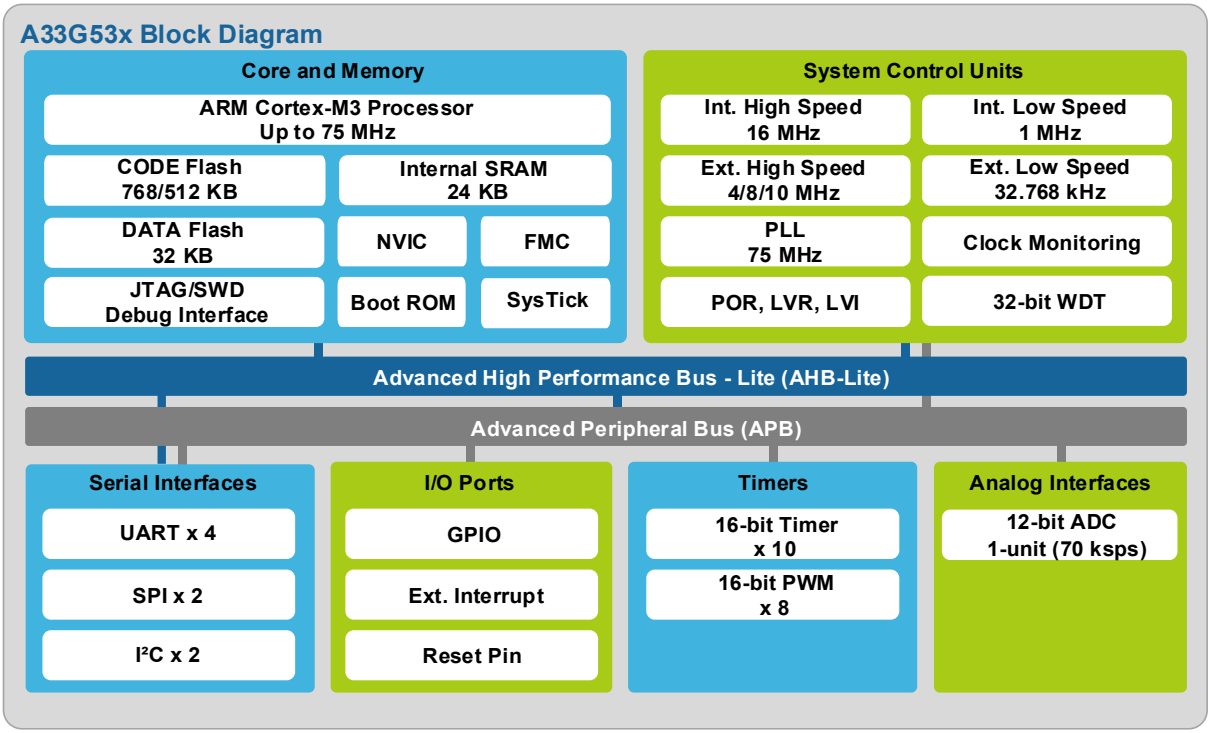
Introduction

A33G53x is a 32-bit microcontroller that is suitable for the main controller of various consumer electronics products, including the Arm's high-performance 32-bit Cortex-M3 core and various peripherals for consumer electronics. A33G53x also has dual-bank flash memory that has versatile uses. For example, two banks can be used as a unified flash memory. Otherwise, one flash bank is used as an upgrade buffer while the other bank is running a code. This feature can switch the applications in each bank with fast transition speed for the upgrade purpose. This microcontroller can also be connected to external devices via various built-in interfaces, including UART, SPI, and I2C.

Figure 1. A33G53x Block Diagram



Reference Document

- Document '**DDI0337H'** is provided by Arm and contains information on Cortex-M3.
- A33G53x Datasheet is provided by ABOV and available at https://www.abovsemi.com/.

Contents

List of Tables

List of Figures

1. Description

A33G53x series is a 32-bit high-performance microcontroller with up to 768 KB of Flash memory and 24 KB SRAM. It is a powerful microcontroller which provides effective solutions to various electrical appliances which require both power consumption and high performance (Arm Cortex-M3 core).

1.1 Product Category Definition

For the list of features per category, see Table 2).

Table 1 shows an overview of memory density versus product line. The present document describes the superset of features for each product category. (For the list of features per category, see Table 2).

Table 1. A33G53x Series Memory Density

Memory density		Category
Flash	RAM	
768 KB	24 KB	A33G539
512 KB	24 KB	A33G538

1.2 Availability of Peripherals

Table 2 summarizes and lists product specific features available in the A33G53x series considering the largest package.

For availability of peripherals and their numbers across all sale types, refer to the datasheet: **Product selection table** or **Ordering information**.

Table 2. A33G53x Series Features and Peripheral Counts

Peripherals		Description
Core	CPU	• Maximum operating frequency: 75 MHz • 32-bit ARM Cortex-M3 CPU • CPU register set: - Uses general-purpose registers specified by the 32-bit Thumb®-2 instruction set - Main stack pointer (MSP) and process stack pointer (PSP): R13 - Link register (LR): R14 - Program counter (PC): R15 • Data ordering format: Little-Endian • AHB / APB
	Interrupt	• NVIC (Nested-Vectored Interrupt Controller) • Up to 64 peripheral interrupts supported. • 3-bit width of Group Priority: 8-step priority.
System Memory	BOOT ROM	• Executes the processor's boot mode when receiving an input at the nBOOT pin from an external circuit • SPI and UART boot modes • In-system programming • A user can program data into the internal Flash memory by setting an application board.
	SRAM	• Capacity: 24 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 2. A33G53x Series Features and Peripheral Counts (continued)

Peripherals		Description
Flash Memory	Code Flash	• Capacity: - A33G539: 768 KB Code Flash memory - A33G538: 512 KB Code Flash memory • A high-capacity built-in Code Flash memory - Max 25 MHz Flash access speed - 1-word (4 bytes) programs - 512 bytes, 2 KB erases - Bulk erase • Read protection • Self-programming (Supports to update data in some Code Flash memory region during execution of user program in Code area.) • CRC code generation and verification for the Flash memory • Endurance: 10,000 cycles • Lifetime: 10 years
	Data Flash	• Capacity: 32 KB - Max 25 MHz access speed - 1 byte program - 1-word (4 bytes) programs (for use ECC) - 512 bytes, 2 KB erases • CRC code generation and verification for the Flash memory • Endurance: 100,000 Cycles • Lifetime: 10 years
Power Management Unit (PMU)	Operation Modes	• Run Mode (Run) • Sleep Mode (Idle) • Deep-Sleep Mode (Power-Down)
	Clock	• 1 MHz Internal Ring Oscillator (RINGOSC) • 16 MHz Internal Oscillator (IOSC16) • 4~10 MHz External Main Oscillator (MXOSC) • 32.768 kHz External Auxiliary Oscillator (SXOSC) • PLL Frequency synthesizer (PLL) • High frequency operation (Max. 75 MHz Output Frequency) • 1 MHz-unit fine-tuning of output frequency. • Main system clock monitoring. • External oscillation clock monitoring and non-oscillation error handling function. Frequency dividing function for main system clock (HCLK) and peripheral device module clock (PCLK).
	Reset	• Reset events are occurred by below reset sources: - Main Clock Fail

Table 2. A33G53x Series Features and Peripheral Counts (continued)

Peripherals		Description
		- External nRESET Pin - Core reset - Software Reset - POR (Power-On Reset) - LVD (Low-Voltage Detector) Reset - External main oscillation error
	LDO	• Integrated LDO (Low Drop-Out) for low-power operation
	POR	• Internal core voltage monitoring and reset signal generation
	LVD	• 8-step voltage detection level • LVD (Low Voltage Detector) reset • LVD Interrupt • Wake-up by LVD function after sleep or deep-sleep mode.
	Low-Power Consumption	• Low-power operation mode • Sleep mode (Idle) • Deep Sleep Mode (Power Down)
	Wake-up Event	• Wake-up events are occurred by below wake-up sources : - GPIOA~GPIOF - FRT - Failure of external main oscillation - WDT - LVD - Rapid wake-up operation with internal oscillator and external clock source
Port Map Controller (PMC) General Purpose I/O (GPIO)		• General purpose I/O Ports • 100-pin LQFP/MQFP - I/O pins: 90 • 80-pin MQFP - I/O pins: 71 • 64-pin LQFP - I/O pins: 60 • Configuration of pin mode - Push-Pull Output - Open-Drain - Logic Input - Analog Input • Setting pin function using MUX • High/Low Level detection and Interrupt • Rising/Falling edge detection and interrupt • Setting Pull-up/Pull-down/Debounce

Table 2. A33G53x Series Features and Peripheral Counts (continued)

Peripherals		Description
		• Large-current Port (Port D) • Separate bit set/reset function • Wake-up event by external asynchronous input
TIMER	16-bit Timer	• General-purpose 16-bit up-count timer • 10 channels • TnC: Timer input 10-ch • TnO: Timer output 10-ch • Timer operation modes - Periodic timer mode - Counter mode - PWM mode - Capture mode • Interrupt Events - Timer/Counter match interrupt - Timer overflow interrupt • Timer Input clock - MXOSC/IOSC16/SXOSC/RINGOSC • Timer input by external TnC pin in capture mode • Output timer clock signal by external TnO pin • 10-bit prescaler
	WDT	• 32-bit down-count timer • Reset event and periodic Interrupt • MXOSC/IOSC16/SXOSX/RINGOSC clock source selection • 8-step prescaler
	FRT	• 32-bit Free-run Timer • System internal time calculation • 32-bit up-count timer • Periodic interrupt mode • Occurred timer interrupt according to the time interval set by the user. • Integrated comparator for match interrupt • Overflow interrupt • 8-step prescaler
PWM Pulse-Width Modulation	PWM	• PWM generator with 8 channels • PWM signal with 16-bit independent counter • consists of 1-unit per 4 channels. • 8-bit prescaler per 1 unit • Configuration of duty and period of PWM output signal • 1/2, 1/4, 1/8, 1/16 clock divider

Table 2. A33G53x Series Features and Peripheral Counts (continued)

Peripherals		Description
		• 16-bit period and count value set • built-in channels that outputs inverted PWM signal
Serial Interface	UART	• 16550/16450 compatible asynchronous serial communication port • 4 channels • 16550- compliant device with 2-channel FIFO • 16450-compliant device with two-channel double buffer • Built-in fractional point divider to improve baud rate accuracy. • Supports UART0 channel when entering BOOT mode. • Single/Multi-Sampling of receiving data
	SPI	• 2 channels synchronous serial communication port • Double buffer structure for high-speed transmission • Master / slave selection function of communication channel • Setting function for transmission data • Number of bits (8/9/16/17 bit variable) • SPI clock speed • LSB-first or MSB first transmission • Supports SPI0 channel when entering BOOT mode.
	I2C	• Standard I2C communication specification • 2 channels • Supports master/slave per channel • 7-bit slave address • Byte-by-byte data communication by interrupt and polling type • I2C Max. transfer rate: 400kbps • Configuring I2C clock and data signal latency
12-bit A/D Converter	ADC	• 12-bit resolution • Single SAR A/D converter • Built-in Analog MUX for multiple input • Input voltage for ADC conversion - GNDV ~ AVDDV • 70kSPS A/D conversion time • Max. 15us/channel (AVDD=5.0V, A/D conversion clock=4MHz) • Interrupt of end of A/D conversion • Trigger source of A/D conversion: Timer 7-channels • A/D conversion by internal START bit or external trigger source • Channels • 100-pin: 16-ch x 1-unit • 80 / 64-pin: 10-ch x 1-unit
Operating Voltage		3.0 V to 5.5 V

Table 2. A33G53x Series Features and Peripheral Counts (continued)

Peripherals	Description
Operating temperature	Commercial grade (−40°C to +85°C)
Package (Debug interface)	Three types of package options 100-pin MQFP/LQFP (Trace / JTAG / SWD) 80-pin LQFP (JTAG / SWD) 64-pin LQFP (JTAG / SWD)

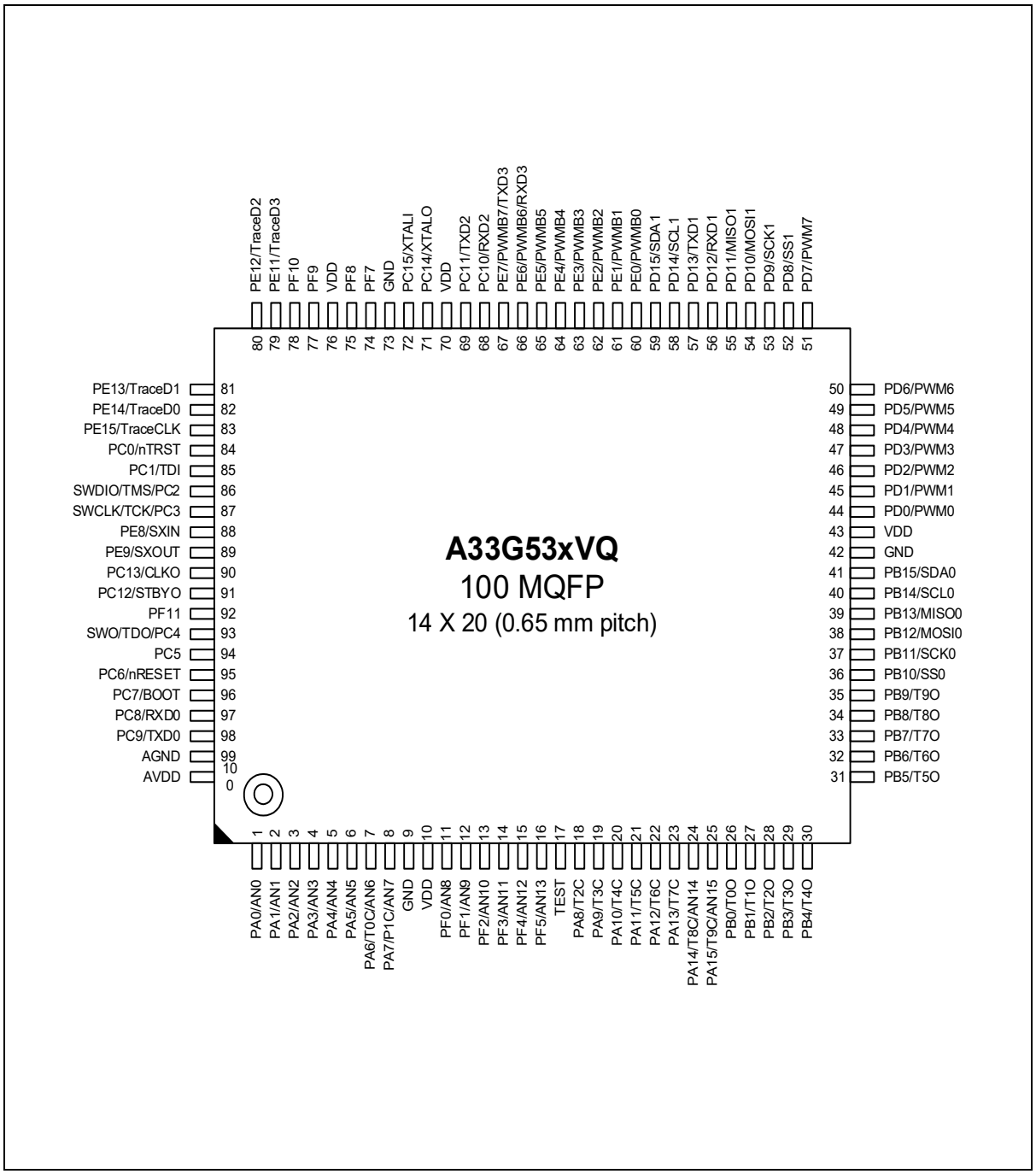
2. Pinouts and Pin Descriptions

The pinouts and pin descriptions of the A33G53x series are described in this chapter.

2.1 Pinouts

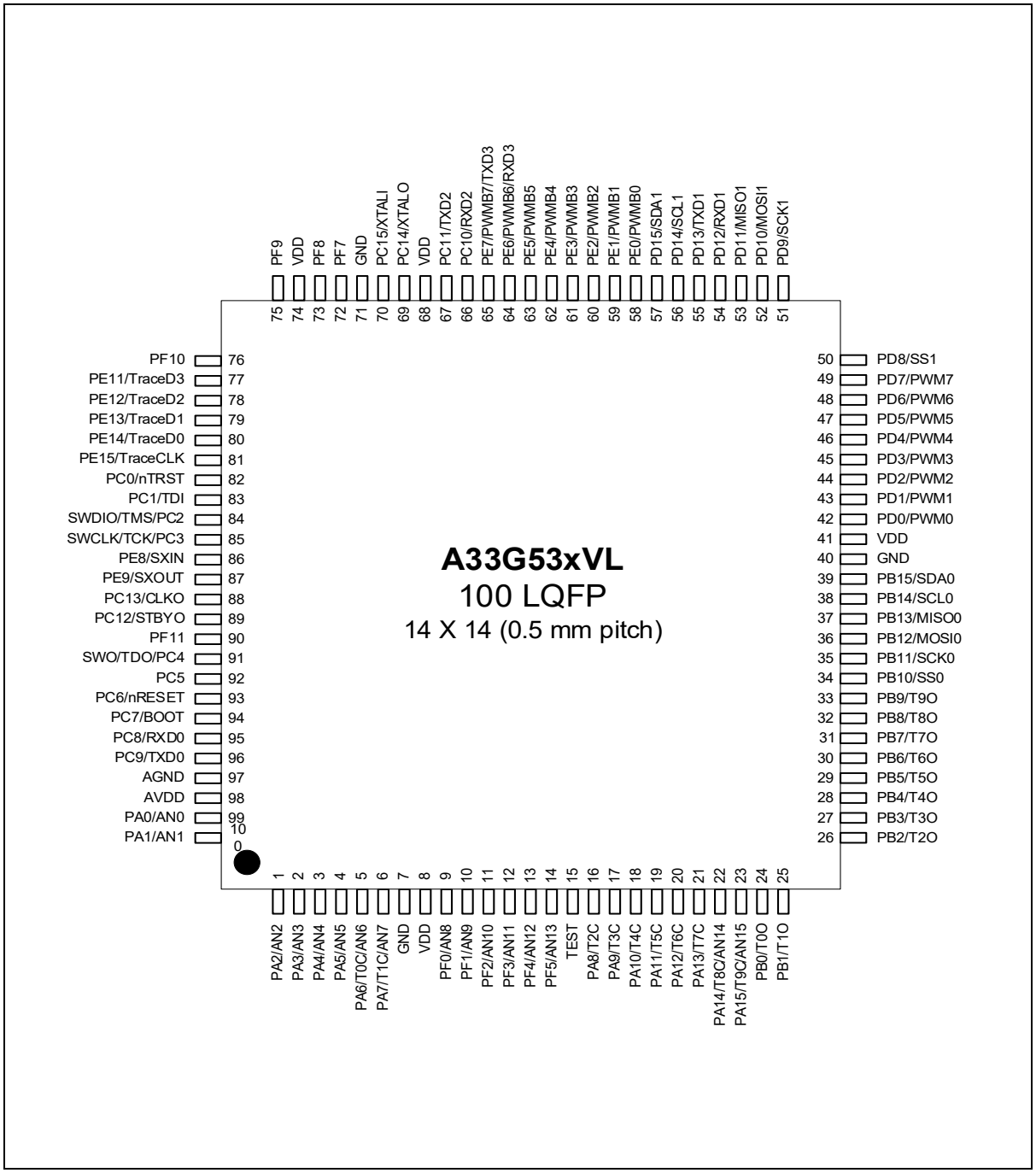
2.1.1 A33G539VQ / A33G538VQ (100 MQFP)

Figure 2. 100 MQFP Pinouts



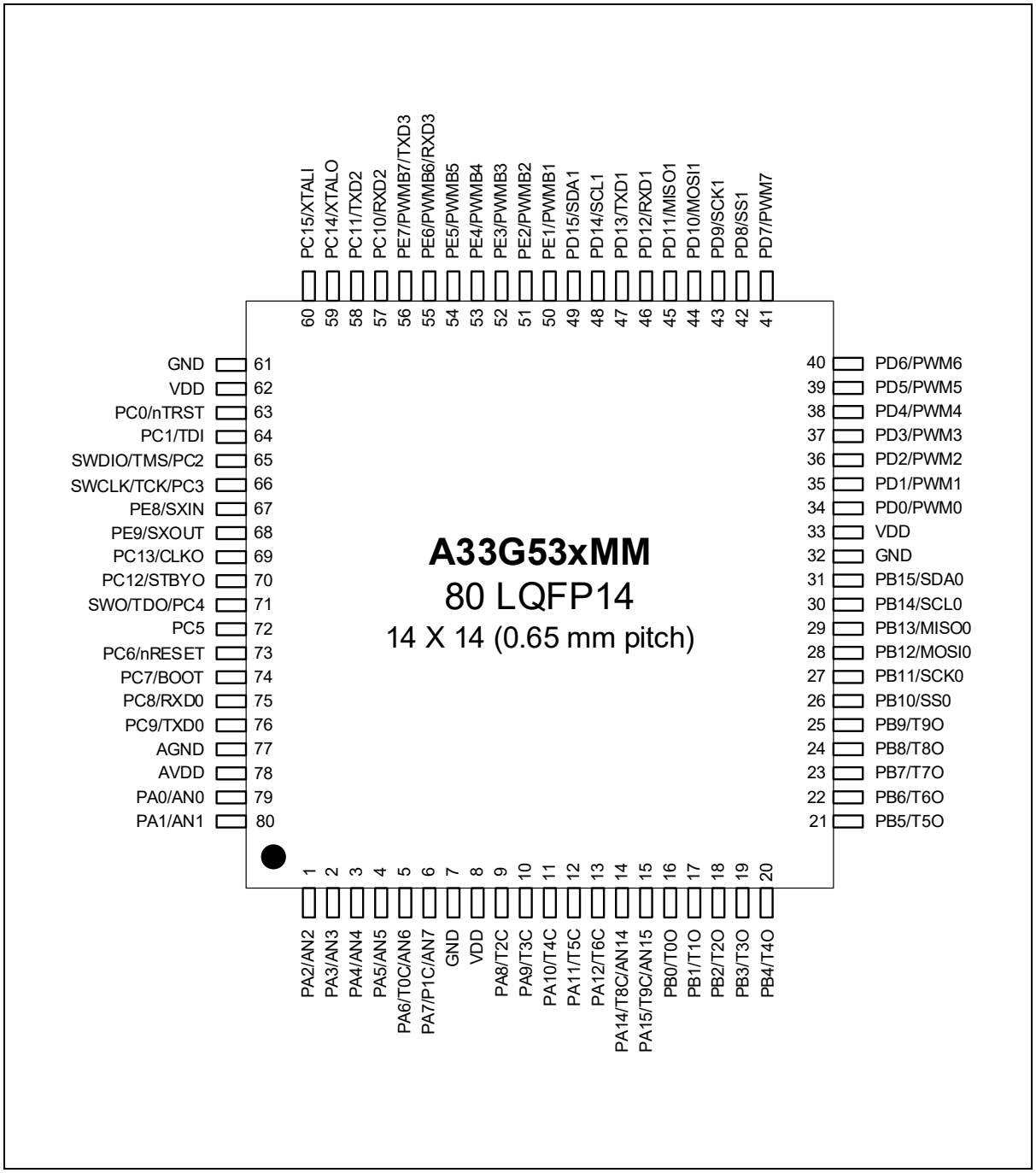
2.1.2 A33G539VL / A33G538VL (100 LQFP)

Figure 3. 100 LQFP Pinouts



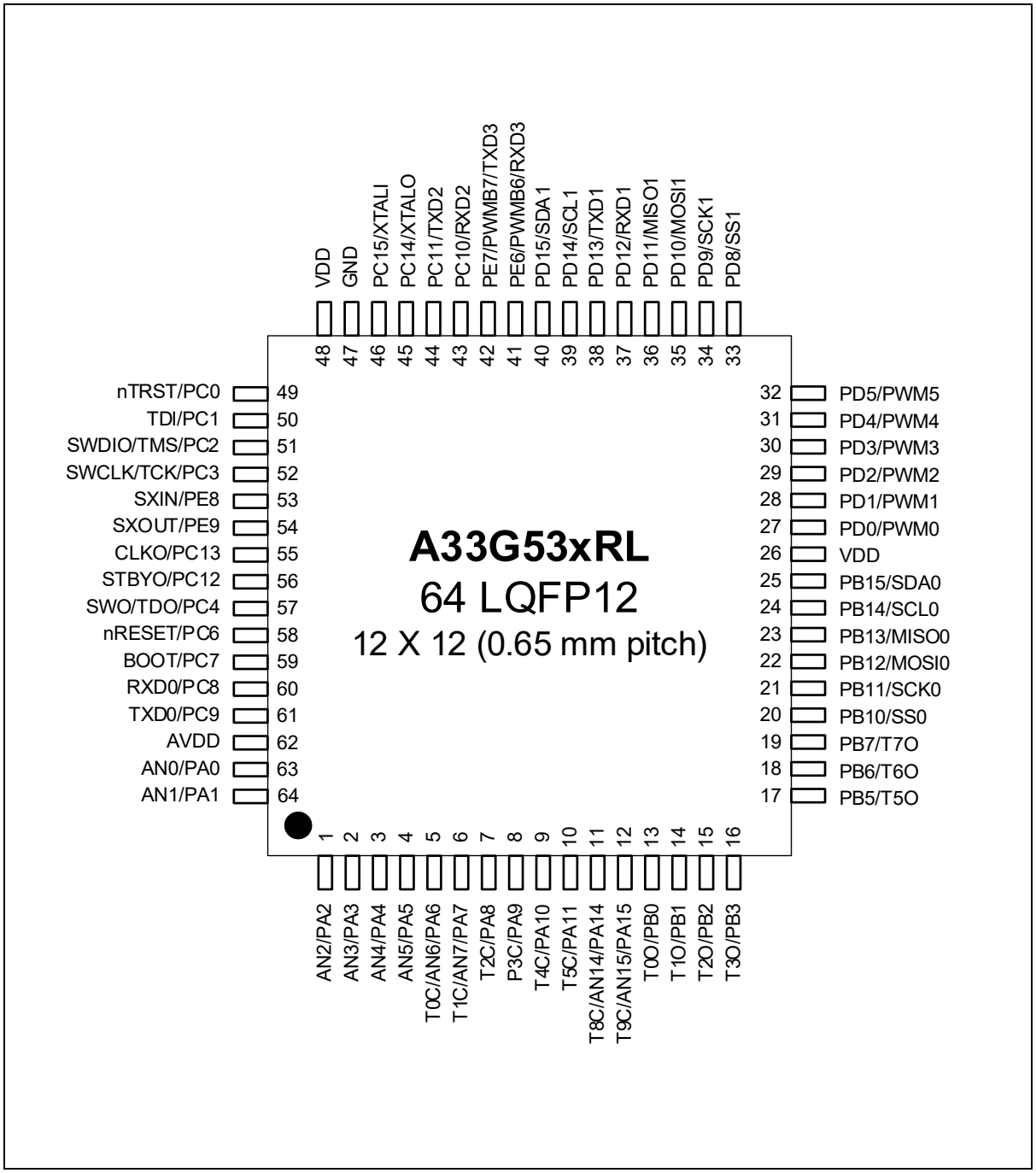
2.1.3 A33G539MM / A33G538MM (80 LQFP14)

Figure 4. 80 LQFP14 Pinouts



2.1.4 A33G539RL / A33G538RL (64 LQFP12)

Figure 5. 64 LQFP12 Pinouts



2.2 Pin Description

Pin configuration information contains two pairs of power, ground, and other dedicated pins. These multi-function pins have up to five 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
100-MQ	100-LQ	80-pin	64-pin				
99	1	79	63	PA0*	IOUS	GPIO A Bit 0 Input / Output	
				AN0	IA	Analog Input 0	
100	2	80	64	PA1*	IOUS	GPIO A Bit 1 Input / Output	
				AN1	IA	Analog Input 1	
1	3	1	1	PA2*	IOUS	GPIO A Bit 2 Input / Output	
				AN2	IA	Analog Input 2	
2	4	2	2	PA3*	IOUS	GPIO A Bit 3 Input / Output	
				AN3	IA	Analog Input 3	
3	5	3	3	PA4*	IOUS	GPIO A Bit 4 Input / Output	
				AN4	IA	Analog Input 4	
4	6	4	4	PA5*	IOUS	GPIO A Bit 5 Input / Output	
				AN5	IA	Analog Input 5	
5	7	5	5	PA6*	IOUS	GPIO A Bit 6 Input / Output	
				T0C	IPUS	Timer 0 Clock/Capture input	
				AN6	IA	Analog Input 6	
6	8	6	6	PA7*	IOUS	GPIO A Bit 7 Input / Output	
				T1C	IUS	Timer 1 Clock/Capture input	
				AN7	IA	Analog Input 7	
7	9	7	-	GND	P	Ground	
8	10	8	-	VDD	P	VDD (3.0 ~ 5.5 V)	
9	11	-	-	PF0*	IOUS	GPIO F Bit 0 Input / Output	
				AN8	IA	Analog Input 8	
10	12	-	-	PF1*	IOUS	GPIO F Bit 1 Input / Output	
				AN9	IA	Analog Input 9	
11	13	-	-	PF2*	IOUS	GPIO F Bit 2 Input / Output	
				AN10	IA	Analog Input 10	
12	14	-	-	PF3*	IOUS	GPIO F Bit 3 Input / Output	
				AN11	IA	Analog Input 11	
13	15	-	-	PF4*	IOUS	GPIO F Bit 4 Input / Output	
				AN12	IA	Analog Input 12	
14	16	-	-	PF5*	IOUS	GPIO F Bit 5 Input / Output	
				AN13	IA	Analog Input 13	
15	17	-	-	TEST	IDS	Test mode input (default Pull-down)	
16	18	9	7	PA8*	IOUS	GPIO A Bit 8 Input / Output	
				T2C	IUS	Timer 2 Clock/Capture input	
17	19	10	8	PA9*	IOUS	GPIO A Bit 9 Input / Output	
				T3C	IUS	Timer 3 Clock/Capture input	
18	20	11	9	PA10*	IOUS	GPIO A Bit 10 Input / Output	
				T4C	IUS	Timer 4 Clock/Capture input	

Table 3. Pin Description (continued)

Pin no.				Pin name	Type	Description	Remark
120-pin	100-pin	80-pin	64-pin				
19	21	12	10	PA11*	IOUS	GPIO A Bit 11 Input / Output	
				T5C	IUS	Timer 5 Clock/Capture input	
20	22	13	-	PA12*	IOUS	GPIO A Bit 12 Input / Output	
				T6C	IUS	Timer 6 Clock/Capture input	
21	23	-	-	PA13*	IOUS	GPIO A Bit 13 Input / Output	
				T7C	IUS	Timer 7 Clock/Capture input	
22	24	14	11	PA14*	IOUS	GPIO A Bit 14 Input / Output	
				T8C	IUS	Timer 8 Clock/Capture input	
				AN14	IA	Analog Input 14	
23	25	15	12	PA15*	IOUS	GPIO A Bit 15 Input / Output	
				T9C	IUS	Timer 9 Clock/Capture input	
				AN15	IA	Analog Input 14	
24	26	16	13	PB0*	IOUS	GPIO B Bit 0 Input / Output	
				T0O	OUS	Timer 0 Output	
25	27	17	14	PB1*	IOUS	GPIO B Bit 1 Input / Output	
				T1O	OUS	Timer 1 Output	
26	28	18	15	PB2*	IOUS	GPIO B Bit 2 Input / Output	
				T2O	OUS	Timer 2 Output	
27	29	19	16	PB3*	IOUS	GPIO B Bit 3 Input / Output	
				T3O	OUS	Timer 3 Output	
28	30	20	-	PB4*	IOUS	GPIO B Bit 4 Input / Output	
				T4O	OUS	Timer 4 Output	
29	31	21	17	PB5*	IOUS	GPIO B Bit 5 Input / Output	
				T5O	OUS	Timer 5 Output	
30	32	22	18	PB6*	IOUS	GPIO B Bit 6 Input / Output	
				T6O	OUS	Timer 6 Output	
31	33	23	19	PB7*	IOUS	GPIO B Bit 7 Input / Output	
				T7O	OUS	Timer 7 Output	
32	34	24	-	PB8*	IOUS	GPIO B Bit 8 Input / Output	
				T8O	OUS	Timer 8 Output	
33	35	25	-	PB9*	IOUS	GPIO B Bit 9 Input / Output	
				T9O	OUS	Timer 9 Output	
34	36	26	20	PB10*	IOUS	GPIO B Bit 10 Input / Output	
				SS0	IOUS	SPI Channel 0 Select Signal Input / Output	
35	37	27	21	PB11*	IOUS	GPIO B bit 11 Input / Output	
				SCK0	IOUS	SPI Channel 0 Clock Signal Input / Output	
36	38	28	22	PB12*	IOUS	GPIO B bit 12 Input / Output	
				MOSI0	IOUS	SPI Channel 0 Master Out/Slave In Signal	
37	39	29	23	PB13*	IOUS	GPIO B bit 13 Input / Output	
				MISO0	IOUS	SPI Channel 0 Master In/Slave Out Signal	
38	40	30	24	PB14*	IOUS	GPIO B bit 14 Input / Output	
				SCL0	OUS	I2C Channel 0 SCL Signal	

Table 3. Pin Description (continued)

