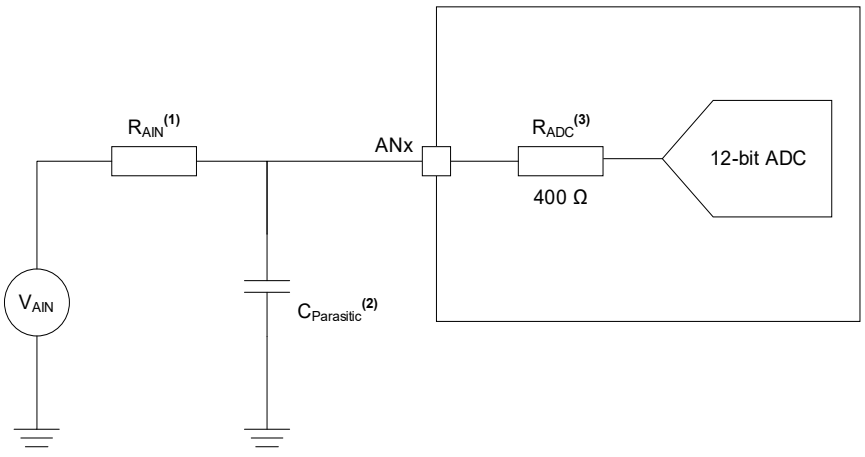
R_{AIN} Max. (kΩ)	t_{CONV} [cycle]	t_{CONV} [μs]	STSEL[4:0]	t_{SMPL} [cycle]	t_{SMPL} [μs]	f_{ADC} [Msps]
0.01	12	0.48	10	12	0.48	1.04
0.1	12	0.48	11	13	0.52	1
0.3	12	0.48	12	14	0.56	0.96
0.5	12	0.48	13	15	0.6	0.92
1.0	12	0.48	14	16	0.64	0.89
1.5	12	0.48	16	18	0.72	0.83
2.0	12	0.48	18	20	0.8	0.78
4.0	12	0.48	22	24	0.96	0.69
5.0	12	0.48	26	28	1.12	0.62

NOTES:

1. The parameters are guaranteed by design.
2. f_{ADC} is an A/D conversion rate from SOC to EOC.
3. See Figure 172 for how to design the R_{AIN} resistor circuit for the ANx analog input.

The circuit of the analog input ANx for ADC to which the source is supplied can be designed as follows. When using I/O set to analog input mode, the ratio of the internal resistance R_{ADC} to the external resistance R_{AIN} must be considered when inputting an external analog source.

Figure 172. Typical Connection Diagram using ADC



NOTES:

1. Refer to Table 138 for more information of the ADC frequency.
2. $C_{Parasitic}$ represents the capacitance of the PCB (Dependent on soldering and PCB layout quality) plus the pad capacitance. A high $C_{Parasitic}$ value downgrades conversion accuracy. To remedy this, f_{ADC} should be reduced.
3. R_{ADC} is the internal resistance value when the I/O pin is set to ANx, which is an analog input state.

15.3.15 Stopping Ongoing Conversion (ASTOP)

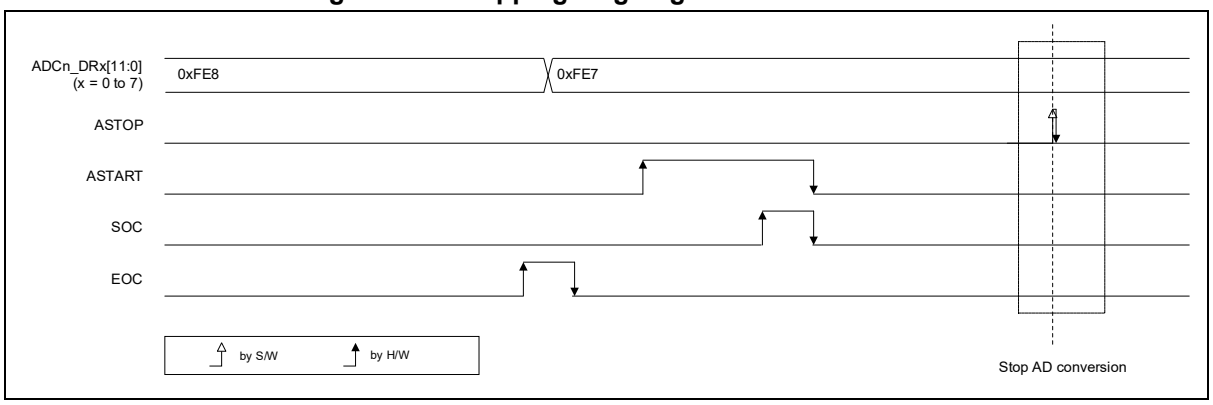
The A/D conversion in progress can be stopped by setting the ASTOP bit in the ADCn_CR register to '1' using software. When the A/D conversion stops, the ADC operation in progress is reset. The ADC can be reconfigured (e.g. channel selection, trigger change) for new input data.

Once the ASTOP bit in the ADCn_CR register is set to '1' by software, all A/D conversions in progress stop and partial results are deleted. (However, the ADCn_DRx (n = 0 to 2, x = 0 to 7) register is not updated due to this deletion.)

During the sequence conversion of the multi-channel, if the ASTOP bit in the ADCn_CR register is set to '1', then the A/D conversion of a sequence stops.

When this procedure is completed, hardware clears the ASTOP bit in the ADCn_CR register, and software needs to clear the EOCIF flag in the ADCn_SR register.

Figure 173. Stopping Ongoing A/D Conversions



15.3.16 Conversion on External Trigger (TRGSEL[1:0], SEQTRGx[3:0], TRGIE, TRGIF)

The A/D conversion can be triggered by the software or the events of the TIMER module and MPWM module (Timer capture with input pin or timer match A interrupt, etc.). When setting the trigger source, it is recommended to stop the A/D conversion by setting the ASTART bit to '0'.

15.3.16.1 ADC Trigger Source Selection (TRGSEL[1:0])

Users can select a trigger source by configuring the TRGSEL[1:0] in ADC_MR register as described in Table 139.

Table 139. External Trigger Source Configuration

TRGSEL[1:0]	Trigger Source	Description
00	ASTART	Disables event triggers and enables the soft trigger only.
01	TIMERn (n = 0 to 9)	Enables the timer event trigger and soft trigger.
10	MPWM0	Enables the MPWM0 event trigger and soft trigger.
11	MPWM1	Enables the MPWM1 event trigger and soft trigger.

To use the TIMER as the ADC trigger source, select the event and polarity of the TIMER channel, then set the ADCTRGEN in TIMER_CR1 register bit to '1'. (n = 0 to 9).

To use the MPWM as the ADC trigger source, configure the MPWM_ATRx register to set a channel, an event, and the point of time for trigger updates. Then, set the corresponding ATRxIE bit in the MPWM_IER register of the event channel to '1' to enable the interrupt. Refer to section 14.6.21 and 14.6.14.

15.3.16.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0])

Once a type of the trigger source is determined, up to 8 trigger sources (As a single trigger source or a sequence of trigger sources per an ADC unit) can be selected by configuring the ADC_TRG register. If a trigger source for a single A/D channel is used, users can use the SEQTRG0[3:0] bits in the ADC_TRG register to select a trigger source.

For each sequence sampling in Sequential / Burst / Multiple mode, assign the corresponding trigger source for the A/D conversion channel using the SEQTRGx[3:0] bits in the ADC_TRG register (x = 0 to 7).

Table 140. Trigger Source Table

Trigger Source (SEQTRGx)	TIMER (TRGSEL[1:0] = '01')	MPWM (TRGSEL[1:0] = '10')	MPWM (TRGSEL[1:0] = '11')
0	TIMER 0	MPWM0_ATR1	MPWM1_ATR1
1	TIMER 1	MPWM0_ATR2	MPWM1_ATR2
2	TIMER 2	MPWM0_ATR3	MPWM1_ATR3
3	TIMER 3	MPWM0_ATR4	MPWM1_ATR4
4	TIMER 4	MPWM0_ATR5	MPWM1_ATR5
5	TIMER 5	MPWM0_ATR6	MPWM1_ATR6
6	TIMER 6	PERIOD / PERIODU	PERIOD / PERIODU
7	TIMER 7	BOTTOM / BOTTOMU	BOTTOM / BOTTOMU
8	TIMER 8	PERIODV	PERIODV
9	TIMER 9	BOTTOMV	BOTTOMV
10	-	PERIODW	PERIODW
11	-	BOTTOMW	BOTTOMW
12	-	-	-
13	-	-	-
14	-	-	-
15	ASTART(1)	ASTART(1)	ASTART(1)

NOTE:

1. The ASTART bit in the ADC_CR register is a software trigger.

15.3.16.3 ADC Trigger Start Control

If a trigger event occurs after the trigger source's timer operation starts, A/D conversion is performed by trigger source.

Table 141. Start Control of AD Trigger Conversion

TRGSEL[1:0]	Start Control Bit
ASTART	ASTART = '1' in ADC_CR register
TIMERn (n = 0 to 9)	TEN = '1' in TIMER_CR2 register
MPWMn (n = 0 to 1)	PWMEN = '1' MPWM_CR1 register

15.3.16.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF)

Interrupts for the ADC trigger events can be enabled by configuring the TRGIE bit in the ADCn_IER register as shown below:

- If TRGIE = 0, Trigger source event interrupt is disabled.
- If TRGIE = 1, Trigger source event interrupt is enabled.

When the ADC trigger interrupt is enabled, the TRGIF flag is generated at the point of time when an ADC trigger source event of each A/D conversion channel is generated. The TRGIF flag is cleared by writing '1' to this bit.

Although the trigger interrupt of MPWM or TIMER is not enabled, TRGIF flag of ADC is generated when trigger source event happens. At this time, If the TRGIE bit was enabled, a trigger event interrupt is generated. If the trigger event occurs while TRGIE bit disabled, the TRGIF bit can be checked by polling with software.

15.3.17 End-of-Conversion, End-of-Sampling Phase (EOC)

The ADC notifies user application at the end of A/D conversion (EOC) event. For the polling operation, set the ASTART bit to '1' and poll until the EOCIF bit is set to '1'. Then read access the ADCn_DDR or ADCn_DRx register to obtain the A/D conversion data.

For the use of interrupts, setting the EOCIE bit in the ADCn_IER register to '1' generates an interrupt when the EOCIF bit in the ADCn_SR register is set to '1' after the A/D conversion is completed. Then read access the ADCn_DDR or ADCn_DRx (n = 0 to 2, x = 0 to 7) register to obtain the A/D conversion data. The EOCIF flag in the ADCn_SR register is cleared by writing '1' to this bit using software.

NOTE:

1. The initial state of the End-of-Conversion (EOC) of the ADC is an unknown status when the microcontroller is first powered on. In this case, leakage current may flow if the microcontroller enters SLEEP mode or DEEP-SLEEP mode. To avoid this situation, set the ASTART bit in the ADCn_CR register to '1' and perform two dummy transfers of the ADCn_DRx (n = 0 to 2, x = 0 to 7) register to clear the EOC. And clear also the EOCIF flag in the ADCn_SR register. Users must consider this situation when initializing the ADC, since it cannot be processed by hardware.

15.3.18 End-of-Sequence Conversion (EOSIE, EOSIF)

The ADC module allows the sequence setup for the A/D conversion of analog inputs in multi-channels. A sequence can be configured with up to 8 channels using the SEQCNT[2:0] bits in the ADCn_MR register as shown below:

- If SEQCNT[2:0] = '000': 1 Sequence ADC (single mode conversion).
- If SEQCNT[2:0] = '001': 2 Sequences ADC.
- If SEQCNT[2:0] = '010': 3 Sequences ADC.
- If SEQCNT[2:0] = '011': 4 Sequences ADC.
- If SEQCNT[2:0] = '100': 5 Sequences ADC.
- If SEQCNT[2:0] = '101': 6 Sequences ADC.
- If SEQCNT[2:0] = '110': 7 Sequences ADC.
- If SEQCNT[2:0] = '111': 8 Sequences ADC.