Pin no.				Pin name	Type	Description	Remark
100-MQ	100-LQ	80-pin	64-pin				
39	41	31	25	PB15*	IOUS	GPIO B bit 15 Input / Output	
				SDA0	OUS	I2C Channel 0 SDA Signal	
40	42	32	-	GND	P	Ground	
41	43	33	26	VDD	P	VDD (3.0 ~ 5.5V)	
42	44	34	27	PD0*	IOUC	GPIO D bit 0 Input / Output	
				PWM0	OUC	PWM Channel 0 Output	
43	45	35	28	PD1*	IOUC	GPIO D bit 1 Input / Output	
				PWM1	OUC	PWM Channel 1 Output	
44	46	36	29	PD2*	IOUC	GPIO D bit 2 Input / Output	
				PWM2	OUC	PWM Channel 2 Output	
45	47	37	30	PD3*	IOUC	GPIO D bit 3 Input / Output	
				PWM3	OUC	PWM Channel 3 Output	
46	48	38	31	PD4*	IOUC	GPIO D bit 4 Input / Output	
				PWM4	OUC	PWM Channel 4 Output	
47	49	39	32	PD5*	IOUC	GPIO D bit 5 Input / Output	
				PWM5	OUC	PWM Channel 5 Output	
48	50	40	-	PD6*	IOUC	GPIO D bit 6 Input / Output	
				PWM6	OUC	PWM Channel 6 Output	
49	51	41	-	PD7*	IOUC	GPIO D bit 7 Input / Output	
				PWM7	OUC	PWM Channel 7 Output	
50	52	42	33	PD8*	IOUC	GPIO D bit 8 Input / Output	
				SS1	IOUC	SPI Channel 1 Select Signal Input / Output	
51	53	43	34	PD9*	IOUC	GPIO D bit 9 Input / Output	
				SCK1	IOUC	SPI Channel 1 Clock Signal Input / Output	
52	54	44	35	PD10*	IOUC	GPIO D bit 10 Input / Output	
				MOSI1	IOUC	SPI Channel 1 Master Out/Slave In Signal	
53	55	45	36	PD11*	IOUC	GPIO D bit 11 Input / Output	
				MISO1	IOUC	SPI Channel 1 Master In/Slave Out Signal	
54	56	46	37	PD12*	IOUC	GPIO D bit 12 Input / Output	
				RXD1	IUC	UART Channel 1 Receive Data Input	
55	57	47	38	PD13*	IOUC	GPIO D bit 13 Input / Output	
				TXD1	OUC	UART Channel 1 Transmit Data Output	
56	58	48	39	PD14*	IOUC	GPIO D bit 14 Input / Output	
				SCL1	OUC	I2C Channel 1 SCL Signal	
57	59	49	40	PD15*	IOUC	GPIO D bit 15 Input / Output	
				SDA1	OUC	I2C Channel 1 SDA Signal	
58	60	-	-	PE0*	IOUS	GPIO E bit 0 Input / Output	
				PWMB0	OUS	PWM Channel 0 Inversion Output	
59	61	50	-	PE1*	IOUS	GPIO E bit 1 Input / Output	
				PWMB1	OUS	PWM Channel 1 Inversion Output	
60	62	51	-	PE2*	IOUS	GPIO E bit 2 Input / Output	
				PWMB2	OUS	PWM Channel 2 Inversion Output	

Table 3. Pin Description (continued)

Pin no.				Pin name	Type	Description	Remark
120-pin	100-pin	80-pin	64-pin				
61	63	52	-	PE3*	IOUS	GPIO E bit 3 Input / Output	
				PWMB3	OUS	PWM Channel 3 Inversion Output	
62	64	53	-	PE4*	IOUS	GPIO E bit 4 Input / Output	
				PWMB4	OUS	PWM Channel 4 Inversion Output	
63	65	54	-	PE5*	IOUS	GPIO E bit 5 Input / Output	
				PWMB5	OUS	PWM Channel 5 Inversion Output	
64	66	55	41	PE6*	IOUS	GPIO E bit 6 Input / Output	
				PWMB6	OUS	PWM Channel 6 Inversion Output	
				RXD3	IUS	UART Channel 3 Receive Data Input	
65	67	56	42	PE7*	IOUS	GPIO E bit 7 Input / Output	
				PWMB7	OUS	PWM Channel 7 Inversion Output	
				TXD3	OUS	UART Channel 3 Transmit Data Output	
66	68	57	43	PC10*	IOUS	GPIO C bit 10 Input / Output	
				RXD2	IUS	UART Channel 2 Receive Data Input	
67	69	58	44	PC11*	IOUS	GPIO C bit 11 Input / Output	
				TXD2	OUS	UART Channel 2 Transmit Data Output	
68	70	-	-	VDD	P	VDD (3.0 ~5.5 V)	
69	71	59	45	PC14*	IOUS	GPIO C bit 14 Input / Output	
				XTALO	IA	Main Crystal Oscillator Output (4/8/10MHz)	
70	72	60	46	PC15*	IOUS	GPIO C bit 14 Input / Output	
				XTALI	IA	Main Crystal Oscillator Input (4/8/10MHz)	
71	73	61	47	GND	P	Ground	
72	74	-	-	PF7	IOUS	GPIO F bit 7 Input / Output	
73	75	-	-	PF8	IOUS	GPIO F bit 8 Input / Output	
74	76	62	48	VDD	P	VDD (3.0 ~5.5 V)	
75	77	-	-	PF9	IOUS	GPIO F bit 9 Input / Output	
76	78	-	-	PF10	IOUS	GPIO F bit 10 Input / Output	
77	79	-	-	PE11*	IOUS	GPIO E bit 11 Input / Output	
				TraceD3	IOUS	ETM Trace Data 3	
78	80	-	-	PE12*	IOUS	GPIO E bit 12 Input / Output	
				TraceD2	IOUS	ETM Trace Data 2	
79	81	-	-	PE13*	IOUS	GPIO E bit 13 Input / Output	
				TraceD1	IOUS	ETM Trace Data 1	
80	82	-	-	PE14*	IOUS	GPIO E bit 14 Input / Output	
				TraceD0	IOUS	ETM Trace Data 0	
81	83	-	-	PE15*	IOUS	GPIO E bit 15 Input / Output	
				TraceCLK	IOUS	ETM Trace Clock	
82	84	63	49	PC0	IOUS	GPIO C bit 0 Input / Output	
				nTRST*	IUS	JTAG nTRST Signal Input	Pull-up
83	85	64	50	PC1	IOUS	GPIO C bit 1 Input / Output	
				TDI*	IUS	JTAG TDI Signal Input	Pull-up

Table 3. Pin Description (continued)

Pin no.				Pin name	Type	Description	Remark
100-MQ	100-LQ	80-pin	64-pin				
84	86	65	51	PC2	IOUS	GPIO C bit 2 Input / Output	
				TMS / SWDIO *	IUS	JTAG TMS, SWD Data Input / Output	Pull-up
85	87	66	52	PC3	IOUS	GPIO C bit 3 Input / Output	
				TCK / SWCLK *	IUS	JTAG TCK, SWD Clock Input	Pull-up
86	88	67	53	PE8*	IOUS	GPIO E bit 8 Input / Output	
				SXIN	IA	Sub Crystal Oscillator Signal Input (32.768kHz)	
87	89	68	54	PE9*	IOUS	GPIO E bit 9 Input / Output	
				SXOUT	IA	Sub Crystal Oscillator Signal Output (32.768kHz)	
88	90	69	55	PC13*	IOUS	GPIO C bit 13 Input / Output	
				CLKO	OUS	External Clock Output	
89	91	70	56	PC12*	IOUS	GPIO C bit 12 Input / Output	
				STBYO	OUS	Stand-by(Power-down) Indication Output Signal	
90	92	-	-	PF11	IOUS	GPIO F bit 11 Input / Output	
91	93	71	57	PC4	IOUS	GPIO C bit 4 Input / Output	
				TDO / SWO *	OUS	JTAG TDO, SWO Signal Output	
92	94	72	-	PC5	IOUS	GPIO C bit 5 Input / Output	
93	95	73	58	PC6	IOUS	GPIO C bit 6 Input / Output	
				nRESET*	IUS	Reset Input	Pull-up
94	96	74	59	PC7	IOUS	GPIO C bit 7 Input / Output	
				BOOT*	IUS	Boot Signal input (input mode when reset)	
95	97	75	60	PC8	IOUS	GPIO C bit 8 Input / Output	
				RXD0*	IUS	UART Channel 0 Receive Data Input	
96	98	76	61	PC9	IOUS	GPIO C bit 9 Input / Output	
				TXD0*	OUS	UART Channel 0 Transmit Data Input	
97	99	77	-	AGND	P	ADC/Analog Ground	
98	100	78	62	AVDD	P	ADC/Analog VDD	

NOTES:

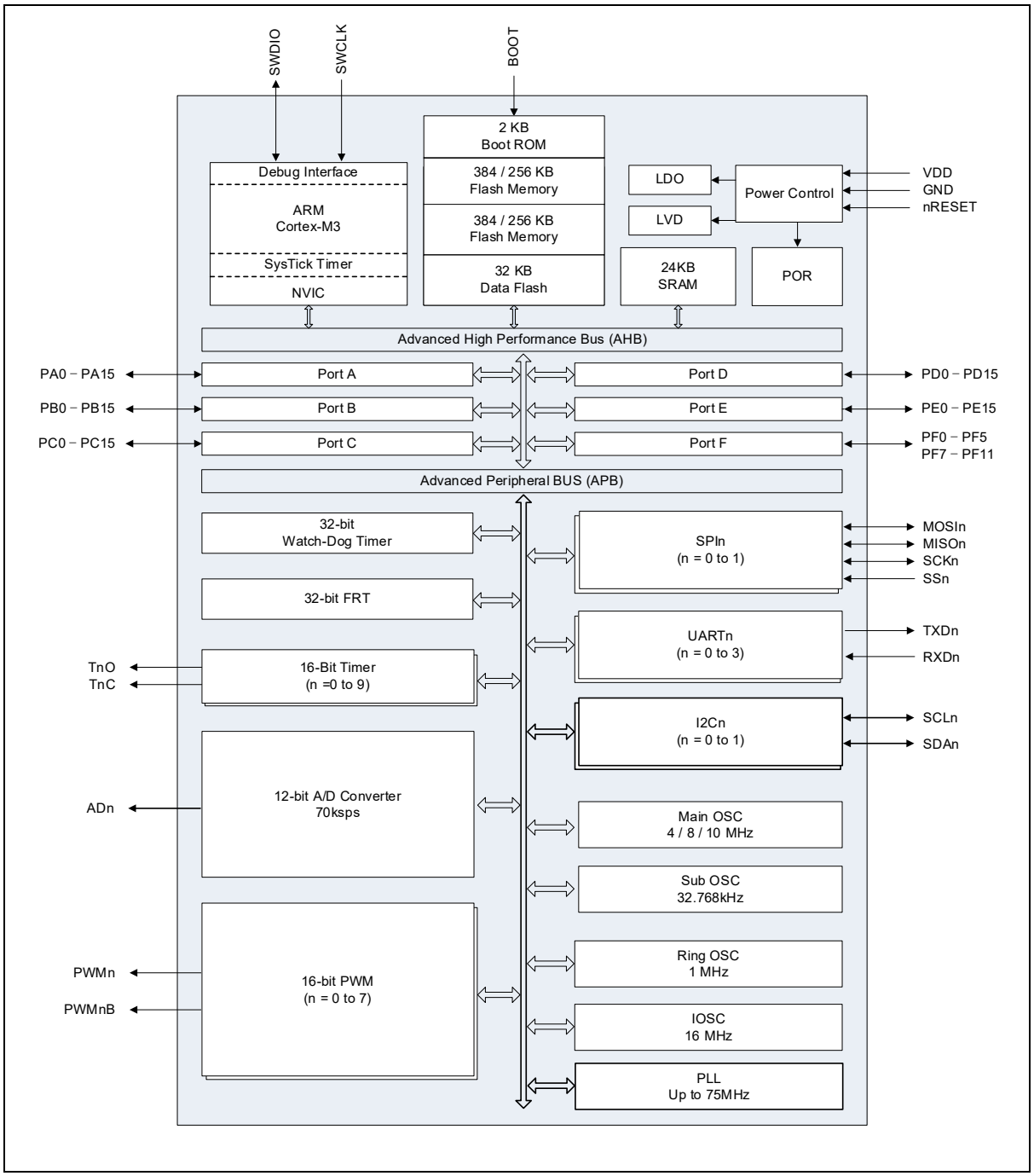
1. I = Input, O = Output, U = Pull-up, D = Pull-down, S = Schmitt-Trigger Input Type, C = CMOS Input Type, A = Analog, P = Power
2. * means 'selected pin function after reset condition', all pins except for certain pin will be Hi-Z state
3. Pin order may be changed with revision notice.
4. PC0 (nTRST), PC1 (TDI), PC2 (TMS / SWDIO), PC3(TCK / SWCLK) and PC6 (nRESET) are the default pull-up pins.
5. Do not configure unused pins as floating inputs.
6. After a reset, the boot pin is floating input. So the boot pin need an external pull down 10kohm.
7. After a reset, the internal pull-up for the serial wire clock (SWCLK) and the serial wire data I/O (SWDIO) is enabled.
8. The SWCLK and SWDIO pins should not be switched to other functions while they are being used.
9. When the PC14 (XTALO), PC15 (XTALI), PE8 (SXIN), and PE9 (SXOUT) 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.
10. TEST pin needs external pull-down (0 ~ 10 kΩ) or connect to GND.

3. System and Memory Overview

The system architecture and memory organization of the A33G53x series are described in this chapter.

3.1 System Architecture

Figure 6. System Block Diagram



3.1.1 Cortex-M3 Core

The A33G53x series is based on the Cortex-M3 core, Arm's high-performance 32-bit microprocessor. The Cortex-M3 is a processor designed for the Harvard Architecture and supports the Thumb-2 instruction set.

For detailed information on the Cortex-M3, refer to document DDI0337H provided by the Arm.

3.1.2 Interrupt Controller

A33G53x has the NVIC (Nested-Vectored Interrupt Controller) interrupt controller in the Cortex-M3 core. NVIC is designed to handle interrupts simultaneously in the CPU more effectively. The NVIC module in the Cortex-M3 makes interrupt processing faster and more efficient.

IRQ interrupts are disabled when the A33G53x system is initialized. To utilize the various interrupts supported by the A33G53x, every interrupt must be activated. Even if the interrupt is enabled in the peripheral side, the interrupts will not take effect if the IRQ interrupt is not enabled in NVIC.

The interrupt vector map of A33G53x is defined as follows:

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

Table 5. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
0	0x0000_0040	LVDFAIL
1	0x0000_0044	MXOSCFAIL
2	0x0000_0048	Reserved
3	0x0000_004C	WDT
4	0x0000_0050	FRT
5	0x0000_0054	TIMER0
6	0x0000_0058	TIMER1
7	0x0000_005C	TIMER2
8	0x0000_0060	TIMER3
9	0x0000_0064	TIMER4
10	0x0000_0068	TIMER5
11	0x0000_006C	TIMER6
12	0x0000_0070	TIMER7
13	0x0000_0074	TIMER8
14	0x0000_0078	TIMER9
15	0x0000_007C	MCKFAIL
16	0x0000_0080	GPIOA
17	0x0000_0084	GPIOB
18	0x0000_0088	GPIOC
19	0x0000_008C	GPIOD
20	0x0000_0090	GPIOE
21	0x0000_0094	GPIOF
22	0x0000_0098	Reserved
23	0x0000_009C	
24	0x0000_00A0	PWM0
25	0x0000_00A4	PWM1
26	0x0000_00A8	PWM2
27	0x0000_00AC	PWM3
28	0x0000_00B0	PWM4
29	0x0000_00B4	PWM5
30	0x0000_00B8	PWM6

Table 6. Interrupt Vector Map (continued)

Priority	Priority	Priority
31	0x0000_00BC	PWM7
32	0x0000_00C0	SPI0
33	0x0000_00C4	SPI1
34	0x0000_00C8	Reserved
35	0x0000_00CC	
36	0x0000_00D0	I2C0
37	0x0000_00D4	I2C1
38	0x0000_00D8	UART0
39	0x0000_00DC	UART1
40	0x0000_00E0	UART2
41	0x0000_00E4	UART3
42	0x0000_00E8	Reserved
43	0x0000_00EC	ADC
44	0x0000_00F0	FMC
45	0x0000_00F4	Reserved
46	0x0000_00F8	
47	0x0000_00FC	
48	0x0000_0100	
49	0x0000_0104	
50	0x0000_0108	
51	0x0000_010C	
52	0x0000_0110	
53	0x0000_0114	
54	0x0000_0118	
55	0x0000_011C	
56	0x0000_0120	
57	0x0000_0124	
58	0x0000_0128	
59	0x0000_012C	
60	0x0000_0130	
61	0x0000_0134	
62	0x0000_0138	
63	0x0000_013C	

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 is the most significant byte.

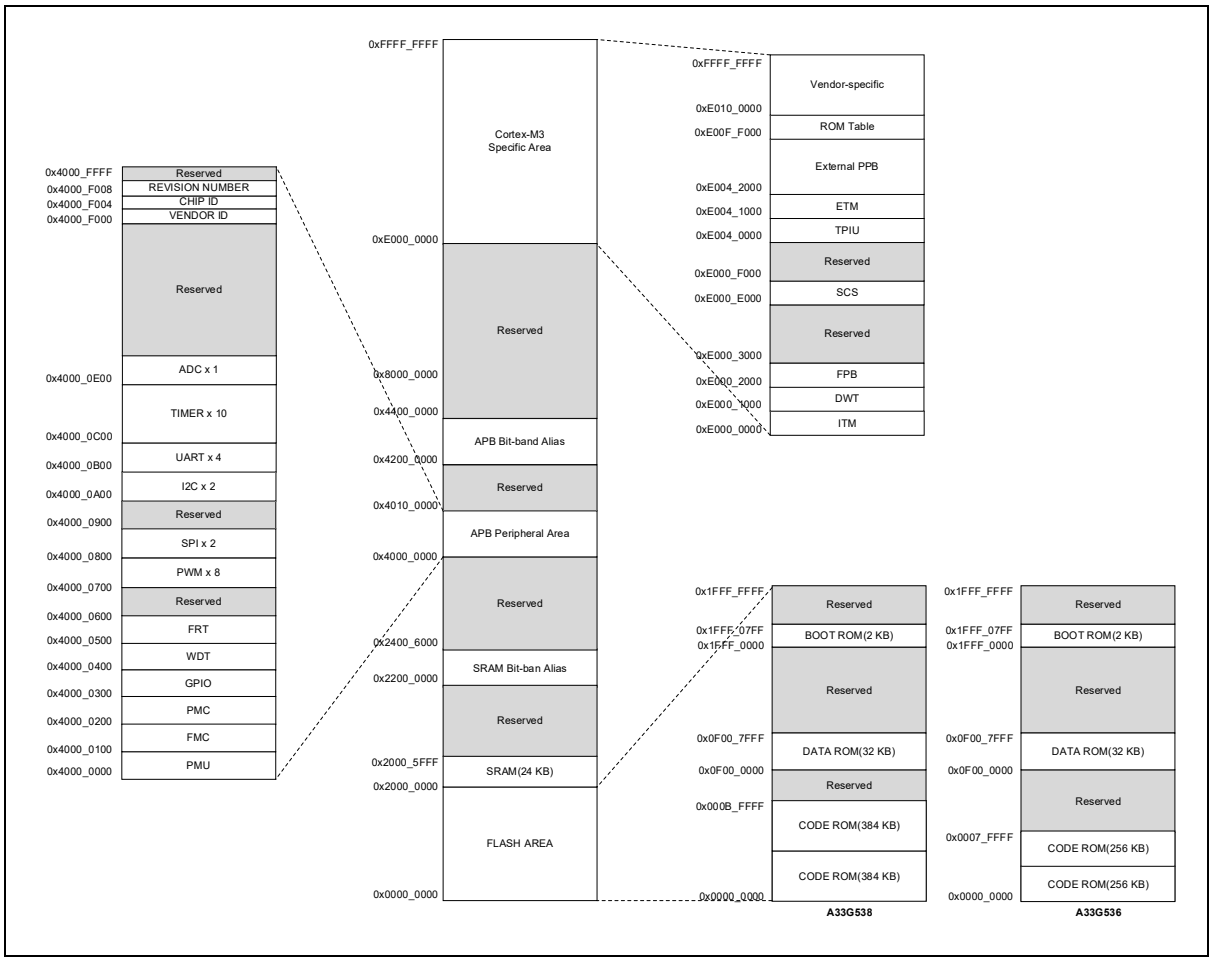
The code flash area is assigned differently depending on the device name.

- A33G539 (768 KB, 384KB + 384 KB)
- A33G538 (512 KB, 256KB + 256 KB)

3.2.2 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 memory map.

Figure 7. Memory Map



3.2.2.1 Memory Map of Internal Peripherals

The peripheral memory area of the A33G53x is assigned to each peripheral of the microcontroller. All the peripheral registers for controlling the peripheral devices are allocated in this area. For information on the peripheral registers for each peripheral device, refer to the register description section in this document.

The following table shows the memory map of the peripherals available in the devices.

Figure 8. Memory Map of Internal Peripherals

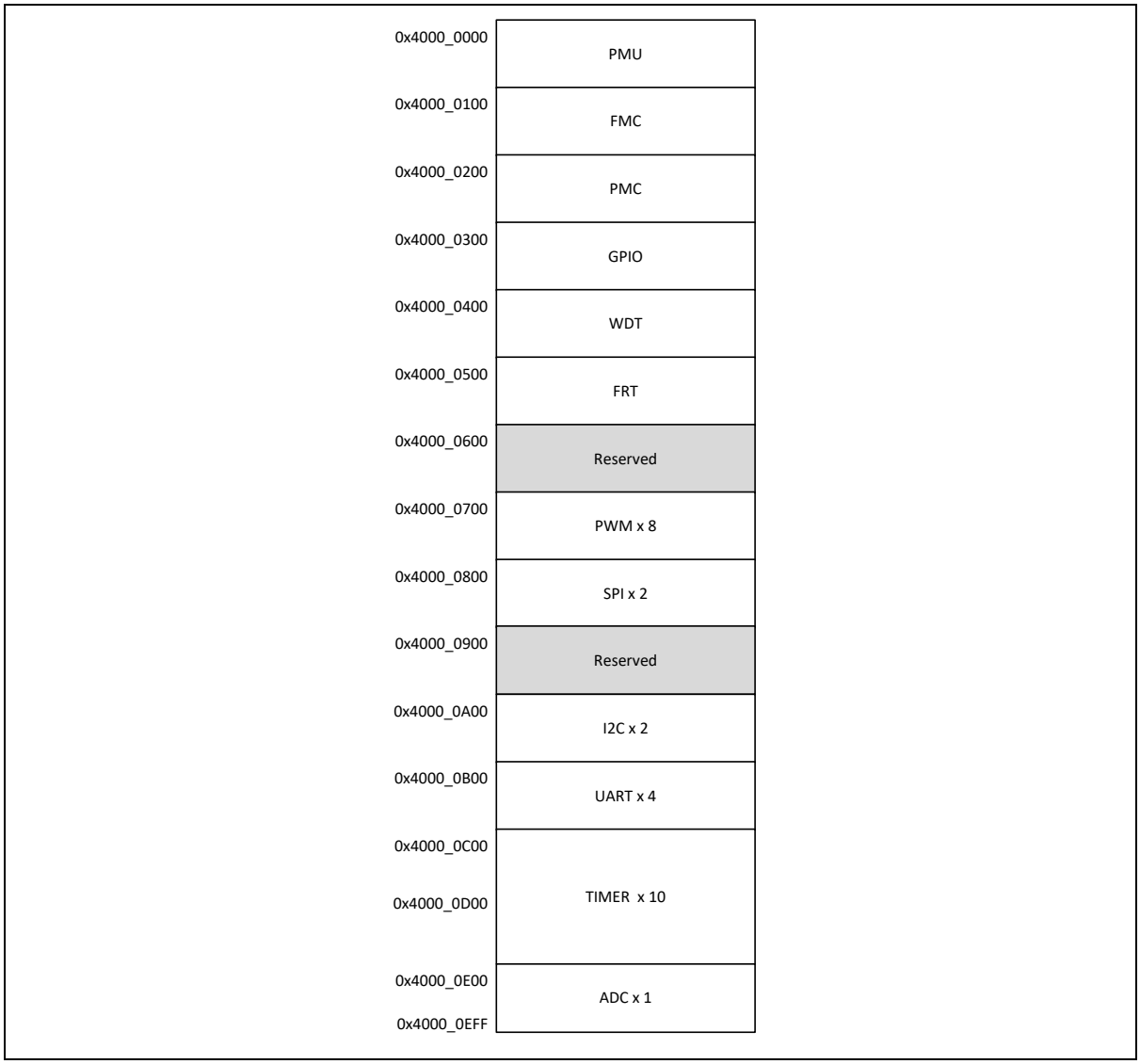


Table 6. Memory Map and Peripherals

Start Address	Peripheral
0x4000_0000	PMU
0x4000_0100	FMC
0x4000_0200	PMC
0x4000_0300	GPIO
0x4000_0400	WDT
0x4000_0500	FRT
0x4000_0700	PWM 0/1/2/3/4/5/6/7
0x4000_0800	SPI 0/1
0x4000_0A00	I2C 0/1
0x4000_0B00	UART 0/1/2/3
0x4000_0C00	TIMER 0/1/2/3/4/5/6/7/8/9
0x4000_0E00	ADC
0x2000_0000	Internal SRAM

3.2.2.2 Memory Map of Cortex-M3

The A33G53x has a fixed memory area for Cortex-M3 from 0xE000_0000 to 0xFFFF_FFFF, and the sections are divided according to the interfaces to the internal and private peripheral buses (PPB) supported by this CPU.

- Interfaces accessible to internal-only peripheral buses
  - Instrumentation Trace Macrocell (ITM)
  - Data Watchpoint and Trace (DWT)
  - Flashpatch and Breakpoint (FPB)
  - System Control Space (SCS) : Memory Protection Unit (MPU) + Nested Vectored Interrupt Controller (NVIC).
- Interfaces accessible to external dedicated peripheral buses
  - Trace Point Interface Unit (TPIU)
  - Embedded Trace Macrocell (ETM)
  - ROM table.
  - Implementation-specific areas of the PPB memory map.

Figure 9. Memory Map of Cortex-M3

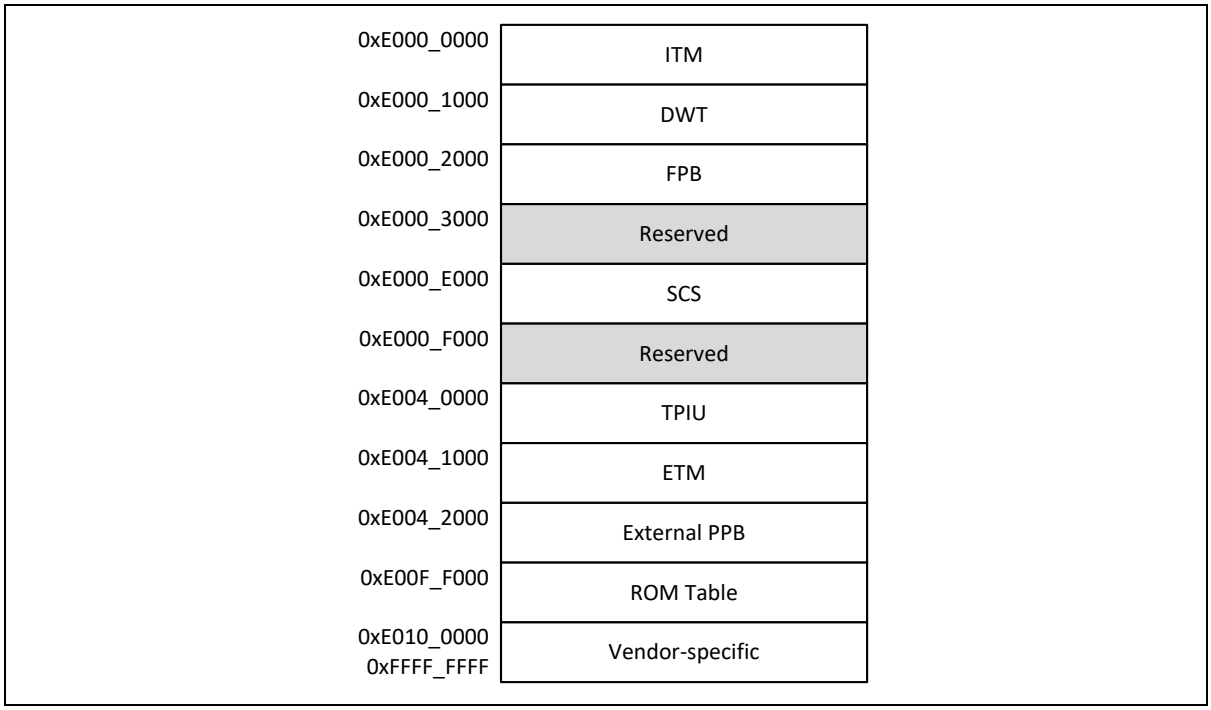


Table 7. Cortex-M3 dedicated Memory Map

Start Address	Block Name
0xE000_0000	ITM
0xE000_1000	DWT
0xE000_2000	FPB
0xE000_E000	SCS
0xE004_0000	TPIU
0xE004_1000	ETM
0xE004_2000	External PPB
0xE00F_F000	ROM Table
0xE010_0000	Vendor-specific

3.3 Memory Map

3.3.1 Embedded SRAM

The A33G53x series has a block of 0-wait on-chip SRAM. Size of the SRAM is 24 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 erase / programming operation.

3.3.2 Flash Memory Overview

The A33G538x series has an internal 512 KB / 768 KB code flash memory block and the controller suitable for controlling the general system. Self-programming is available, and ISP and SWD programming is also supported in boot mode and in debugging mode.

CPU can access Flash memory with one wait state up to 25 MHz bus frequency.

3.4 Boot Configuration

The A33G53x series has a boot mode option to program internal flash memory. Boot mode can be entered by setting BOOT pin to ‘H’ at reset timing (Normal state is ‘L’).

Boot mode supports both of UART boot and SPI boot:

- UART boot uses TXD0 / RXD0 pins
- SPI boot uses MOSI0 / MISO0 / SCK0 / SS0 pins

The boot mode uses the pins listed in Table 8.

Table 8. Boot Mode Pin List

Block	Pin name	Pin direction	Description
SYSTEM	nRESET/PC6	Input	Reset input signal
	BOOT/PC7	Input	Boot mode setting pin
UART0	RXD0/PC8	Input	UART boot receive data
	TXD0/PC9	Output	UART boot transmit data
SPI0	SS0/PB10	Input	SPI boot slave select
	SCK0/PB11	Input	SPI boot clock input
	MOSI0/PB12	Input	SPI boot data input
	MISO0/PB13	Output	SPI boot data output

4. Power Management Unit (PMU)

4.1 PMU Introduction

PMU, the Power Management Unit of A33G53x microcontroller, can control and monitor the source clock of the chip, and set system operation speed. In addition, the power consumption of the application can be controlled by setting the operation mode of the microcontroller. There is also a function to prevent malfunction of user application by setting reset.

It also provides the function to enable the peripherals' clocks and devices in the Power Management Unit (PMU) block to enable control of each peripheral device in the microcontroller.

Table 9. Operation Summary

Item	Pin Name	Remark
Clock usage	Main XTAL Sub XTAL RingOSC IOSC16 HCLK PCLK	Clock settings
Reset Source	External Reset Soft-Reset LVD (Low Voltage Detector) Clock Fail	PMU_CFGR LVD_CON PMU_CMR
Reset Generation	External Reset Soft-Reset LVD Clock Fail	PMU_CFGR LVD_CON PMU_CMR
Interrupt Generation	LVD (IRQ0) MXOSCFAIL (IRQ1)	LVD_CON PMU_CMR
Interrupt Clear Method	Writing '1' to the interrupt bit	PMU_CMR, LVD_CON

4.2 PMU Main Features

The features of A33G53x series are as follows:

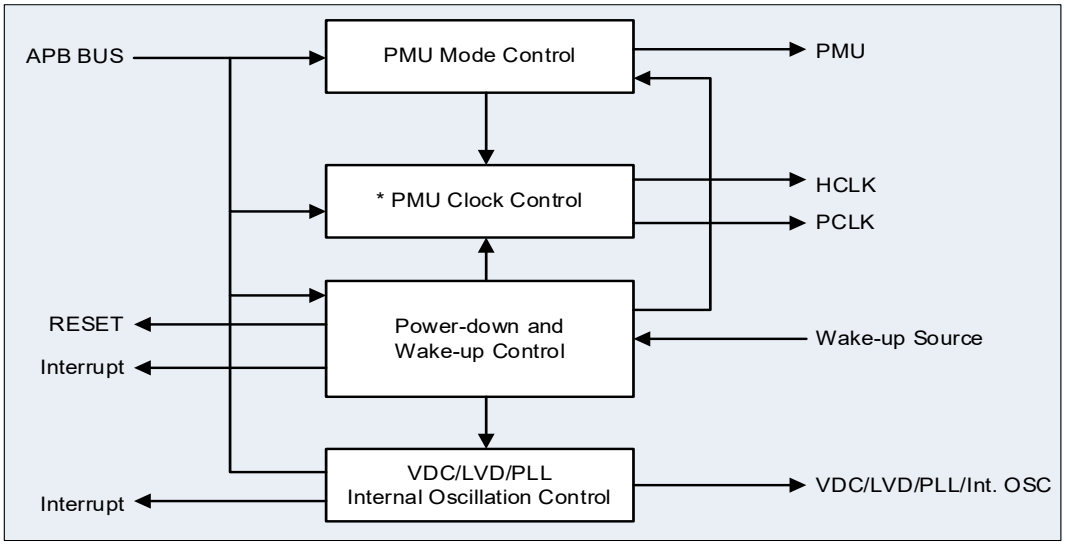
- Operating mode for low power consumption
  - Run mode (Run)
  - Sleep mode (Idle)
  - Deep-sleep mode (Power-Down)
- Input clock sources for microcontroller operation
  - 1 MHz internal ring oscillator (RINGOSC)
  - 16 MHz internal oscillator (IOSC16)
  - 4 / 8 / 10 MHz external main oscillator (MXOSC)
  - 32.768 kHz external auxiliary oscillator (SXOSC)
- PLL frequency output
  - High-speed clock driving up to 75 MHz frequency output
  - Fine adjustment in 1 MHz unit
- Clock monitoring and non-oscillation error check
  - Main clock and external main oscillator clock monitoring
  - No-oscillation error handling function
- Reset source and wake-up event
- Others
  - LDO (Low DropOut Regulator)
  - POR (Power-On Reset)
  - LVD (Low-Level Detector)

4.3 PMU Functional Description

4.3.1 PMU Block Diagram

Figure 10 shows the PMU block diagram.

Figure 10. PMU Block Diagram



NOTE: * For details on the SCU Clock Control, see section 4.4.2 Clock Tree Configuration.

4.3.2 PMU Pins and Type

The Power Management Unit (PMU) of the A33G53x has input / output pins that can be set.

The input signals that can be set in the PMU include reset input (nRESET), main oscillator input (XTALI / XTALO), and auxiliary oscillator input (SXIN / SXOUT).

The output signals that can be set by the PMU include the STBYO pin indicating the power down state and the CLKO pin for dividing the internal PLL or main clock.

Table 10. PMU Pins

Pin name	Type	Description
nRESET	Input	Initialize the entire chip. This pin is Schmitt-Trigger type input.
XTALI/XTALO	Analog	Main crystal oscillator pin
SXIN/SXOUT	Analog	32 kHz crystal sub-oscillator pin
STBYO	Output	Indicator pin of Stand-by mode
CLKO	Output	Clock output pin of internal PLL

4.4 Clocks

4.4.1 Overview

The ring oscillator (RINGOSC) of about 1 MHz and a 16 MHz RC oscillator (IOSC16) are built in the A33G53x microcontrollers. The external main crystal (MXOSC) oscillation can be used as a clock input, and the clock can be used to 75 MHz using the internal PLL. In addition, the external clock 32.768 kHz crystal (SXOSC) oscillation as an auxiliary clock, more time-critical periodic time calculation is possible.

The clock sources of the A33G53x series are described in Table 8.

Table 11. Clock Sources

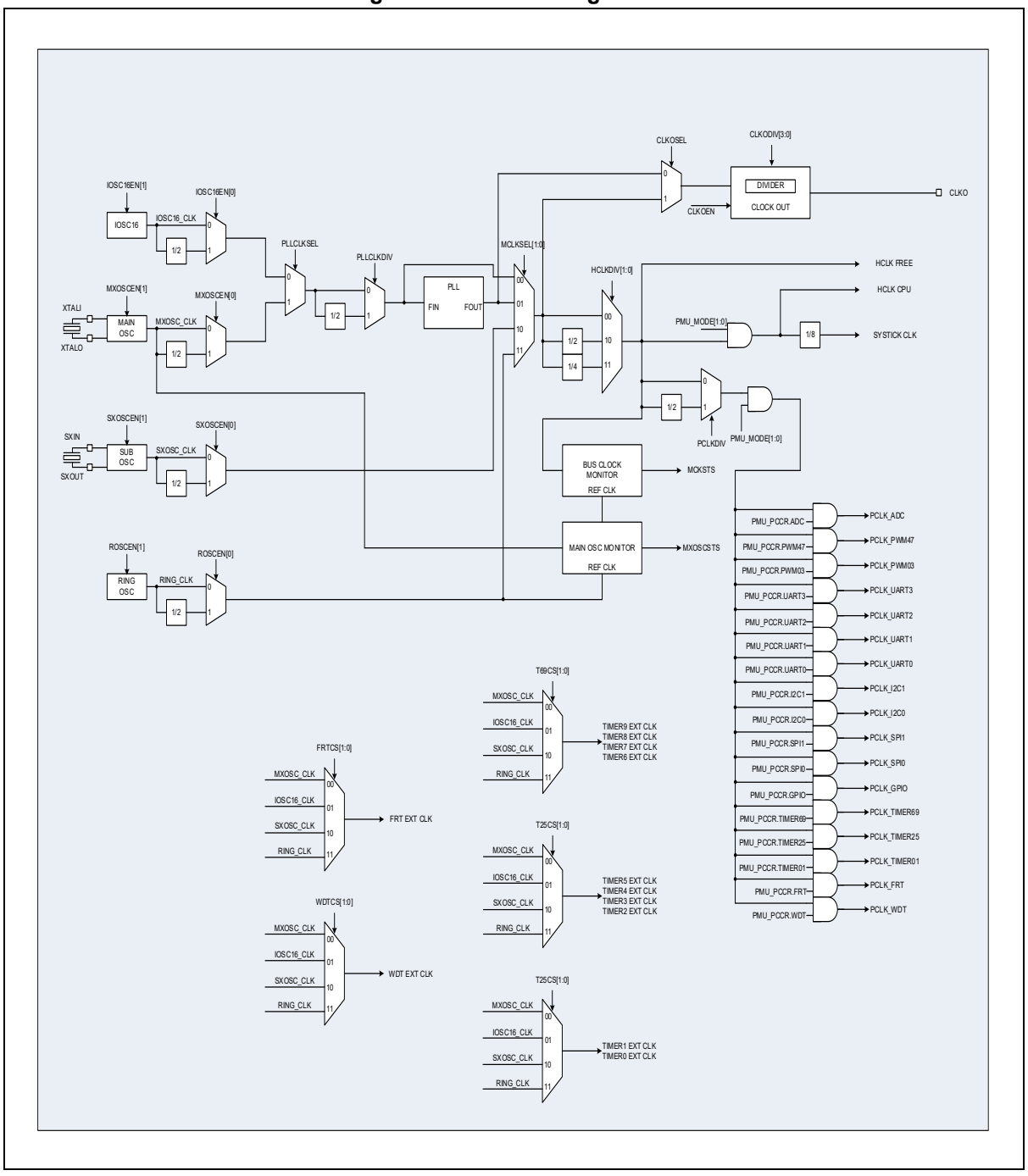
Clock name	Frequency	Description
RINGOSC	1 MHz (±50%)	Clock monitoring for internal system Used for WDT, Clock monitoring (Large variations in temperature and voltage)
IOSC16	16 MHz (±3%)	Internal main clock (This clock can be used instead of XTAL)
MXOSC (MainOSC)	XTAL (4 MHz to 10 MHz)	External Main clock
SXOSC (SubOSC)	SXTAL (32.768 kHz)	External auxiliary clock (for real time clock)
PLL	8 MHz to 75 MHz	Multiplier and divider of PLL clock High frequency output XTAL or IOSC16 input as clock source

4.4.2 Clock Tree Configuration

Users can maintain the clock system variation under software control.

Figure 11 shows the overall connection diagram of the clock system on the A33G53x device.

Figure 11. Clock Configuration



All multiplexers switching clock sources have a glitch-free circuit in each. Therefore, clocks can be switched without glitch problem. When changing a clock mux control, both of clock sources must be alive. If any of the source clocks is not alive, the clock change operation is stopped, and the system will be halted and not be recovered.

The five pins in Table 12 are assigned for the clock function.

Table 12. Clock Generation Circuit Pins

Pin Name	Type	Description
XTALI	Analog Input	External Crystal Oscillator (4 MHz to 10 MHz)
XTALO	Analog Output	
SXIN	Analog Input	External sub-Crystal Oscillator (32.768 kHz)
SXOUT	Analog Output	
CLKO	Output	Clock Output Monitoring Signal

4.4.3 MXOSC Clock

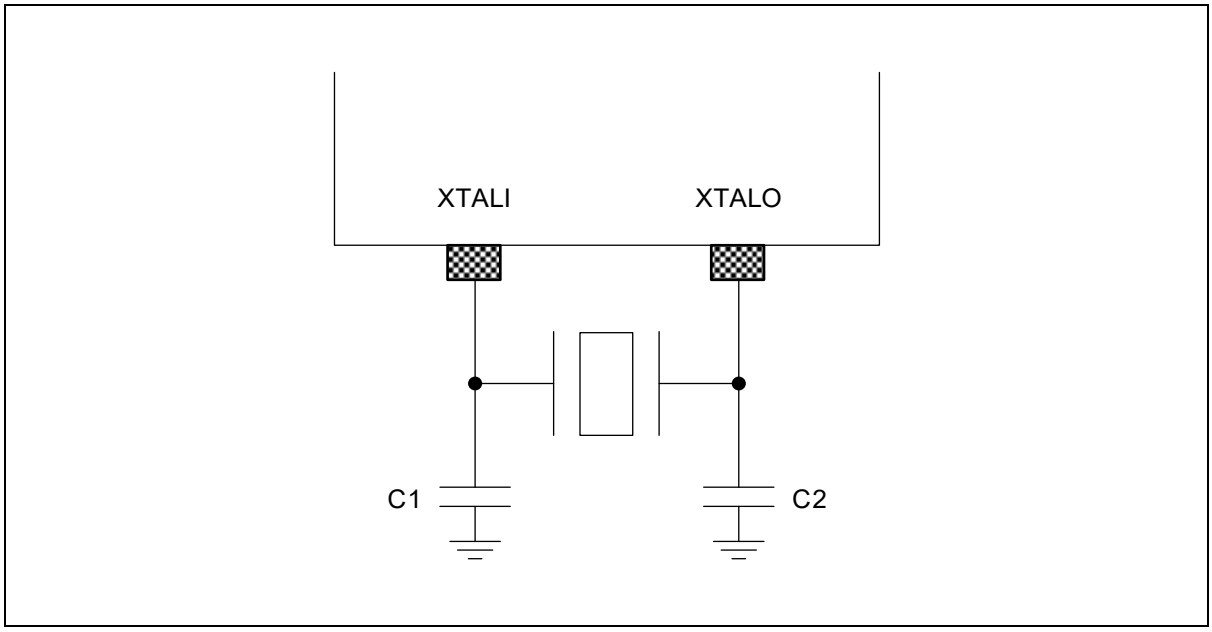
The external main oscillator clock signal (MXOSC) can be generated from two clock sources below:

XTALI and XTALO pins are assigned for the MXOSC block, connected to the external main crystal or ceramic oscillator.

Table 13. MXOSC Pins

Pin Name	Type	Description
XTALI	Analog Input	These pins are connected to external crystal or ceramic Oscillator (4 MHz to 10 MHz). For details, see Figure 12.
XTALO	Analog Output	

Figure 12. External main Crystal / Ceramic Oscillator



4.4.4 IOSC16 Clock

The internal main clock signal (IOSC16) is generated from the internal 16 MHz oscillator.

- 16 MHz (±3% at −20°C to +70°C, ±4% at −40°C to +85°C)

4.4.5 PLL Clock

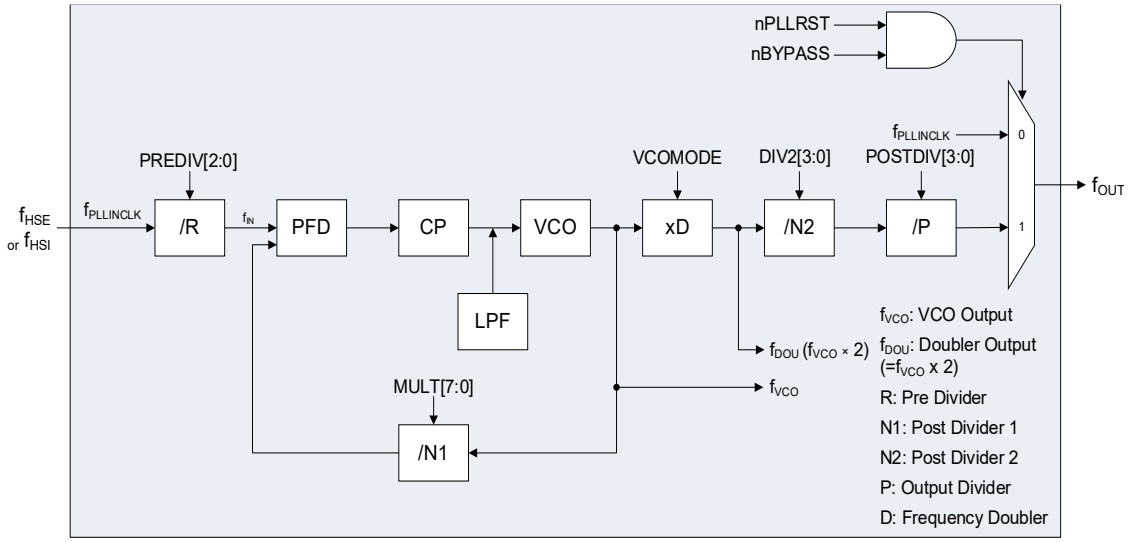
Phase-locked Loop (PLL) can be used to multiply the MXOSC or IOSC16 output clock frequencies.

The PLL frequency generator generates a high-speed clock (up to 75 MHz). For more information on the PLL frequency generator, refer to **Figure 11. Clock Configuration** and **section 4.8.20.**

Users must set up the PLL configuration including selecting the input clock and multiplication factors before enabling the PLL.

The configuration values must not be changed while the PLL is enabled.

Figure 13. PLL Block Diagram



NOTES:

1. Setting PLL Freq. to PMU_PLLCON[31:0].
2. Wait 500 μs for the stabilization of PLL output.
3. Wait until LOCKSTS flag is occurred.
4. The output calculation formula is as follows:

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

register	bit	Symbol	Description
PMU_PLLCON	PREDIV[2:0]	R	PLLINCLK Pre Divider Value
	MULT[7:0]	N1	Post Multiplier Value
	DIV[3:0]	N2	Post Divider Value
	POSTDIV[3:0]	P	Output Divider Value
	VCOMODE	D	Frequency Doubler Value

5. VCOMODE (D) must be '0'.

4.4.5.1 PLL Output Frequency Settings

The PLL of the A33G53x can accurately set the output frequency, F_{OUT}, in 1 MHz increments. To control the PLL, first write 0x80750000 in the PMU_PLLCON register and calculate the parameter value for the PLL output frequency according to the following formula and write to each corresponding bit. The formula for the F_{IN} input to the F_{VCO} input of the PLL is as follows, and the input range of the F_{IN} frequency is between 1 MHz and 3 MHz, the recommended value of the input frequency (F_{IN}) is 2 MHz.

$$\mathrm{FIN} = \frac{\mathrm{PLLINCLK}}{(\mathrm{R}+1)}, \qquad 1\mathrm{MHz} \le \mathrm{FIN} \le 3\mathrm{MHz}\ (\mathrm{Recommanded\ FIN} = 2\ \mathrm{MHz})$$

The VCO's output frequency (f_{VCO}) must be set between 50 MHz and 200 MHz, and calculated by the formula below:

$$\mathrm{VCO} = \mathrm{FIN} \times (\mathrm{N}_1 + 1), \quad 50\ \mathrm{MHz} \le \mathrm{VCO} \le 200\ \mathrm{MHz}\ if\ \mathrm{D} = 0$$

The PMU_PLLCON register has a "doubler" option that doubles the VCO output based on the settings of the VCOMODE bit. When the doubler option is enabled, the output frequency of VCO × 2 must be between a minimum of 100 MHz and a maximum of 250 MHz.

The formula below calculates the VCO × 2:

$$\mathrm{VCO} \times 2 = \mathrm{VCO} \times (\mathrm{D}+1), \qquad 100\ \mathrm{MHz} \le \mathrm{VCO} \times 2 \le 250\ \mathrm{MHz}\ if\ \mathrm{D} = 1$$

Overall, FOUT, the final frequency of PLL, can be obtained using the formula above and using the formula below.

$$\mathrm{FOUT} = \frac{\mathrm{PLLINCLK} \times (\mathrm{N}_1 + 1)}{(\mathrm{R}+1) \times (\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1) = \frac{\mathrm{FIN} \times (\mathrm{N}_1 + 1)}{(\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1)$$

NOTE:

If the main clock is changed from RINGOSC (1 MHz) to PLL, it is recommended to change the clock gradually in the order below:

4.4.5.2 Example of MXOSC PLL (75 MHz) Clock Changing

- Set up XTALI and XTALO pins and enable MXOSC clock source. (Assume $f_{PLLINCLK}$ is 8 MHz.)
  - $f_{PLLINCLK}$ = 8 MHz / 1 = 8 MHz (MXOSCEN[1:0] = '10' in PMU_CCR register)
- Configure the PMU_BCCR register:
  - Set PLLCLKDIV bit to '0'
  - Set PLLCLKSEL bit to '1'.
  - Set MCLKSEL[1:0] bits to '11'.
- Set R to 3 (PREDIV[2:0] = 0x3). This allows the f_{IN} value to be between 1 MHz and 3 MHz.
  - $f_{IN} = f_{PLLINCLK} / (R + 1) = 8\ MHz / (3 + 1) = 2\ MHz$
- Set N1 to the value (MULT = 0x4A) that allows f_{VCO} to be less than 200 MHz.
  - $f_{VCO} = f_{IN} \times (N1 + 1) = 2\ MHz \times (74 + 1) = 150\ MHz$
- Set D to the value (VCOMODE = '0') that allows f_{VCO}×2 to be less than 250 MHz.
  - $f_{VCO}\times 2 = f_{VCO} \times (D + 1) = 150\ MHz \times (0 + 1) = 150\ MHz$
- By adjusting the values of N2 (DIV = 0x0) and P (POSTDIV= '1'), set f_{PLLOUT} to be 75 MHz.
  - $f_{PLLOUT} = f_{VCO} / \{(N2 + 1) \times (P + 1)\} = 150\ MHz / \{(0 + 1) \times (1 + 1)\} = 75\ MHz$
- Configure the PMU_PLLCON register:
  - Set PLLEN and nPLLRST bits to enable PLL and PLL Reset.
  - Set nBYPASS bit to '1' to enable PLL Output mode.
  - Set the PLL divider values to PMU_PLLCON register that you defined for the PLL divider in the step 3 to 6.
  - Update the PMU_PLLCON register.
  - Wait for about 500 μs stabilization time
  - Wait until LOCKSTS flag indicating the PLL is locked.
- If the LOCKSTS bit is set, configure the PMU_BCCR register so that the system main clock becomes the MXOSC (HSE) PLL mode:
  - Set PLLCLKSEL bit to '1'.
  - Set MCLKSEL[1:0] bits to '10'.
- Update each global variable representing HCLK and PCLK to have the system operation frequency value.

4.4.6 SXOSC Clock

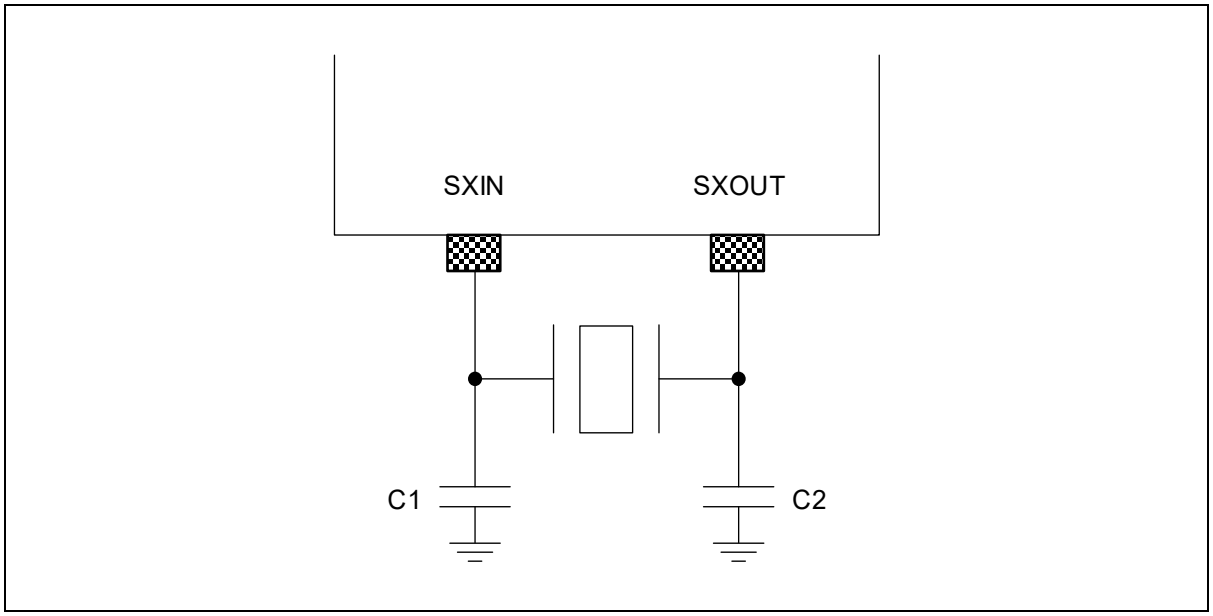
The external sub-oscillator (SXOSC) is a 32.768 kHz crystal or ceramic resonator. It has the advantage of implementing a low-power mode but highly accurate clock source to the clock peripheral for a watch, a calendar, or other time-related functions.

- External Sub-oscillator (LSE): 32.768 kHz

Table 14. LSE Clock Pins

Pin name	Type	Description
SXIN	Analog Input	These pins are connected to a 32.768 kHz crystal resonator.
SXOUT	Analog Output	

Figure 14. LSE External Clock



4.4.7 RINGOSC Clock

After the microcontroller is powered on, the RINGOSC (1 MHz) is initially enabled as a system clock source by default in the system operation sequence. Other clock sources are set by users while the system is clocked by the RINGOSC.

For more details, refer to the electrical characteristics chapter in A33G53x Series.

- Internal Ring oscillator (RINGOSC)
- 1 MHz (±50% @−40°C to +85°C)

4.4.8 Two Main Operating Clocks: HCLK and PCLK

A33G53x series has two main operating clocks. One is HCLK, the clock signals both for CPU core and AHB bus; the other is PCLK, the clock signals for peripherals.

4.4.8.1 HCLK Clock

The Cortex-M3 core requires two clocks, the HCLK and FCLK. The HCLK is fed to the CPU and AHB. The FCLK is always enabled except in DEEP-SLEEP mode, whereas the HCLK can be disabled in Idle mode.

The buses and memories are clocked by the HCLK. As the bus clock frequency is limited to a maximum of 75 MHz, the HCLK frequency must not exceed 75 MHz.

4.4.8.2 PCLK Clock

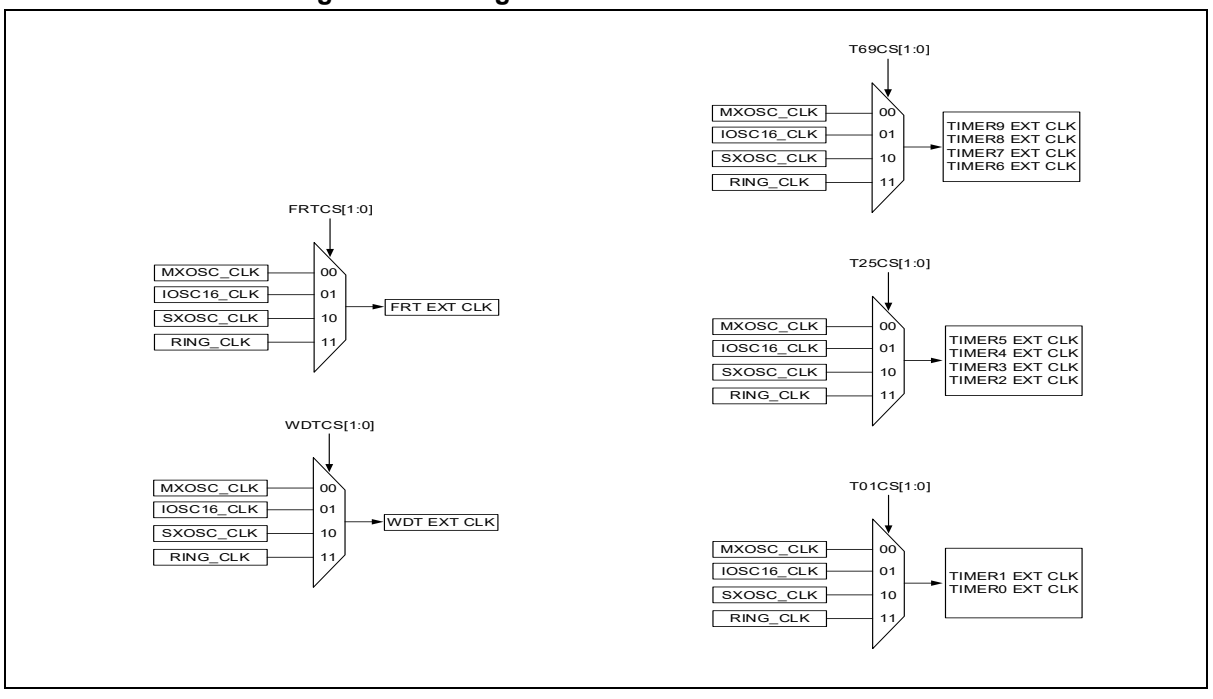
The PCLK can be used as a clock for any peripherals. Whether to enable or disable the PCLK for each peripheral is determined with the PMU_PCCR registers; each peripheral block's registers cannot be read unless its PCLK input is enabled. And the PCLK stops operating in DEEP-SLEEP mode.

4.4.9 Configuration of Miscellaneous Clocks

The A33G53x series supports the "miscellaneous clocks" feature, which allows some peripheral to be assigned with a different clock source (MXOSC, IOSC16, SXOSC and RINGOSC) at a different frequency division ratio.

Users can set each peripheral (FRT, WDT, 16-bit Timer 0 to 9) clock source in the corresponding PMU_PCSR register. (See chapter 4.8.18)

Figure 15. Configuration of Miscellaneous Clocks



4.4.10 Peripheral Clocks

Table 15 describes the peripheral's clock and the register selection:

Table 15. Peripheral Clock Selection

Peripheral	PMU_PCSR	PCLK
Systick	N/A	N/A
WDT	PMU_PCSR[1:0]	O
FRT	PMU_PCSR[3:2]	O
TIMER 0 / 1	PMU_PCSR[5:4]	O
TIMER 2 / 3 / 4 / 5	PMU_PCSR[7:6]	O
TIMER 6 / 7 / 8 / 9	PMU_PCSR[9:8]	O
ADC	N/A	O
UART 0 / 1 / 2 / 3	N/A	O
I2C 0 / 1	N/A	O
SPI 0/1	N/A	O
GPIO	N/A	O
PWM 0 / 1 / 2 / 3	N/A	O
PWM 4 / 5 / 6 / 7	N/A	O

NOTE: Systick clock source is HCLK.

4.4.11 Clock Monitoring

Using the clock monitoring function of A33G53x, a reset or interrupt occurs when the oscillator stops oscillating or erroneous oscillation occurs due to external noise, etc.

When the clock monitoring event occurs, the core clock must not be the main oscillator. This function uses the internal ring oscillator for monitoring and clock switching. To enable this feature, the internal ring oscillator must be always enabled.

Procedures for main oscillator oscillation and monitoring function are shown Figure 16.

Figure 16. The Main Oscillator Operating Procedures

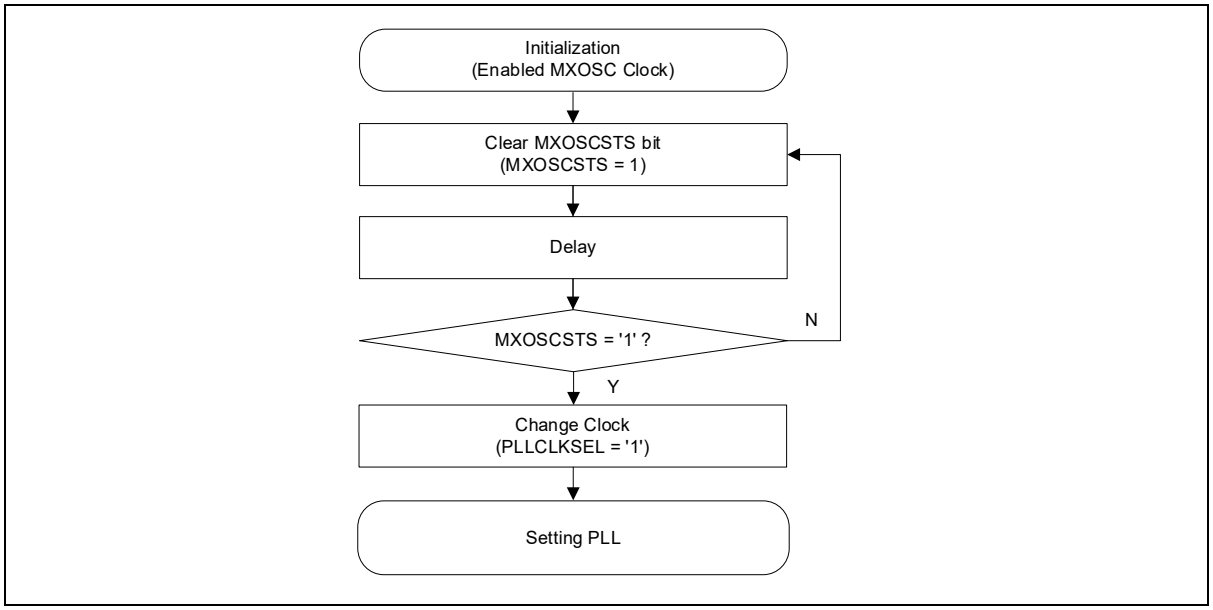
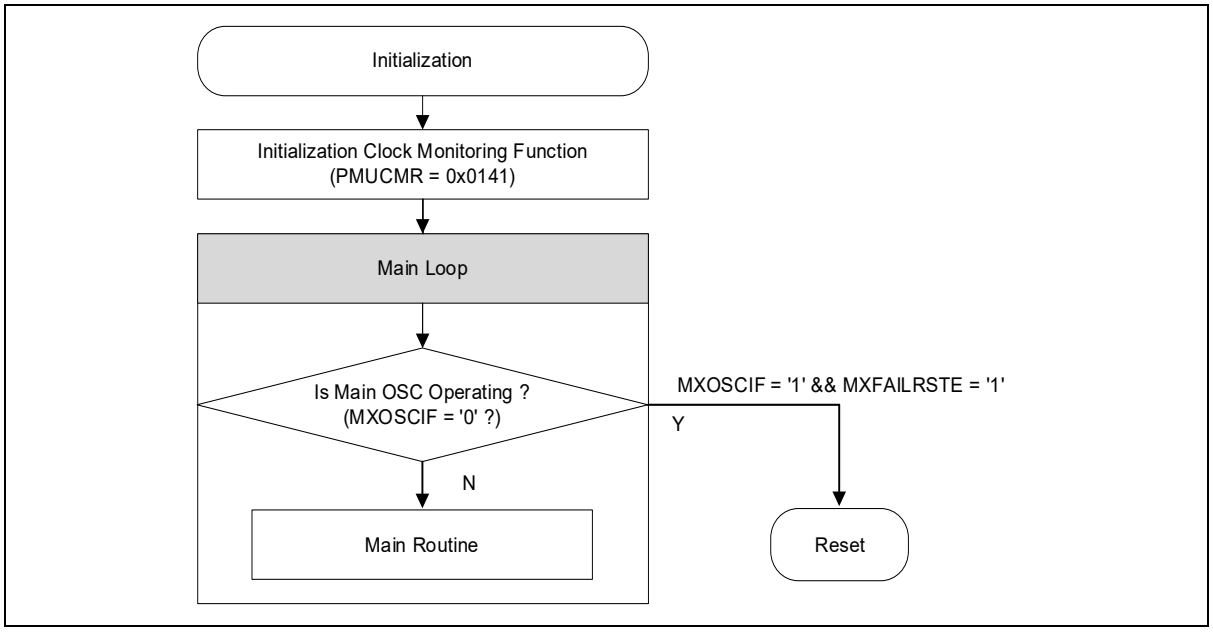


Figure 17. MXOSC Clock Monitoring Workflow



4.4.12 Setting Example

<Example1> Set ring oscillator to system clock

WDT_CON<WDH> = ‘0’	: Disables WDT (Watchdog timer)
PMU_CCR<ROSCEN[7:6]> = ‘10’	: Oscillation 1:1 internal ringosc
PMU_BCCR<PCLKDIV> = ‘0’	: PCLK = HCLK
PMU_BCCR<HCLKDIV[9:8]> = ‘00’	: HCLK = Main Clock
PMU_BCCR<MCLKSEL[1:0]> = ‘00’	: Sets ring oscillator to system clock

<Example2> 8 MHz external main crystal oscillation and monitoring setup.

PC_MR<P15[31:30]> = ‘01’	: Sets the function of PC15 to XTALI
PC_MR<P14[29:28]> = ‘01’	: Sets the function of PC14 XTALO
PC_CR<P15[31:30]> = ‘10’	: Sets the direction of XTALI port to analog input
PC_CR<P14[29:28]> = ‘10’	: Set the direction of XTALO to analog input
PC_PCR<D15> = ‘0’	: Select pull-up resistor of XTALI
PC_PCR<P15> = ‘0’	: Disable pull-up/pull-down register of XTALI
PC_PCR<D14> = ‘0’	: Select pull-up resistor of XTALO
PC_PCR<P14> = ‘0’	: Disable pull-up/pull-down register of XTALO
PMU_CCR<MXOSCEN[1:0]> = ‘10’	: Enable oscillation of external main crystal (1:1)
PMU_CMR<MXOSCMNT> = ‘1’	: Enable monitoring of external main oscillatior
[READ] PMU_CMR<MXOSCSTS>	: Initialize status of external clock oscillation (Read Clear)

<Example3> PLL output frequency setting (74 MHz)

FM_CFG<CWAIT[4:0]> = '00011’	: Set wait-time of code flash.
FM_CFG<DWAIT[12:8]> = ‘00011’	: Set wait-time of data flash.
PLL_CON = ‘0x80750000’	: Set key value 0x80750000 for using A33G53x PLL.
PLL_CON<PLLRESB>= ‘1’	: Enable PLL reset
PLL_CON<PLLEN> = ‘1’	: Enable PLL operation
PLL_CON<PREDIV[10:8]> = ‘011’	: Set the value of pre-divider (R = 3)
PLL_CON<MULT[27:20]> = ‘00110110’	: Set the value of VCO (N1 = 36)
PLL_CON<DIV[19:16]> = ‘0000’	: Set the value of VCO divider (N2 = 0)
PLL_CON<POSTDIV[3:0]> = ‘0000’	: Set the value of Post-Divider (P = 0)
PLL_CON<VCOMODE> = ‘0’	: Set the value of VCO doubler (D = 0)
PLL_CON<BYPASS> = ‘1’	: Set PLL clock as multiplier output

4.4.13 Clock Configuration Procedure

After the microcontroller is powered on, the RingOSC (1 MHz) is initially enabled as a system clock source by default in the system operation sequence. Other clock sources should be set by users while the system is clocked by the RingOSC. The IOSC16 (16 MHz) can be enabled with PMU_CCR register (Clock control register).