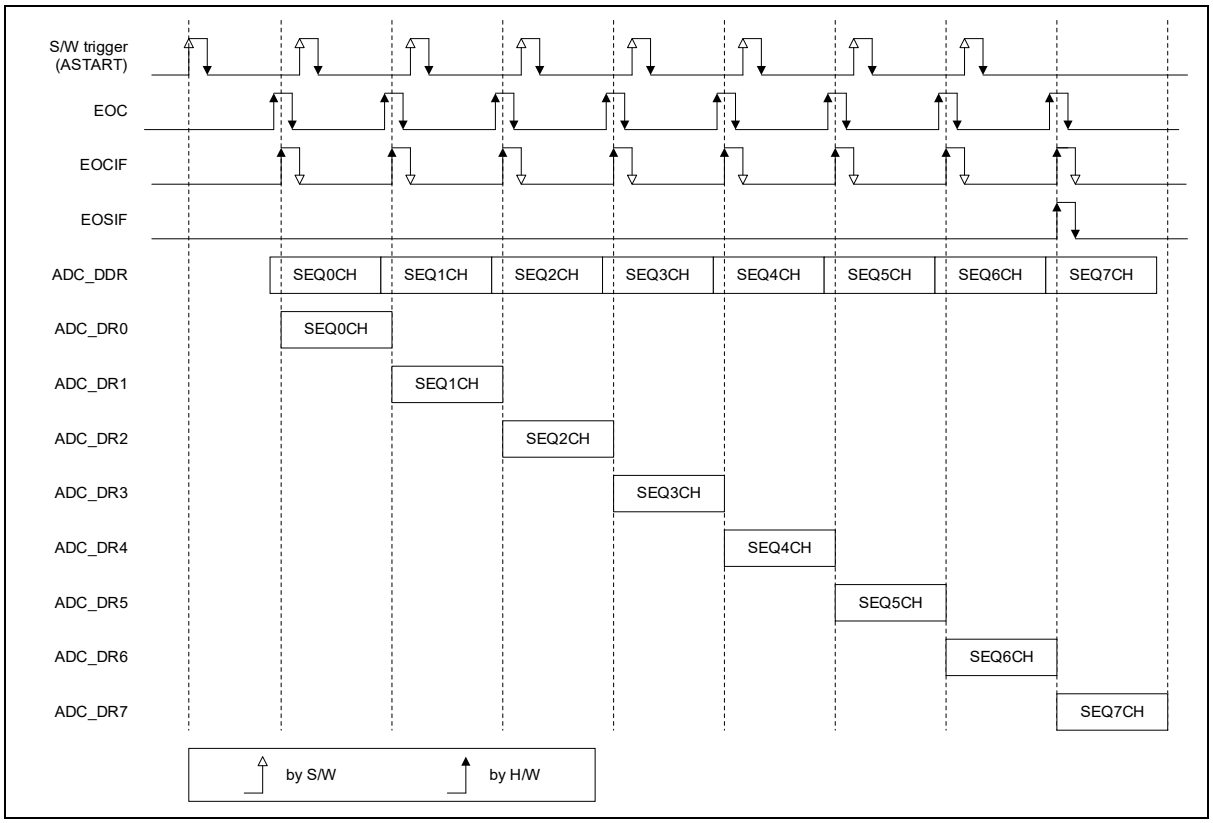
After setting the number of channels, users can assign the channels that are performed in each sequence using the ADCn_SCSR1 register (SEQ0CH to SEQ3CH) and the ADCn_SCSR2 (SEQ4CH to SEQ7CH) register. Refer to section 15.3.7.

Sequences in sequential mode are performed by following the ascending order of the sequence trigger source number set in the SEQTRGx[3:0] bits in the ADCn_TRG register. In multiple mode, the sampling order is determined by the timing of each trigger source's event.

Users can adjust the trigger timing to set the sampling period for the ADC sequence. Refer to section 15.3.16.

For the ADC sequence conversion interrupt operation, enable the EOSIE bit in the ADCn_IER register. If the EOSIF = '1', the A/D conversion is performed for the specified channel of a sequence. When the A/D conversion is completed for each channel, the EOCIF flag in the ADCn_SR register is set to '1'.

Figure 174. Example of A/D Conversion Timing in Sequence Mode



15.3.19 Data Management and Information

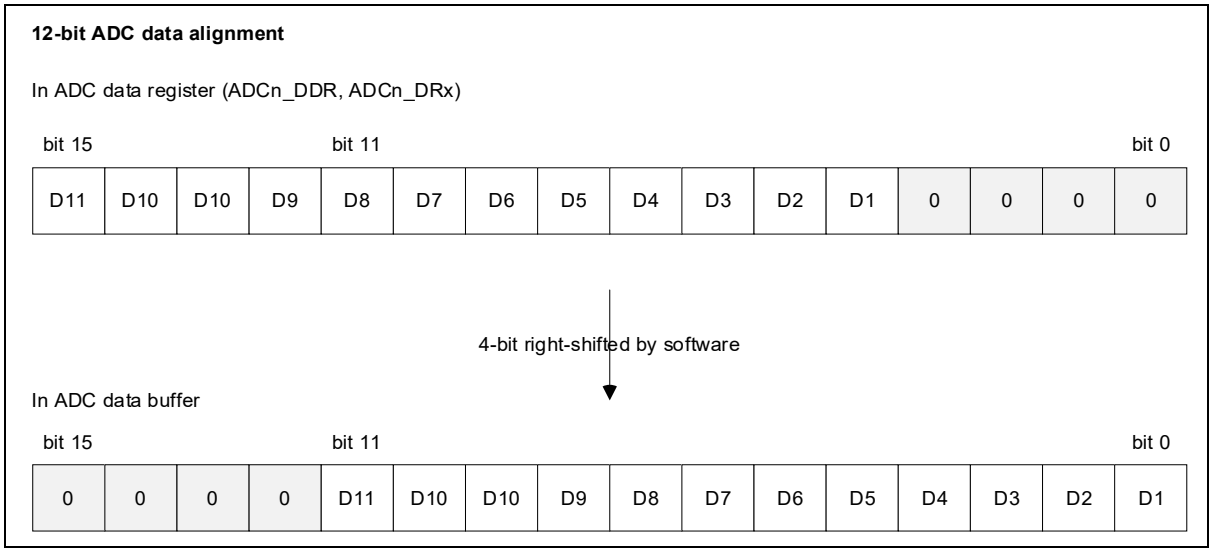
15.3.19.1 Data Register and Data Alignment

At the EOC event of each A/D conversion channel, the result is stored in ADC data registers such as ADCn_DDR, ADCn_DR0, ADCn_DR1, … and ADCn_DR7.

- ADCn_DDR: This register stores the A/D data value of a channel where the A/D conversion was completed most recently.
- ADCn_DRx: This register stores the A/D conversion result of a channel set in each sequence.

As shown in Figure 175, the A/D conversion results are left-aligned in the lower 16 bits of the data registers and then shifted by software to the right by 4 bits, generating a 12-bit value:

Figure 175. Left Alignment (unsigned value)



15.3.19.2 Channel and Trigger Information

Users can get information of the A/D conversion data channel and the trigger sources by checking the upper 16 bits of the ADC data registers.

- Set the CHINFO bit in the ADCn_MR register to '1' to store the A/D conversion channel information to the ACH[4:0] bit field in the ADC_DRx register.
- Set the TRGINFO bit in the ADCn_MR bit to '1' to store the trigger source information for the channel where the A/D conversion is performed to the TRGINFOx (x = 0 to 7) bit in the ADC_DRx register.

The TRGINFOx bit in the ADC_DDR register (x = 0 to 7) shows that the lower the channel number set in a sequence, the higher the channel priority and is executed first.

Figure 176 shows the configuration of the channel and trigger information in the ADC_DDR and ADC_DRx registers. As shown in the figure, the SEQxCH (x = 0 to 7) sequence is configured with 8 channels adc_ch0 to adc_ch7, allowing users to check the channel and trigger information in the in the ADC_DDR and ADC_DRx registers.

Figure 176. Channel and Trigger Source Information of A/D Conversion Sequence

	SEQ0CH	SEQ1CH	SEQ2CH	SEQ3CH	SEQ4CH	SEQ5CH	SEQ6CH	SEQ7CH
	adc_ch0	adc_ch1	adc_ch2	adc_ch3	adc_ch4	adc_ch5	adc_ch6	adc_ch7
ACH[4:0] (Channel number of AD conversion)	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
TRGINFO[7:0] (Trigger source priority of AD conversion)	0x01	0x02	0x04	0x08	0x10	0x20	0x40	0x80

15.3.20 ADC Low-Power Modes

Table 142. Effect of Low-Power Mode on ADC

Mode	Description
SLEEP	No effect, ADC used with DMA
DEEP-SLEEP	The ADC module control is reset in DEEP-SLEEP mode. ADC module must be reinitialized after waking up from DEEP-SLEEP mode.

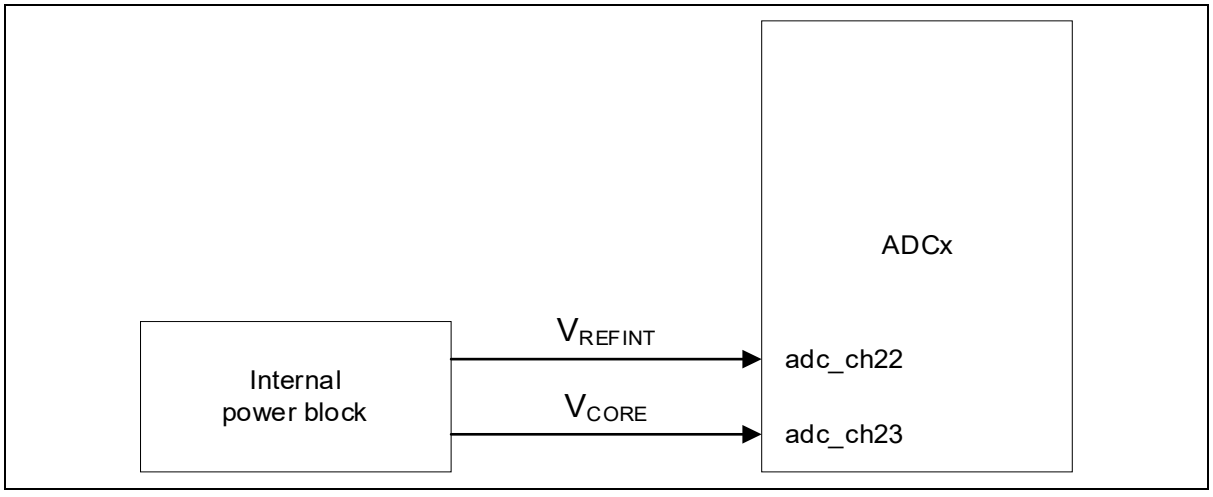
NOTE:

1. Applications using ADC modules must clear the EOC at initialization, when using DEEP-SLEEP mode. Please refer to 15.3.17.

15.3.21 Monitoring the Internal Voltage Reference

It is possible to have reference point for the evaluation of external analog input voltage level by monitoring the internal voltage reference V_{CORE} or V_{REFINT}. The analog signals of the internal voltage references V_{CORE} outputting Typ. 1.5 V and the V_{REFINT} outputting Typ. 1.0 V are entered into the adc_ch23 and adc_ch22 of all ADC unit.

Figure 177. Channel Block Diagram of V_{CORE} and V_{REFINT}



15.3.22 A/D Conversion using the DMA

The ADC module sends a request for data transfer to the DMA module when the A/D conversion of the analog inputs is completed and the EOCIF flag in the ADCn_SR register is set to '1'. The DMA module transfers data to the specified memory area upon receiving this request.

To use the A/D conversion functionality, enable the ADC and DMA using the SCU and then activate the DMAEN bit in the DMAn_SR register to '1' with the ADEN bit in the ADCn_MR register to '1' as listed below:

- If DMAEN = '0', ADC DMA is disabled.
- If DMAEN = '1', ADC DMA is enabled.

To connect the ADC to the DMA channel, set the PERISEL[4:0] bits in the DMA_CR register to select the ADC, referring to the number below:

- PERISEL[4:0] = '01101' (0xD, Selected ADC module of DMA transferring) in DMA_CR register.

Set the SIZE[1:0] bits in the DMAn_CR register to half-word for transferring DMA data of the A/D conversion data, and write the number of DMA data to the TRANSCNT[11:0] bits in the DMAn_CR register (n = 0 to 15). For example, if users want to transfer 64 A/D conversion data values to the DMA, the DMA data width is set to 16-bit, and the number of the DMA data transfers is set to 64 as shown below:

- SIZE[1:0] = '01' (half-word, 16-bit DMA data width)
- TRANSCNT[11:0] = 0x40 (64 half-word DMA data)

Set the register address of an ADC unit that is to be transferred by DMA to the PAR[15:0] bits in the DMAn_PAR register and the DMA transfer's destination memory (SRAM) address in the MAR[15:0] bits in the DMAn_MAR register.

- PAR[15:0] = 0xB02C (0x4000_B02C, ADCn_DDR register address)
- MAR[15:0] = 0x3000 (0x2000_3000, buffer data address)

The direction of the DMA transfer is determined based on the memory and can be controlled by setting the DIR bit in the DMAn_CR register. For example, to store the A/D conversion results in memory, users must set the DIR = '1'.

- If DIR = '0', Memory to Peripheral (Tx)
- If DIR = '1', Peripheral to Memory (Rx)

The ADC DMA interrupt can be controlled by setting the DMAIE bit in the ADCn_IER.