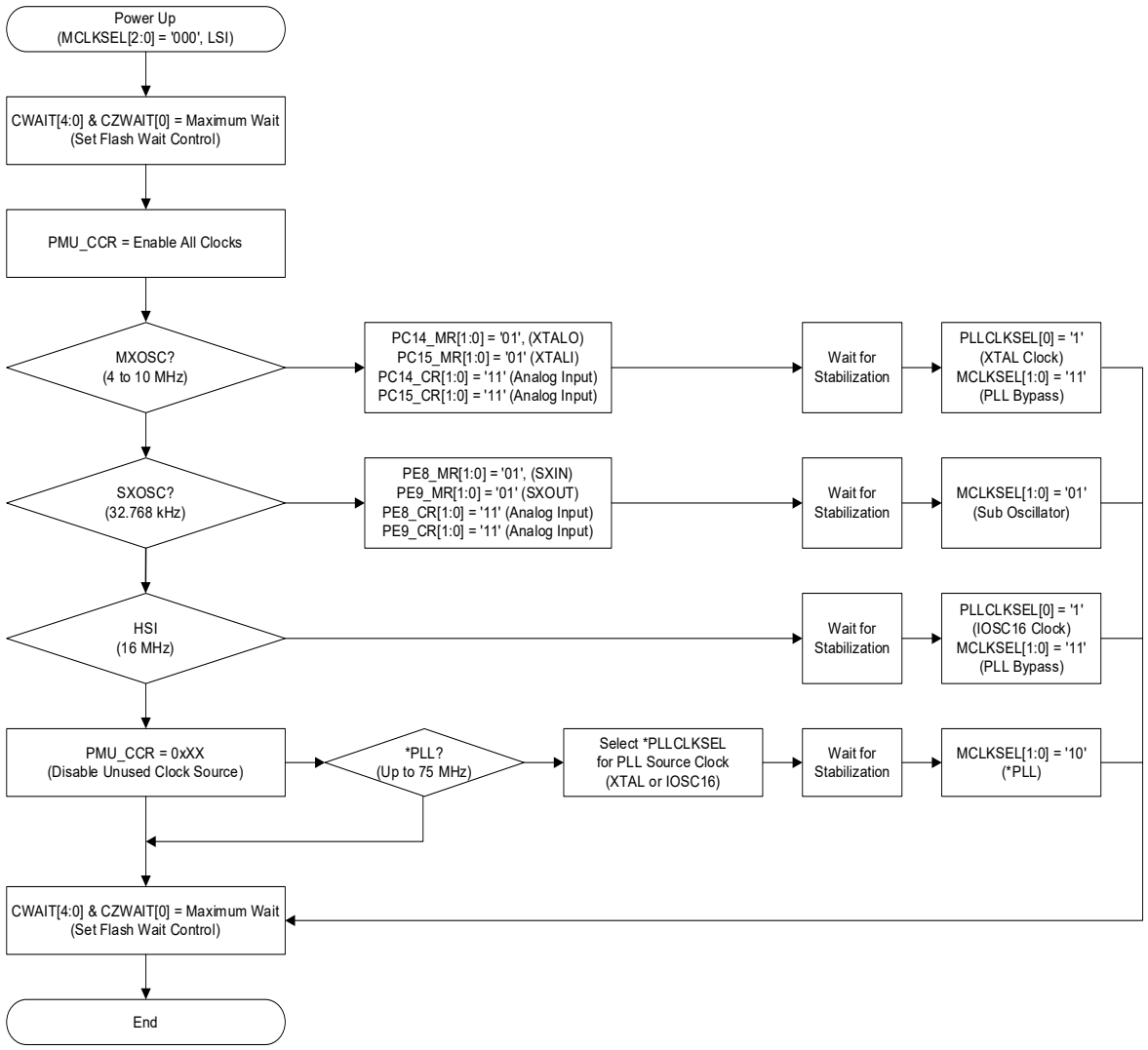
Before enabling the MainOSC block, the relevant pins must be configured as XTALI and XTALO, and the user must be careful not to affect other bits of the PC_MR and PC_CR registers. Once the MainOSC block has been enabled, the system must wait for the crystal oscillation to stabilize.

The sub-oscillator (32.768 kHz) clock can be enabled with the PMU_CCR register (Clock Control Register). Likewise with XTALI and XTALO pins for the MainOSC block, the SubOSC must be enabled after the pin multiplexer configuration is completed for SXIN and SXOUT pins and then the stabilizing time is elapsed.

The MCLK can be changed with the PMU_BCCR register (PMU Bus Clock Control register).

Figure 18 shows an example sequence of how the system clock changes.

Figure 18. Clock Change Procedure



NOTE: For details on the SCU Clock Control, see section 4.4.5 PLL Clock.

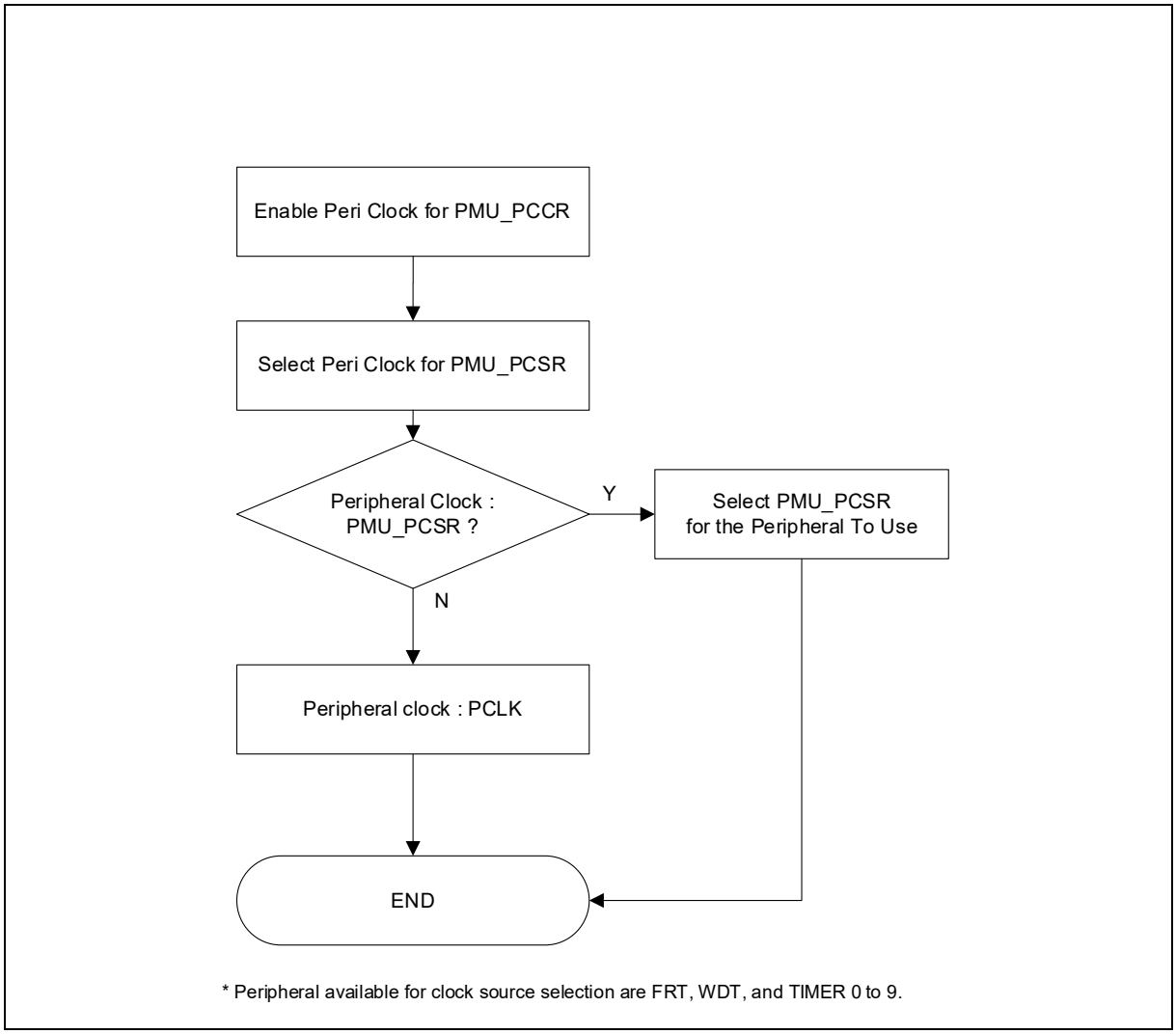
To speed up the system clock to the maximum operating frequency, users must check the Flash wait control configuration. Flash read access time is one of limitation factor for performance. The wait control recommendation is suggested in Table 16.

Table 16. Flash Wait Control Recommendation

Bus Clock (HCLK)	CZWAIT	CWAIT or DWAIT	Operating Cycle	Flash Access Timing	Max. Access Timing
8 MHz	0	0	1.5	5.3 MHz	25 MHz
8 MHz	1	X (Don’t Care)	0.5	16 MHz	16 MHz
16 MHz	0	0	1.5	10.6 MHz	25 MHz
16 MHz	1	X (Don’t Care)	-	Do not use	Do not use
20 MHz	0	0	1.5	13.3 MHz	25 MHz
20 MHz	0	1	2.5	8 MHz	25 MHz
20 MHz	1	X (Don’t Care)	-	Do not use	Do not use
25 MHz	0	0	1.5	16.6 MHz	25 MHz
25 MHz	0	1	2.5	10 MHz	25 MHz
25 MHz	1	X (Don’t Care)	-	Do not use	Do not use
32 MHz	0	0	1.5	21.3 MHz	25 MHz
32 MHz	0	1	2.5	12.8 MHz	25 MHz
32 MHz	0	2	3.5	9.14 MHz	25 MHz
74 MHz	0	2	3.5	21.1 MHz	25 MHz
74 MHz	0	3	4.5	16.4 MHz	25 MHz
75 MHz	0	2	3.5	21.4 MHz	25 MHz
75 MHz	0	3	4.5	16.6 MHz	25 MHz

Figure 19 shows how to set the peripheral clock. Typical selection of peripheral clocks is to use PMU_PCSR.

Figure 19. Peripheral Clock Select



4.5 Reset

4.5.1 Overview

The reset features of A33G53x series are as follows:

- Power-on Reset (POR)
- Low-Voltage Detect Reset (LVR)
- nRESET pin Reset
- WatchDog Tiimer Reset (WDT)
- Software Reset
- Clock oscillating error Reset
- CPU Request Reset

The A33G53x series has two system reset options. One is a cold reset, which is effective during power-up or power-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 Pin Description)

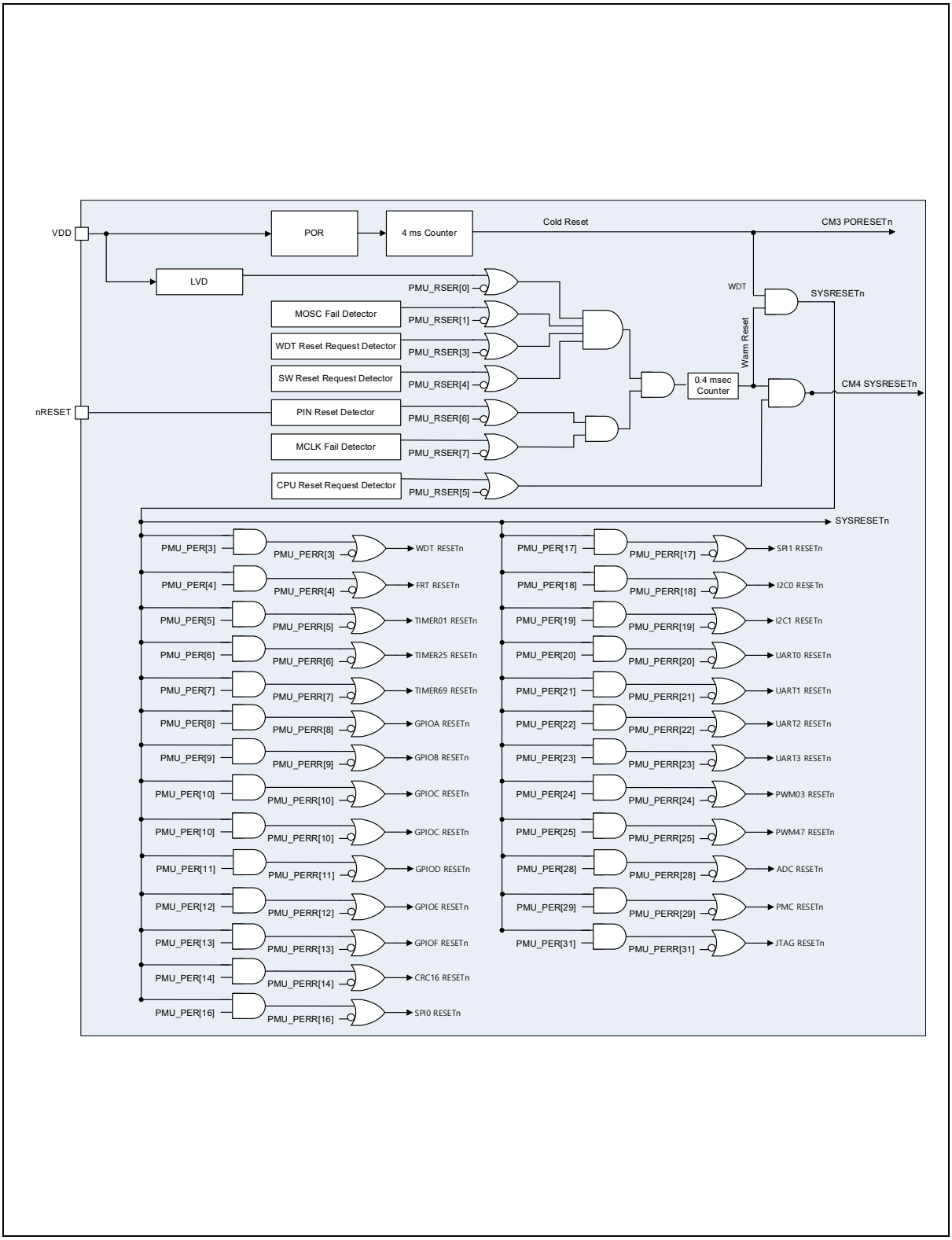
A reset event makes a chip to turn to an initial state. The reset sources of the cold reset and the warm reset are listed in Table 17.

Table 17. Reset Sources of Cold Reset and Warm Reset

	Cold Reset	Warm Reset
Reset Sources	POR (Power-on Reset)	nRESET Pin LVD Reset WatchDog Timer Reset Software Reset Clock oscillating error Reset CPU Request Reset

4.5.2 Reset Tree Configuration

Figure 20. Reset Tree Configuration



4.5.3 Cold Reset

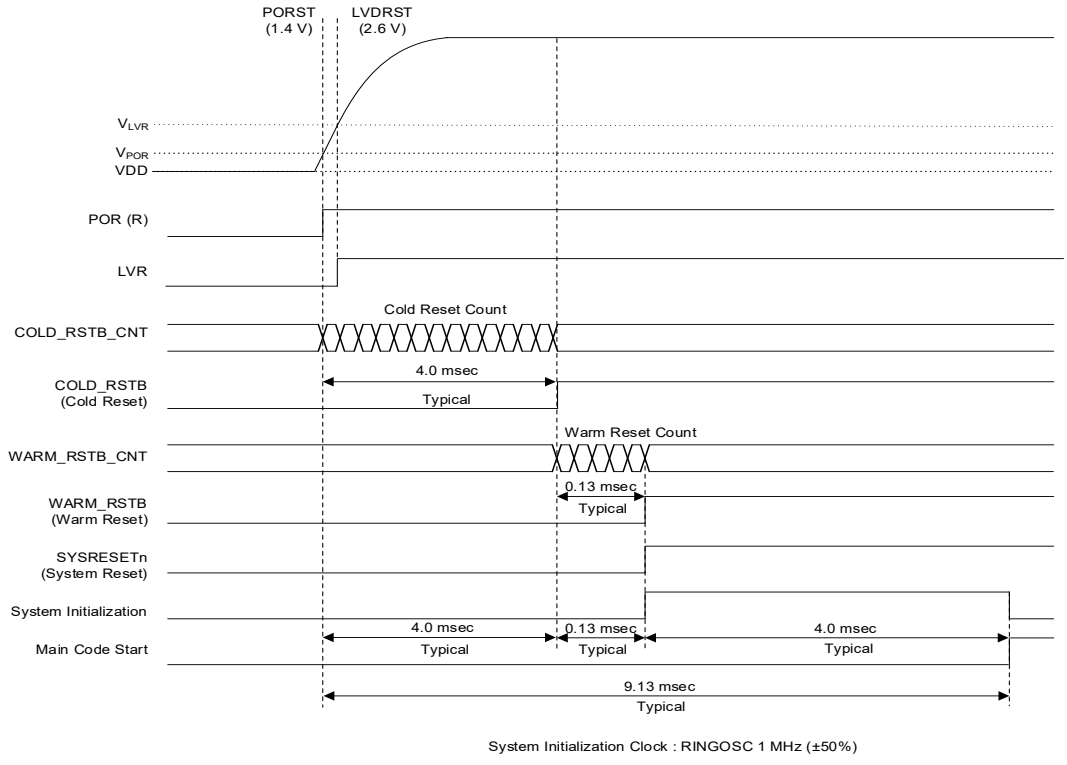
The A33G53x offers a variety of reset functions to protect the entire system from malfunctions in a service routine or core and to ensure the operation of the microcontrollers that the user can trust. Since the reset timing of the A33G53x is based on the ring oscillator (RINGOSC), it is necessary to consider the error rate (±50%) of RINGOSC when applying the reset function to the user application.

The A33G53x supports two types of reset: cold-reset, which initializes the system by turning the system power off and on, and warm reset, which initializes the system without changing system power. When the system boots normally after the reset occurs, user code is executed.

When the system power is turned on and the power reaches 1.4 V, a power-on reset (POR) reset occurs, and when the voltage value set by LVD reset is detected, a LVD reset occurs. After performing a cold-reset, a warm-reset occurs, and the system is initialized and the boot ROM and main code are executed.

Figure 21 below shows the A33G53x cold reset timing diagram.

Figure 21. Timing Diagram of Power-up Procedure



NOTE:

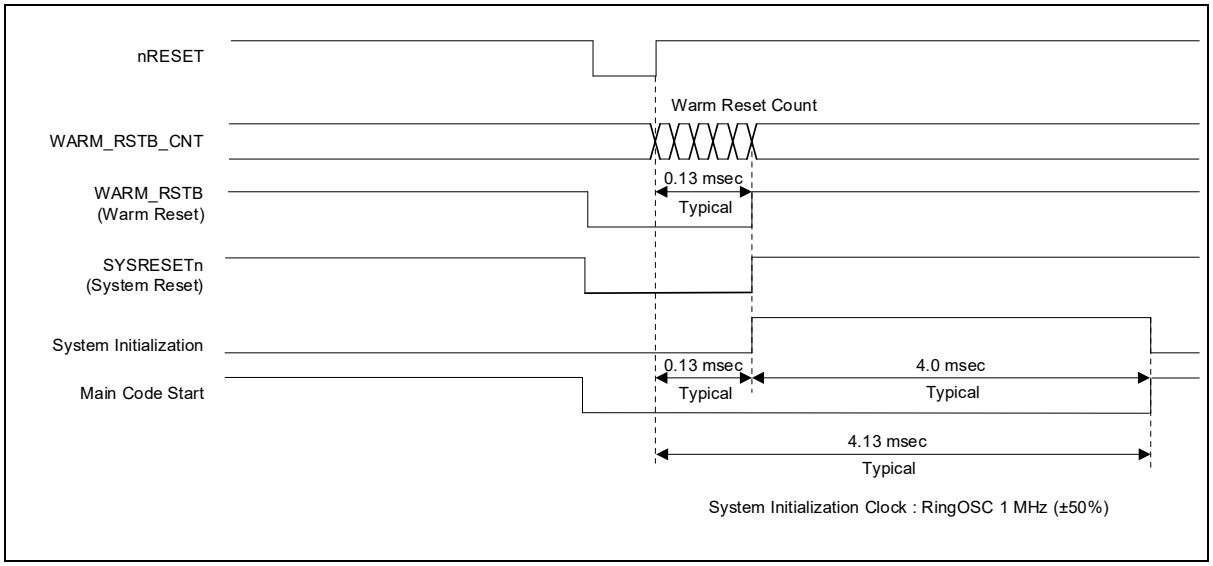
For details on the electrical characteristics, refer to the datasheet.

4.5.4 Warm Reset

Warm reset is also called a soft reset. While the microcontroller is operating at normal voltage, a reset can occur by running a specific routine or by configuring a reset source. When this warm reset is happened, the microcontroller restarts from the boot ROM and runs a main code.

A reset event can be performed for each module by setting the reset source provided by PMU_RSER (PMU Reset Source Enable Register). The figure below shows the timing of the warm reset.

Figure 22. Warm Reset 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. The user can configure the LVD block to generate a reset or an interrupt, when an LVR event is triggered.

LVD reset must be set early in main code. When any reset occurs, the LVD setting returns to the default value.

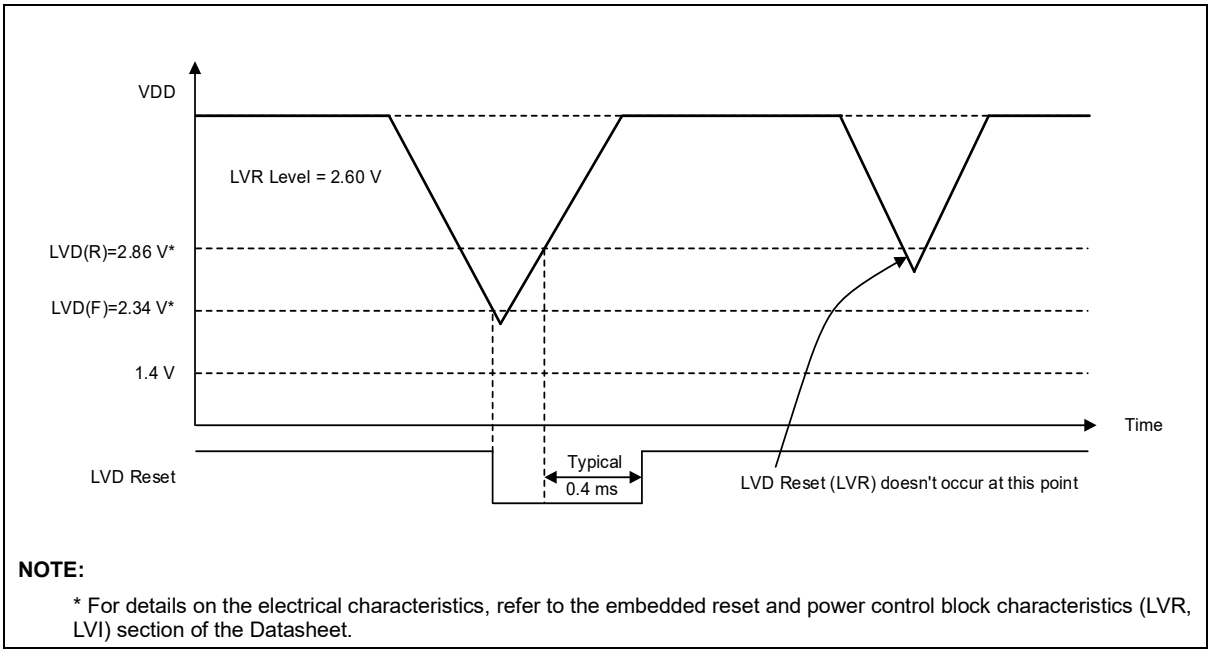
For example, while executing the main code, the LVD reset level is set to 4.0 V, and if VDD drops from 5 V to 3.5 V, an LVD reset occurs, and the LVD reset level is initialized to default (2.6 V) and the main code restarts. Then, the LVD reset level is set to 4 V by the main code again, the LVD is reset, and the operation is repeated until VDD reaches 4.0 V. To prevent this, LVD reset must be set early in the main code.

The LVR in Table 18 is assigned for LVR level.

Table 18. LVD Reset Level

LVD reset level	Description
2.60 V ~ 4.50 V	LVD Reset Level (8-step) This LVR level is set by LVDRL[2:0] in PMU_LVDCON register.

Figure 23. LVR Diagram (LVR level = 2.60 V)



NOTE:

* For details on the electrical characteristics, refer to the embedded reset and power control block characteristics (LVR, LVI) section of the Datasheet.

4.5.6 nRESET Pin Reset

This reset signal is generated by the external nRESET pin. When the nRESET pin is driven low, currently running process is aborted and the microcontroller enters reset state.

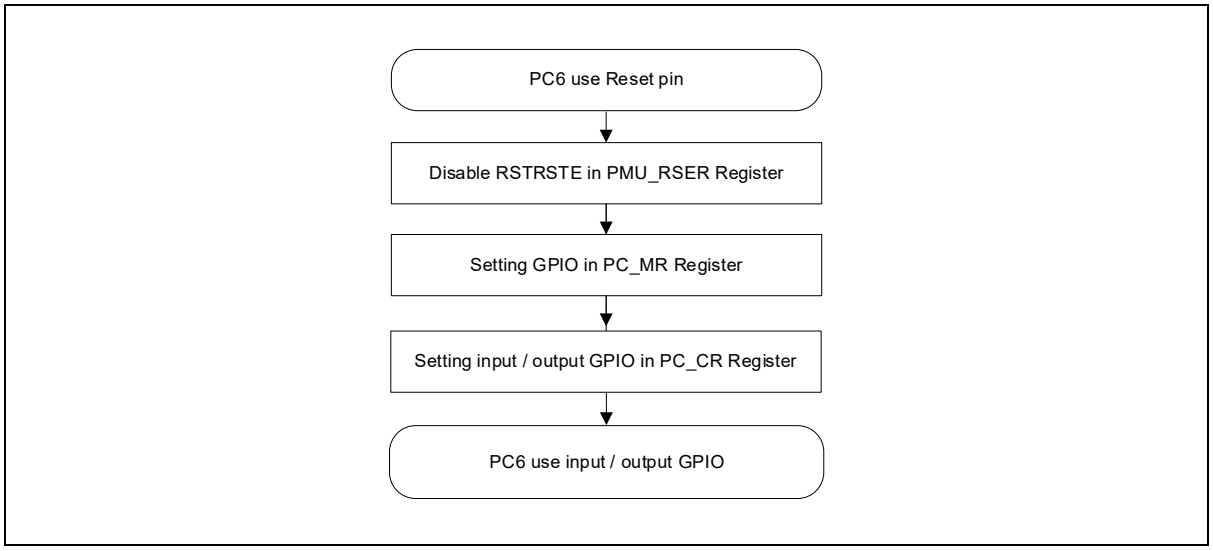
The nRESET pin in Table 19 is assigned as a RESET input by default.

Table 19. nRESET Pin

Pin name	Type	Description
PC6/nRESET	Input	External Reset Input

To use PC6 (nRESET) as a GPIO, the RSTRSTE of PMU RSER register must be disabled.

Figure 24. Changing nRESET Pin to GPIO Pin



4.5.7 WatchDog Timer Reset

WatchDog Timer (WDT) monitors the operation of the microcontroller and is typically used to detect software malfunctions. When the microcontroller becomes uncontrollable due to a malfunction, the WDT restarts the microcontroller to recover it.

The A33G53x series has a built-in WDT module, which functions as a 32-bit downcounter. When WDT is the reset source, the microcontroller will reset when the WDT counts down to zero.

For details on the WatchDog Timer Reset, refer to the chapter 8.

4.5.8 Software Reset

A system reset sets all registers to their reset values except the PMU_RSSR, PMU_MR, PMU_PERR, PMU_LVDCON, PMU_IOSC16TRIM, PMU_EOSCCON register.

4.5.9 Main Oscillator (XTAL) Fail Reset

The MXFAILRSTE bit in PMU_RSER register is used to configure input signals that trigger a reset event.

Setting the MXFAILRSTE bit to ‘1’ enables the corresponding reset signal to trigger a reset. Setting this bit to ‘0’ disables the reset signal, masking the reset event.

The MXFAILRST bit in PMU_RSSR register indicates the occurrences of reset event. A ‘1’ value in this bit indicates that the corresponding reset source has triggered a reset.

4.5.10 CPU Request Reset

The SYSRSTE bit in PMU_RSER register is used to configure input signals that trigger a reset event.

Setting the SYSRSTE bit to ‘1’ enables the corresponding reset signal to trigger a reset. Setting this bit to ‘0’ disables the reset signal, masking the reset event.

The SYSRST bit in PMU_RSSR register indicates the occurrences of reset events. A ‘1’ value in this bit indicates that the corresponding reset source has triggered a reset.

4.6 Power Supply

The device requires an operating voltage supply (VDD) of 3.0 to 5.5 V, and analog peripherals are supplied through separated power pins: AVDD and AGND.

Table 20. Power Supply of VDD and AVDD

Parameter	Symbol	Min.	Max.	Unit
Supply Voltage	VDD	3.0	5.5	V
	AVDD (Analog VDD)	3.0	5.5	V

In a power-up sequence, the reset signal is essential and affects the entire system booting process. The A33G53x series has two power-related reset features as described below:

- POR_RST (Power-On Reset) is activated when the voltage is less than 1.4 V.
- LVD_RST (Low-Voltage Detect Reset) is activated when the voltage is less than 2.6 V (Default).

If the voltage level is higher than the POR threshold level and below the flash memory operating voltage (Min. 1.35 V), the code fetch operation may malfunction. To prevent this abnormal code fetch operation, set the LVR level above the flash memory operating voltage. Then, the LVD_RST will generate the internal SYSRESETn signal when the voltage is under the threshold level, and the microcontroller stays in the reset mode.

The minimum level of the LVR is 2.6 V, which satisfies the minimum operating voltage of flash memory.

NOTE:

Electrical characteristics of the POR and LVR are available in the “3.3.2 Power on reset characteristics” and “3.3.4 Embedded reset and power control block characteristics (LVR, LVI)” sections in the datasheet.

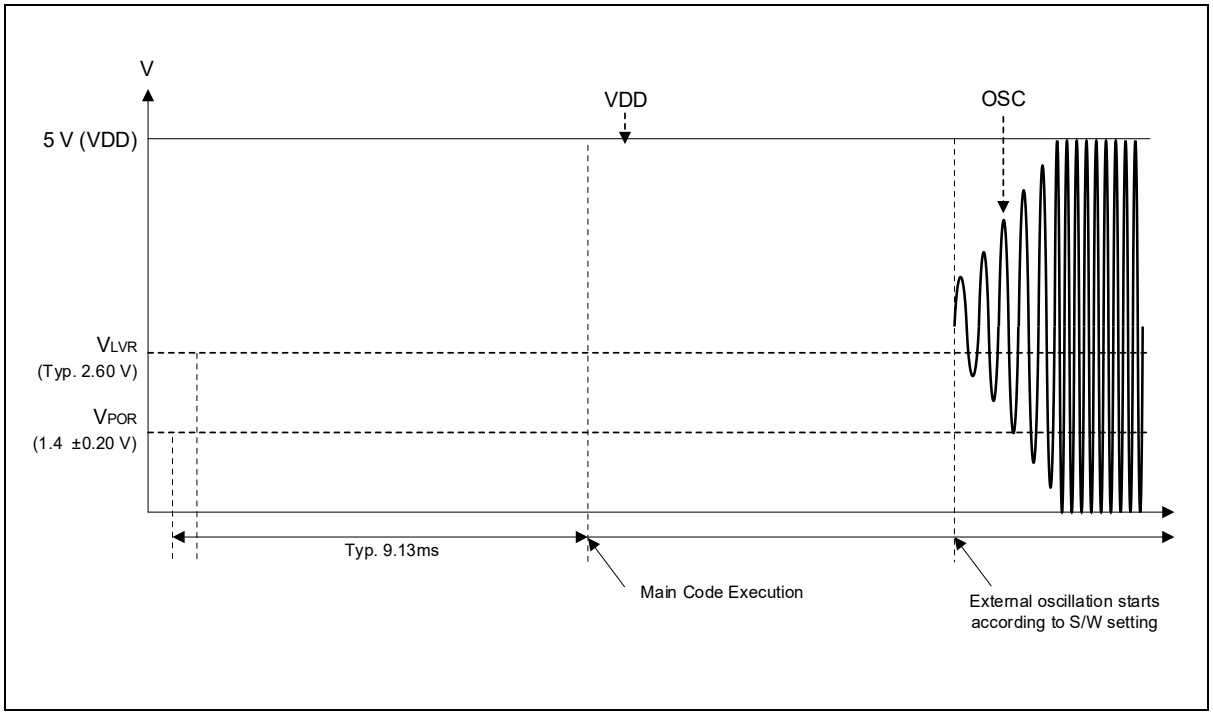
An LVD reset event is triggered when the operating voltage drops below the selected LVD level while the microcontroller is running. The user can configure the microcontroller to issue a LVD reset by LVR level setting. Low-Voltage Detect Reset is a warm reset. Refer to the description on 4.5.4 Warm Reset for details.

Figure 25 shows a timing diagram for the operation sequence after the POR and LVR. In this timing diagram, main code execution and oscillator operation after power-on could be estimated.

As power is applied, Figure 25 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 A33G53x, note that external oscillation is set to operate by software setting.

Figure 25. POR and LVR Timing Diagram



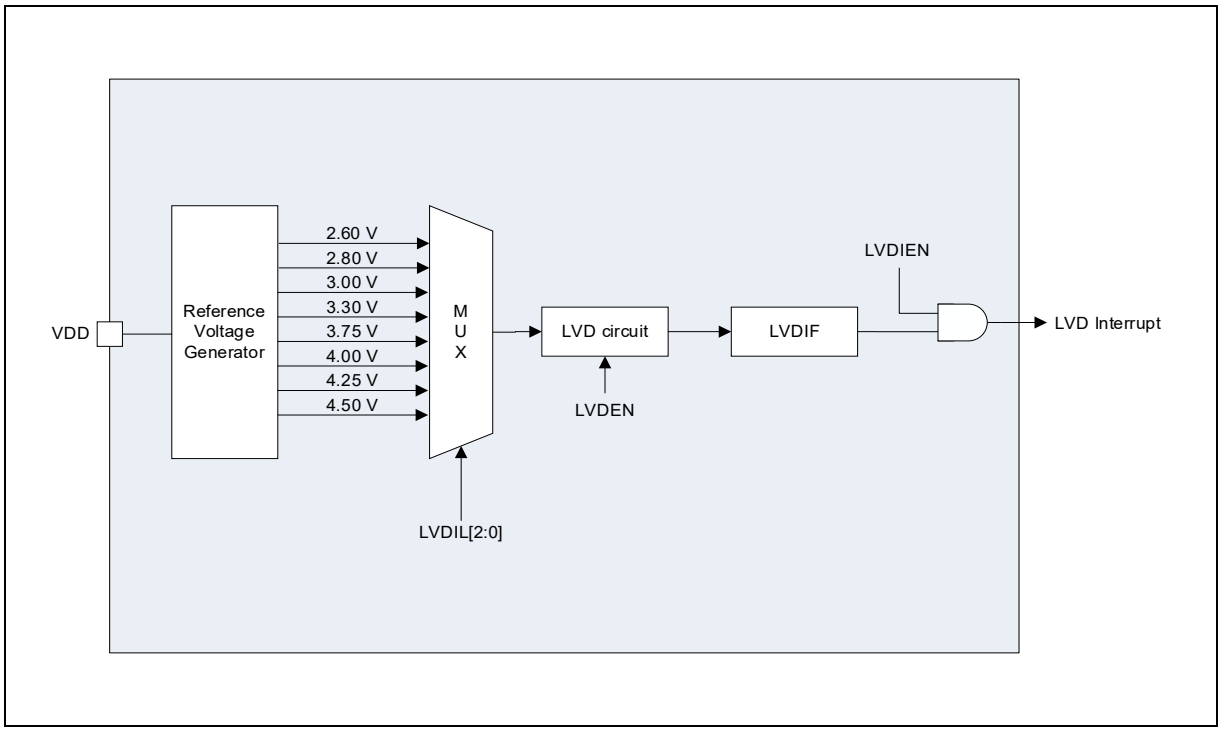
4.6.1 LVD Interrupt Block Diagram

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

Figure 26 shows a block diagram of the LVD Interrupt.

NOTE: For information of the LVD signal operation, see chapter 4.8.21.

Figure 26. LVI Block Diagram



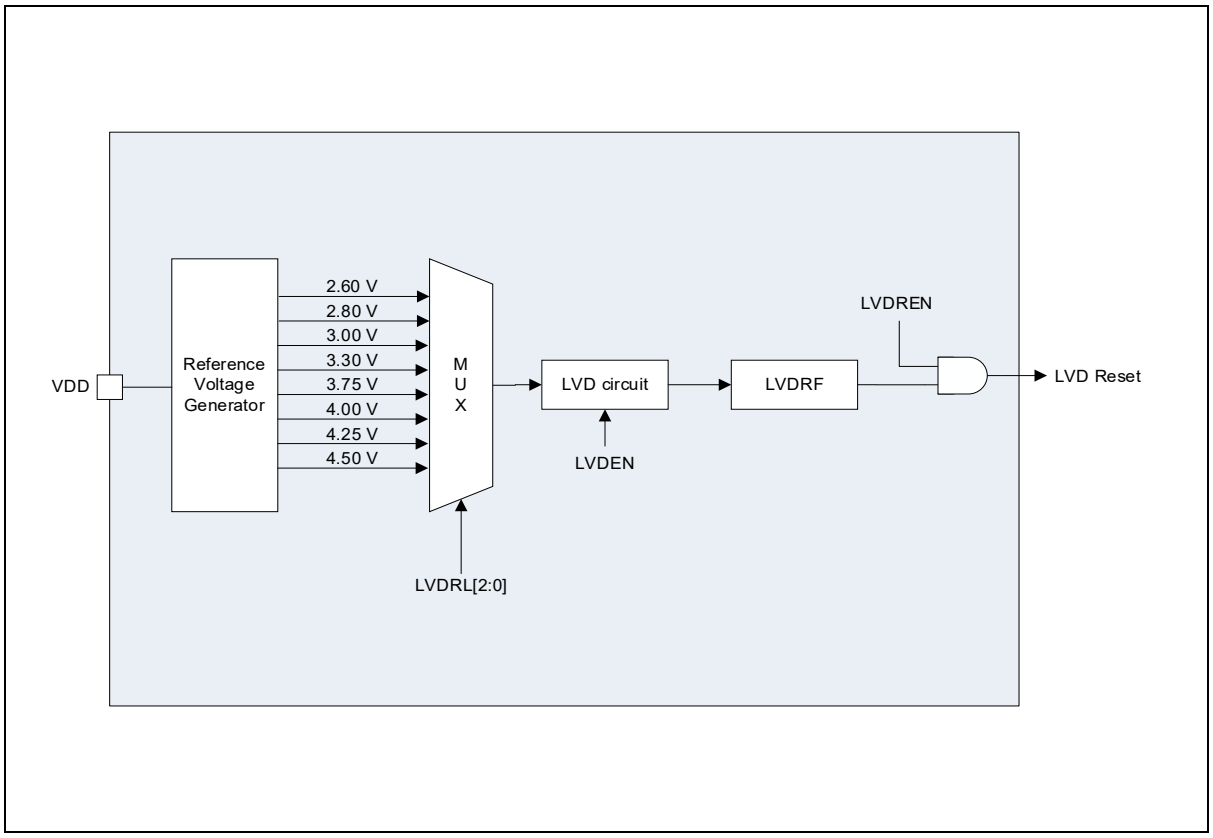
4.6.2 LVD Reset Block Diagram

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.

NOTE: For information of the LVR signal operation, see chapter 4.8.21.

Figure 27. LVR Block Diagram

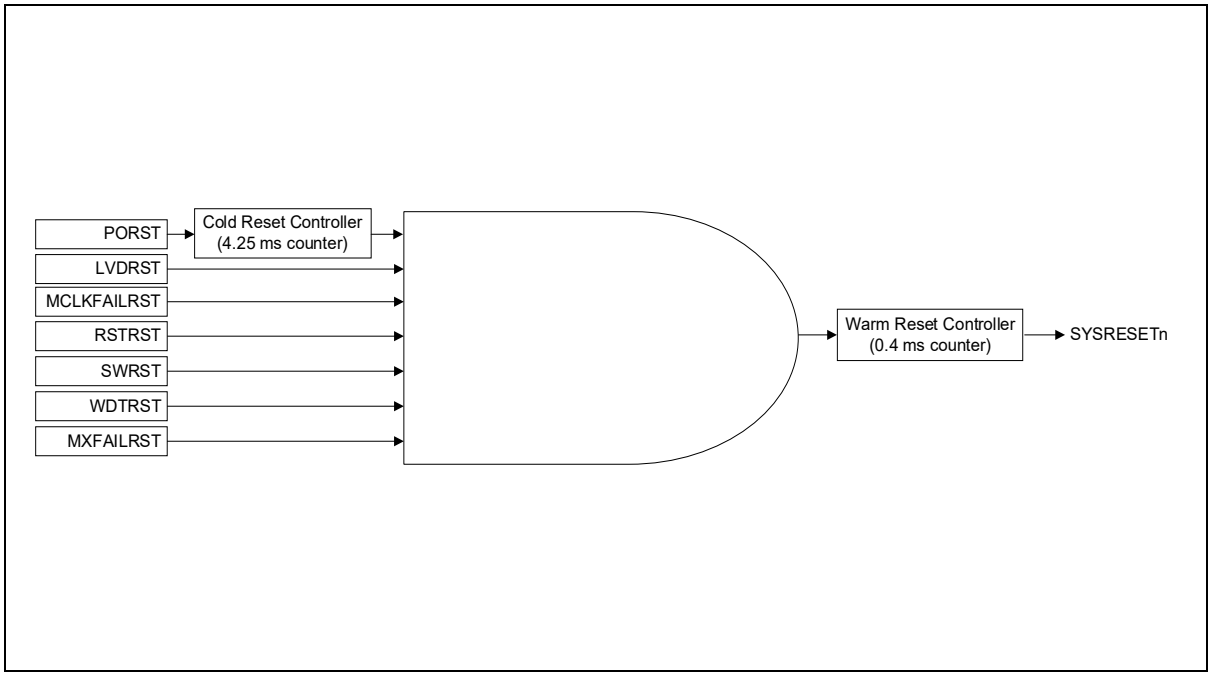


4.6.3 POR, LVR

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. 2.6 V) is set to improve stability and prevent operation errors such as code read.

NOTE: Detailed information for the POR / LVR, see chapter 4.5 Reset.

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 A33G53x series operates in three modes below:

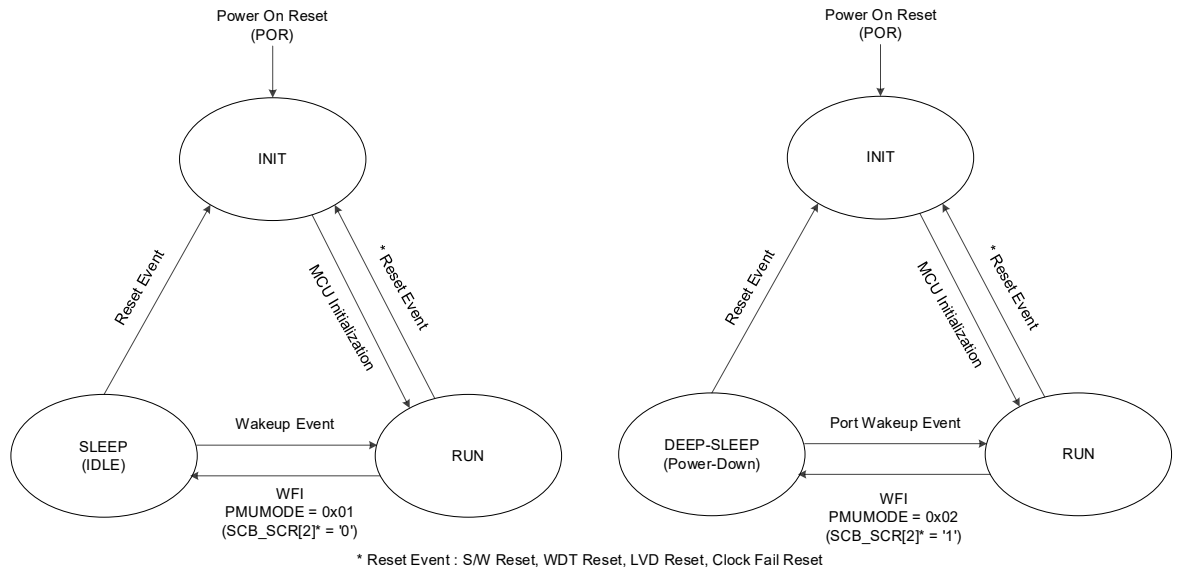
- RUN mode
- SLEEP (IDLE) mode
- DEEP-SLEEP (Power-Down) mode

4.7.2 Transition of Operation Mode

INIT mode is an 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



NOTE: * means that SCB_SCR is System Register in System Control Block.

Table 21. Operation Mode

Mode	Condition	After wake-up event	After reset event
RUN	POWER-ON	N/A	INIT
SLEEP (IDLE)	WFI (Wait for Interrupt) SCB_SCR[2]* = 0 (PMUMODE = 0x01)	RUN	INIT
DEEP-SLEEP	WFI (Wait for Interrupt): SCB_SCR[2]* = 1, (PMUMODE = 0x02)	RUN	INIT

NOTE: *SCB_SCR is System Control Register in System Control Block

4.7.3 RUN Mode

This mode is the normal operating mode of A33G53x microcontrollers. Once activated by an external power supply, the power-on reset (POR) holds the microcontroller in a reset state until it stabilizes. Upon its stabilization, the microcontroller initializes all clock sources and registers in initialization mode and then enters run mode. In run mode, all clock sources, buses, and peripherals can be enabled, and the current consumption will vary with their operating frequencies.

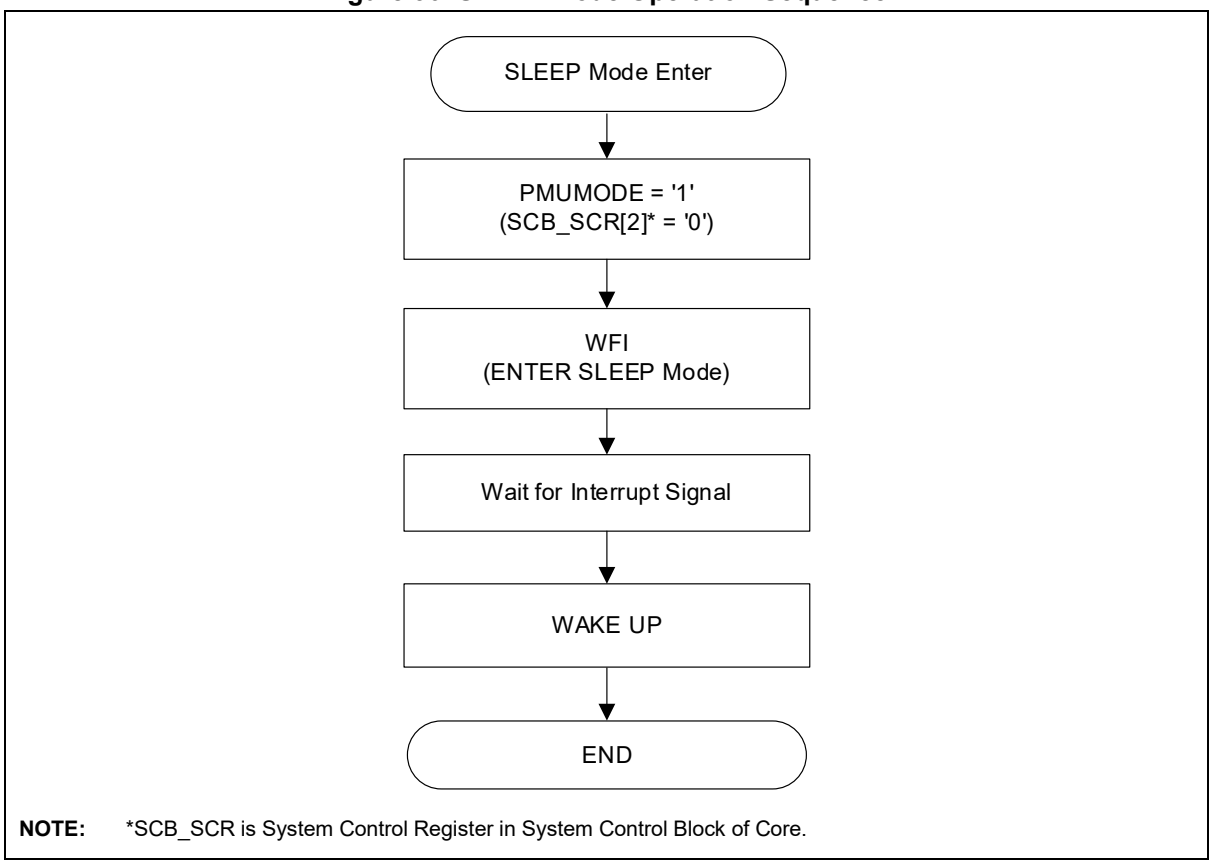
4.7.4 SLEEP (IDLE) Mode

When in RUN mode, writing 0x01 to the PMUMODE of the PMU_MR register triggers the microcontroller to enter sleep mode, in which the clock input to the CPU becomes disabled.

Using the wake-up source enable register (PMU_WSER), the user can select the wake-up sources from the internal peripherals, including GPIOA through GPIOF (IRQ5–10), FRT (IRQ4), WDT (IRQ3), main oscillator error (IRQ2), and LVD (IRQ0). When a wake-up event occurs based on the settings, the microcontroller will return to RUN mode.

In addition, the clock for each peripheral device can be individually enabled or disabled to supply power to only the minimum number of peripherals, thereby reducing power consumption.

Figure 30. SLEEP Mode Operation Sequence



NOTE: *SCB_SCR is System Control Register in System Control Block of Core.

4.7.5 DEEP-SLEEP (Power-Down) Mode

An A33G53x microcontroller operating in run mode can be switched to deep-sleep (power-down) mode by writing 0x00 to the main clock select (MCLKSEL) of the Power Management Unit's (PMU's) bus clock control register (PMU_BCCR), which will set the ring oscillator as the main clock source, and then writing 0x02 to the PMUMODE of the PMU Mode Register (PMU_MR). In deep-sleep mode, the PLL and its clock source, the internal oscillator, are disabled, along with the system clock (HCLK) and peripheral clock (PCLK). This system stop allows for operation with minimal power consumption.

From among the various wake-up sources for peripherals provided by the Wake-up Source Enable Register (PMU_WSER), and wake-up event in deep-sleep mode can be used only for the external GPIOA through GPIOF (bit-5 to -10).

By default, this mode automatically powers off the main oscillator (MXOSC) and sub-oscillator (SXOSC). Therefore, peripherals that are supplied with a clock signal from the main or sub-oscillator will stop operating in deep-sleep mode. To keep those peripherals running in this mode, the microcontroller can be configured in either of the following ways:

1) Set the ring oscillator as the clock source to feed the peripherals that need to continue running in deep-sleep mode.

2) To continue using the main or sub-oscillator in deep-sleep mode, set the ECLKMD bit of the PMU mode register (PMU_MR) to "1." This will prevent the main and sub-oscillators from automatically being powered off. Next, configure the PMU clock control register (PMU_CCR) manually to disable the clock sources that are not in use in deep-sleep mode. However, this method requires special care to ensure effective power saving in deep-sleep mode.

4.7.6 Operation Mode Summary

The following table describes the configurable clocks in each operating mode.

Table 22. Operability of the Clock Sources, Buses, and Modules in each Operating Mode

Functions		Microcontroller modes		
		Run	Sleep	Deep Sleep
Clock Source	IOSC16	O	O	X
	RING	O	O	X
	MXOSC	O	O	△
	SXOSC	O	O	△
	PLL	O	O	X
Buses	AHB	O	O	X
	APB	O	O	X
Modules	CORE	O	X	X
	PMU	O	O	X
	FLASH	O	O	X
	WDT	O	O	X
	FRT	O	O	X
	TIMER	O	O	X
	PWM	O	O	X
	GPIO (PCU)	O	O	X*
	ADC	O	O	X
	UART	O	O	X
	I2C	O	O	X
	SPI	O	O	X

NOTES:

O: Operable
X: Not operable
△: Operability depends on the settings of PMU_MR<ECLKMD>.
*: If the GPIO's wake-up source is set in the PMU_WSER register, its level event is available.

4.8 PMU Registers

The base address and register map of the CHIPCONFIG are as follows:

Table 23. Base Addresses of CHIPCONFIG

Name	Base address
CHIPCONFIG	0x4000_F000

Table 24. CHIPCONFIG Register Map

Name	Offset	Type	Description	Reset value	Reference
VENDIDR	0x0000	RO(1)	Vendor Identification Register	0x4142_4F56	4.8.1
CHIPIDR	0x0004	RO(1)	Chip Identification Register.	0x4733_538x	4.8.2
REVIDR	0x0008	RO(1)	Revision Number Register	0x0000_00xx	4.8.3

NOTE: 'RO' means 'Read Only'.

Base address and register map of the PMU are described in the followings:

Table 25. Base Address of PMU

Name	Base address
PMU	0x4000_0000

Table 26. PMU Register Map

Name	Offset	Type	Description	Reset value	Reference
PMU_IDR	0x0000	RO	PMU ID Register	0xCEDA_0000	4.8.4
PMU_MR	0x0004	RW	PMU Mode Register	0x0000_0000	4.8.5
PMU_CFGR	0x0008	RW	PMU Configuration Register	0x0000_0000	4.8.6
PMU_WSER	0x0010	RW	PMU Wake-up Source Enable Register	0x0000_0000	4.8.7
PMU_WSSR	0x0014	RO	PMU Wake-up Source Status Register	0x0000_0000	4.8.8
PMU_RSER	0x0018	RW	PMU Reset Source Enable Register	0x0000_0069	4.8.9
PMU_RSSR	0x001C	RW	PMU Reset Source Status Register	0x0000_0040*	4.8.10
PMU_PERR	0x0020	RW	PMU Peripheral Event Reset Register	0xB3FF_3FF8	4.8.11
PMU_PER	0x0024	RW	PMU Peripheral Enable Register	0xB3FF_3FF8	4.8.12
PMU_PCCR	0x0028	RW	PMU Peripheral Clock Control Register	0x0000_0118	4.8.13

Table 26. PMU Register Map (continued)

Name	Offset	Type	Description	Reset value	Reference
PMU_CCR	0x0030	RW	PMU Clock Control Register	0x0000_0080	4.8.14
PMU_CMR	0x0034	RW	PMU Clock Monitoring Register	0x0000_0000	4.8.15
PMU_MCMR	0x0038	RW	PMU Main Clock Monitoring Register	0x0000_0000	4.8.16
PMU_BCCR	0x003C	RW	PMU Bus Clock Control Register	0x0000_0000	4.8.17
PMU_PCSR	0x0040	RW	PMU Peripheral Clock Select Register	0x0000_0000	4.8.18
PMU_COR	0x0044	RW	PMU Clock Output Register	0x0000_0000	4.8.19
PMU_PLLCON	0x50	RW	PLL Control Register	0x0000_0000	4.8.20
PMU_LVDCON	0x54	RW	LVD (Low Voltage Detector) Control Register	0x0000_8800	4.8.21
PMU_VDCCON	0x58	RW	VDC/LVD Trimming Register	0xXXXX_XXXX	4.8.22
PMU_IOSC16TRIM	0x5C	RW	Internal Oscillator (16MHz) Trimming Register	0x0000_XXXX	4.8.23
PMU_EOSCCON	0x60	RW	Main Oscillator (XTAL) Control Register	0x0000_0001	4.8.24
PMU_EXTMODER	0x70	RW	External Mode Pin Read Register	0x0000_0000	4.8.25

NOTE : Marked with an asterisk (*) are reset POR reset only applies to register.

4.8.1 CHIPCONFIG_VENDIDR: Vendor ID Register

The CHIPCONFIG_VENDIDR 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
VENDIDR[31:0]																															
0x4142_4F56																															
RO																															

31 0	VENDIDR[31:0]	Vendor identification bits

4.8.2 CHIPCONFIG_CHIPIDR: Chip ID Register

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

CHIPCONFIG_CHIPIDR=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
CHIPIDR[31:0]																															
0x4733_538x																															
RO																															

31 0	CHIPIDR[31:0]	Chip ID bit	
		0x4733_5381	768 KB Flash memory / 24 KB SRAM
		0x4733_5380	512 KB Flash memory / 24 KB SRAM

4.8.3 CHIPCONFIG_REVIDR: Revision Number Register

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

CHIPCONFIG_CHIPIDR=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			REVIDR[7:0]
-			0xXX
-			RO

31 0	REVIDR[7:0]	Revision number bits These bits are modified by a manufacturer.

4.8.4 PMU_IDR: PMU ID Register

PMUIDR is a 32-bit read-only register that stores the product ID number and revision number. The revision number is CHIPREV value at the least significant byte of this register.

PMU_IDR=0x4000_0000

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
CHIPID[31:8]			CHIPREV
0xCEDA00			00
RO			RO

31 8	CHIPID[31:8]	The value of the A33G53x identification number
7 0	CHIPREV	The value of the A33G53x revision number

4.8.5 PMU_MR: PMU Mode Register

PMU_MR is a register that can display the previous operation mode of the microcontroller or set the current operation mode. For example, to enter power-down mode using the FRT as a wake-up source, ECLKMD bit must be set to "1". For details on how to use the microcontroller operation mode, see chapter 4.7.2.

PMU_MR=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				PVDCLP	PREVMODE	ECLKMD	VDCLP	PMUMODE
-				0	00	0	0	00
-				RO	RO	RW	RW	RW

6	PVDCLP	VDC low power status of the previous state
	0	VDC previous state is Normal mode before power-down
	1	VDC previous state is Low-power mode before power-down
5 4	PREVMODE	The previous microcontroller status before reset or Wake-up
	00	RUN mode
	01	Sleep mode
	10	Power-down mode
	11	Initialization mode
3	ECLKMD	In Power-down mode, External Clock Oscillator (Main / Sub) is automatically OFF or set to allow user setting.
	0	MOSC / SOSC automatically OFF in Power-down mode
	1	MOSC / SOSC User Setting in Power-down mode
2	VDCLP	HSI enablement in DEEP-SLEEP mode
	0	Normal VDC mode in power-down
	1	Low power VDC mode in power-down
1 0	PMUMODE	Indicates the current operating mode of the microcontroller, that the value of this mode when you write. However, writing '11' is prohibited.
	00	RUN mode
	01	Sleep mode
	10	Power-down mode
	11	Initialization mode (Writing prohibited)

NOTE:

To enter the power-down mode, the main clock must be set before the RingOSC.(BCCR = 0x00)

4.8.6 PMU_CFGR: PMU Configuration Register

This register supports the software reset function and the STBYO output polarity selection feature.

PMU_CFGR=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				STBYOP	Reserved	Reserved	Reserved	SOFTRST
-				0	-	-	-	O
-				RW	-	-	-	WO

4	STBYOP	Output polarity of STBY pin (except for sleep mode)	
		0	Low-Active output in power-down
		1	High-Active output in power-down
0	SOFTRST (WARMRESET)	Setting for internal soft-reset	
		0	Normal operation
		1	Internal soft-reset (auto-clear)

4.8.7 PMU_WSER: PMU Wakeup Source Enable Register

PMU_WSER provides a wake-up function that allows the A33G53x to change its operation to the Run state after entering the sleep mode or power-down mode. The user can choose GPIOA to GPIOF, FRT, WDT, main oscillator, LVD as wakeup source. To wake up using a specific wake-up source, the wake-up source bit must be set to '1'.

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			GPIOFE	GPIOEE	GPIODE	GPIOCE	GPIOBE	GPIOAE	FRTE	WDTE	Reserved	MXFAILE	LVDE
-			0	0	0	0	0	0	0	0	-	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW

10	GPIOFE	Whether to 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.
9	GPIOEE	Whether to 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.
8	GPIODE	Whether to 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.
7	GPIOCE	Whether to 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.
6	GPIOBE	Whether to 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.
5	GPIOAE	Whether to 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.
4	FRTE	Whether to use the FRT 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	WDTE	Whether to 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.
1	MXFAILE	Whether to use the MXFAIL 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	LVDE	Whether to use the LVD 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.

4.8.8 PMU_WSSR: PMU Wakeup Source Status Register

This register reports the event source of the recent wake-up condition. When a wake-up event occurs, relevant bit will be '1'. The condition will be cleared to '0' by writing '1' to the relevant bit or by clearing the event.

PMU_WSSR=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			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT	WDT	Reserved	MXFAIL	LVD
-			0	0	0	0	0	0	0	0	-	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW

10	GPIOF	Whether the GPIOG port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
9	GPIOE	Whether the GPIOE port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
8	GPIOD	Whether the GPIOD port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
7	GPIOC	Whether the GPIOC port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
6	GPIOB	Whether the GPIOB port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
5	GPIOA	Whether the GPIOA port event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
4	FRT	Whether the FRT event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
3	WDT	Whether the WDT event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
1	MXFAIL	Whether the MXFAIL event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.
0	LVD	Whether the LVD event has been triggered for wake-up
	0	The wake-up event has not been triggered.
	1	The wake-up event has been triggered.

4.8.9 PMU_RSER: PMU Reset Source Enable Register

PMU_RSER is a register to set the microcontroller reset event. When the bit of the corresponding reset source is set to '1', the microcontroller is reset when an event condition occurs.

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			MCKFAILRSTE	RSTRSTE	SYSRSTE	SWRSTE	WDTRSTE	Reserved	MXFAILRSTE	LVDRSTE
-			0	1	1	0	1	-	0	1
-			RW	RW	RW	RW	RW	-	RW	RW

7	MCKFAILRSTE	Whether to enable or disable the main clock oscillation failure reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
6	RSTRSTE	Whether to enable or disable the external pin reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
		NOTE: To use PC6 (nRESET) as a GPIO, you must disable the RSTRSTE bit in the PMU RSER register before changing to GPIO.
5	SYSRSTE	Whether to enable or disable the core system reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
4	SWRSTE	Whether to enable or disable the software reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
3	WDTRSTE	Whether to enable or disable the WDT reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
1	MXFAILRSTE	Whether to enable or disable the external main oscillator (XTAL) error reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
0	LVDRSTE	Whether to enable or disable the LVD reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.

4.8.10 PMU_RSSR: PMU Reset Source Status Register

The PMU_RSSR register is a register that informs the source of the reset event when the PMU (Power Management Unit) has reset by an event reset. This register is set to '1' when an event of the corresponding reset source occurs, and cleared when the bit is set to '0'.

PMU_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			MCKFAILRST	RSTRST	SYSRST	SWRST	WDTRST	Reserved	MXFAILRST	LVDRST
-			0	0	0	0	0	-	0	0
-			RW	RW	RW	RW	RW	-	RW	RW

7	MCKFAILRST	Whether the main clock oscillation failure reset has occurred
	0	Read: The reset has not occurred. Write: N/A
	1	Read: The reset has occurred. Write: Clears the flag.
6	RSTRST	Whether 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.
5	SYSRST	Whether the core system reset has occurred
	0	Read: The reset has not occurred. Write: N/A
	1	Read: The reset has occurred. Write: Clears the flag.
4	SWRST	Whether 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.
3	WDTRST	Whether 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.
1	MXFAILRST	Whether the external main oscillator (XTAL) 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	Whether 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.

4.8.11 PMU_PERR: PMU Peripheral Event Reset Register

The PMU_PERR register is used by the Power Management Unit (PMU) to reset peripherals in the microcontroller when an event reset occurs. An event reset can determine whether to reset each peripheral. If a peripheral bit is set to '1', the peripheral is reset when a reset event occurs. If set to '0', a reset event does not affect the peripheral.

PMU_PERR=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
JTAGEN	Reserved	PMC	ADC	Reserved	Reserved	PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0	Reserved	CRC16	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	TIMER69	TIMER25	TIMER01	FRT	WDT	Reserved	Reserved	Reserved
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	-	-	-
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	-	-	-

31	JTAGEN	Enable JTAG reset
29	PMC	Enable PMC (Port Map Controller) reset
28	ADC	Enable ADC reset
25	PWM47	Enable PWM4, 5,6,7 reset
24	PWM03	Enable PWM0, 1,2,3 reset
23	UART3	Enable UART3 reset
22	UART2	Enable UART2 reset
21	UART1	Enable UART1 reset
20	UART0	Enable UART0 reset
19	I2C1	Enable I2C1 reset
18	I2C0	Enable I2C0 reset
17	SPI1	Enable SPI1 reset
16	SPI0	Enable SPI0 reset
14	CRC16	Enable CRC16 reset
13	GPIOF	Enable GPIOF reset
12	GPIOE	Enable GPIOE reset
11	GPIOD	Enable GPIOD reset
10	GPIOC	Enable GPIOC reset
9	GPIOB	Enable GPIOB reset
8	GPIOA	Enable GPIOA reset
7	TIMER69	Enable TIMER 6, 7, 8, 9 reset
6	TIMER25	Enable TIMER 2, 3, 4, 5 reset
5	TIMER01	Enable TIMER 0, 1 reset
4	FRT	Enable Free-run Timer reset
3	WDT	Enabled Watchdog Timer reset

4.8.12 PMU_PER: PMU Peripheral Enable Register

The PMU_PER register is used to set whether to use peripherals of the microcontroller. When a specific bit is set to '1', the relevant peripheral will be enabled. When a specific bit is set to '0', the bit peripheral will be disabled and set to the Stop (Reset) state. To enable peripherals for an application, you must set to '1' the peripherals bits in the PMU_PER and PMU_PCCR registers.

PMU_PER=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
JTAGEN	Reserved	PMC	ADC	Reserved	Reserved	PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0	Reserved	CRC16	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	TIMER69	TIMER25	TIMER01	FRT	WDT	Reserved	Reserved	Reserved
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	-	-	-
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	-	-	-

31	JTAGEN	Enable / Reset JTAG
29	PMC	Enable / Reset PMC (Port Map Controller)
28	ADC	Enable / Reset ADC
25	PWM47	Enable / Reset PWM 4, 5, 6, 7
24	PWM03	Enable / Reset PWM 0, 1, 2, 3
23	UART3	Enable / Reset UART3
22	UART2	Enable / Reset UART2
21	UART1	Enable / Reset UART1
20	UART0	Enable / Reset UART0
19	I2C1	Enable / Reset I2C1
18	I2C0	Enable / Reset I2C0
17	SPI1	Enable / Reset SPI1
16	SPI0	Enable / Reset SPI0
14	CRC16	Enable / Reset CRC16
13	GPIOF	Enable / Reset GPIOF, PMC_F
12	GPIOE	Enable / Reset GPIOE, PMC_E
11	GPIOD	Enable / Reset GPIOD, PMC_D
10	GPIOC	Enable / Reset GPIOC, PMC_C
9	GPIOB	Enable / Reset GPIOB, PMC_B
8	GPIOA	Enable / Reset GPIOA, PMC_A
7	TIMER69	Enable / Reset TIMER 6, 7, 8, 9
6	TIMER25	Enable / Reset TIMER 2, 3, 4, 5
5	TIMER01	Enable / Reset TIMER 0, 1
4	FRT	Enable / Reset Free-run Timer
3	WDT	Enable / Reset Watchdog Timer

4.8.13 PMU_PCCR: PMU Peripheral Clock Control Register

The PMU_PCCR register is a register that sets the clock supply to the internal peripheral device. When the bit is set to 1, the peripheral device is clocked. Writing a 0 to the bit will stop the clock supply to the device and cause the device to stop functioning. To use the peripherals required by your application, you must set the bits of the peripheral device to '1' in the PMU_PER register and the PMU_PCCR register. Minimizing the clock supply by deactivating the clocks of peripheral devices not used by user applications can reduce the microcontroller's current consumption.

PMU_PER=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
Reserved			ADC	Reserved		PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0	Reserved	CRC16	Reserved					GPIO	TIMER69	TIMER25	TIMER01	FRT	WDT	Reserved		
-			0	-		0	0	0	0	0	0	0	0	0	0	-	0	-					1	0	0	0	1	1	-		
-			RW	-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	-					RW	RW	RW	RW	RW	RW	-		

28	ADC	Enable / Disable ADC clock
25	PWM47	Enable/Disable PWM4, 5,6,7 clock
24	PWM03	Enable/Disable PWM0, 1,2,3 clock
23	UART3	Enable/Disable UART3 clock
22	UART2	Enable/Disable UART2 clock
21	UART1	Enable/Disable UART1 clock
20	UART0	Enable/Disable UART0 clock
19	I2C1	Enable/Disable I2C1 clock
18	I2C0	Enable/Disable I2C0 clock
17	SPI1	Enable/Disable SPI1 clock
16	SPI0	Enable/Disable SPI0 clock
14	CRC16	Enable/Disable CRC16 clock
8	GPIO	Enable/Disable GPIO clock
7	TIMER69	Enable/Disable TIMER6, 7,8,9 clock
6	TIMER25	Enable/Disable TIMER2, 3,4,5 clock
5	TIMER01	Enable/Disable TIMER0, 1 clock
4	FRT	Enable/Disable Free-run Timer clock
3	WDT	Enable/Disable Watch-dog Timer clock

4.8.14 PMU_CCR: PMU Clock Control Register

The PMU_CCR register can control the clock oscillation of the microcontroller internal and external oscillators. The internal oscillator has a RINGOSC that outputs a 1 MHz waveform and an internal oscillator (IOSC16) that outputs 16 MHz. The external oscillator uses the output signals of the external main crystal (SXTAL) outputting a 32.768 kHz RTC (Real-Time Clock) and the external main crystal (XTAL) outputting 4 MHz to 10 MHz frequency.