- If DMAIE = '0', Disable ADC DMA done interrupt
- If DMAIE = '1', Enable ADC DMA done interrupt

When the DMAEN bit in the DMAn_SR is set to '1', the DMA is enabled, and the A/D conversion results are transferred to the address specified by the DMA controller using software. The value in the TRANSCNT[11:0] bits in the DMAn_CR register is decreased by one each time a DMA transfer of a

specified size is completed.

When the DMA transfer is completed by the data size requested to the DMA, and the TRANSCNT[11:0] bits in the DMAn_CR register is set to '0', the interrupt is generated with the EOT flag in the DMAn_SR register of the DMA module and the DMAF flag in the ADCn_SR register indicating the DMA data transfer is done of the ADC module.

- If DMAF = 0, There is DMA data to transfer.
- If DMAF = 1, There is no DMA data to transfer. The DMA transfer is complete.
- The DMAF is cleared by writing '1' to this bit.

The EOT flag in the DMAn_SR register of the DMA module is enabled when the transmission is complete and the TRANSCNT[11:0] value becomes 0. The EOT flag is cleared by writing a number larger than zero to the TRANSCNT[11:0] bits in the DMAn_CR register.

The A/D conversion results before the generation of the EOCIF can be referenced only by accessing the ADCn_DRx and ADCn_DDR registers in the ADC module, since the latest results are always updated in the registers.

By using the DMA functionality, users can store the A/D conversion results as much as the size set by software and prevent loss of already stored data.

15.3.23 ADC Overrun (OVR) in DMA Mode

The DMA data transfer requests and ACK signals between the ADC and the DMA modules occur in the following order:

1. When the ADC completes the A/D conversion and the EOCIF flag in the ADCn_SR register is set to '1', the ADC sends a data transfer request to the DMA.
2. The DMA transfers the A/D conversion results to the specified SRAM memory address and sends the ACK signal to the ADC.
3. When the TRANSCNT[15:0] bits in the DMAn_CR register value becomes '0', the EOT flag in the DMAn_SR register signal is sent to the ADC module to inform that all DMA data transmissions are complete.

When the DMA does not complete the transmission yet and cannot send the ACK signal to the ADC, if it receives another transfer request from the ADC, the DMA overrun flag (DOVRUN) in the ADCn_SR register) is set to '1'. DOVRUN does not support interrupt function.

If the overrun flag is generated because the DMA cannot process the DMA transfer requests on time, the ADC stops requesting transfer to the DMA and the DMA does not transfer new ADC conversion results. In other words, all data transferred to SRAM can be considered valid. The DMA transfer request is blocked until software clears the DOVRUN flag in the ADCn_SR register.

Even if the overrun flag is generated, the ADC maintains operating. Unless software sets the ASTOP bit in the ADCn_CR register to '1' to stop and reset the ADC sequence, the ADC continues to perform the A/D conversions to store the latest results in the ADCn_DDR register until the EOCIF flag in the

ADCn_SR register is generated.

15.3.24 Comparison of ADC Sensing Value

The ADC has a built-in comparator that compares the sampling results of the AD inputs and software set values.

Users can compare the value set by CVAL[11:0] with the value A/D conversion channel specified by CCH[4:0] based on the VDD reference voltage.

Set the CCH[4:0] bits in the ADCn_CMPR register to select a target ADC channel for comparison and enter the comparison value in the CVAL[11:0] bits in the ADCn_CMPR register.

- If CMPEN = 0, ADC comparator operation is disabled.
- If CMPEN = 1, ADC comparator operation is enabled.

Figure 178. Block diagram of ADC internal comparator

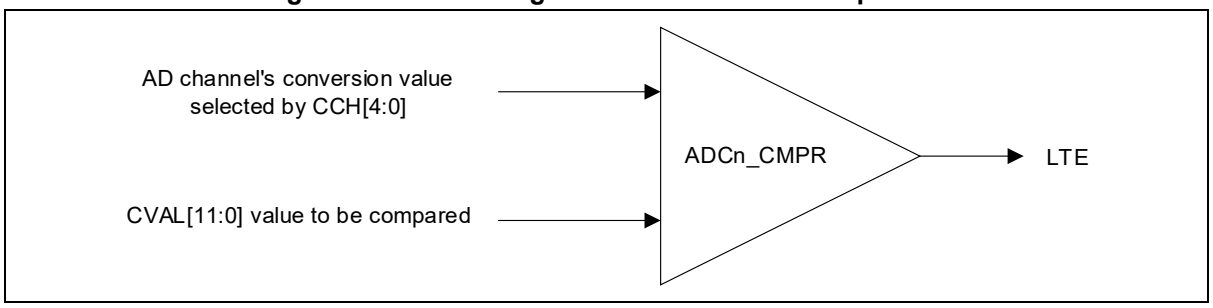


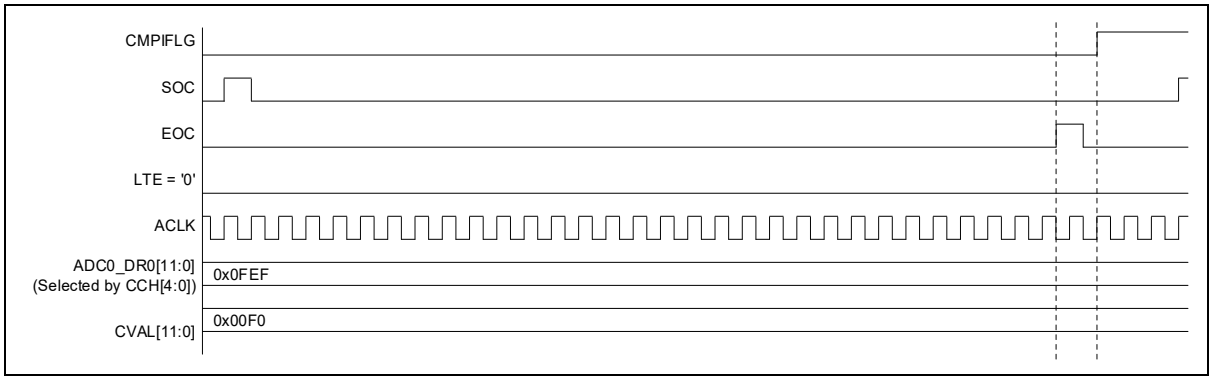
Table 143 shows the conditions under which the CMPIFLG flag occurs by comparing the CVAL[11:0] and the ADC channel values set in the CCH[4:0] bits in the ADCn_CMPR register according to the LTE bit in the ADCn_CMPR register.

Table 143. Comparison and Flag Generation Conditions by LTE

LTE bit	Comparison Condition	CMPIFLG Occurrence Condition
0	ADC value of CCH[4:0] channel > CVAL[11:0]	1
0	ADC value of CCH[4:0] channel ≤ CVAL[11:0]	0
1	ADC value of CCH[4:0] channel > CVAL[11:0]	0
1	ADC value of CCH[4:0] channel ≤ CVAL[11:0]	1

Since the ASTART bit in the ADCn_CR register is set to ‘1’, when the EOCIF flag in the ADCn_SR register indicating the A/D conversion completion is set to ‘1’, two input signals are compared internally and the CMPIFLG flag in the ADCn_SR register indicating the comparison completion is generated (CMPIFLG = ‘1’).

Figure 179. Timing Diagram of Comparison between A/D Conversion Results and Reference Value



The comparison results based on the LTE bit in the ADCn_CMPR register value affect the flag setting condition of the CMPIFLG flag in the ADCn_SR register as shown below:

- If LTE = 0, the interrupt is generated when the A/D conversion result is larger than the comparison value (CVAL[11:0]).
- If LTE = 1, the interrupt is generated when the A/D conversion result is less than or equal to the comparison value (CVAL[11:0]).

The interrupts for the comparison results can be enabled by setting the CMPIEN bit in the ADCn_CMPR register to ‘1’ as shown below:

- If CMPIEN = 0, ADC compare interrupt is disabled.
- If CMPIEN = 1, ADC compare interrupt is enabled.

15.4 ADC Interrupts

For each ADC unit, an interrupt can be generated by setting below:

- End-of-Single data conversion (EOCIE)
- End-of-Sequence of conversion (EOSIE)
- Interrupt by trigger source (TRGIE)
- End-of-DMA transfer of A/D conversion data (DMAIE)
- ADC conversion data comparison (CMPIEN)

Each interrupt enable bits are available for flexibility.

Table 144. Control Bits and Event Flags for ADC Interrupts

Interrupt Event	Event Flag	Enable Control Bit
End of conversion of single data	EOCIF	EOCIE
End of sequence conversion	EOSIF	EOSIE
End of conversion by timer trigger source	TRGIF	TRGIE
End of DMA transfer of ADC conversion data	DMADF	DMAIE
ADC conversion data comparison	CMPIFLG	CMPIEN

15.5 ADC Registers

The base addresses of the ADCs and register map are described in the followings:

Table 145. Base Address of ADC

Name	Base Address
ADC0	0x4000_B000
ADC1	0x4000_B100
ADC2	0x4000_B200

Table 146. ADC Register Map

Name	Offset	Type	Description	Reset Value	Reference
ADCn_MR	0x0000	RW	ADC n Mode Register	0x0000_0000	15.5.1
ADCn_CSCR	0x0004	RW	ADC n Current Sequence and Channel Register	0x0000_0000	15.5.2
ADCn_CCR	0x0008	RW	ADC n Clock Control Register	0x0000_0080	15.5.3
ADCn_TRG	0x000C	RW	ADC n Trigger Select Register	0x0000_0000	15.5.4
ADCn_SCSR1	0x0018	RW	ADC n Channel Select 1 Register	0x0000_0000	15.5.5
ADCn_SCSR2	0x001C	RW	ADC n Channel Select 2 Register	0x0000_0000	15.5.6
ADCn_CR	0x0020	RW	ADC n Control Register	0x0000_0000	15.5.7
ADCn_SR	0x0024	RW	ADC n Status Register	0x0000_0000	15.5.8
ADCn_IER	0x0028	RW	ADC n Interrupt Enable Register	0x0000_0000	15.5.9
ADCn_DDR	0x002C	RO	ADC n DMA Data Register	0x0000_0000	15.5.10
ADCn_DR0	0x0030	RO	ADC n Sequence 0 Data Register	0x0000_0000	15.5.11
ADCn_DR1	0x0034	RO	ADC n Sequence 1 Data Register	0x0000_0000	15.5.11
ADCn_DR2	0x0038	RO	ADC n Sequence 2 Data Register	0x0000_0000	15.5.11
ADCn_DR3	0x003C	RO	ADC n Sequence 3 Data Register	0x0000_0000	15.5.11
ADCn_DR4	0x0040	RO	ADC n Sequence 4 Data Register	0x0000_0000	15.5.11
ADCn_DR5	0x0044	RO	ADC n Sequence 5 Data Register	0x0000_0000	15.5.11
ADCn_DR6	0x0048	RO	ADC n Sequence 6 Data Register	0x0000_0000	15.5.11
ADCn_DR7	0x004C	RO	ADC n Sequence 7 Data Register	0x0000_0000	15.5.11
ADCn_CMPR	0x0070	RW	ADC n Channel Compare Register	0x0000_0000	15.5.12

NOTE:

1. n = 0, 1, and 2.

15.5.1 ADCn_MR: ADC n Mode Register

The ADCn_MR register configures the modes of the ADC module. Users should set this register prior to all other ADC registers according to the intended use of the ADC module.