PMU_CCR=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			ROSCEN	IOSC16EN	SXOSCEN	MXOSCEN
-			10	00	00	00
-			RW	RW	RW	RW

7	ROSCEN	Enable internal ring oscillator (RINGOSC)	
		0x	Internal ring oscillator stop
		10	Internal ring oscillator oscillating – 1:1 output
		11	Internal ring oscillator oscillating – 1/2 output
5	IOSC16EN	Enable internal 16MHz oscillator (IOSC16)	
		0x	Internal 16 MHz oscillator stop
		10	Internal 16 MHz oscillator oscillation – 1:1 output
		11	Internal 16 MHz oscillator oscillation – 1/2 output
3	SXOSCEN	Enable external sub-crystal oscillator (SXOSC)	
		0x	External sub-crystal oscillator stop
		10	External sub-crystal oscillator – 1:1 output
		11	External sub-crystal oscillator – 1/2 output
1	MXOSCEN	Enable external main crystal oscillator (MXOSC)	
		0x	External crystal oscillator stop
		10	External crystal oscillator – 1:1 output
		11	External crystal oscillator – 1/2 output

4.8.15 PMU_CMR: PMU Clock Monitoring Register

The PMU_CMR register supports an oscillator activated in the PMU_CCR (PMU Clock Control Register) to monitor or verify the normal oscillation of the external main crystal (MXOSC) using an internal ring oscillator (RINGOSC). MXOSCFAIL (1) occurs when the oscillation of the external main source oscillates abnormally after activating the external main crystal oscillation interrupt.

PMU_CMR=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
Reserved			MXOSCIE	Reserved	MXOSCMNT	Reserved	MXOSCIF	Reserved	MXOSCSTS
-			0	-	0	-	0	-	0
-			RW		RW	-	RWC1		RWC1

14	MXOSCIE	Enable/Disable the interrupt event of monitoring external main crystal oscillation (AND with MXOSCSTS)
	0	Disable
	1	Enable
8	MXOSCMNT	Enable/Disable the monitoring function of external main crystal oscillator
	0	Disable
	1	Enable
6	MXOSCIF	Failure detection flag of external main crystal oscillation (To clear this flag, write '1' to the bit)
	0	No failure detected
	1	Failure detected
0	MXOSCSTS	The oscillation status of external main crystal oscillation (To clear this flag, write '1' to the bit)
	0	No oscillation
	1	Oscillation detected

4.8.16 PMU_MCMR: PMU Main Clock Monitoring Register

This register can set the monitoring function and interrupt function to check the oscillation status of the system main clock (HCLK). When using this register, be sure to set the RECOVER bit to 1 to automatically switch the main clock to the internal ring oscillator when the main clock oscillation error occurs. MCKFAIL (15) interrupt occurs when main clock oscillation becomes abnormal opereation after enabling main clock oscillation monitoring interrupt.

PMU_MCMR=0x4000_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
Reserved		RECOVER	MCKIE	Reserved	MCKMNT	Reserved	MCKIF	Reserved	MCKSTS
-		0	0	-	0	-	0	-	0
-		RW	RW	-	RW	-	RW C1	-	RWC 1

16	RECOVER	Automatically switches the internal ring oscillator when the main clock oscillator error occurred.
	0	Automatic switching function is disable
	1	Automatic switching function is enable
14	MCKIE	Main oscillator monitoring interrupt enable / disable selection (MCKSTS and use)
	0	MMCKFAIL interrupt disable
	1	MMCKFAIL interrupt enable
8	MCKMNT	Main oscillator monitoring enable
	0	Disable
	1	Enable
6	MCKIF	Main oscillator failure detection interrupt flag. (cleared by written with '1')
	0	No failure
	1	Failure was detected
0	MCKSTS	Main oscillator operating state If MCKMNT bit is activated, it will be displayed in real time. (cleared by written with '1').
	0	Main oscillator operation error
	1	Main oscillator is operating normally

4.8.17 PMU_BCCR PMU Bus Clock Control Register

This register is used to set the main system clock (HCLK). By setting the PLLCLKSEL bit value, IOSC16 or external XTAL can be selected as the PLL input clock source. At this time, IOSC16 must be turned on even if external XTAL is set as PLL input.

PMU_BCCR=0x4000_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
Reserved			PCLKDIV	HCLKDIV	Reserved	PLLCLKDIV	PLLCLKSEL	Reserved	MCLKSEL
-			0	00	-	0	0	-	00
-			RW	RW	-	RW	RW	-	RW

10	PCLKDIV	PCLK clock divider setting
	0	PCLK = HCLK
	1	PCLK = HCLK/2
9 8	HCLKDIV	HCLK clock divider setting
	0x	HCLK is System Clock (1:1)
	10	HCLK is System Clock/2
	11	HCLK is System Clock/4
5	PLLCLKDIV	PLL input clock divider
	0	1:1 input
	1	PLL input/2
4	PLLCLKSEL	PLL input clock selection (IOSC16 must be turned on even if external XTAL is set as PLL input)
	0	IOSC16 clock
	1	XTAL clock
1 0	MCLKSEL	Main clock source selection
	00	The ring oscillator
	01	Sub Oscillator
	10	PLL output
	11	PLL bypass (PLL input clock)

4.8.18 PMU_PCSR PMU Peripheral Clock Source Register

This register selects the source clock to be supplied to each peripheral device.

PMU_PCSR=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			T69CS	T25CS	T0CS	FRTCS	WDTCS
-			00	00	00	00	00
-			RW	RW	RW	RW	RW

9 8	T69CS	Select the timer 6,7,8,9 input clock	
		00	Main XTAL
		01	IOSC16 (slower than PCLK/2)
		10	Sub XTAL (slower than PCLK/2)
		11	RingOSC (slower than PCLK/2)
7 6	T25CS	Select timer 2,3,4,5 input clock	
		00	Main XTAL
		01	IOSC16 (slower than PCLK/2)
		10	Sub XTAL (slower than PCLK/2)
		11	RingOSC (slower than PCLK/2)
5 4	T01CS	Select timer 0,1 input clock	
		00	Main XTAL
		01	IOSC16 (slower than PCLK/2)
		10	Sub XTAL (slower than PCLK/2)
		11	RingOSC (slower than PCLK/2)
3 2	FRTCS	Select FRT input clock	
		00	Main XTAL
		01	IOSC16
		10	Sub XTAL
		11	RingOSC
1 0	WDTCS	Select WDT input clock	
		00	Main XTAL
		01	IOSC16
		10	Sub XTAL
		11	RingOSC

4.8.19 PMU_COR PMU Clock Output Register

The PMU_COR register is used to set the CLKO and TRACE functions.

The CLKO function is used to monitor the internal main system clock through the CLKO port.

The TRACE function represents the TRACE function of the debugger.

When changing the clock or delay of TRACE, you must write the TRACEDIVKEY key value together. TRACECLK INV and TRACEDATA DELAY are used to fine-tune the TRACE signal.

PMU_COR=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		TRACEDIVKEY	Reserved	TRACECLKINV	TRACEDATADEL	CLKOSEL	CLKOEN	CLKODIV
-		0000	-	0	0	0	0	0000
-		WO	-	RW	RW	RW	RW	RW

15 12	TRACEDIVKEY	To change the value of TRACECLKINV, you must use “0xA” at the same time.
		0xA TRACE key value when changing the trace clock (TRACECLKINV) or data delay (TRACEDATADELAY)
7	TRACECLKINV	TRACECLK output Inversion.
		0 Original TRACECLK output
		1 TRACECLK output Inversion
6	TRACEDATADELAY	Internal TRACE data delay selection
		0 TRACE delay is 1/1 output (No delay)
		1 TRACE delay is 1/4 output
5	CLKOSEL	Select the CLKO output source
		00 CLKO output is PLL Clock
		01 CLKO output is MCLK
3 0	CLKODIV	CLKOEN is activated, the external output clock, set the value of the division of CLKO. Clock divider will be calculated by the following formula.
		If CLKODIV = 0, CLKO = PLLOUT (or MCLK) If CLKODIV > 0, $CLKO(Hz) = \frac{\text{PLLIOUT or MCLK}}{2 \times \text{CLKODIV}}$

4.8.20 PMU_PLLCON PLL Control Register

PMU_PLLCON is a register for PLL frequency output. It can output precise frequency in 1 MHz unit by setting the value of PLL block of A33G53x. To control the PLL, first write 0x80750000 in the PMU_PLLCON register and calculate the parameter value for the PLL output frequency according to the following formula and write to each corresponding bit. See chapter 4.4.12 for setting the PLL frequency.

PMU_PLLCON=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	VCOMODE	MULT / DIV	nPLLRST	PLLEN	nBYPASS	LOCKSTS	Reserved	PREDIV	Reserved	POSTDIV
-	0	00000000 / 0000	0	0	0	0	-	000	-	0000
-	RW	RW / RW	RW	RW	RW	RO	-	RW	-	RW

28	VCOMODE	To set the operation mode of the VCO Determine the D value of the frequency calculation
		0 Determine the D value of the frequency calculation
		1 Normal VCO mode
27 20	MULT	Set the output multiplication of VCO Determine the N1 value of the frequency calculation
19 16	DIV	Set the output multiplication of VCO Determine the N2 value of the frequency calculation
15	nPLLRST	PLL reset
		0 PLL reset signal
		1 Release PLL reset signal PLL (PLL is enabled.)
14	PLLEN	Enable/Disable PLL Operation
		0 disable
		1 enable
13	nBYPASS	Select PLL output clock
		0 Outputs by bypassing the PLL input clock (FIN). Refer to PMU_BCCR[5:4] in 4.8.17
		1 Outputs PLL frequency (FOUT) calculated by the PLL formula. Refer to 4.4.5
12	LOCKSTS	PLL lock status
		00 PLL is unlock (Unstable output of PLL clock)
		01 PLL is locked (Stable output of PLL clock)
10 8	PREDIV	Pre-Divider division control Determine the R value of the frequency calculation
3 0	CLKODIV	Post divider control. Determine the P value of the frequency calculation
		The final output PLL frequency can be calculated by the following formula. $\mathrm{FOUT} = \frac{\mathrm{PLLINCLK} \times (\mathrm{N}_1 + 1)}{(\mathrm{R}+1) \times (\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1)$

4.8.21 PMU_LVDCON LVD Control Register

This register controls the LVD (Low-Voltage Detector) operation of the microcontroller. After enabling the LVD function, an LVD reset or interrupt will occur if the system voltage detects the level set at LVDIL or LVDRL. This register must be set at the beginning of the main code. (see LVD reset - LVR)

PMU_LVDCON=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		LVDEN	LVDRF	Reserved		LVDREN	LVDRL	Reserved	LVDIF	LVDICS	Reserved	LVDIEN	LVDIL
-		1	0	-		1	000	-	0	0	-	0	000
-		RW	RC	-		RW	RW	-	RC	RO	-	RW	RW

15	LVDEN	Enable/Disable LVD (Low-Voltage Detection)
	0	Disable
	1	Enable
15	LVDRF	LVD reset flag (cleared when written ‘1’)
	0	LVD reset voltage has NOT been detected
	1	LVD reset voltage has been detected
14	LVDREN	Enable/Disable LVD reset
	0	Disable
	1	Enable
10 8	LVDRL	Select LVD Reset voltage level
	000	2.60 V (default)
	001	2.80 V
	010	3.00 V
	011	3.30 V
	100	3.75 V
	101	4.00 V
	110	4.25 V
	111	4.50 V
6	LVDIF	LVD interrupt flag (cleared when writing ‘1’)
	0	LVD interrupt has NOT been detected
	1	LVD interrupt has been detected
5	LVDICS	LVD interrupt current status
	0	Below LVD interrupt voltage
	1	Above LVD interrupt voltage
3	LVDIEN	Enable LVD Interrupt
	0	LVD interrupt disable
	1	LVD interrupt enable
2 0	LVDIL	Select LVD Interrupt voltage level
	000	2.60 V (default)
	001	2.80 V
	010	3.00 V
	011	3.30 V
	100	3.75 V
	101	4.00 V
	110	4.25 V
	111	4.50 V

4.8.22 PMU_VDCCON VDC/LVD Trimming Register

PMU_VDCCON is a register that sets the VDC built in the microcontroller. This register has the internal configuration information of the chip.

This register is not available to general users. (Write prohibited)

PMU_VDCCON=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
DFLVL	Reserved	BMR_TRIM	Reserved	BOPENB	Reserved	VDROP	DFLVL_EN	VDROP_EN	BMRT_EN	LVDT_EN	VDCT_EN	VDCD_EN	LVDTRIM	VDCTRIM	VDCDELAY
0	-	00	-	0	-	0	00	00	0	0	0	0	0000	0000	FF
RW	-	RW	-	RW	-	RW	WO	WO	WO	WO	WO	WO	RW	RW	RW

31	DFLVL	BOD default level (hidden) signal Setting
	0	BODRSEL_I/BODISEL_I Enable (BOD Level = 2.6~4.5 V)
	1	BODRSEL_I/BODISEL_I Disable (BOD Level = 2.2 V)
29 28	BMRTRIM	BMR voltage trimming value Setting
	00	0.9860 V
	01	1.0736 V
	10	0.9173 V
	10	0.9394 V
26	BOPENB	BURST OFF / PULLUP enable / Circuit OFF (Changeable by writing '10 ' to PFLVL_EN field)
	0	Pull-up circuit Enable
	1	Pull-up circuit Disable
24	VDROP	VDC Level Down Mode
	0	VDD = 1.5 V
	1	VDD = 1.39 V (8% down)
23 22	DFLVL_EN	Maintain/Change DFLVL value (Write Only, The value currently being written is applied)
	00 11	Maintain the value of DFLVL, BOPENB
	10	Changeable BOPENB value
	01	Changeable DFLVL value
21 20	VDROP_EN	Maintain/Change VDROP value
	00 x1	Maintain VDROP value
	10	VDROP value can be changed
19	BMRT_EN	Maintain/Change BMRTRIM value
	0	Maintain BMRTRIM value
	1	Changeable BMRTRIM value
18	LVDT_EN	Maintain/Change LVDTRIM (Write Only, The value currently being written is applied)
	0	Maintain LVDTRIM value
	1	Changeable LVDTRIM value
17	VDCT_EN	Maintain/Change VDCTRIM value
	0	Maintain VDCTRIM value
	1	Changeable VDCTRIM value
16	VDCD_EN	Maintain/Change VDCDELY value

		0	Maintain VDCDELY value
		1	Changeable VDCDELY value
15 12	LVDTRIM	Setting LVD (Low-Voltage Detector) voltage trimming value	
11 8	VDCTRIM	Setting VDC voltage trimming value	
7 0	VDCDELAY	Setting VDC output delay time during wake-up from power-down mode	

4.8.23 PMU_IOSC16TRIM IOSC16 Trimming Register

PMU_IOSC16TRIM register is for manufacturing purposes only and is not allowed to write in normal use. Writing to this register may cause the microcontroller to behave abnormally.

PMU_IOSC16TRIM=0x4000_005C

31 30 29 28 27 26 25 24	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	LT_EN	LTM_EN	TSL_EN	UDCH_EN	TCAL_EN	Reserved	LT	LTM	TSL	UDCH	TCAL1
-	0	0	0	0	0	-	1000	10	100	01	011
-	WO	WO	WO	WO	WO	-	RW	RW	RW	RW	RW

20	LT_EN	Maintain/change LT value	
		0	Maintain LT value
		1	Changeable LT value
19	LTM_EN	Maintain/change LTM value	
		0	Maintain LTM value
		1	Changeable LTM value
18	TSL_EN	Maintain/change TSL value	
		0	Maintain TSL value
		1	Changeable TSL value
17	UDCH_EN	Maintain/change UDCH value	
		0	Maintain UDCH value
		1	Changeable UDCH value
16	TCAL_EN	Maintain/change TCAL value	
		0	Maintain TCAL value
		1	Changeable TCAL value
13 10	LT	Period tuning: ±2.8% (default: 1000)	
9 8	LTM	Period tuning: ±0.7% (default: 1000)	
7 5	TSL	Period and temperature tuning: ±1.8% (default: 100)	
4 3	UDCH	Period and temperature tuning (default: 01)	
2 0	TACL1	Temperature tuning (default: 011)	

4.8.24 PMU_EOSCCON External OSC Control Register

This register controls the external oscillator operation.

When changing the field value, PMU_EOSCCON [31:16] must be written with '0x18A2'.

PMU_EOSCCON=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
Reserved		SOSCNFOFF	Reserved	SOSCISEL	Reserved	MOSCNFOF	Reserved	MOSCNFSEL	Reserved	MOSCISEL
-		0	-	00	-	0	-	00	-	01
-		RW	-	RW	-	RW	-	RW	-	RW

15	SOSCNFOFF	Enable or disable Noise Filter of SUB OSC
	0	Enable Nosie Filter
	1	Disable Nosie Filter
13 12	SOSCISEL	Current strength Control of SUB OSC
	00	Strong
	. .	
	11	Weak
7	MOSCNFOFF	Enable or disable Noise Filter of MAIN OSC
	0	Enable Nosie Filter
	1	Disable Nosie Filter
5 4	MOSCNFSEL	Noise Filter Control of MAIN OSC
	00	Strong
	:	
	11	Fast
1 0	MOSCISEL	Current strength Control of MAIN OSC
	00	Strong
	:	
	11	Fast

4.8.25 PMU_EXTMODER External Mode Read Register

The PMU_EXTMODER register reads the external pin value in test mode to determine which mode is set.

This register is not available to general users. (write prohibited)

PMU_EXTMODER=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			FMTEST		Reserved	PDON	XRSTON	ROSCAON	SCANMD	TEST	BOOT
-			0		-	0	0	0	0	0	0
-			WO		-	RO	RO	RO	RO	RO	RO

8	FMTEST	Configure all pins to test mode. (prohibited)
		0 Set all pins to general mode
		1 Set all pins to test mode
5	PDON	Read the value of external PDON pin.
		0 PDON pin = ‘L’
		1 PDON pin = ‘H’
4	XRSTON	Read the value of external XRSTON pin.
		0 XRSTON pin = ‘L’
		1 XRSTON pin = ‘H’
3	ROSCAON	Read the value of external ROSCAON pin.
		0 ROSCAON pin = ‘L’
		1 ROSCAON pin = ‘H’
2	SCANMD	Read the value of external SCANMD pin.
		0 SCANMD pin = ‘L’
		1 SCANMD pin = ‘H’
1	TEST	Read the value of external TEST pin.
		0 TEST pin = ‘L’
		1 TEST pin = ‘H’
0	BOOT	Read the value of external BOOT pin.
		0 BOOT pin = ‘L’
		1 BOOT pin = ‘H’

4.8.26 PMU Register Map Summary

Table 27. PMU 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	VENDIDR	VENDIDR[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	CHIPIDR	CHIPIDR[31:0]																															
	Reset value	0	1	0	0	1	1	0	1	0	0	1	1	0	0	1	1	1	0	1	0	0	0	0	0	x	x	x	x	x	x	x	x
0x08	REVIDR																										REVIDR[7:0]						
	Reset value																										x	x	x	x	x	x	x

Table 28. PMU 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	PMU_IDR	CHIPID[23:0]																								CHIPREV[7:0]							
	Reset value	1	1	0	0	1	1	1	0	1	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	PMU_MR																										PVDCLP	PREVMODE[1:0]		ECLKMD	VDCLP	PMUMODE[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x08	PMU_CFGR																												STBYOP				SWRST
	Reset value																												0				0
0x10	PMU_WSER																						GPIOFE	GPIOEE	GPIODE	GPIOCE	GPIOBE	GPIOAE	FRTE	WDTE		MXFAILE	LVDE
	Reset value																						0	0	0	0	0	0	0	0		0	0
0x14	PMU_WSSR																						GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT	WDT		MXFAIL	LVD
	Reset value																						0	0	0	0	0	0	0	0		0	0
0x18	PMU_RSER																									MCKFAILRST	RSTRSTE	SYSRSTE	SWRSTE	WDTRSTE		MXFAILRSTE	LVDRSTE
	Reset value																									0	1	1	0	1		0	1
0x1C	PMU_RSSR																									MCKFAILRST	RSTRST	SYSRST	SWRST	WDTRST		MXFAILRST	LVDRST
	Reset value																									0	0	0	0	0		0	0

Table 28. PMU 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	PMU_PERR	JTAG		PMC	ADC			PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0		CRC16	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	TIMER69	TIMER25	TIMER01	FRT	WDT			
	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			
0x24	PMU_PER	JTAG		PMC	ADC			PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0		CRC16	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	TIMER69	TIMER25	TIMER01	FRT	WDT			
	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			
0x28	PMU_PCCR				ADC			PWM47	PWM03	UART3	UART2	UART1	UART0	I2C1	I2C0	SPI1	SPI0		CRC16						GPIO	TIMER69	TIMER25	TIMER01	FRT	WDT			
	Reset value				0			0	0	0	0	0	0	0	0	0	0		0						1	0	0	0	1	1			
0x30	PMU_CCR																									ROSCEN [1:0]		IOSC16EN [1:0]		SXOSCEN [1:0]		MXOSCEN [1:0]	
	Reset value																									1	0	0	0	0	0	0	0
0x34	PMU_CMR																		MXOSCIE						MXOSCMNT		MXOSCIF						MXOSCSTS
	Reset value																		0						0		0						0
0x38	PMU_MCMR																	RECOVER	MCKIE						MCKMNT		MCKIF						MCKSTS
	Reset value																	0	0						0		0						0
0x3C	PMU_BCCR																						PCLKDIV	HCLKDIV[1:0]				PLLCLKDIV	PLLCLKSEL			MCLKSEL [1:0]	
	Reset value																						0	0	0		0	0	0			0	0
0x40	PMU_PCSR																							T69CS[1:0]		T25CS[1:0]		T01CS[1:0]		FRTCS[1:0]		WDTCS[1:0]	
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x44	PMU_COR																	TRACEDIVKEY [3:0]								TRACECLKINV	TRACEDATADEL	CLKOSEL	CLKOEN	CLKODIV[3:0]			
	Reset value																	0	0	0	0					0	0	0	0	0	0	0	0
0x50	PMU_PLLCON				VCOMODE	MULT[7:0]								DIV[3:0]				nPLLRST	PLLEN	nBYPASS	LOCKSTS		PREDIV[3:0]							POSTDIV[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
0x54	PMU_LVDCON																	LVDEN	LVDRF			LVDREN	LVDRL[2:0]			LVDIF		LVDICS		LVDIEN		LVDIL[2:0]	
	Reset value																	1	0			1	0	0	0		0	0		0	0	0	0

5. Port Map Controller (PMC)

5.1 Introduction

5.1.1 PMC Introduction

A33G53x series has a Port Map Controller (PMC) module that controls the external input and output (I/O) ports. By setting the PMC registers, users can configure the pins' uses, input / output direction, Pull-up / Pull-down resistor, and debouncing for their applications as needed.

5.2 Features

5.2.1 PMC Features

The Port Map Controller (PMC) configures and controls external I/Os as listed below:

- MUX registers define the use of each pin.
- Direction (Input/Output) setting of each pin
  - Push-pull output
  - Open-drain output
  - Logic input
  - Analog input
- Pull-up/Pull-down resistor setting of 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 F).
- Each pin can be set for debouncing.

5.3 PMC Functional Description

Subject to the specific hardware characteristics of each I/O port, 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
- Input pull-down
- Analog
- 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: nTRST with pull-up
- PC1: TDI with pull-up
- PC2: TMS / SWDIO with pull-up
- PC3: TCK / SWCLK with pull-up
- PC6: nRESET with pull-up

Following figures (Figure 31, Figure 32, Figure 33, Figure 34, Figure 35, Figure 36, Figure 37) show block diagrams of GPIO and external interrupt I/O pins, respectively.