ADC0_MR=0x4000_B000, ADC1_MR=0x4000_B100, ADC2_MR=0x4000_B200

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
Reserved		TRGINFO	CHINFO	Reserved		DMAEN		STSEL[4:0]	Reserved	SEQCNT[2:0]	ADEN	ARST	ADMOD[1:0]	Reserved	TRGSEL[1:0]
-		0	0	-		0		00000	-	000	0	0	00	-	00
-		RW	RW	-		RW		RW	-	RW	RW	RW	RW	-	RW

21	TRGINFO	Enable the trigger information option (In external trigger mode)
	0	Disables the option.
	1	Stores trigger source information in ADCn_DRx [4:0] (n = 0 to 2, x = 0 to 7).
20	CHINFO	Enable the channel information option
	0	Disables the option.
	1	ACH[4:0] in ADCn_DRx register stores the information of the channel through which data conversion has been conducted.
17	DMAEN	DMA enable bit (This bit must be set if ADCEN = 1.) If DMA is enabled, each DMA interrupt request is made when the ADC is signaled by the DMA controller that the previous DMA transfer has been completed (the status bit of the ADC_SR register is '1'). For the multiple transfers, the DMA requests is automatically made at every interrupt and conversion completion (The bit setting is valid in burst mode as well).
	0	Disable
	1	Enable
16 12	STSEL[4:0]	Sampling time selection
		The bits determine the time window in which the next trigger is recognized. This set value is applied right after the occurrence of the current trigger. ADC sampling time is calculated by (2 + STSEL[4:0]) ACLK cycles. The minimum sampling time is two ACLK cycles, and the sampling channel is always active when STSEL[4:0] = '11111'.
10 8	SEQCNT[2:0]	Number of conversions in a sequence If ADMOD[1:0] is '0' and SEQCNT[2:0] is not '0', the CSEQN value increments to the SEQCNT[2:0] value by a trigger event. (The setting of the bit field is valid only in Single and Sequential modes.)
	000	1-sequence ADC
	001	2-sequence ADC
	010	3-sequence ADC
	011	4-sequence ADC
	100	5-sequence ADC
	101	6-sequence ADC
	110	7-sequence ADC
	111	8-sequence ADC
7	ADEN	ADC enable bit
	0	Disable
	1	Enable

6	ARST	Setting bit to restart at the end of the sequence	
		0	Stops at the end of the sequence. (Set the ASTART to '1' to enable the restart.)
		1	Restart at the end of the Sequency.
5 4	ADMOD[1:0](1)	ADC mode selection bit	
		00	Single or Sequential conversion mode
		01	Burst conversion mode
		10	Multiple conversion mode
		11	Not used
1 0	TRGSEL[1:0]	Trigger selection bit	
		00	Event trigger disable / Software trigger only
		01	Timer event trigger / Software trigger
		10	MPWM event trigger / Software trigger
		11	Not used

NOTE:

1. If ADMOD[1:0] has been set for burst conversion mode, the ADC channels are assigned as BST0CH through BST7CH. Burst mode always begins with BST0CH. (In three-burst mode, for example, the analog inputs to the BST0CH, BST1CH, and BST2CH channels are converted in the order of channel number.)

   If ADMOD[1:0] has been set for Multiple mode, any trigger source set enabled in the ADC_TRG register triggers a start of conversion immediately when its triggering conditions are met, regardless of the sequence setting.

15.5.2 ADCn_CSCR: ADC n Current Sequence and Channel Register

The ADCn_CSCR displays the ADC's current sequence and channel. It is comprised of current sequence number bits and current active channel number bits. The CSEQN[2:0] bits in the ADCn_CSCR register enables the user to instantly set the current sequence number.

Before setting this register, users must set the ADEN bit in the ADCn_MR register.

ADC0_CSCR=0x4000_B004, ADC1_CSCR=0x4000_B104, ADC2_CSCR=0x4000_B204

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
Reserved			CSEQN[2:0]	Reserved	CACH[4:0]
-			000	-	00000
-			RW	-	RO

10 8	CSEQN[2:0]	Current sequence number	
		000	Current sequence is 0.
		001	Current sequence is 1.
		010	Current sequence is 2.
		011	Current sequence is 3.
		100	Current sequence is 4.
		101	Current sequence is 5.
		110	Current sequence is 6.
		111	Current sequence is 7.
4 0	CACH[4:0]	Current active channel	
		00000	ADC channel 0 is active.
		00001	ADC channel 1 is active.
		00010	ADC channel 2 is active.

	00011	ADC channel 3 is active.
	00100	ADC channel 4 is active.
	00101	ADC channel 5 is active.
	00110	ADC channel 6 is active.
	00111	ADC channel 7 is active.
	01000	ADC channel 8 is active.
	01001	ADC channel 9 is active.
	01010	ADC channel 10 is active.
	01011	ADC channel 11 is active.
	01111	ADC channel 15 is active.
	10000	ADC channel 16 is active.
	10001	ADC channel 17 is active.
	10010	ADC channel 18 is active.
	10011	ADC channel 19 is active.
	10100	ADC channel 20 is active.
	10101	ADC channel 21 is active.
	10110	ADC channel 22 is active.
	10111	ADC channel 23 is active.
	Others	Reserved

15.5.3 ADCn_CCR: ADC n Clock Control Register

The ADCn_CCR register controls the clock of the ADC module. This register is 16 bits wide.

ADC0_CCR=0x4000_B008, ADC1_CCR=0x4000_B108, ADC2_CCR=0x4000_B208

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
Reserved		ADCPDA	CLKDIV[6:0]	ADCPD	EXTCLK	CLKINVT	Reserved
-		0	0000000	1	0	0	-
-		RW	RW	RW	RW	RW	-

15	ADCPDA	ADC disablement for power saving
	0	Disable (default)
	1	Reserved (Do not set to '1')
14 8	CLKDIV[6:0]	ADC clock division ratio value bit (This bit is enabled when EXTCLK = 0).
		CLKDIV = 0, ADC clock = ADC input clock (bypass) CLKDIV = 1, ADC clock = clock stop CLKDIV ≥ 2, ADC clock = ADC input clock / CLKDIV[6:0] **NOTE: If the CLKDIV[6:0] value is set to two or larger number for the ADC clock division, the ADC clock's frequency must not exceed 42 MHz.**
7	ADCPD	ADC DEEP-SLEEP
	0	ADC normal mode
	1	ADC DEEP-SLEEP mode
6	EXTCLK	ADC external clock configuration
	0	Use Internal clock (Enable CLKDIV[6:0])
	1	Reserved
5	CLKINVT	Divide clock inversion (optional)
	0	Duty ratio of the divided clock is larger than 50%.
	1	Duty ratio of the divided clock is less than 50%.

15.5.4 ADCn_TRG: ADC n Trigger Select Register

The ADCn_TRG register selects trigger sources for the ADC module.

ADC0_TRG=0x4000_B00C, ADC1_TRG=0x4000_B10C, ADC2_TRG=0x4000_B20C

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[3:0]	SEQTRG6[3:0]	SEQTRG5[3:0]	SEQTRG4[3:0]	SEQTRG3[3:0]	SEQTRG2[3:0]	SEQTRG1[3:0]	SEQTRG0[3:0] BSTTRG[3:0]
0000	0000	0000	0000	0000	0000	0000	0000
RW	RW	RW	RW	RW	RW	RW	RW

30 28	SEQTRG7[3:0]	8th sequence trigger source
27 24	SEQTRG6[3:0]	7th sequence trigger source
23 20	SEQTRG5[3:0]	6th sequence trigger source
19 16	SEQTRG4[3:0]	5th sequence trigger source
15 12	SEQTRG3[3:0]	4th sequence trigger source
11 8	SEQTRG2[3:0]	3rd sequence trigger source
7 4	SEQTRG1[3:0]	2nd sequence trigger source
3 0	SEQTRG0[3:0] BSTTRG[3:0]	1st sequence trigger source Burst conversion trigger source

NOTE:

1. In multiple mode, the 1st sequence trigger source is given the highest priority, and the 8th sequence trigger source is given the lowest priority. Table 140 shows trigger sources represented by each value (refer to section 15.3.1615.3.16).

15.5.5 ADCn_SCSR1: ADC n Channel Select 1 Register

ADCn_SCSR1 is the first register for the ADC module's selection of channels.

Each selected sequence's channel works based on the corresponding setting of the sequence's trigger source clock selected by the ADCn_TRG register. To write to this register, you must first enable the ADEN bit in the ADCn_MR register.

ADC0_SCSR1=0x4000_B018, ADC1_SCSR1=0x4000_B118, ADC2_SCSR1=0x4000_B218

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	
Reserved	SEQ3CH[4:0]	Reserved	SEQ2CH[4:0]	Reserved	SEQ1CH[4:0]	Reserved	SEQ0CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ3CH[4:0]	4th conversion sequence channel selection
20 16	SEQ2CH[4:0]	3rd conversion sequence channel selection
12 8	SEQ1CH[4:0]	2nd conversion sequence channel selection
4 0	SEQ0CH[4:0](1)	1st conversion sequence channel selection

NOTE:

1. When setting the ADC mode to Single mode, the channel must be set to the SEQ0CH[4:0].

15.5.6 ADCn_SCSR2: ADC n Channel Select 2 Register

The ADCn_SCSR2 is the first register for the ADC module's selection of channels.

Each selected sequence's channel works based on the corresponding setting of the sequence's trigger source clock selected by the ADCn_TRG register. To write to this register, users must first enable the ADEN bit in the ADCn_MR register.

ADC0_SCSR2=0x4000_B01C, ADC1_SCSR2=0x4000_B11C, ADC2_SCSR2=0x4000_B21C

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	
Reserved	SEQ7CH[4:0]	Reserved	SEQ6CH[4:0]	Reserved	SEQ5CH[4:0]	Reserved	SEQ4CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ7CH[4:0]	8th conversion sequence channel selection
20 16	SEQ6CH[4:0]	7th conversion sequence channel selection
12 8	SEQ5CH[4:0]	6th conversion sequence channel selection
4 0	SEQ4CH[4:0]	5th conversion sequence channel selection

15.5.7 ADCn_CR: ADC n Control Register

The ADCn_CR is used to control the operation of ADC module.

ADC0_CR=0x4000_B020, ADC1_CR=0x4000_B120, ADC2_CR=0x4000_B220

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
Reserved			ASTOP	Reserved	TRGCLR	ASTART
-			0	-	0	0
-			WO	-	RW	RW

7	ASTOP	ADC STOP bit
	0	No effect
	1	Stops ADC conversion (Cleared at the next ADC clock cycle) If ASTOP sets enabled after a conversion cycle starts, the current conversion is completed.
1	TRGCLR	ADC all trigger flags clear option
	0	No clear
	1	Clear all trigger flags generated during the previous ADC operation.
0	ASTART	ADC START bit
	0	No ADC conversion
	1	Starts ADC conversion (ASTART bit will be clear at the next ADC clock cycle). To start ADC conversion, the ADEN bit must be set to '1'. If ARST is '0' in trigger event mode, setting ASTART to '1' will start ADC conversion, which then will be performed as many times as set with the SEQCNT[2:0] in the ADC_MR register.

15.5.8 ADCn_SR: ADC n Status Register

The ADCn_SR register displays the status of the ADC.

ADC0_SR=0x4000_B024, ADC1_SR=0x4000_B124, ADC2_SR=0x4000_B224

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
Reserved			CMPIFLG	Reserved		DOVRUN	DMAF	TRGIF	EOSIF	Reserved	EOCIF
-			0	-		0	0	0	0	-	0
-			RWC1	-		RO	RO	RWC1	RWC1	-	RWC1

8	CMPIFLG	Compare interrupt flag
	0	The interrupt has not occurred.
	1	The interrupt has occurred (writing a '1' to the bit clears the flag).
5	DOVRUN	DMA overrun flag
	0	Flag is not detected.
	1	Flag is detected.
4	DMAF	DMA done received flag (DMA transfer is completed.)
	0	Flag is not detected.
	1	Flag is detected.
3	TRGIF	ADC trigger interrupt flag
	0	Flag is not detected.
	1	Flag is detected (writing '1' clears the flag).
2	EOSIF	Sequence end interrupt flag
	0	Flag is not detected.
	1	Flag is detected (writing '1' clears the flag).
	0	Interrupt is not generated.
0	EOCIF(1)	Sequence conversion end interrupt flag
	0	Flag is not detected.
	1	Flag is detected (writing '1' clears the flag).

NOTE:

1. To poll the flag, you must use the EOCIF bit.

   Example:

```
while ( (ADC_SR & 0x01) == 1 )       // EOCIF flag checking
{
   ADC_SR = 0x1;  // EOCIF flag clear
}
```

   …

15.5.9 ADCn_IER: ADC n Interrupt Enable Register

The ADCn_IER register determines whether to enable the ADC interrupts.

ADC0_IER=0x4000_B028, ADC1_IER=0x4000_B128, ADC2_IER=0x4000_B228

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
Reserved				DMAIE	TRGIE	EOSIE	Reserved	EOCIE
-				0	0	0	-	0
-				RW	RW	RW	-	RW

4	DMAIE	DMA done interrupt enable
		0 Disable
		1 Enable
3	TRGIE	ADC trigger conversion interrupt enable
		0 Disable
		1 Enable
2	EOSIE(1)	ADC sequence conversion interrupt enable
		0 Disable
		1 Enable
0	EOCIE	ADC single conversion interrupt Enable sequence end interrupt enable
		0 Disable
		1 Enable

NOTE:

1. Burst mode sets the EOSIE = '1' and check the EOSIF flag in the ADC_SR register.

15.5.10 ADCn_DDR: ADCn DMA Data Register

The ADCn_DDR register manages the ADC module's DMA data. It displays the results of the ADC module's DMA conversions.

ADC0_DDR=0x4000_B02C, ADC1_DDR=0x4000_B12C, ADC2_DDR=0x4000_B22C

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
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Reserved	ADMACH[4:0]	ADDMAR[11:0] ADC DMA temporary data	Reserved
0	0	0	0	0	0	0	0	-	0x00	0x000	-
R0	R0	R0	R0	R0	R0	R0	R0	-	R0	R0	-

31 24	TRGINFOx (x = 0 to 7)	ADC trigger information - Indicates whether each trigger source has been captured until the end of conversion (EOC). - To use this bit field, you must set the TRGINFOx bit in the ADCn_MR register to '1' to be enabled. **For Multiple mode:** The lower the TRGINFOx number, the higher priority it has. If a trigger is pending due to another ongoing trigger, multiple TRGINFOx can be read as 1. (x = 0 to 7) **For Single / Sequential modes:** The bit field displays which trigger sources are pending due to another ongoing trigger. You can find out which trigger source is currently being processed by referring to the CSEQN[2:0] bits in the ADCn_CSCRx register. (x = 1 to 2)
20 16	ADMACH[4:0]	DMA ADC channel indicator To use this bit field, you must set the CHINFO bit in the ADCn_MR register to '1' to be enabled.
15 4	ADDMAR[11:0]	DMA ADC conversion result data (12-bit)

NOTE:

1. Even when DMA is inactive, data is temporarily stored in this register before stored in the designated buffer. Additionally, the register also displays which channel the data belongs to.

15.5.11 ADCn_DRx: ADC n Data 0 to 7 Register

The ADCn_DRx register displays the results of ADC conversions. There are eight of these registers.

ADC0_DR0=0x4000_B030, ADC0_DR1=0x4000_B034, ADC0_DR2=0x4000_B038, ADC0_DR3=0x4000_B03C, ADC0_DR4=0x4000_B040, ADC0_DR5=0x4000_B044, ADC0_DR6=0x4000_B048, ADC0_DR7=0x4000_B04C, ADC1_DR0=0x4000_B130, ADC1_DR1=0x4000_B134, ADC1_DR2=0x4000_B138, ADC1_DR3=0x4000_B13C ADC1_DR4=0x4000_B140, ADC1_DR5=0x4000_B144, ADC1_DR6=0x4000_B148, ADC1_DR7=0x4000_B14C, ADC2_DR0=0x4000_B230, ADC2_DR1=0x4000_B234, ADC2_DR2=0x4000_B238, ADC2_DR3=0x4000_B23C, ADC2_DR4=0x4000_B240, ADC2_DR5=0x4000_B244, ADC2_DR6=0x4000_B248, ADC2_DR7=0x4000_B24C

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
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Reserved			ACH[4:0]					ADDATA[11:0]												Reserved			
0	0	0	0	0	0	0	0	-			0x00					0x000												-			
R0	R0	R0	R0	R0	R0	R0	R0	-			R0					R0												-			

31 24	TRGINFOx (x = 0 to 7)	ADC trigger information - Indicates whether each trigger source has been captured until the end of conversion (EOC). - To use this bit field, you must set the TRGINFOx bit in the ADC_MR register to be enabled. **For Multiple mode:** The lower the TRGINFOx number, the higher priority it has. If a trigger is pending due to another ongoing trigger, multiple TRGINFOx can be read as '1'. **For Single / Sequential modes:** The bit field displays which trigger sources are pending due to another ongoing trigger. You can find out which trigger source is being currently processed by referring to the CSEQN[2:0] bits in the ADC_CSCR register.
20 16	ACH[4:0]	ADC channel indicator To use this bit field, you must set the CHINFO bit in the ADCn_MR register to '1' to be enabled.
15 4	ADDATA[11:0]	ADC conversion result data (12-bit)

15.5.12 ADCn_CMPR: ADC n Channel Comparison 0 Register

The ADCn_CMP0R register controls the comparison between the ADC channels.

ADC0_CMPR=0x4000_B070, ADC1_CMPR=0x4000_B170, ADC2_CMPR=0x4000_B270

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
Reserved	CMPIEN	CMPEN	Reserved	LTE	CCH[4:0]	CVAL[11:0]	Reserved
-	0	0	-	0	00000	0x000	-
-	RW	RW	-	RW	RW	RW	-

24	CMPIEN	Compare interrupt enable bit	
		0	Disable
		1	Enable
23	CMPEN	Compare operation enable bit	
		0	Disable
		1	Enable
21	LTE	ADC compare mode bit	
		0	Output when flag or interrupt when ADC value is higher than CVAL[11:0] value.
		1	Output when ADC value is lower than CVAL[11:0] value.
20 16	CCH[4:0]	Compare channel	
		00000	Compare channel is ADC channel 0.
		00001	Compare channel is ADC channel 1.
		00010	Compare channel is ADC channel 2.
		00011	Compare channel is ADC channel 3.
		00100	Compare channel is ADC channel 4.
		00101	Compare channel is ADC channel 5.
		00110	Compare channel is ADC channel 6.
		00111	Compare channel is ADC channel 7.
		01000	Compare channel is ADC channel 8.
		01001	Compare channel is ADC channel 9.
		01010	Compare channel is ADC channel 10.
		01011	Compare channel is ADC channel 11.
		01111	Compare channel is ADC channel 15.
		10000	Compare channel is ADC channel 16.
		10001	Compare channel is ADC channel 17.
		10010	Compare channel is ADC channel 18.
		10011	Compare channel is ADC channel 19.
		10100	Compare channel is ADC channel 20.
		10101	Compare channel is ADC channel 21.
		10110	Compare channel is ADC channel 22.
		10111	Compare channel is ADC channel 23.
		Others	Reserved
15 4	CVAL[11:0]	Compare value bit	

15.5.13 ADC Register Map Summary

Table 147. ADC Register Map Summary

Offset	Register name	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
0x00	ADCn_MR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TRGINFO	CHINFO	Res	Res	DMAEN	STSEL[4:0]					Res	SEQCNT[2:0]			ADEN	ARST	ADMOD[1:0]			TRGSEL[1:0		
	Reset value											0	0			0	0	0	0	0	0		0	0	0	0	0	0	0			0	0
0x04	ADCn_CSCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CSEQN[2:0]			Res	Res	Res	CACH[4:0]				
	Reset value																						0	0	0				0	0	0	0	0
0x08	ADCn_CCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ADCPDA	CLKDIV[6:0]							ADCPD	EXTCLK	CLKINVT	Res	Res	Res	Res	Res
	Reset value																	0	0	0	0	0	0	0	0	1	0	0					
0x0C	ADCn_TRG	SEQTRG7[3:0]				SEQTRG6[3:0]				SEQTRG5[3:0]			SEQTRG4[3:0]					SEQTRG3[3:0]				SEQTRG2[3:0]				SEQTRG1[3:0]			SEQTRG0[3:0] BSTTRG[3:0]			
	Reset value	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
0x18	ADCn_SCSR1	Res	Res	Res	SEQ3CH[4:0]					Res	Res	Res	SEQ2CH[4:0]					Res	Res	Res	SEQ1CH[4:0]					Res	Res	Res	SEQ0CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x1C	ADCn_SCSR2	Res	Res	Res	SEQ7CH[4:0]					Res	Res	Res	SEQ6CH[4:0]					Res	Res	Res	SEQ5CH[4:0]					Res	Res	Res	SEQ4CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x20	ADCn_CR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ASTOP	Res	Res	Res	Res	Res	Res	TRGCLR	ASTART
	Reset value																								0							0	0
0x24	ADCn_SR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CMPIFLG		Res	Res	DOVRUN	DMAF	TRGIF	EOSIF	Res	EOCIF
	Reset value																							0				0	0	0	0		0
0x28	ADCn_IER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMAIE	TRGIE	EOSIE	Res	EOCIE
	Reset value																												0	0	0		0
0x2C	ADCn_DDR	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Res	Res	Res	ADMACH[4:0]					ADDMAR[11:0]																
	Reset value	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				
0x30-0x4C	ADCn_DRx	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0			Res	ACH[4:0]					ADDDATA[11:0]																
	Reset value	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				
0x70	ADCn_CMPR	Res	Res	Res	Res	Res	Res	Res	CMPIEN	CMPEN	Res	LTE	CCH[4:0]					CVAL[11:0]															
	Reset value								0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

16. Random Number Generator (RNG)

16.1 RNG Introduction

The A34M420 has a built-in Random Number Generator (RNG) that operates with an RNG clock, which users can select the clock source.

The RNG generates a random number based on the set seed value. If the users read data before the random number generation is completed, an error interrupt is flagged to prevent the users from reading the false data.

16.2 20.2 RNG Main Features

The RNG of A34M420 series features the followings:

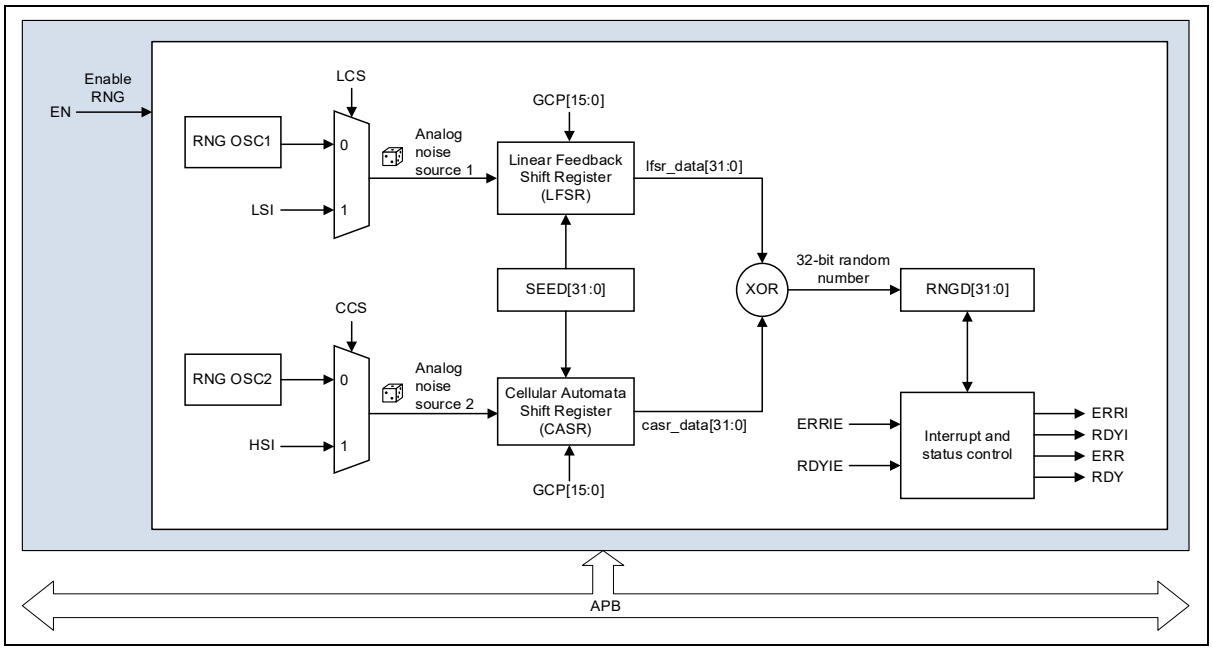
- Generation of 32-bit random number data
- Interrupt events:
  - Error interrupt
  - Ready interrupt
- Programmable random number-generating seed

16.3 RNG Functional Description

16.3.1 RNG Block Diagram

Figure 180 shows a block diagram of the RNG.

Figure 180. RNG Block Diagram



16.3.2 RNG Internal Signals

Table 148. RNG Internal Input and Output Signals

Signal Name	Signal Type	Description
rng_it	Digital output	RNG interrupt request
rng_pclk	Digital input	APB clock
internal_LFSR_ clock	Digital input	RNG dedicated clock, asynchronous to rng_pclk (Selectable between RNG OSC1 or LSI)
Internal_CASR_clock	Digital input	RNG dedicated clock, asynchronous to rng_pclk (Selectable between RNG OSC2 or HSI)

16.3.3 Random Number Generation

The 32-bit hardware RNG is based on a Linear Feedback Shift Register (LFSR) and a Cellular Automata Shift Register (CASR). Each shift register is clocked by an independent ring oscillator, and the output is sampled only when a new number is requested.

The LFSR has 43-bits, and a characteristic polynomial of $X^{43} + X^{41} + X^{20} + X + 1$. This is a primitive polynomial and gives a period length of 2^{43} - 1. The hardware RNG uses a 37-bit CASR, and its maximum length is $2^{37} – 1$.

To generate a random number, 32bits of the LFSR and CASR are selected and permuted, and then XORed together.

The ERR flag in the RNG_STAT register shows the error status indicating whether an error has occurred while generating a random number.

Users can find more information about the error status, by referring to section 16.3.8.