Figure 31. PMC Block Diagram

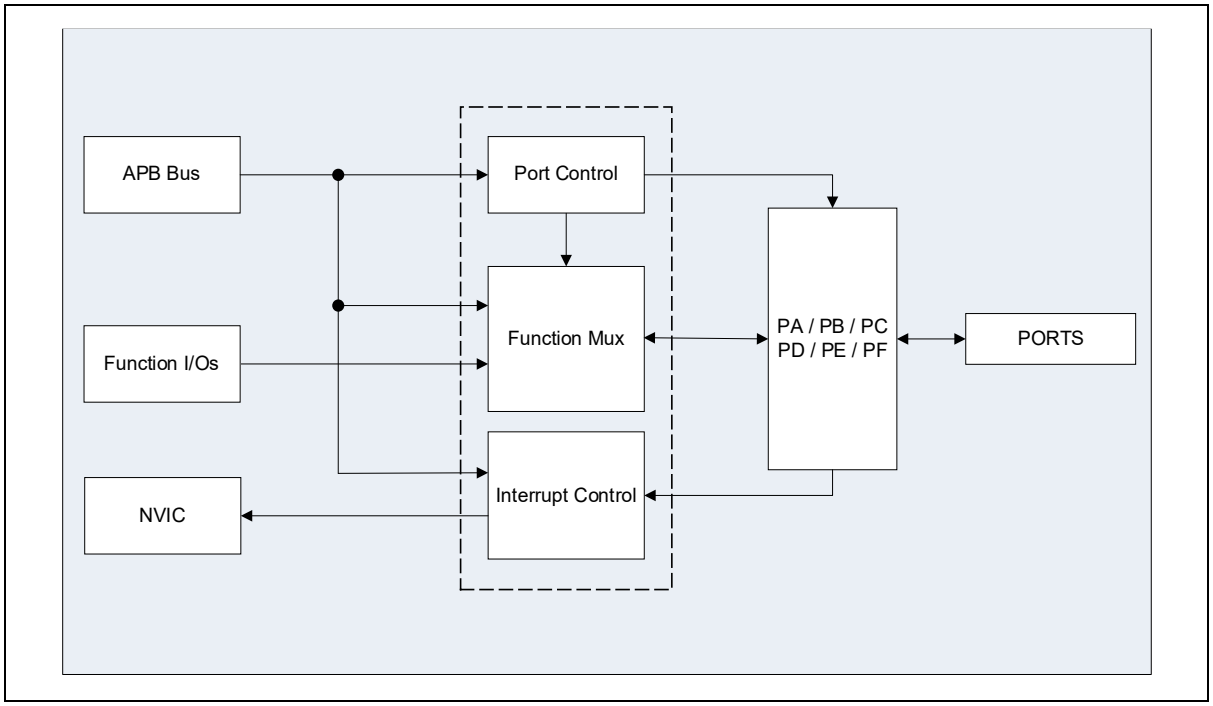


Figure 32. PA Port Block Diagram

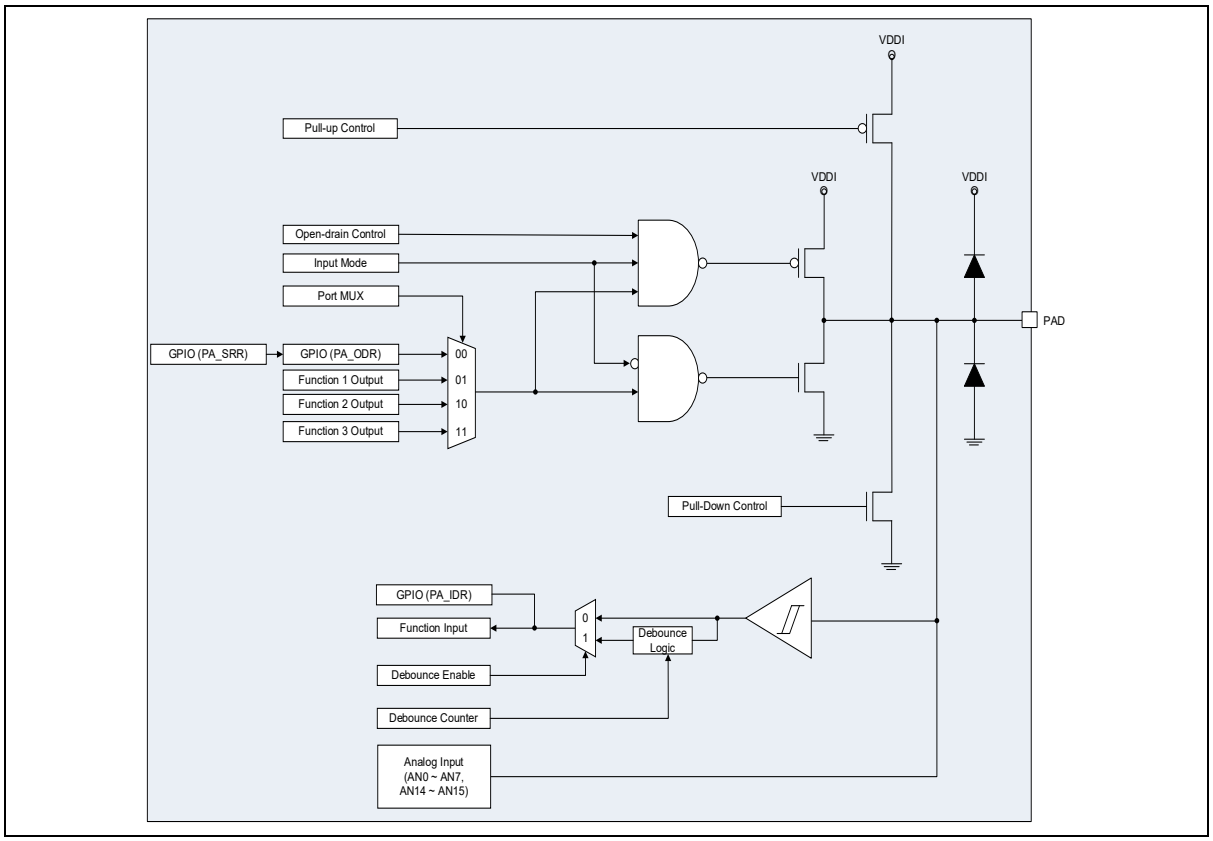


Figure 33. PB Port Block Diagram

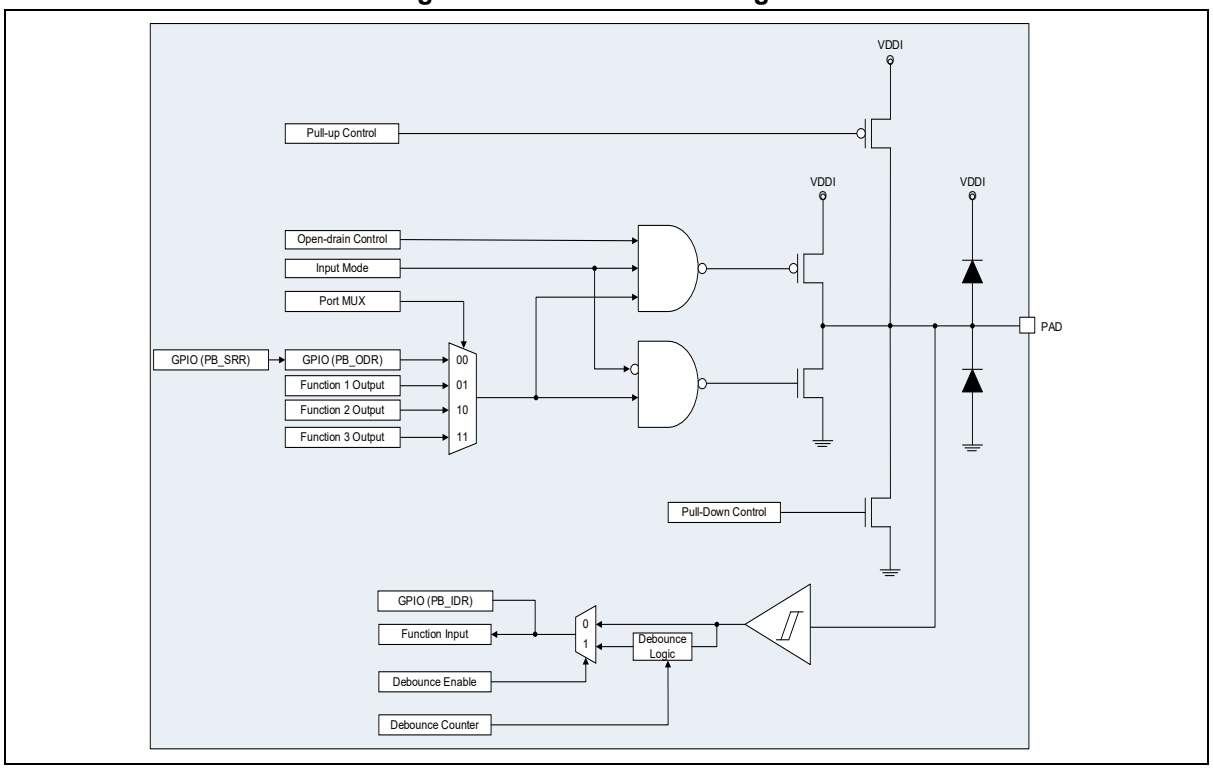


Figure 34. PC Port Block Diagram

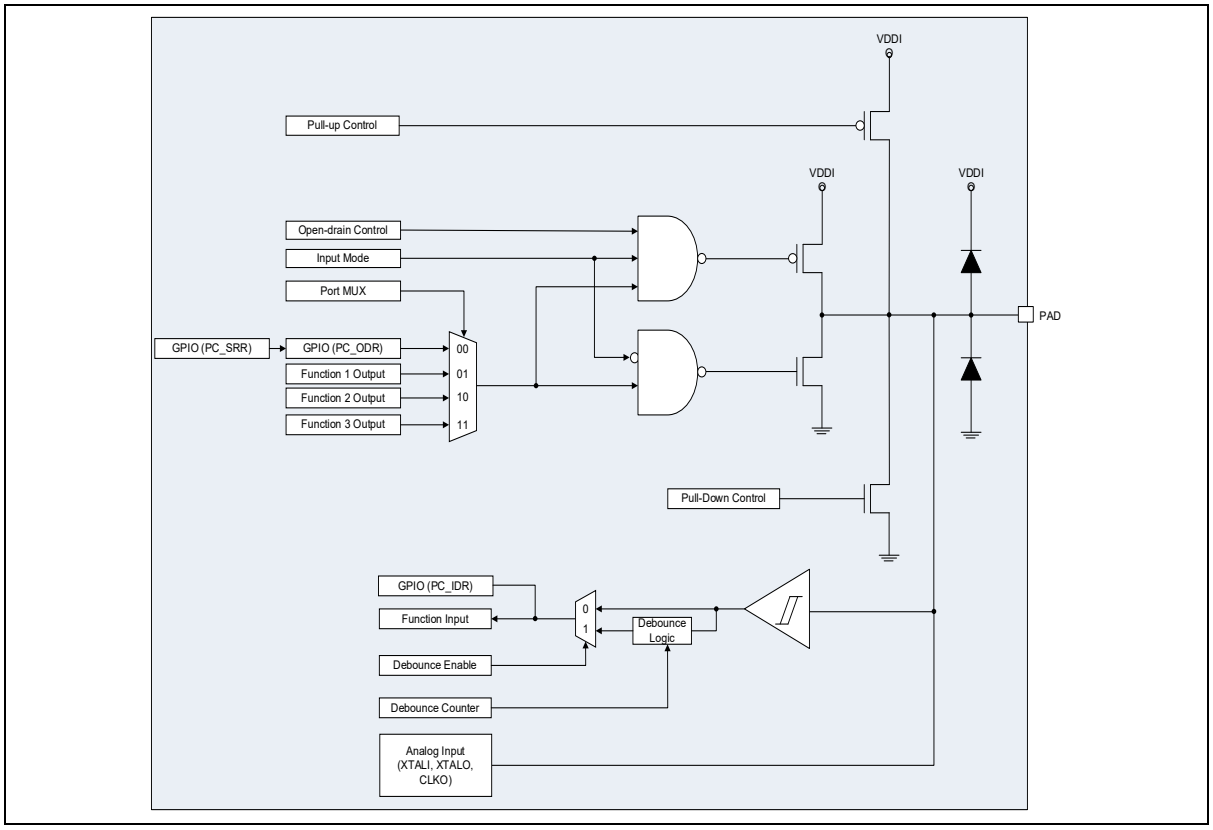


Figure 35. PD Port Block Diagram

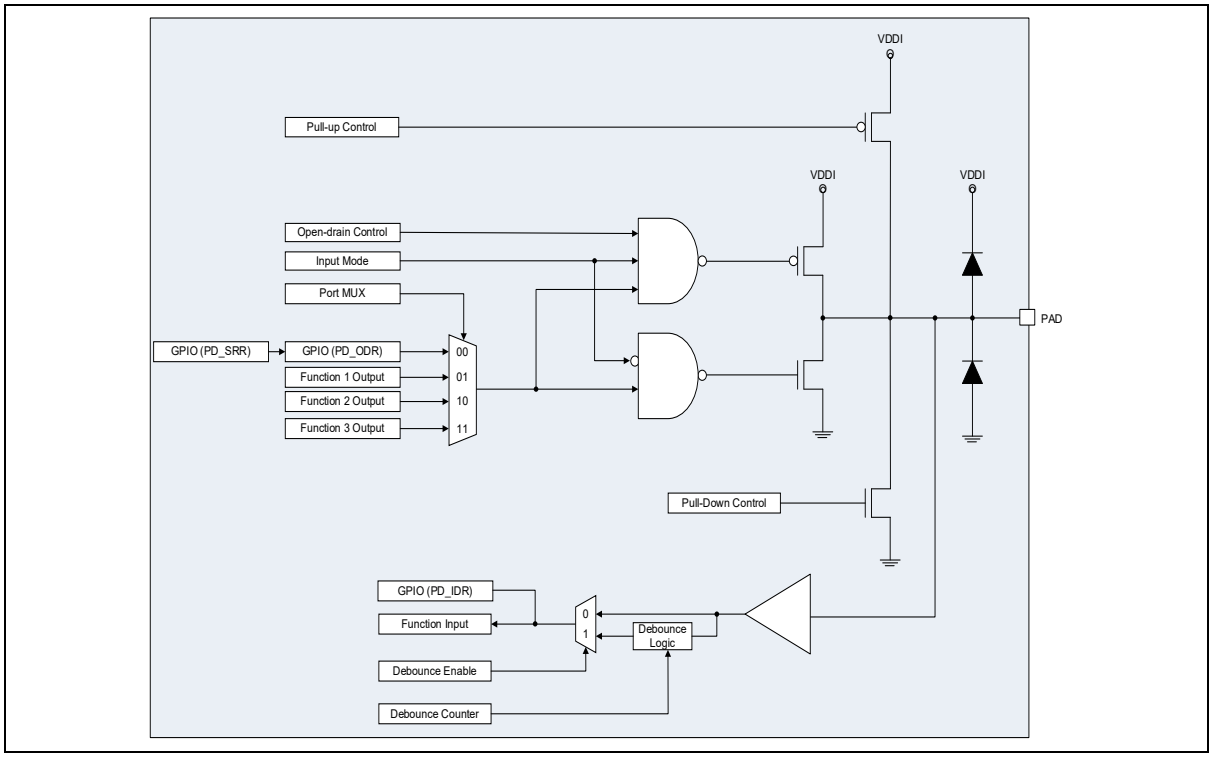


Figure 36. PE Port Block Diagram

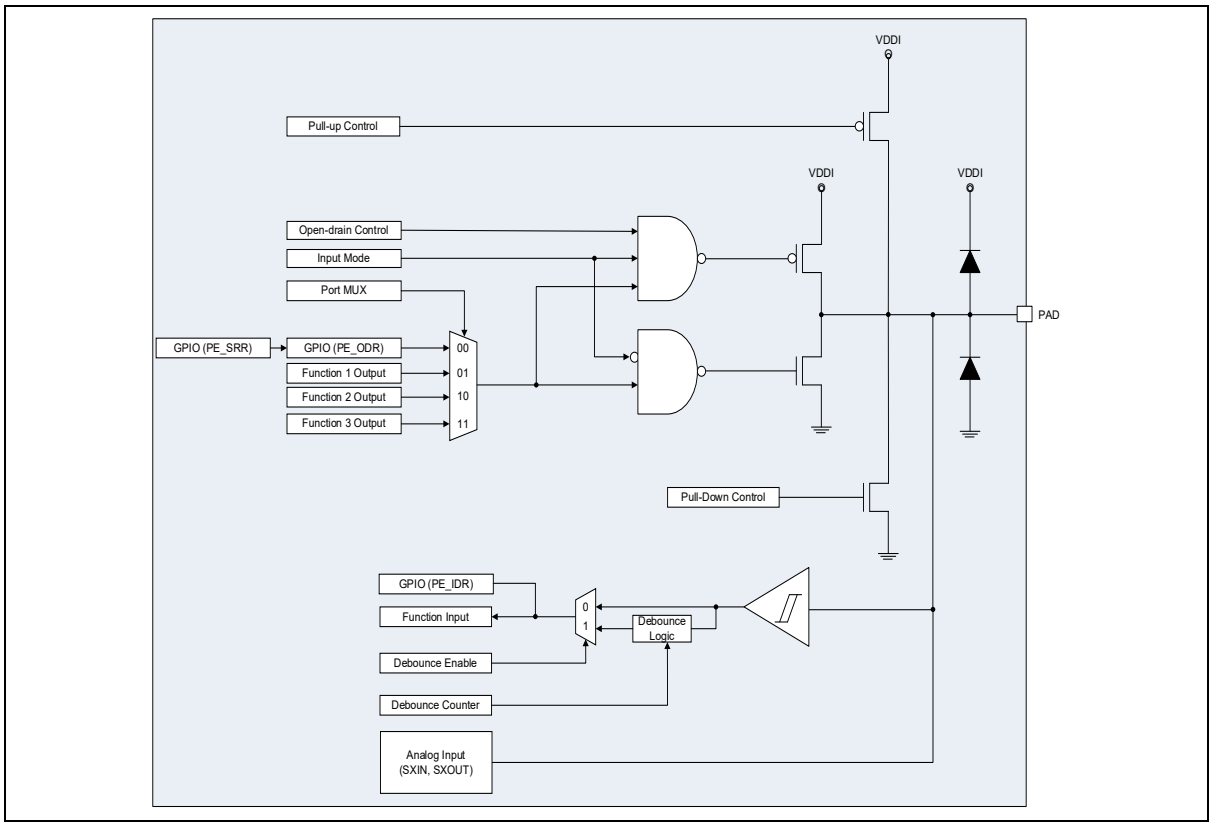
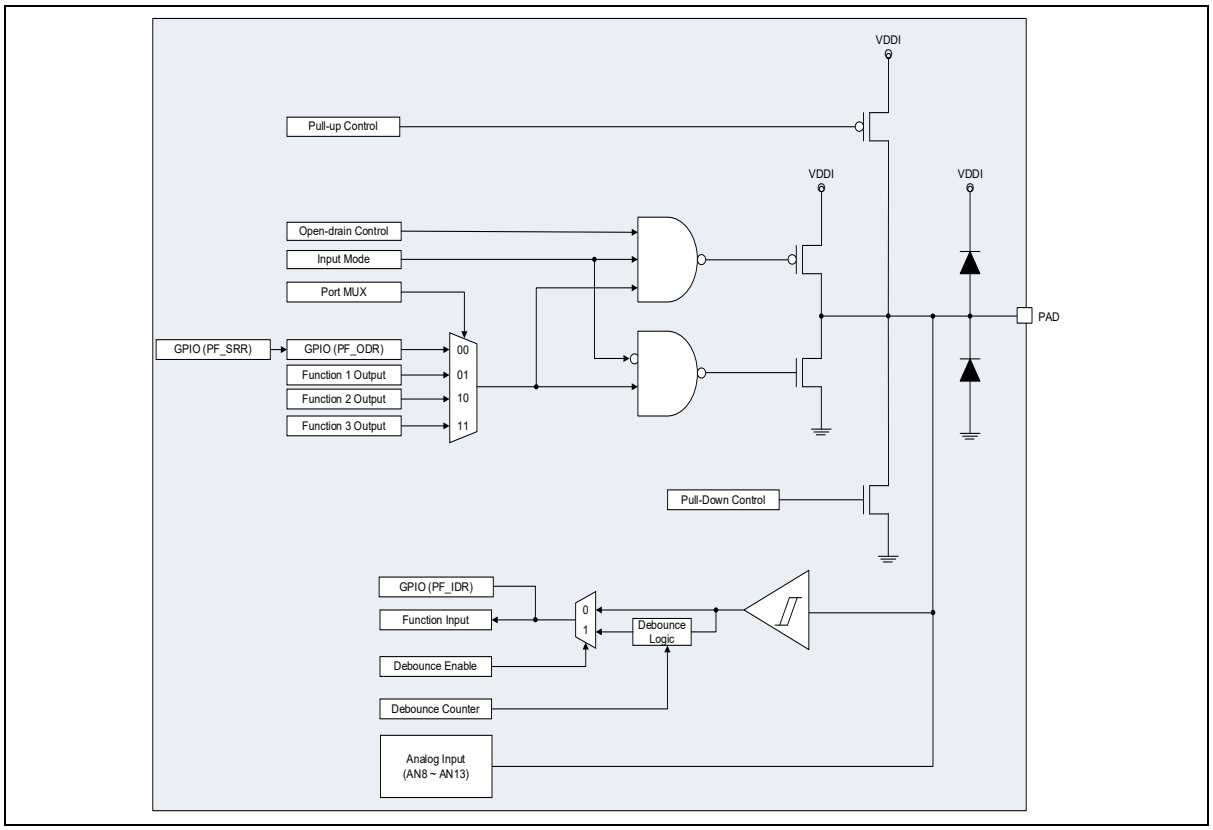


Figure 37. PF Port Block Diagram



5.3.1 PMC and GPIO Pins and Internal Signals

The PMC and GPIO internal common signal is described in Table 29.

Table 29. PMC and GPIO Internal Signal

Signal name	Signal type	Description
APB	Clock	APB clock

The pins assigned for PMC and GPIO module are described in Table 30.

Table 30. PMC and GPIO Pins

Pin name	Type	Description
PA	IO	PA0 to PA15
PB	IO	PB0 to PB15
PC	IO	PC0 to PC15
PD	IO	PD0 to PD15
PE	IO	PE0 to PE9, PE11 to PE15
PF	IO	PF0 to PF5, PF7 to PF11

5.3.2 I/O Port Control Registers

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

- The Pn_CR register selects the I/O mode from the logic Input, analog input (ADC / OSC), output, and output open-drain modes.
- The Pn_PCR register controls the pull-up and pull-down resistors of all ports I/Os.

5.3.3 I/O Port Data Registers

Each GPIO has two 16-bit memory mapped 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. This register is read-only.

5.3.4 I/O Alternative Function

Each GPIO has four 32-bit memory mapping alternative function registers that operate as shown below:

- The Pn_MR register selects an alternative function for each port I/O pin (pin 0 to 15).

NOTE:

Depending on the Pn_MR setting, Pn_CR must also be set. In particular, logic input and analog input can be set for each pin. Pn_MR must be configured as an analog input when using ADCs and oscillators, in which case the logic portion of that pin will be disabled.

Table 31 describes the GPIO alternative functions.

Table 31. GPIO Alternative Function

Pin name	Alternative function			
	00	01	10	11
PA0	PA0 (1)			AN0
PA1	PA1 (1)			AN1
PA2	PA2 (1)			AN2
PA3	PA3 (1)			AN3
PA4	PA4 (1)			AN4
PA5	PA5 (1)			AN5
PA6	PA6 (1)	T0C		AN6
PA7	PA7 (1)	T1C		AN7
PA8	PA8 (1)	T2C		
PA9	PA9 (1)	T3C		
PA10	PA10 (1)	T4C		
PA11	PA11 (1)	T5C		
PA12	PA12 (1)	T6C		
PA13	PA13 (1)	T7C		
PA14	PA14 (1)	T8C		AN14
PA15	PA15 (1)	T9C		AN15
PB0	PB0 (1)	T0O		
PB1	PB1 (1)	T1O		
PB2	PB2 (1)	T2O		
PB3	PB3 (1)	T3O		
PB4	PB4 (1)	T4O		
PB5	PB5 (1)	T5O		
PB6	PB6 (1)	T6O		
PB7	PB7 (1)	T7O		
PB8	PB8 (1)	T8O		
PB9	PB9 (1)	T9O		
PB10	PB10 (1)	SS0		
PB11	PB11 (1)	SCK0		
PB12	PB12 (1)	MOSI0		
PB13	PB13 (1)	MISO0		
PB14	PB14 (1)	SCL0		
PB15	PB15 (1)	SDA0		

Table 31. GPIO Alternative Function (continued)

Pin name	Alternative function			
	00	01	10	11
PC0	PC0	nTRST (1)		
PC1	PC1	TDI (1)		
PC2	PC2	TMS (SWDIO) (1)		
PC3	PC3	TCK (SWCLK) (1)		
PC4	PC4	TDO (SWO) (1)		
PC5	PC5 (1)			
PC6	PC6	nRESET (1)		
PC7	PC7/BOOT (1)			
PC8	PC8 (1)	RXD0		
PC9	PC9 (1)	TXD0		
PC10	PC10 (1)	RXD2		
PC11	PC11 (1)	TXD2		
PC12	PC12 (1)	STBYO		
PC13	PC13 (1)	CLKO		
PC14	PC14 (1)	XTALO		
PC15	PC15 (1)	XTALI	CLKIN	
PD0	PD0 (1)	PWMA0		
PD1	PD1 (1)	PWMA1		
PD2	PD2 (1)	PWMA2		
PD3	PD3 (1)	PWMA3		
PD4	PD4 (1)	PWMA4		
PD5	PD5 (1)	PWMA5		
PD6	PD6 (1)	PWMA6		
PD7	PD7 (1)	PWMA7		
PD8	PD8 (1)	SS1		
PD9	PD9 (1)	SCK1		
PD10	PD10 (1)	MOSI1		
PD11	PD11 (1)	MISO1		
PD12	PD12 (1)	RXD1		
PD13	PD13 (1)	TXD1		
PD14	PD14 (1)	SCL1		
PD15	PD15 (1)	SDA1		

Table 31. GPIO Alternative Function (continued)

Pin name	Alternative function			
	00	01	10	11
PE0	PE0 (1)	PWM0B		
PE1	PE1 (1)	PWM1B		
PE2	PE2 (1)	PWM2B		
PE3	PE3 (1)	PWM3B		
PE4	PE4 (1)	PWM4B		
PE5	PE5 (1)	PWM5B		
PE6	PE6 (1)	PWM6B	RXD3	
PE7	PE7 (1)	PWM7B	TXD3	
PE8	PE8 (1)	SXIN		
PE9	PE9 (1)	SXOUT		
-	-	-	-	-
PE11	PE11 (1)	TraceD3		
PE12	PE12 (1)	TraceD2		
PE13	PE13 (1)	TraceD1		
PE14	PE14 (1)	TraceD0		
PE15	PE15 (1)	TraceCLK		
PF0	PF0 (1)			AN8
PF1	PF1 (1)			AN9
PF2	PF2 (1)			AN10
PF3	PF3 (1)			AN11
PF4	PF4 (1)			AN12
PF5	PF5 (1)			AN13
-	-	-	-	-
PF7	PF7 (1)			
PF8	PF8 (1)			
PF9	PF9 (1)			
PF10	PF10 (1)			
PF11	PF11 (1)			
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

NOTES:

1. This means ‘selected pin function after reset condition’. (The initial value of the pin is different depending on the package type)
2. It is recommended that unused pins are set to output-“Low” by firmware.
3. PE10, PF6, and PF12 to PF15 pins are not available.

5.3.5 Interrupt Functionality

The GPIO module has six interrupt sources, and each port can be an interrupt source. To set each GPIO pins 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 32. Interrupt Sources and Corresponding Pins in GPIO Module

Interrupt Name	Pins	Configurable Pins	Remark
GPIOA	16	Port A[15:0]	Output or Schmitt-trigger input
GPIOB	16	Port B[15:0]	Output or Schmitt-trigger input
GPIOC	16	Port C[15:0]	Output or Schmitt-trigger input
GPIOD	16	Port D[15:0]	Strong output or CMOS input
GPIOE	15	Port E[9:0], Port E[15:11]	Output or Schmitt-trigger input
GPIOF	6	Port F[5:0], Port F[11:7]	Output or Schmitt-trigger input

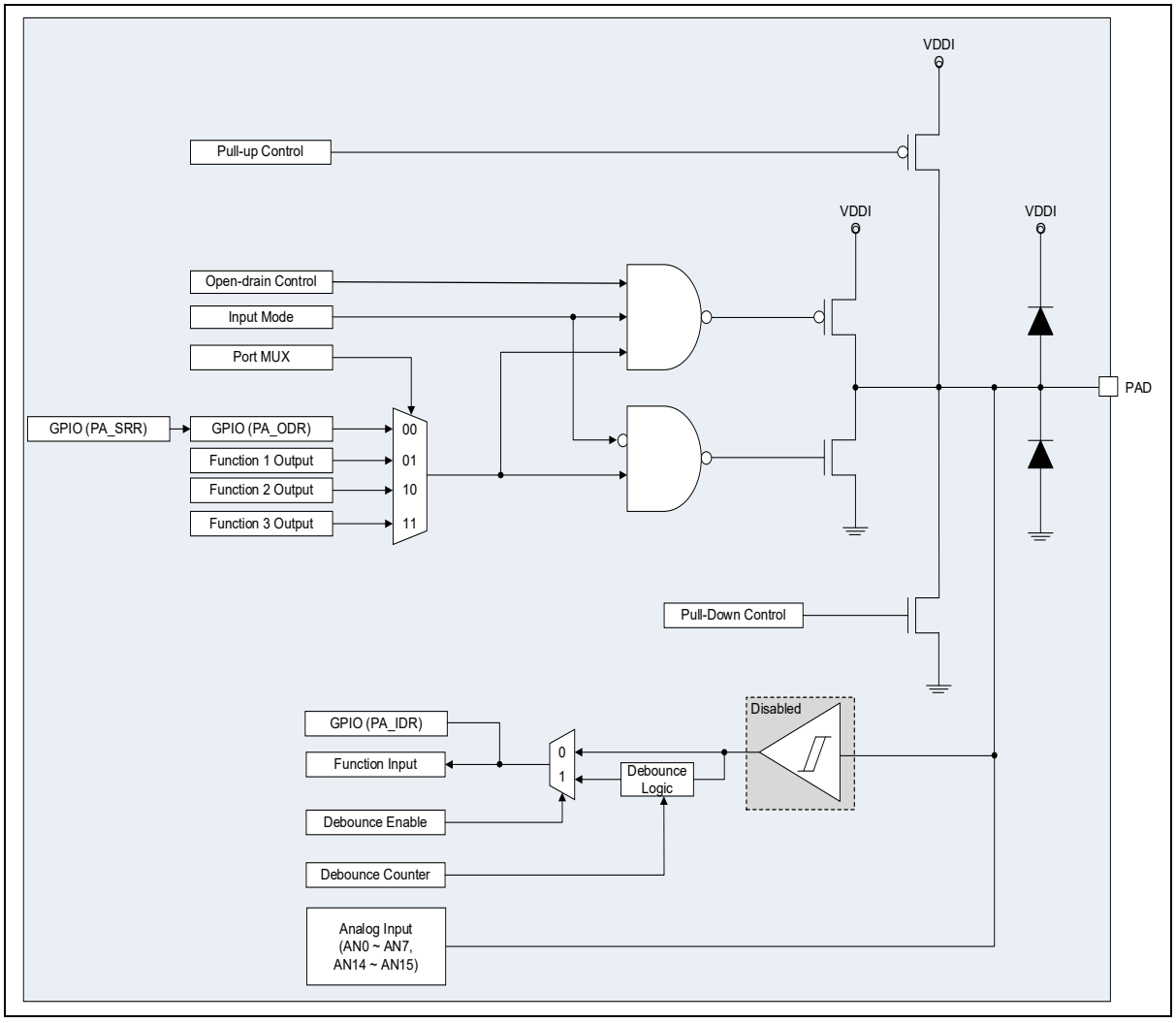
5.3.6 Output Configuration

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

- The push-pull driver is turned on or off depending on the Pn_ODR register or function output value.
- Open-drain mode is available.
- The use of the pull-up and pull-down resistors is enabled by the Pn_PCR register.
- Path of analog signal must be opened for the individual module control.

Figure 38 shows a block diagram of an output port.

Figure 38. Output Port Block Diagram (Example: PA)



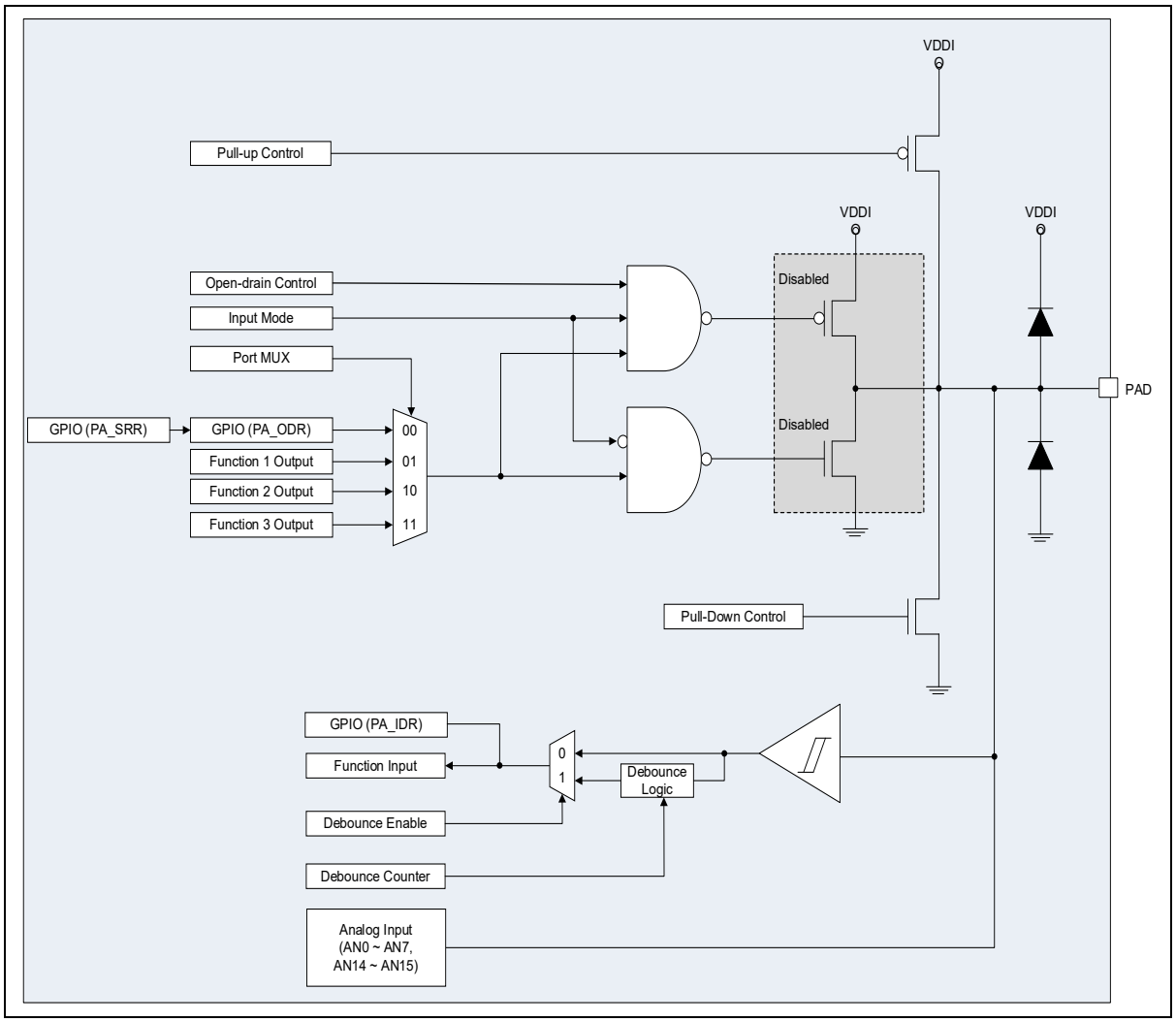
5.3.7 Input Configuration

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

- Output push-pull driver is turned off, and schmitt-trigger input buffer passes the input.
- Input data is displayed in the Pn_IDR register or input signal enters toward functions.
- Path of analog signal must be opened for the individual module control.

Figure 39 shows a block diagram of an input port.

Figure 39. Input Port Block Diagram (Example: PA)

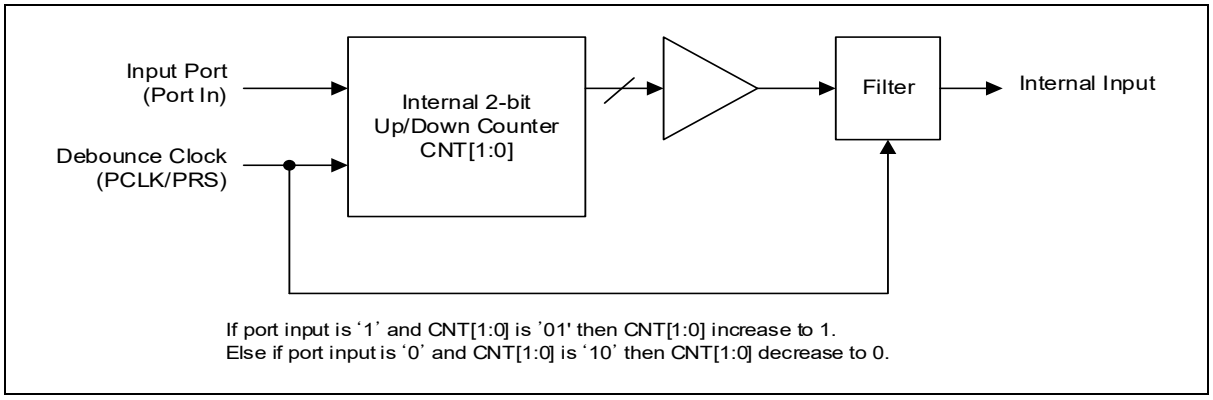


5.3.8 Debouncing Functionality

The input debounce function is supported on each port of the A33G53x. The debounce function is used to filter the interference noise of the input port to filter out the normal signal.

The user can set the Pn_DPR register to adjust the filtering level of each 16-pin port and turn on or off the filter for each pin. The filtering level uses a clock that divides the PCLK clock by a 16-bit counter and can set this value. Figure 40 shows a block diagram of a simplified debouncing logic.

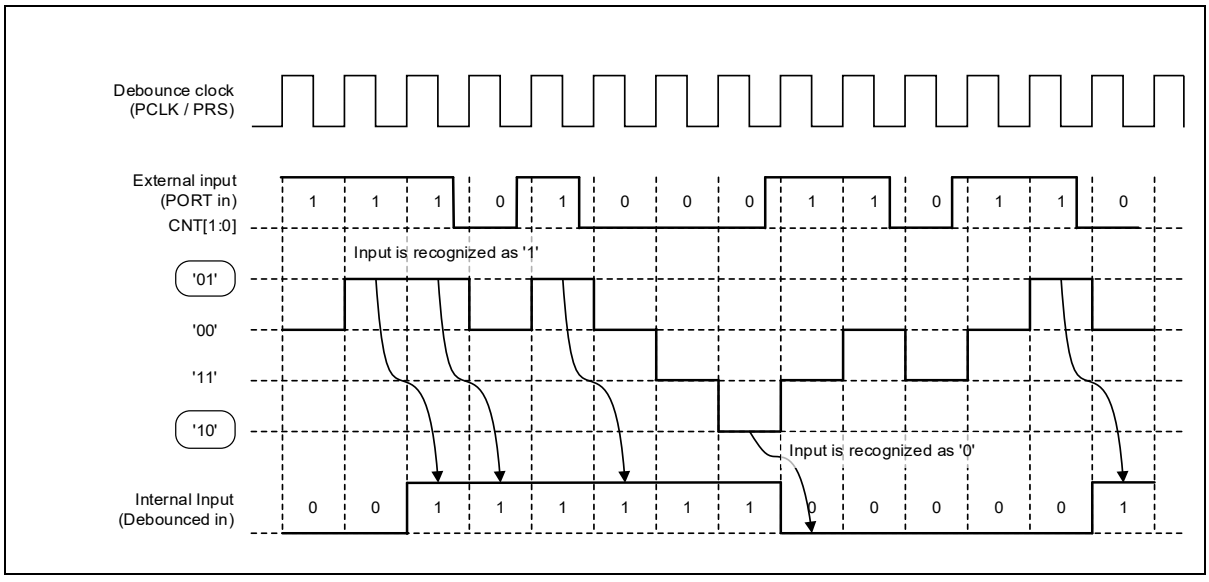
Figure 40. Debouncing Logic Block Diagram



When the debounce logic of the logic input is activated, the debounce function operates on the basis of the prescaled clock by DPR[4: 0] bits of the Pn_DPR register.

Debounce will change the input recognition value if the changed input is maintained for a maximum of three clocks continuously. If the internal counter is '01', the input is set to '1'. If the internal counter is '10', it is set to '0'. Otherwise, it retains its previous value without change. Figure 41 shows a timing diagram of a port debouncing.

Figure 41. Example Timing Diagram of Port Debouncing



5.3.9 Setting Example

<Example 1> Example of edge interrupt setting of PA0 input port

PA_MR<P0[1:0]> = “00”	: Sets PA0 port to GPIO
PA_CR<P0[1:0]> = “10”	: Sets the direction of PA0 port to logic input
PA_PCR<P0>= “1”	: Enables pull-up/pull-down of PA0
PA_PCR<D0>= “0”	: Enables pull-up resistor of PA0
PA_IER<P0[0:1]> = “11”	: Enables edge interrupt of PA0
PA_ICR<P0[0:1]> = “11”	: Enables Rising and Falling edge interrupt
PA_ISR<P0[0:1]> = “11”	: Enables rising and falling edge interrupt event display

< Example 2> Example of debounce of PA port input port

PA_DPR<DPR[0:4]> = “00000”	: Sets debounce prescaler of PA port (53ns @75MHz)
PA_DER<P0>= “1	: Enables debounce function of PA0 port

5.4 PMC Registers

The base addresses and register map of PMC are as follows:

Table 33. Base Addresses of PCU

Name	Base address	Description
PA	0x4000_0200	General Port A
PB	0x4000_0220	General Port B
PC	0x4000_0240	General Port C
PD	0x4000_0260	General Port D
PE	0x4000_0280	General Port E
PF	0x4000_02A0	General Port F

Table 34. PCU and GPIO Register Map

Name	Offset	Type	Description	Reset value	Reference
PA_MR	0x0000	RW	Port A Pin MUX Register	0x00000000	5.4.1
PB_MR	0x0000	RW	Port B Pin MUX Register	0x00000000	5.4.2
PC_MR	0x0000	RW	Port C Pin MUX Register	0x00001155	5.4.3
PD_MR	0x0000	RW	Port D Pin MUX Register	0x00000000	5.4.4
PE_MR	0x0000	RW	Port E Pin MUX Register	0x00000000	5.4.5
PF_MR	0x0000	RW	Port F Pin MUX Register	0x00000000	5.4.6
Pn_CR	0x0004	RW	Port n Control Register (Except PC)	0xFFFFFFFF	5.4.7
PC_CR	0x0004	RW	Port C Control Register	0xFFFFECAA	5.4.8
Pn_PCR	0x0008	RW	Port n Pull-up/Pull-down Control Register (Except PC)	0x00000000	5.4.9
PC_PCR	0x0008	RW	Port C Pull-up/Pull-down Control Register	0x0000004F	5.4.10
Pn_DER	0x000C	RW	Port n Debounce Enable Register	0x00000000	5.4.11
Pn_IER	0x0010	RW	Port n Interrupt Enable Register	0x00000000	5.4.12
Pn_ISR	0x0014	RWC1	Port n Interrupt Status Register	0x00000000	5.4.13
Pn_ICR	0x0018	RW	Port n Interrupt Configuration Reigster	0x00000000	5.4.14
Pn_DPR	0x001C	RW	Port n Debounce Prescaler	0x00000000	5.4.15

NOTE: n = A, B, C, D, E, and F.

5.4.1 PA_MR Port A Pin MUX Register

This register selects a function of the multi-function pins of port A for the purpose.

PA_MR=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
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]
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

Pin name	Multifunction (bit value)			
	00	01	10	11
PA0	PA0			AN0
PA1	PA1			AN1
PA2	PA2			AN2
PA3	PA3			AN3
PA4	PA4			AN4
PA5	PA5			AN5
PA6	PA6	T0C		AN6
PA7	PA7	T1C		AN7
PA8	PA8	T2C		
PA9	PA9	T3C		
PA10	PA10	T4C		
PA11	PA11	T5C		
PA12	PA12	T6C		
PA13	PA13	T7C		
PA14	PA14	T8C		AN14
PA15	PA15	T9C		AN15

5.4.2 PB_MR Port B Pin MUX Register

This register selects a function of the multi-function pins of port B for the purpose.