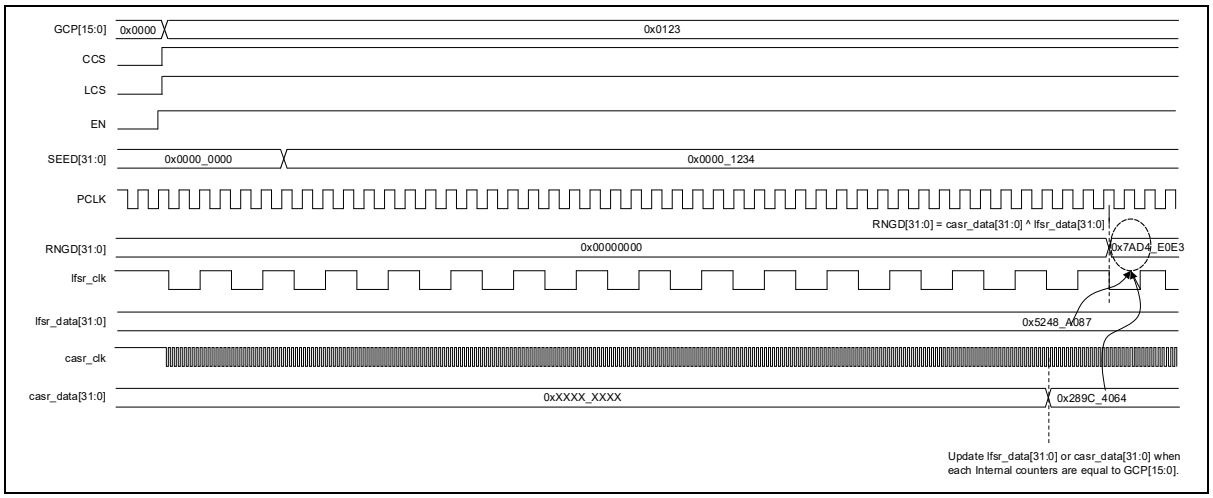
The interrupt can be generated when an error is detected.

16.3.4 RNG Initialization

Figure 181 shows that a series of events shown below occur after enabling the RNG (the EN bit in the RNG_CTRL register is set to ‘1’).

- Register settings:
  - RNG_SEED register = 0x0000_1234
  - GCP[15:0] in RNG_CTRL register = 0x0123
  - LCS in RNG_CTRL register = ‘1’ (clock selection: LSI)
  - CCS in RNG_CTRL register = ‘1’ (clock selection: HSI)

Figure 181. Event Occurrences with the EN in RNG_CTRL Register Set to ‘1’



1. By referencing the Generation Counter Parameter set in the GCP[15:0] bits in the RNG_CTRL register using the seed value set in the RNG_SEED register, the internal LFSR_DATA and CASR_DATA are generated. At this time, the RDY flag in the RNG_STAT register is set to ‘1’, and the LFSR_DATA and CASR_DATA, which are just generated, are XORed and output to the RNG_RNGD register.
2. Once the RNG value is generated, the RDY flag in the RNG_STAT register becomes ‘0’ when reading the RNG_RNGD register, and the LFSR counter, CASR counter, and the RNG_RNGD register are cleared to ‘0’.

16.3.5 RNG Operation

16.3.5.1 Normal Operations

1. To use interrupts to run the RNG, we recommend that users follow the order below:
2. Set the seed value by configuring the RNG_SEED register.
3. Set the GCP[15:0] bits in the RNG_CTRL register to determine the LFSR and CASR generation time.
4. Set the ERRIE and RDYIE bits in the RNG_CTRL register to enable interrupts. At the same time, set the EN bit in the RNG_CTRL register to '1' to activate the RNG.
5. Once steps 1, 2, and 3 are set, an interrupt is generated when a random number is ready, or an error occurs. Therefore, check the following for each interrupt.
   A. No error occurred. The ERRI flag in the RNG_STAT register is set to '0'.
   B. A random number is ready. The RDYI flag in the RNG_STAT register is set to '1'.
      i. If the two conditions (step 1 and step 2) are true, the RNG_RNGD register can be read to obtain the random number generated by the RNG.
      ii. If one or both two conditions (step 1 and step 2) are false, the RNG_RNGD register must not be read.
      iii. Refer to section 16.3.8.

The RNG in Polling mode requires the sequence below for execution:

1. Set the seed value by configuring the RNG_SEED register.
2. Set the GCP[15:0] bits in the RNG_CTRL register to determine the LFSR and CASR generation time.
3. Set the EN bit in the RNG_CTRL register to '1' to activate the RNG.
4. Read the RNG_STAT register and check the following.
   A. No error occurred. The ERR flag in the RNG_STAT register is set to '0'.
   B. A random number is ready. The RDY flag in the RNG_STAT bit is set to '1'.
      i. If the two conditions (step 1 and 2) are true, the RNG_RNGD register can be read to obtain the random number generated by the RNG.
      ii. If one or both two conditions (step 1 and 2) are false, the RNG_RNGD register must not be read.
      iii. Refer to 16.3.8 Error management to run the error recovery sequence in the event of an error.

NOTE:

1. If data is not ready (RDY = '0'), the RNG_RNGD register returns '0'.

16.3.5.2 Low-Power Operations

When entering DEEP-SLEEP mode, the RNG goes under Low-Power mode and the EN bit in the RNG_CTRL register is set to ‘0’, which stops the RNG_STAT and RNG_RGND registers, the RNG counter, and the internal RNG OSC1 and RNG OSC2 outputs.

When the DEEP-SLEEP mode is released by the Wake-up signal, the EN bit in the RNG_CTRL register is set back to ‘1’ and the RNG resumes operation.

16.3.6 RNG Clocking

The RNG operates on two different clocks: the APB clock and the dedicated RNG clock. The APB clock is used to configure each component of the APB Bank Register such as the RNG_CTRL, RNG_SEED, RNG_RNGD, and RNG_STAT registers. The dedicated RNG clock is used for noise source sampling.

The RNG module can be reset by disabling and enabling again the RNG bit in the SCU_PER2 and SCU_PCER2 register, respectively.

16.3.7 RNG Processing Time

RNG processing time varies based on the GCP[15:0] bits in the RNG_CTRL register value. The RDY flag in the RNG_STAT register is set to ‘1’ only when the internal LFSR counter and CASR counter match to the value set in the GCP[15:0] bits in the RNG_CTRL register field to generate two match signals.

The LFSR clock and CASR clock can be set by configuring the LCS and CCS bits in the RNG_CTRL register.

16.3.8 Error Management

If the RNG_RNGD is read before the RDY flag in the RNG_STAT register set to ‘1’, the ERR flag in the RNG_STAT register becomes ‘1’ to indicate the RNG error detection. In this case, if the ERRIE bit in the RNG_CTRL register is enabled, the interrupt is generated to detect the RNG error as soon as the ERRI flag in the RNG_STAT register is set to ‘1’.

16.3.9 RNG Low-Power Consumption in RUN Mode

If power consumption is important in the overall operation, users can deactivate the RNG by setting the EN bit in the RNG_CTRL register to ‘0’ when the RDY flag in the RNG_STAT register is ‘1’.

When the RNG is deactivated, analog seed generators and all logics using the RNG clock are disabled.

16.4 RNG Interrupts

The RNG generates interrupts from the events shown below:

- Error interrupt
- Ready interrupt

Table 149 shows the RNG interrupt requests. The RNG interrupts can be used with dedicated interrupt enable control bits.

Table 149. RNG Interrupt Requests

Interrupt Source	Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
RNG	Error flag	ERRI	ERRIE	Write '1' to ERRI
	RNGD ready flag	RDYI	RDYIE	Write '1' to RDYI

Users can independently enable or disable each interrupt source by changing the ERRIE and RDYIE bits in the RNG_CTRL register. The status of individual interrupt sources can be read from the RNG_STAT register.

NOTE:

1. The interrupts are generated only if the RNG is enabled.

16.5 RNG Registers

The RNG generates interrupts from the events shown below:

The base address and register map of the AES module are described in the following tables:

Table 150. Base Address of RNG

Name	Base Address
RNG	0x4000_0A00

Table 151. RNG Register Map

Name	Offset	Type	Description	Reset Value	Reference
RNG_CTRL	0x0000	RW	RNG Control Register	0xFFFF_0000	16.5.1
RNG_SEED	0x0004	RW	RNG Seed Register	0x0000_0000	16.5.2
RNG_RNGD	0x0008	RO	RNG Random Number Data Register	0x0000_0000	16.5.3
RNG_STAT	0x000C	RW	RNG Status Register	0x0000_0000	16.5.4

16.5.1 RNG_CTRL: RNG Control Register

The RNG_CTRL register is used to control the RNG. It is 32 bits wide.

RNG_CTRL = 0X4000_0A00

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
GCP[15:0]		CCS	LCS	Reserved	ERRIE	RDYIE	Reserved	EN
0xFFFF		0	0	-	0	0	-	0
RW		RW	RW	-	RW	RW	-	RW

31 16	GCP[15:0]	Generation Counter Parameter Determines linear feedback shift register (LFSR) and cellular automata shift register (CASR) generation time.
15	CCS	CASR clock selection 0 RNG OSC2 1 HSI
14	LCS	LFSR clock selection 0 RNG OSC1 1 LSI
9	ERRIE	Enable the error interrupt. (Occurs when RNG_RNGD is read while its value is not ready) 0 Disable 1 Enable
8	RDYIE	Enable the RNGD ready interrupt. 0 Disable 1 Enable
0	EN	Enable the RNG. 0 Disable 1 Enable

16.5.2 RNG_SEED: RNG Seed Register

The RNG_SEED register is used to set the RNG seed. It is 32 bits wide.

RNG_SEED=0x4000_0A04

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
SEED[31:0]			
0x0000_0000			
RW			

31 0	SEED[31:0]	RNG seed setting bit Sets the RNG seed.

16.5.3 RNG_RNGD: RNG Random Number Data Register

The RNG_RNGD register displays the number data that has been generated by the RNG. It is 32 bits wide.

RNG_RNGD=0x4000_0A08

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
RNGD[31:0]			
0x0000_0000			
RO			

31 0	RNGD[31:0]	Random number generation data bit Displays the random number data that has been generated by the RNG.

16.5.4 RNG_STAT: RNG Status Register

The RNG_STAT register displays the operating status of the RNG. It is 32 bits wide.

RNG_STAT=0x4000_0A0C

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
Reserved			ERRI	RDYI	Reserved	ERR	RDY
-			0	0	-	0	0
-			RWC1	RWC1	-	RWC1	RO

9	ERRI	Error interrupt flag
	0	Not flagged.
	1	Flagged (writing a ‘1’ to the bit clears the flag).
8	RDYI	RNGD ready interrupt flag
	0	Not flagged.
	1	Flagged (writing a ‘1’ to the bit clears the flag).
1	ERR	Error status
	0	The error has not occurred.
	1	The error has occurred (writing a ‘1’ to the bit clears the flag).
0	RDY	RNGD ready status
	0	Not ready
	1	Ready

16.5.5 RNG Register Map Summary

Table 152. RNG Register Map Summary

Offset	Register name	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
0x00	RNG_CTRL	GCP[15:0]																CCS	LCS	Res	Res	Res	Res	ERRIE	RDYIE	Res	Res	Res	Res	Res	Res	Res	EN
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0					0	0								0
0x04	RNG_SEED	SEED[31:0]																															
	Reset value	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
0x08	RNG_RNGD	RNGD[31:0]																															
	Reset value	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
0x0C	RNG_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ERRI	RDYI	Res	Res	Res	Res	Res	Res	ERR	RDY
	Reset value																							0	0							0	0

17. Cyclic Redundancy Check (CRC) Calculation Module

17.1 CRC Introduction

A Cyclic Redundancy Check (CRC) module is used to get 32-bit, 16-bit IBM, 8-bit and 7-bit CRC codes from 8-bit or 32-bit data size. Application programs employ CRC-based technologies to examine the integrity of data transfers, storages, and Flash memories in conformance with functional safety standards.

Using this module, it is possible to examine the integrity of transfer data and flash memory data.

Specifically, the CRC module of the A34M420 supports DMA.

17.2 CRC Main Features

The CRC of the A34M420 series features the followings:

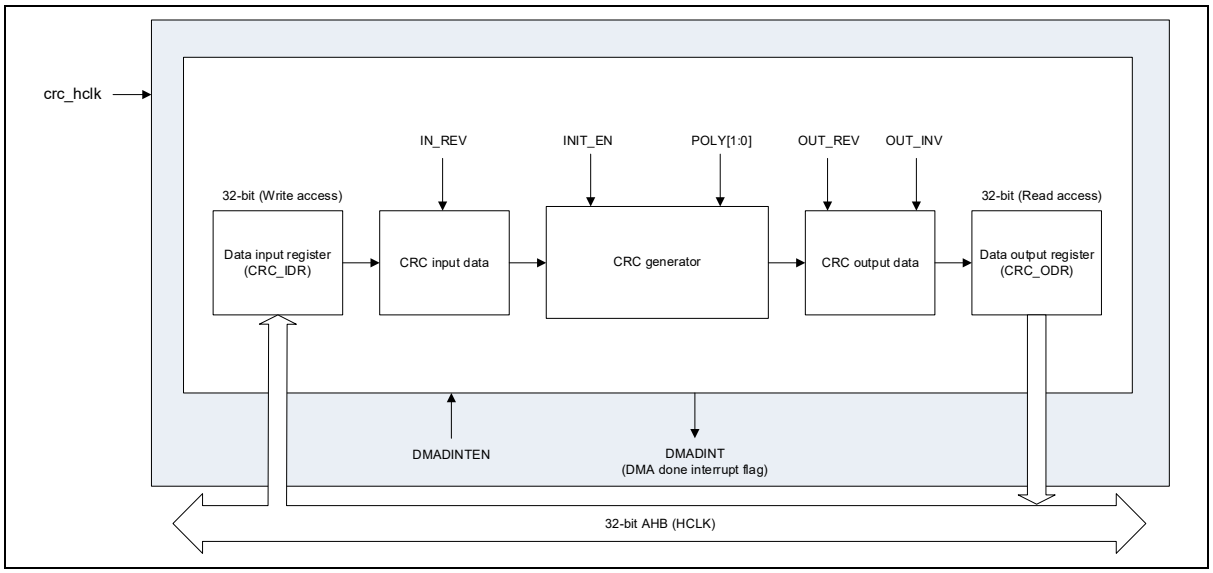
- Handles 8-, 32-bit data size
- Input buffer to avoid bus stall during calculation
- CRC calculation automatically done for word/byte input data size
  - Four AHB clock cycles (HCLK) for 32-bit
  - Two AHB clock cycles (HCLK) for 16-bit
  - One AHB clock cycle (HCLK) for 8-bit
- CRC code selectable by POLY[1:0]
  - CRC-32 (0x04C1_1DB7) when POLY[1:0] is set to ‘0’.
  - CRC-16 IBM (0x8005) when POLY[1:0] is set to ‘1’.
  - CRC-8 (0x07) when POLY[1:0] is set to ‘2’.
  - CRC-7 (0x09) when POLY[1:0] is set to ‘3’.
- Reversibility option on Input data by the IN_REV
- Reversibility option on Output data by the OUT_REV
- Invert option on Output data by the OUT_INV

17.3 CRC Functional Description

17.3.1 CRC Block Diagram

Figure 182 shows a block diagram of the A34M420 CRC module.

Figure 182. CRC Block Diagram



17.3.2 CRC Internal Signals

Table 153. CRC Internal Input/Output Signals

Signal Name	Signal Type	Description
crc_hclk	clock	AHB clock

17.3.3 Timing Diagram of CRC Internal Signals for Data Size

Figures in this section show timing diagrams of the CRC's internal operations. Each figure describes AHB clock cycles for different CRC data sizes.

Figure 183 shows a timing diagram of 8-bit size input data and CRC-7 or CRC-8 calculation (outputs). Signals in Figure 183 such as the HCLK (crc_hclk), CRC enable bit (crc_cal_en), input data (IDR[7:0]), and output data (ODR[7:0] are internal signals, and show the cycles required for internal calculation.

The CRC-7 and CRC-8 calculated values are output every 1 AHB clock cycle.

Figure 183. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-7/8 Outputs

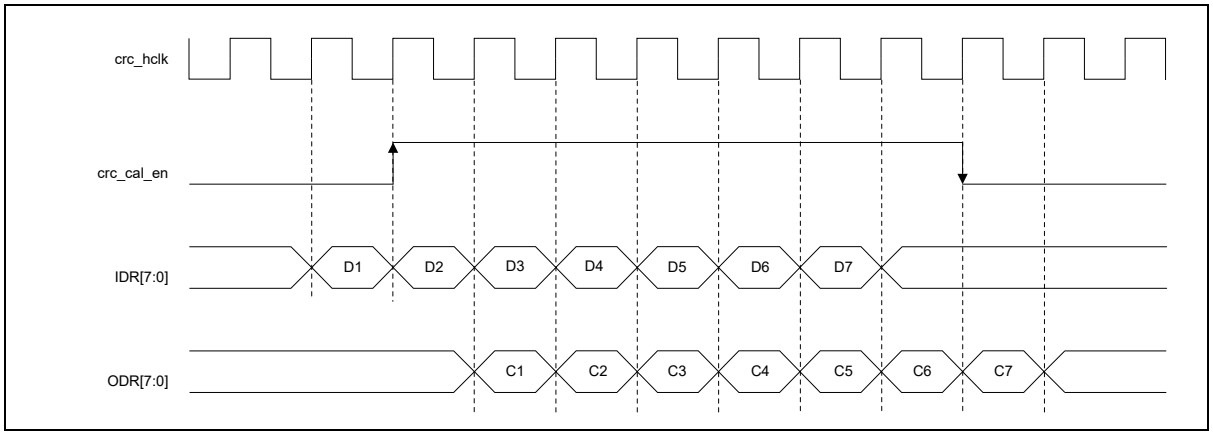


Figure 184 shows a timing diagram of 32-bit size input data and the CRC-7 and CRC-8 calculations (outputs).

Signals in Figure 184 show that the 32-bit size input data enters at once and the CRC-7 and CRC-8 calculated values are output from the point of single AHB clock cycle.

Figure 184. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-7/-8 Outputs

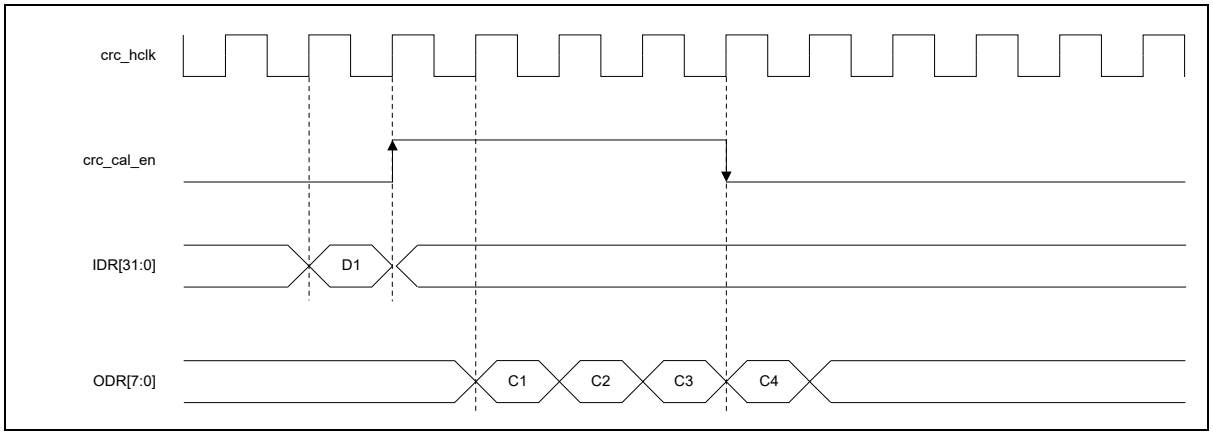


Figure 185 shows a timing diagram of 32-bit size input data and CRC-16 calculation (outputs).

Signals in Figure 185 show that the 32-bit size input data enters at once and two CRC-16 values are output. The first CRC-16 calculated value is output at the point of two AHB clock cycles and the second CRC-16 calculated in parallel is output at the final point of 2 + 2 AHB clock cycles.

Figure 185. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs

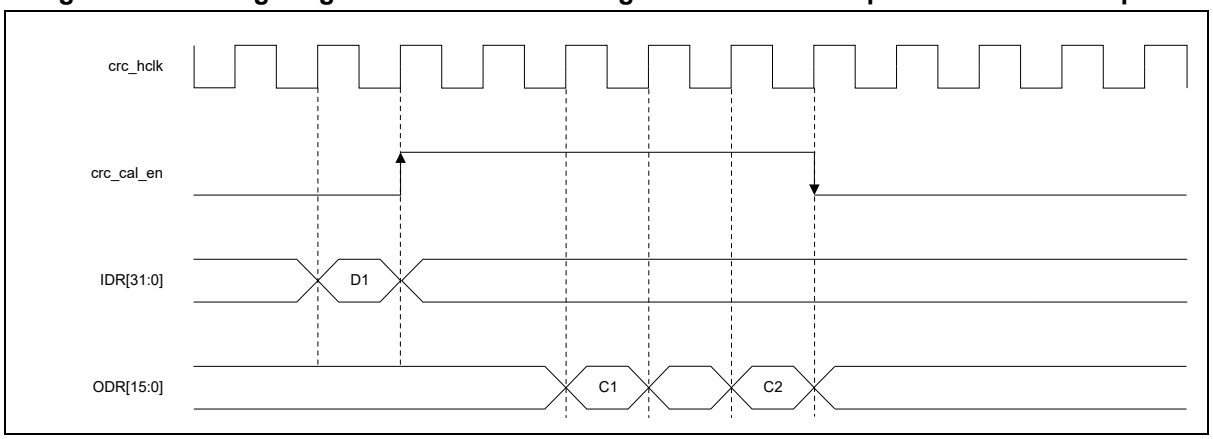
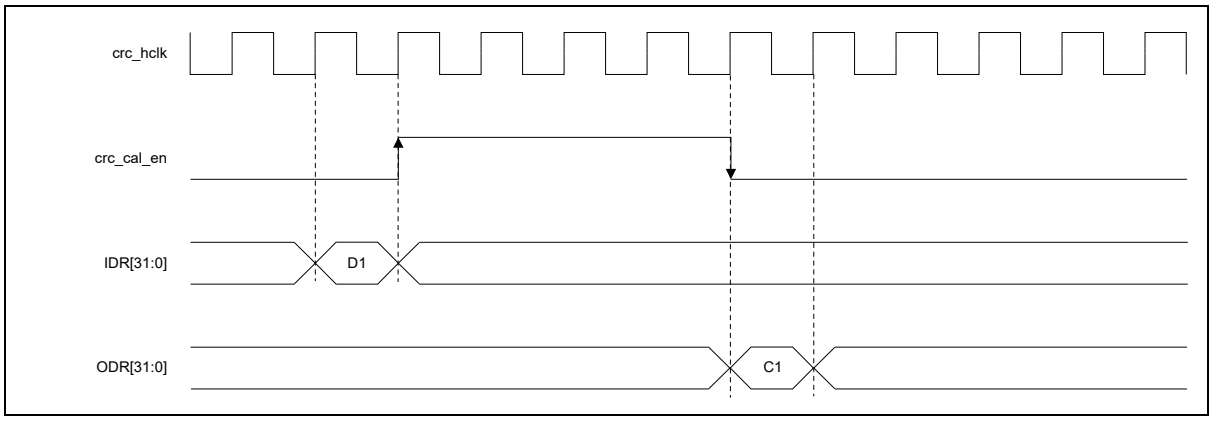


Figure 186 shows a timing diagram of 32-bit size input data and CRC-32 calculation (outputs).

Signals in Figure 186 show that the 32-bit size input data enters at once and the CRC-32 calculated values are output from the point of four AHB clock cycles.

Figure 186. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs



17.3.4 CRC Operation

The CRC calculated value can be obtained with 32-bit, 16-bit, 8-bit, 7-bit polynomial.

The CRC module must be enabled before working, so the CRC bit in the SCU_PER2 and SCU_PCER2 register must be set to ‘1’.

New input data is written to the CRC_IDR register (Input data register) and the previous CRC calculated value is read from the CRC_ODR register (output data register).

The CRC_IDR register is available for 8-bit and 32-bit size input data. For the 16-bit CRC, the CRC module receives 32-bit input data and internally processes the CRC calculation of 16-bit data two times.

17.3.4.1 How to Obtain the CRC Calculation Result

This section describes how to obtain the CRC calculated value as shown below:

1. Set each bit of the CRC_CTRL register (CRC control register) as needed.
2. Update the CRC_IDR register (Input data register) to have a new CRC calculated value in combination with the previous CRC values.
3. Read the input data from the CRC_IDR register in the order below.
    A. For the 32-bit data size, for example, 0x1122_3344 is read from the least significant bit and calculated in the order of 0x44, 0x33, 0x22, and 0x11.
    B. For the 8-bit data size, user can input 0x44, 0x33, 0x22, and 0x11 data by using byte pointer for the CRC_IDR register. These written data is read and calculated in the order of 0x44, 0x33, 0x22, and 0x11.
4. The CRC calculation rate depends on the data width shown below.
    A. Four AHB clock cycles for 32-bit
    B. Two AHB clock cycles for 16-bit
    C. One AHB clock cycles for 8-bit

The CRC initial value can be programmed in the CRC_INIT register (CRC initial data register), and its reset initial value is defined as 0xFFFF_FFFF. Before setting this register, the INIT_EN bit in the CRC_CTRL register must be enabled.

The CRC module can reverse input data or output data and control the output data inversion. The OUT_REV or IN_REV bits in the CRC_CTRL register is converted using bit reversal. For example, ‘0x1122_3344’ is converted to ‘0x22CC_4488’.

The OUT_INV bit in the CRC_CTRL register can enable the inversion of CRC output data. For example, if the CRC output data is ‘0xAA55_FF00’, it is converted to ‘0x55AA_00FF’.

If the OUT_INV bit is set to ‘1’, each byte of CRC output data is XORed with ‘0xFF’ before it is output; while if the OUT_INV bit is set to ‘0’, each byte of CRC output data is XORed with ‘0x00’ before it is output.

The CRC module uses the following polynomial coefficients:

- CRC-32 (0x04C1_1DB7) when POLY[1:0] bits is set to ‘0’.
- CRC-16 IBM (0x8005) when POLY[1:0] bits is set to ‘1’.
- CRC-8 (0x7) when POLY[1:0] bits is set to ‘2’.
- CRC-7 (0x09) when POLY[1:0] bits is set to ‘3’.

17.3.5 CRC using DMA

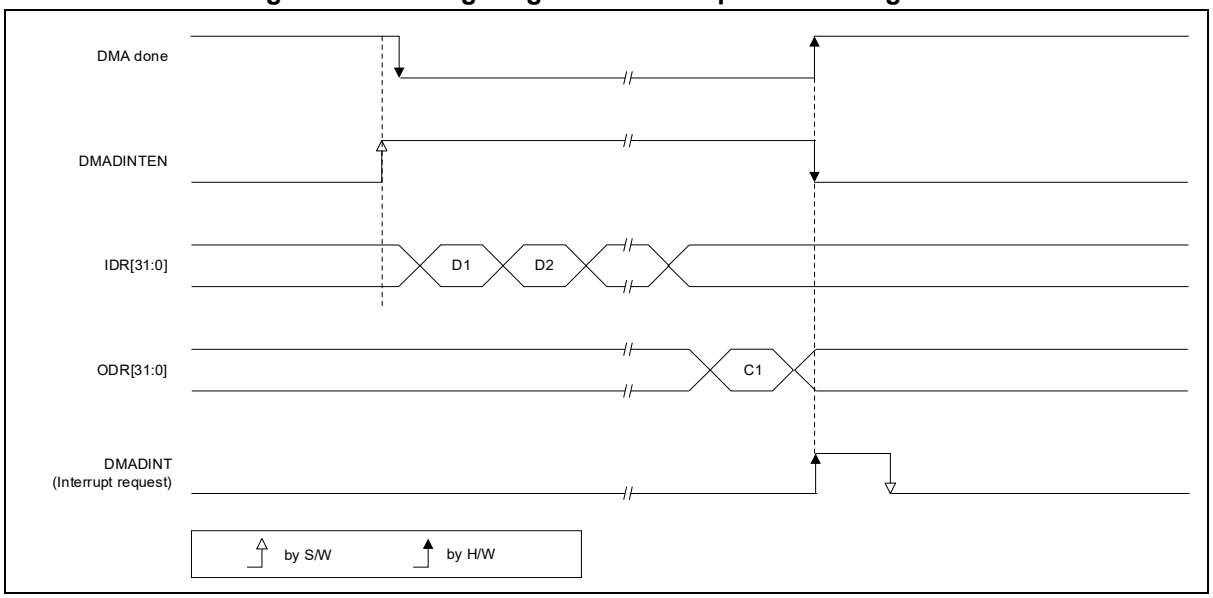
The CRC module supports DMA function.

The following procedure allows users to learn how to obtain the CRC calculated values using the DMA function. The DMA function is used with the CRC_CTRL register set to default.

1. Write CRC input data into memory.
2. Configure each bit of the DMAn_CR register by referring to the followings.
   A. The TRANSCNT[11:0] bits determine the number of DMA transfers.
   B. The PERISEL[4:0] bits select the CRC_Tx (24).
   C. The SIZE[1:0] bits select the size of a data transfer among byte and word.
   D. The DIR bit sets the direction from memory to peripheral as TX.
3. Set the DMAn_PAR register to 0x4000_2008, the address of the CRC_IDR register.
4. To the DMAn_MAR register, write the initial address of the memory to be read.
5. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR regist er to transfer the CRC value using DMA.
6. Once the DMA transfer is started and when value of the TRANSCNT[11:0] bits in the DMAn_ CR register reaches ‘0’, the EOT flag in the DMAn_SR register is enabled indicating that the DMA transfer ends.
7. For detailed information of the DMA operation, refer to chapter 7.

Figure 187 shows a timing diagram of the CRC operation using DMA.

Figure 187. Timing Diagram of CRC Operation using DMA



17.4 CRC Registers

The base address and register map of the CRC module are described in the following tables.

Table 154. Base Address of CRC

Name	Base Address
CRC	0x4100_2000

Table 155. CRC Register Map

Name	Offset	Type	Description	Reset Value	Reference
CRC_CTRL	0x0000	RW	CRC Control Register	0x0000_0000	17.4.1
CRC_INIT	0x0004	RW	CRC Initial Data Register	0xFFFF_FFFF	17.4.2
CRC_IDR	0x0008	WO	CRC Input Data Register	0x0000_0000	17.4.3
CRC_ODR	0x0008	RO	CRC Output Data Register	0xFFFF_FFFF	17.4.4
CRC_STAT	0x000C	RWC1	CRC Status Register	0x0000_0000	17.4.5

17.4.1 CRC_CTRL: CRC Control Register

The CRC_CTRL register controls the CRC calculation module.

CRC_CTRL=0x4100_2000

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
Reserved		OUT_INV	OUT_REV	Reserved	IN_REV	Reserved	DMADINTEN	Reserved	POLY[1:0]	INIT_EN
-		0	0	-	0	-	0	-	00	0
-		RW	RW	-	RW	-	RW	-	RW	WO

21	OUT_INV	Enable CRC output data inversion	
		0	Normal
		1	Inverted
20	OUT_REV	Reverse CRC output data	
		0	Not affected (LSB first)
		1	Bit-reversed output format (MSB first)
16	IN_REV	Reverse CRC input data	
		0	Not affected (LSB first)
		1	Reverse the input data. (MSB first)
8	DMADINTEN	Enable the DMA done interrupt	
		0	Disable
		1	Enable
2 1	POLY[1:0]	Polynomial selection bit	
		00	CRC32 (0x04C1_1DB7)
		01	CRC16 (0x8005)
		10	CRC8 (0x07)
		11	CRC7 (0x09)
0	INIT_EN	Whether to apply the CRC initial value register (auto clear)	
		0	No Effect
		1	Applies the CRC_INIT register value.

17.4.2 CRC_INIT: CRC Initial Data Register

The CRC initial value data is written on the CRC_INIT register.

CRC_INIT=0x4100_2004

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
INIT[31:0]			
0xFFFF_FFFF			
RW			

31 0	INIT[31:0](1)	CRC initial value

NOTE:

1. To write INIT[31:0] data to the internal CRC initial register, the INIT_EN bit in the CRC_CTRL register must be enabled. For example, writing '0x8005' to CRC_INIT register and writing a '1' to the INIT_EN bit in the CRC_CTRL register changes the internal CRC initial register to '0x8005'.

17.4.3 CRC_IDR: CRC Input Data Register

The CRC_IDR register can be accessed by 32-, 16- and 8-bit. This register is 32 bits wide.

CRC_IDR=0x4100_2008

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
IDR[31:0]			
0x0000_0000			
WO			

31 0	IDR[31:0]	CRC input data bits Once a data is written in this bit field, its polynomial result is automatically output to the CRC_ODR register.

17.4.4 CRC_ODR: CRC Output Data Register

The CRC_ODR register can be accessed by 32-, 16- and 8-bit. This register is 32 bits wide.

CRC_ODR=0x4100_2008

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
ODR[31:0]			
0xFFFF_FFFF			
RO			

31 0	ODR[31:0]	CRC output data bits

17.4.5 CRC_STAT: CRC Status Register

The CRC_STAT register displays the operating status of the CRC.

CRC_STAT = 0x4100_200C

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
Reserved			DMADINT	Reserved
-			0	-
-			RWC1	-

8	DMADINT	DMA Done Interrupt Flag Bit	
		0	DMA transfer is not done.
		1	DMA transfer is done. (Writing to the bit clears the flag.)

17.4.6 CRC Register Map Summary

Table 156. CRC Register Map Summary

Offset	Register name	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
0x00	CRC_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUT_INV	OUT_REV	Res.	Res.	Res.	IN_REV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMADINTEN	Res.	Res.	Res.	Res.	Res.	POLY[1:0]		INIT_EN
	Reset value											0	0				0								0						0	0	0
0x04	CRC_INIT	INIT[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x08	CRC_IDR	IDR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	CRC_ODR	ODR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	CRC_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMADINT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																								0								

Abbreviation for Registers

The following abbreviations are used in register descriptions:

Table 157. Abbreviation

Abbreviation	Description
RW (Read / write)	Software can read and write to this bit.
RO (Read only)	Software can only read this bit.
WO (Write-only)	Software can only write to this bit. Reading this bit returns the reset value.
RC (Read / clear)	Software can read to clear this bit.
RWC0 (Read / write 0 clear)	Software can read and clear this bit by writing ‘0’.
RWC1 (Read / write 1 clear)	Software can read and clear this bit by writing ‘1’.
Res. (Reserved)	Reserved bit, must be kept at reset value.

Glossary

This section gives a brief definition of acronyms, abbreviations, and terminology used in this document:

- Word: Data of 32-bit length
- Half-word: Data of 16-bit length
- Byte: Data of 8-bit length
- PGM: Flash Memory Programming
- ERS: Flash Memory Erasing
- AHB: Advanced High-performance Bus
- APB: Advanced Peripheral Bus
- CRC: Cyclic Redundancy Check
- DMA: Direct Memory Access
- FRT: Free-Run Timer
- HSE: High-Speed External
- HSI: High-Speed Internal
- I2C: Inter-Integrated Circuit
- LSB: Least Significant Bit
- LQFP: Low-profile Quad Flat Package
- LSE: Low-Speed External
- LSI: Low-Speed Internal
- LVI: Low-Voltage Indicator
- LVR: Low-Voltage Reset
- MPWM: Motor Pulse-Width Modulation
- MSB: Most Significant Bit
- OPAMP: Operational Amplifier
- PGM: Programming
- PLL: Phase-Locked Loop
- POR: Power-On Reset
- SCU: System Control Unit
- SPI: Serial Peripheral Interface
- UART: Universal Asynchronous Receiver Transmitter
- WDT: WatchDog Timer

Revision History

Revision	Date	Notes
1.00	Dec. 6, 2023	First release.

Korea

Regional Office, Seoul
R&D, Marketing & Sales
8th Fl., 330, Yeongdong-daero,
Gangnam-gu, Seoul,
06177, Korea

Tel: +82-2-2193-2200
Fax: +82-2-508-6903
www.abovsemi.com

HQ, Ochang
R&D, QA, and Test Center
37, Gangni 1-gil, Ochang-eup,
Cheongwon-gun,
Chungcheongbuk-do,28126, Korea

Tel: +82-43-219-5200
Fax: +82-43-217-3534
www.abovsemi.com

Domestic Sales Manager
Tel: +82-2-2193-2206
Fax: +82-2-508-6903
Email: sales_kr@abov.co.kr

Global Sales Manager
Tel: +82-2-2193-2281
Fax: +82-2-508-6903
Email: sales_gl@abov.co.kr

China Sales Manager
Tel: +86-755-8287-2205
Fax: +86-755-8287-2204
Email: sales_cn@abov.co.kr

ABOV Disclaimer

IMPORTANT NOTICE – PLEASE READ CAREFULLY

ABOV Semiconductor ("ABOV") reserves the right to make changes, corrections, enhancements, modifications, and improvements to ABOV products and/or to this document at any time without notice. ABOV does not give warranties as to the accuracy or completeness of the information included herein. Purchasers should obtain the latest relevant information of ABOV products before placing orders. Purchasers are entirely responsible for the choice, selection, and use of ABOV products and ABOV assumes no liability for application assistance or the design of purchasers' products. No license, express or implied, to any intellectual property rights is granted by ABOV herein. ABOV disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of ABOV products in such unauthorized applications. ABOV and the ABOV logo are trademarks of ABOV. All other product or service names are the property of their respective owners. Information in this document supersedes and replaces the information previously supplied in any former versions of this document.