PB_MR=0x4000_0220

31 30	29 28	27 26	25 24	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]
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

Pin name	Multifunction (bit value)			
	00	01	10	11
PB0	PB0	T0O		
PB1	PB1	T1O		
PB2	PB2	T2O		
PB3	PB3	T3O		
PB4	PB4	T4O		
PB5	PB5	T5O		
PB6	PB6	T6O		
PB7	PB7	T7O		
PB8	PB8	T8O		
PB9	PB9	T9O		
PB10	PB10	SS0		
PB11	PB11	SCK0		
PB12	PB12	MOSI0		
PB13	PB13	MISO0		
PB14	PB14	SCL0		
PB15	PB15	SDA0		

5.4.3 PC_MR Port C Pin MUX Register

This register selects a function of the multi-function pins of port C for the purpose.

PC_MR=0x4000_0240

31 30	29 28	27 26	25 24	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]
00	00	00	00	00	00	00	00	00	01	00	01	01	01	01	01
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Pin name	Multifunction (bit value)			
	00	01	10	11
PC0	PC0	nTRST		
PC1	PC1	TDI		
PC2	PC2	TMS (SWDIO)		
PC3	PC3	TCK (SWCLK)		
PC4	PC4	TDO (SWO)		
PC5	PC5			
PC6	PC6 [1)]	nRESET		
PC7	PC7/BOOT			
PC8	PC8	RXD0		
PC9	PC9	TXD0		
PC10	PC10	RXD2		
PC11	PC11	TXD2		
PC12	PC12	STBYO		
PC13	PC13	CLKO		
PC14	PC14	XTALO		
PC15	PC15	XTALI	CLKIN	

NOTES:

1. To use PC6/nRESET as GPIO, RSTRSTE of PMU RSER register must be disabled. See Figure 24
2. The initial function of the PC0, PC1, PC2, PC3, PC4 pins are JTAG (nTRST, TDI, TMS, TCK, TDO).
3. When PC0/nTRST pin is set to GPIO, JTAG input pins are fixed as below:

Pin Name	Fixed Level	Remark
nTRST	H	JTAG Enable
TDI	H	-
TMS (SWDIO)	H	-
TCK (SWCLK)	H	-
TDO (SWO)	X	-

4. To use SWD (Serial Wire Debug) interface, the port is handled as follows.

Pin Name	Fixed Level	Remark
SWDIO	H	-
SWCLK	H	-
SWO	X	Optional function

5.4.4 PD_MR Port D Pin MUX Register

This register selects a function of the multi-function pins of port D for the purpose.

PD_MR=0x4000_0260

31 30	29 28	27 26	25 24	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]
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

Pin name	Multifunction (bit value)			
	00	01	10	11
PD0	PD0	PWMA0		
PD1	PD1	PWMA1		
PD2	PD2	PWMA2		
PD3	PD3	PWMA3		
PD4	PD4	PWMA4		
PD5	PD5	PWMA5		
PD6	PD6	PWMA6		
PD7	PD7	PWMA7		
PD8	PD8	SS1		
PD9	PD9	SCK1		
PD10	PD10	MOSI1		
PD11	PD11	MISO1		
PD12	PD12	RXD1		
PD13	PD13	TXD1		
PD14	PD14	SCL1		
PD15	PD15	SDA1		

5.4.5 PE_MR Port E Pin MUX Register

This register selects a function of the multi-function pins of port E for the purpose.

PE_MR=0x4000_0280

31 30	29 28	27 26	25 24	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]	Reserved	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]
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

Pin name	Multifunction (bit value)			
	00	01	10	11
PE0	PE0	PWM0B		
PE1	PE1	PWM1B		
PE2	PE2	PWM2B		
PE3	PE3	PWM3B		
PE4	PE4	PWM4B		
PE5	PE5	PWM5B		
PE6	PE6	PWM6B	RXD3	
PE7	PE7	PWM7B	TXD3	
PE8	PE8	SXIN		
PE9	PE9	SXOUT		
PE11	PE11	TraceD3		
PE12	PE12	TraceD2		
PE13	PE13	TraceD1		
PE14	PE14	TraceD0		
PE15	PE15	TraceCLK		

5.4.6 PF_MR Port F Pin MUX Register

This register selects a function of the multi-function pins of port F for the purpose.

PF_MR=0x4000_02A0

31	30	29	28	27	26	25	24	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		Reserved		Reserved		Reserved		P11[1:0]		P10[1:0]		P9[1:0]		P8[1:0]		P7[1:0]		Reserved		P5[1:0]		P4[1:0]		P3[1:0]		P2[1:0]		P1[1:0]		P0[1:0]	
-		-		-		-		00		00		00		00		00		-		00		00		00		00		00		00	
-		-		-		-		RW		RW		RW		RW		RW		-		RW		RW		RW		RW		RW		RW	

Pin name	Multifunction (bit value)			
	00	01	10	11
PF0	PF0			AN8
PF1	PF1			AN9
PF2	PF2			AN10
PF3	PF3			AN11
PF4	PF4			AN12
PF5	PF5			AN13
PF7	PF7			
PF8	PF8			
PF9	PF9			
PF10	PF10			
PF11	PF11			

5.4.7 Pn_CR: PORT n Control Register (Except PC)

Pn_CR is a register that controls I / O settings for each pin of the port.

push-pull output, open drain output, logic-input and analog input can be set for each pin. This register can set the input/output of GPIO or various peripheral devices. The ADC and Oscillator must be set to analog inputs, in this case the logic part of that pin is disabled.

PA_CR=0x4000_0204, PB_CR=0x4000_0224
PD_CR=0x4000_0264, PE_CR=0x4000_0284, PF_CR=0x4000_02A4

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	Px[1:0]	00	Push-pull output
		01	Open-drain output
		10	Logic input
		11	Analog Input (ADC / OSC)

5.4.8 PC_CR: PORT n Control Register

PC_CR is a register that controls I / O settings for each pin of the port C.

Push-pull output, open drain output, logic-input and analog input can be set for each pin. This register can set the input/output of GPIO and peripherals. The oscillator must be set to analog inputs, in this case the logic part of that pin is disabled.

PC_CR=0x4000_0244

31 30	29 28	27 26	25 24	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	10	10	11	00	10	10	10	10
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	Px[1:0]	00	Push-pull output
		01	Open-drain output
		10	Logic input
		11	Analog Input (ADC / OSC)

5.4.9 Pn_PCR: PORT n Pull-up/Pull-down Resistor Control Register (Except PC)

Pn_PCR is a register to selects the pull-up option of each pin of port C. Pull-up and pull-down resistors for each pin can be turned on or off. Lower 16 bits are used to determine whether pull-up/pull-down is enable or disable, and Upper 16 bits are used to determine use of pull-up or pull-down resistor.

PA_PCR=0x4000_0208, PB_PCR=0x4000_0228
PD_PCR=0x4000_0268, PE_PCR=0x4000_0288, PF_PCR=0x4000_02A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
0	0	0	0	0	0	0	0	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	Dx	Select pull-up or pull-down resistor, x = 0 to 15	
		0	Select pull-up resistor
		1	Select pull-down resistor
x	Px	Enable or disable pull-up/pull-down, x = 0 to 15	
		0	Pull-up/pull-down disable
		1	Pull-up/pull-down enable

5.4.10 PC_PCR: PORT n Pull-up/Pull-down Resistor Control Register

Pn_PCR is a register to select the pull-up option of each pin of port C. Pull-up and pull-down resistors for each pin can be turned on or off. Lower 16 bits are used to determine whether pull-up/pull-down is enabled or disabled, and Upper 16 bits are used to determine use of pull-up or pull-down resistor.

PC_PCR=0x4000_0248

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
0	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	0	0	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	RW	RW	RW	RW	RW	RW	RW	RW

x	Dx	Select pull-up or pull-down resistor, x = 0 to 15	
		0	Select pull-up resistor
		1	Select pull-down resistor
x	Px	Enable or disable pull-up/pull-down, x = 0 to 15	
		0	Pull-up/pull-down disable
		1	Pull-up/pull-down enable

5.4.11 Pn_DER: Port n Debouncing Enable Register

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

PA_DER=0x4000_020C, PC_DER=0x4000_022C, PC_DER=0x4000_024C
PD_DER=0x4000_026C, PE_DER=0x4000_028C, PF_DER=0x4000_02AC

31 30 29 28 27 26 25 24	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		P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
-		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	Px	Enable/disable debounce of pin, x = 0 to 15
		0 Disables the debouncing filter of Px
		1 Enables the debouncing filter of Px

5.4.12 Pn_IER: Port n Interrupt Enable Register

These registers are for enabling of the GPIO interrupt for each pin and can set the interrupt type such as level interrupt or edge interrupt. Level interrupt occur when a high or low level signal is detected through the input pin. The edge interrupt occurs when an edge that made is detected when the input signal changes from low to high or from high to low.

PA_IER=0x4000_0210, PB_IER=0x4000_0230, PC_IER=0x4000_0250
PD_IER=0x4000_0270, PE_IER=0x4000_0290, PF_IER=0x4000_02B0

31 30	29 28	27 26	25 24	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]
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	Px[1:0]	The interrupt source to be enabled for the pin, x = 0 to 15	
		x0	Disable the GPIO interrupt source.
		01	Enable level interrupt (‘H’ or ‘L’)
		11	Enable edge interrupt (‘H’ → ‘L’ or ‘L’ → ‘H’)

5.4.13 Pn_ISR: Port n Interrupt Status Register

This register indicates the GPIO interrupt status for all pins of each port. All pins of each port can be GPIO interrupt source.

The interrupt signal generated onthe port is sent to the interrupt controller.

Because interrupt controller just can detect ‘H’ level only, PMC block sends the signal generated level interrupt or edge interrupt, or ‘H’ or ‘L’ signal interrupt by trigger to interrupt controller.

Since the interrupt controller can detect the interrupt signal only at the 'H' level, the PMC block sends the interrupt signal generated from all triggers ('H' or 'L') of the level or edge to the interrupt controller.

When the port interrupt is set to edge interrupt, if an interrupt event occurs, the event information is displayed in the status register. If the port interrupt is set to level interrupt, the event information is not displayed in the status register.

To initialize the state of any port in this register, a ‘1’ must be written to the corresponding bit.

PA_ISR=0x4000_0214, PB_ISR=0x4000_0234, PC_ISR=0x4000_0254
PD_ISR=0x4000_0274, PE_ISR=0x4000_0294, PF_ISR=0x4000_02B4

31 30	29 28	27 26	25 24	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]
00	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	RWC1

2x+1 2x	Px[1:0]	indicates an interrupt request according to the condition of each bit, x = 0 to 15 Write ‘1’ to the flag to clear the flag.	
		00	No interrupt occurred
		01	‘L’ level or falling edge interrupt occurred
		10	‘H’ level or rising edge interrupt occurred
		11	Either of rising or falling edge interrupt occurred (Not available on level mode interrupt)

5.4.14 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_0218, PB_ICR=0x4000_0238, PC_ICR=0x4000_0258
PD_ICR=0x4000_0278, PE_ICR=0x4000_0298, PF_ICR=0x4000_02B8

31 30	29 28	27 26	25 24	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]
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	Px[1:0]	Pin interrupt mode, x = 0 to 15	
		00	Disable interrupt
		01	'L' level or falling edge interrupt input
		10	'H' level or rising edge interrupt input
		11	Either of rising or falling edge interrupt (Not available on level mode interrupt)

5.4.15 Pn_DPR: Port n Debounce Prescaler Register

PnDPR is a 5-bit register that divides the debounce reference clock for each port.

PA_DPR=0x4000_021C, PB_DPR=0x4000_023C, PC_DPR=0x4000_025C
PD_DPR=0x4000_027C, PE_DPR=0x4000_029C, PF_DPR=0x4000_02BC

31 30 29 28 27 26 25 24	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			DPR
-			0x00000
-			RW

5 0	DPR	The debounce clock input divider. The debounce clock can be obtained by dividing the main clock by two to the power of DPR, as in the following formula: $Debounce\ Clock = \frac{\mathrm{PCLK}}{2^{DPR}}$

Input debounce clock function is specified for an input signal based on four-consecutive clock When you get the same value as the input values for the recognition function. Therefore, the input filter time,

$$Tfilter = \frac{4}{Debounce\ Clock}$$

Debounce filter window of the prescaler in the following table.

Table 35. The Examples of Debounce Filter Windows

DPR	PCLK Count	Filtering Window (16 MHz)	Filtering Window (75 MHz)	DPR	PCLK Count	Filtering Window (16 MHz)	Filtering Window (75 MHz)
0	4	250 ns	53 ns	16	256 K	16 ms	3.4 ms
1	8	500 ns	106 ns	17	512 K	32 ms	6.9 ms
2	16	1 µs	213 ns	18	1,024 K	65 ms	13.9 ms
3	32	2 µs	426 ns	19	2,048 K	131 ms	27.9 ms
4	64	4 µs	853 ns	20	4,096 K	262 ms	55.9 ms
5	128	8 µs	1.7 µs	21	8,192 K	524 ms	111 ms
6	256	16 µs	3.4 µs	22	16 M	1.0 s	223 ms
7	512	32 µs	6.8 µs	23	32 M	2.0 s	447 ms
8	1,024	64 µs	13.6 µs	24	64 M	4.1 s	894 ms
9	2,048	128 µs	27.3 µs	25	128 M	8.3 s	1.7 s
10	4,096	256 µs	54.6 µs	26	512 M	16 s	3.5 s
11	8,192	512 µs	109 µs	27	1 G	33 s	7.1 s
12	16,384	1 ms	218 µs	28	2 G	67 s	14.3 s
13	32,768	2 ms	436 µs	29	4 G	134 s	28.6 s
14	65,536	4 ms	873 µs	30	8 G	268 s	57.2 s
15	128 K	8.1 ms	1.74 ms	31	16 G	536 s	114.5 s

5.4.16 PMC Register Map Summary

Table 36. PMC 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	PA_MR	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	PB_MR	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	PC_MR	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	1	0	1
0x00	PD_MR	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	PE_MR	P15[1:0]		P14[1:0]		P13[1:0]		P12[1:0]		P11[1:0]		Res		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	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	PF_MR	Res		Res		Res		Res		P11[1:0]		P10[1:0]		P9[1:0]		P8[1:0]		P7[1:0]		Res		P5[1:0]		P4[1:0]		P3[1:0]		P2[1:0]		P1[1:0]		P0[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
0x04	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
0x04	PC_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	0	1	0	1	1	0	0	1	0	1	0	1	0	1	0
0x08	Pn_PCR	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
	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	PC_PCR	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
	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	1	0	0	1	1	1	1

Table 36. PMC Register Map Summary (continued)

0x0C	Pn_DER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	Pn_IER	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	0	0	0	0	0	0	0	0	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	Pn_ISR	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	0	0	0	0	0	0	0	0	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	Pn_ICR	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	Pn_DPR	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.	DPR[4:0]			
	Reset value																													0	0	0	0	0

6. General-Purpose I/O (GPIO)

6.1 Introduction

6.1.1 GPIO Introduction

Pins, except for VDD, GND, and 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 read for their input status in the control registers.

6.2 Main Features

6.2.1 GPIO Features

The GPIO module controls the GPIO ports as listed below:

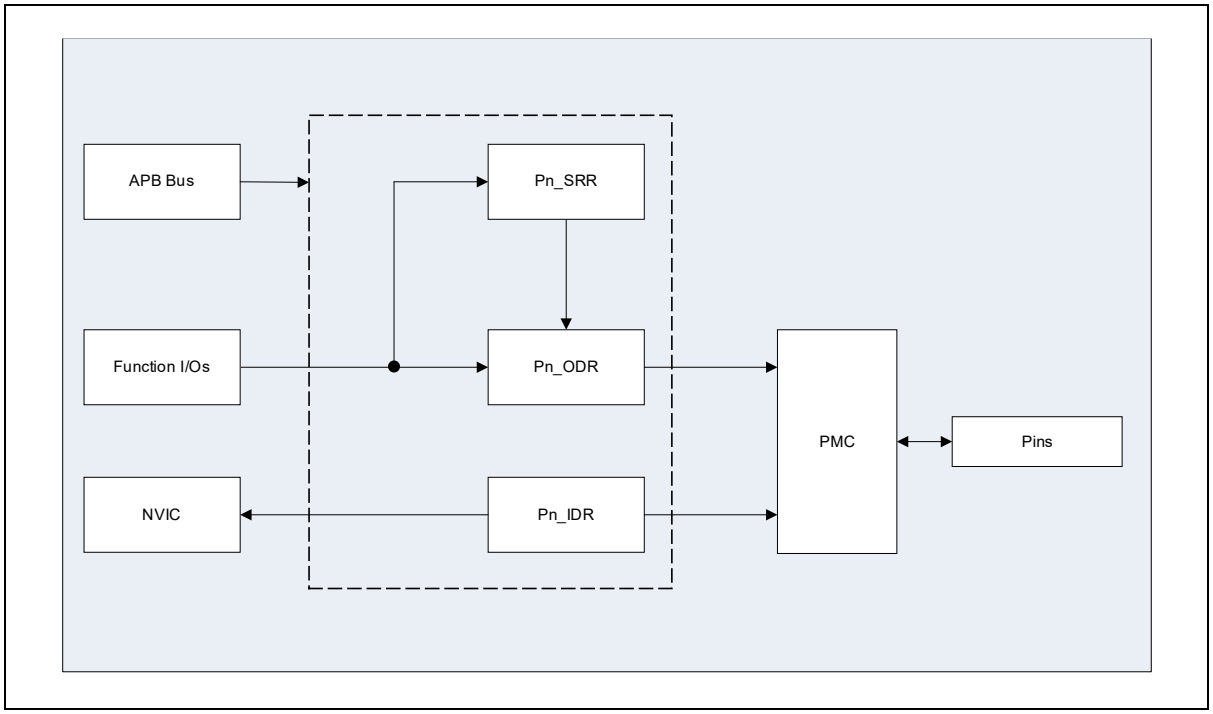
- Output value (High/Low) selection function of each pin
- Function to check logic input status of each pin
- General-Purpose GPIO usage
- Pin function selection by PMC register

6.3 GPIO Functional Description

All pins other than A33G53x function pins can be set to GPIO (General-Purpose Input Output). To use GPIO, first set the pin MUX register on the port map controller (PMC) of that port to GPIO. (See **5.3** PMC Functional Description)

Following figures (Figure 42) show block diagrams of GPIO.

Figure 42. GPIO Block Diagram



6.3.1 Setting Example

<Example 1> Function of PD0 pin, input / output direction, pull-up / pull-down resistor setting

PD_MR<P0[1:0]> = “00” PD_CR<P0[1:0]> = “00” PD_PCR<P0>= “1” PD_PCR<D0>= “0”	: Set PD0 Pin to GPIO (General Purpose I/O). : Set the directionof PD0 pin to push-pull output. : Enable the pull-up/pull-down function of PD0 pin. : Select pull-up resistor of PD0 pin.

<Example 2> Output data setting of PD0 pin

PD_MR<P0[1:0]> = “00” PD_CR<P0[1:0]> = “00” PD_ODR<ODR[15:0]> = “00000000 00000001”	: Set PD0 Pin to GPIO (General Purpose I/O). : Set the directionof PD0 pin to push-pull output. : Set the output signal of PD0 pin to high.

<Example 3> PD0 value set / reset

PD_SRR<BSR[15:0]> = “00000000 00000001” PD_SRR<BRR[31:16]> = “00000000 00000001”	: When the BSR bit of the PD0 port is '1', the bit value 'set' : When the BRR bit of the PD0 port is '0', the bit value 'reset'

6.4 GPIO Registers

The base addresses and register map of GPIO are as follows:

Table 37. Base Addresses of PCU

Name	Base address	Description
PA	0x4000_0300	General Port A
PB	0x4000_0310	General Port B
PC	0x4000_0320	General Port C
PD	0x4000_0330	General Port D
PE	0x4000_0340	General Port E
PF	0x4000_0350	General Port F

Table 38. PCU and GPIO Register Map

Name	Offset	Type	Description	Reset value	Reference
PA_ODR	0x00	RW	Port A Output Data Register	0x00000000	6.4.1
PA_IDR	0x04	RO	Port A Input Data Register	-	6.4.2
PA_SRR	0x08	WO	Port A Pin Set/Reset Register	-	6.4.3
PB_ODR	0x10	RW	Port B Output Data Register	0x00000000	6.4.1
PB_IDR	0x14	RO	Port B Input Data Register	-	6.4.2
PB_SRR	0x18	WO	Port B Pin Set/Reset Register	-	6.4.3
PC_ODR	0x20	RW	Port C Output Data Register	0x00000000	6.4.1
PC_IDR	0x24	RO	Port C Input Data Register	-	6.4.2
PC_SRR	0x28	WO	Port C Pin Set/Reset Register	-	6.4.3
PD_ODR	0x30	RW	Port D Output Data Register	0x00000000	6.4.1
PD_IDR	0x34	RO	Port D Input Data Register	-	6.4.2
PD_SRR	0x38	WO	Port D Pin Set/Reset Register	-	6.4.3
PE_ODR	0x40	RW	Port E Output Data Register	0x00000000	6.4.1
PE_IDR	0x44	RO	Port E Input Data Register	-	6.4.2
PE_SRR	0x48	WO	Port E Pin Set/Reset Register	-	6.4.3
PF_ODR	0x50	RW	Port F Output Data Register	0x00000000	6.4.1
PF_IDR	0x54	RO	Port F Input Data Register	-	6.4.2
PF_SRR	0x58	WO	Port F Pin Set/Reset Register	-	6.4.3

NOTE: n = A, B, C, D, E, and F.

6.4.1 Pn_ODR: Port n Output Data Register

When the bit of the corresponding port is set as the output, the value of this register is output to each port. This register outputs a signal when the corresponding pin is set to GPIO.

PA_ODR=0x4000_0300, PB_ODR=0x4000_0310, PC_ODR=0x4000_0320
PD_ODR=0x4000_0330, PE_ODR=0x4000_0340, PF_ODR=0x4000_0350

31 30 29 28 27 26 25 24	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		ODR	
-		0x0000000000000000	
-		RW	

15 0	ODR	If the port is output mode, ODR value is output.

6.4.2 Pn_IDR: Port n Input Data Register

This register indicates the current input state of the port and indicates the input when the GPIO and pins are set to logic inputs of other functions. When the pin is set to a special function such as ADC or clock, '1' is always displayed.

If the level interrupt function is enabled on all ports via the PMC_ISR register, the level value can be checked by reading the Pn_IDR register value for the input level value.

PA_IDR=0x4000_0304, PB_IDR=0x4000_0314, PC_IDR=0x4000_0324
PD_IDR=0x4000_0334, PE_IDR=0x4000_0344, PF_IDR=0x4000_0354

31 30 29 28 27 26 25 24	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		IDR	
-		0xXXXXXXXXXXXXXXXX	
-		RO	

15 0	ODR	Indicates the current status of a port.

NOTE:

If the external pin of the microcontroller is set as an input when it is in the floating state, it will be in an unknown state that recognizes the level as 0 or 1 regardless of the value of the input signal. In this floating state, determining the level value recognized from the Pn_IDR register and executing the program to perform certain functions may malfunction. Therefore, It is recommended to design a pull-up or pull-down for pins that receive signals from the outside.

6.4.3 Pn_SRR: Port n Set/Reset Register

This register can be set or reset for the value of each pin of the corresponding port. This register can only change the PnODR value to a specific bit. If the values of BRR and BSR are '1' at the same time, the current output value is inverted and output.

PA_SRR=0x4000_0308, PB_SRR=0x4000_0318, PC_SRR=0x4000_0328
PD_SRR=0x4000_0338, PE_SRR=0x4000_0348, PF_SRR=0x4000_0358

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

15 0	ODR	If the bit is '1', the corresponding bit will be reset.
15 0	BSR	If the bit is '1', the corresponding bit will be set.

NOTE: When BSR and BRR are set to '1' at the same time, the output of the corresponding pin is inverted.

6.4.4 GPIO Register Map Summary

Table 39. 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_ODR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ODR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	Pn_IDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDR[15:0]															
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x08	Pn_SRR	BRR[15:0]																BSR[15:0]															
	Reset value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

7. FMC (Flash Memory Controller)

7.1 Introduction

The A33G53x microcontroller contains a non-volatile memory, code flash and data flash, which are electrically able to erase, program, each with a 2 KB sector size. A33G53x series offers a high-capacity code flash area, you can use the flash area according to the purpose of user application.

Code and data flash memory of A33G53x also supports CRC-16 (Cyclic Redundancy Check 16) function, which enables error detection on data written to flash memory.

When the Extended mode is activated by setting the EX bit value in the FMC_TEST register, functions such as Self-Program, Self-Erase (Page, Sector), 512-byte Page Erase, and Boot Block Protection are also available.

In addition, it supports limitations on read / program / erase for specific areas of memory, providing security features such as flash memory protection, external access protection, and security.

The Flash memory controller of A33G53x series supports the followings:

- Erase / Read / Write functions in code and data flash memory areas
- Read-Protection function
- Write / Erase Protection function
- Bank swap and RWW (Read-While-Write) in code flash memory
- ECC function

7.2 Features

The Flash memory of A33G53x series has several key features as listed below:

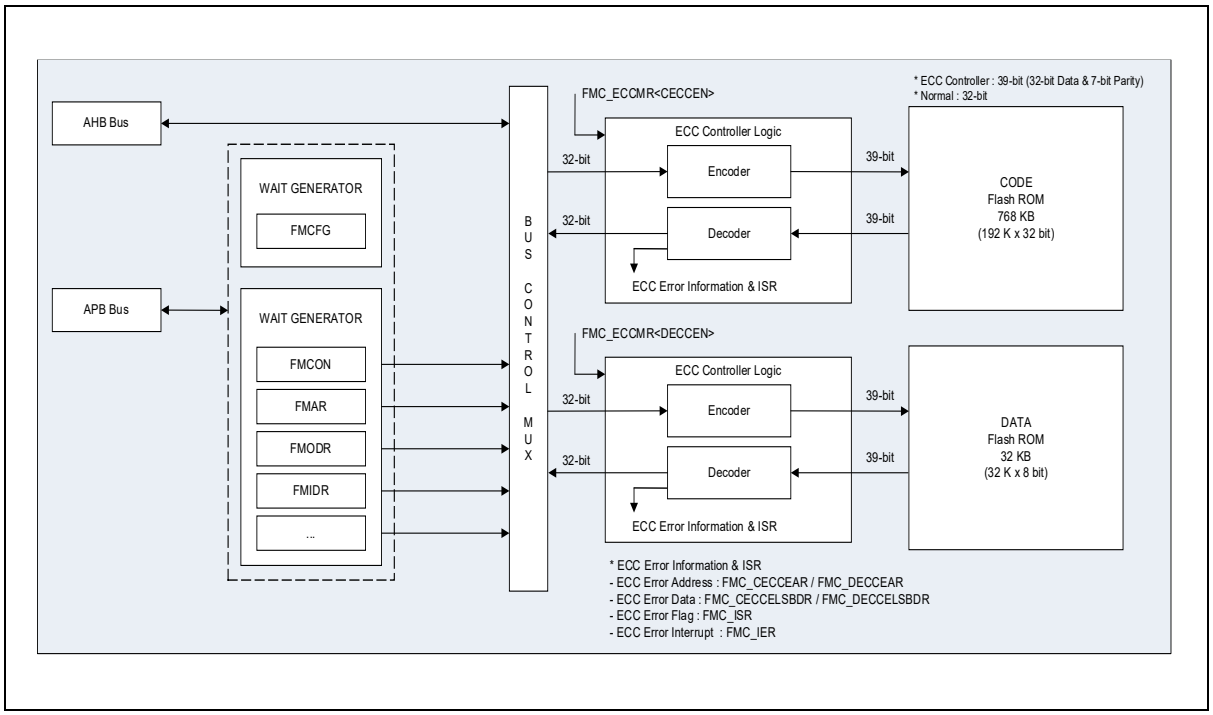
- Code flash memory capacity: Built-in high-capacity code flash memory
  - A33G539 (768 KB, 384 sectors)
  - A33G538 (512 KB, 256 sectors)
- Data Flash Capacity: 32 KB (32 sectors)
- Up to 25 MHz flash memory access timing
- Program size unit
  - Code flash memory: 1 word (4 bytes)
  - Data flash memory: 1 byte (Data Flash ECC Disable) / 1 word (Data Flash ECC Enable)
- 512 bytes, 2 KB sectors and bulk erase
- Flash Self-PGM (Program) of code flash memory
  - Supports to update data in some code flash memory region during execution of user program in code flash area.
  - Self-PGM only supports code flash.
- CRC-16 generation and verification for error detection
- The restrict a specific area of flash memory for protection and security
- Bank Swap and RWW (Read-While-Write) in code flash memory
- ECC (Error Correct Code)
- Endurance
  - Code flash memory: 10,000 cycles
  - Data flash memory: 100,000 cycles
- Lifetime: 10 years

7.3 Functional Description

7.3.1 Block Diagram

The flash memory is composed of code flash memory and data flash memory; where code flash memory stores the application code or constant data and data flash memory mainly stores parameters or log data.

Figure 43. FMC Block Diagram



7.3.1.1 Code Flash Memory

Code flash memory has key features as shown below:

- Flash code memory for 512 KB and 768 KB with write protection bits
- Page (512 bytes), sector (2 KB) and bulk erases
- Up to 25 MHz flash access timing
- Read-Protection functionality

Table 40. Code Flash Memory Controller Features

Item	Description	
Size	512 KB	768 KB
Start address	0x0000_0000	0x0000_0000
End address	0x0007_FFFF	0x000B_FFFF
Page size	512 bytes	512 bytes
Total page count	1,024 pages	1,536 pages
Sector size	2 KB	2 KB
Total sector count	256 sectors	384 sectors
PGM unit	4 bytes (1 word)	4 bytes (1 word)
Erase unit	512 bytes 2 KB Bulk	512 bytes 2 KB Bulk

7.3.1.2 Data Flash Memory

Data Flash memory has key features as shown below:

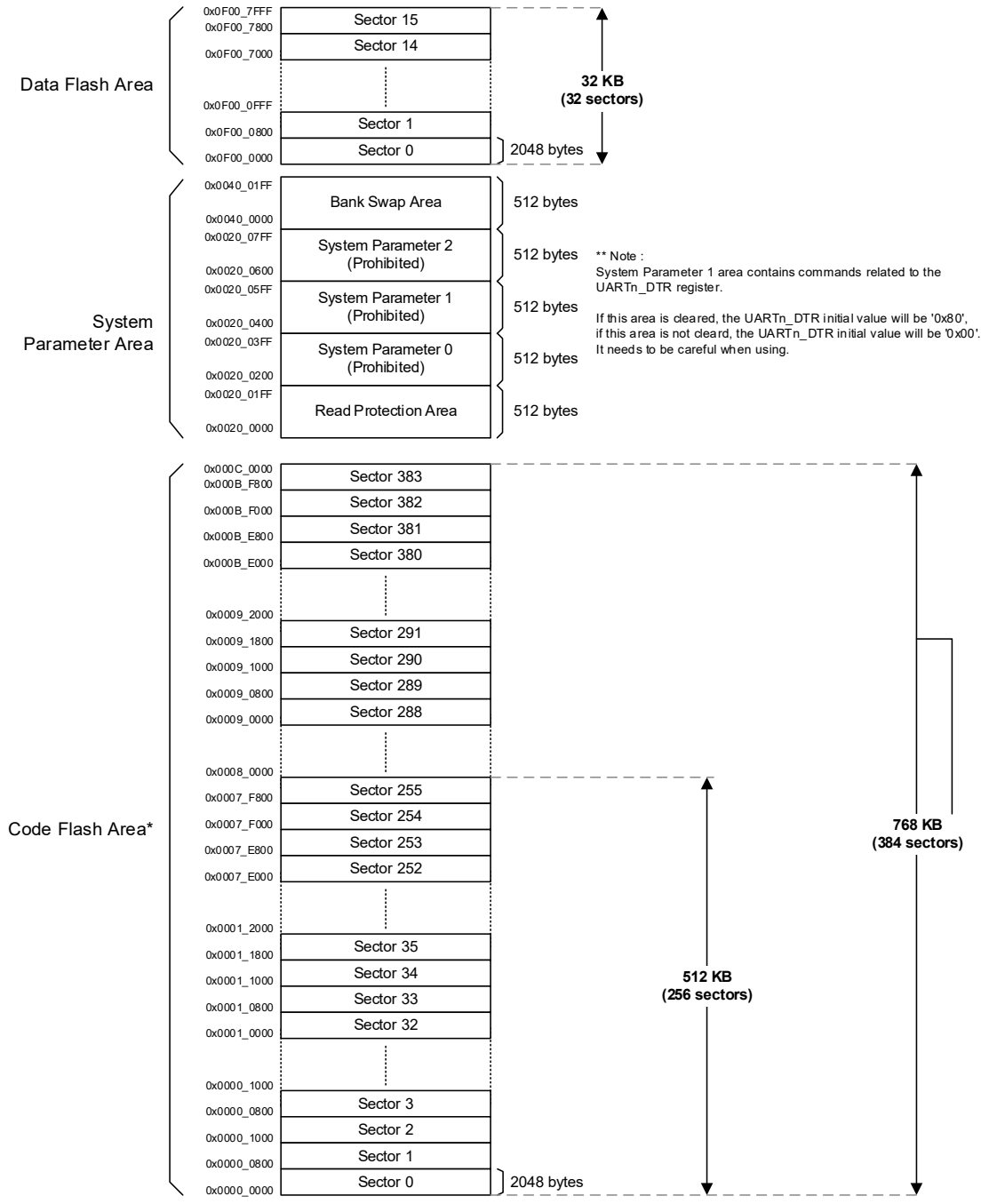
- Flash data memory for 32 KB with write protection bits
- Page (512 bytes), sector (2 KB), and bulk erases
- Up to 25 MHz flash access timing

Table 41. Data Flash Memory Controller Features

Item	Description
Size	32 KB
Start Address	0x0F00_0000
End Address	0x0F00_7FFF
Page Size	512 bytes
Total Page Count	64 pages
Sector size	2 KB
Total sector count	16 sectors
PGM Unit (Data flash ECC disable)	1 byte
PGM Unit (Data flash ECC enable)	4 bytes (1 word)
Erase Unit	512 bytes / 2 KB / bulk

7.3.1.3 Flash Memory Map

Figure 44. Flash Memory Map



NOTE: To write or erase data in the flash memory area, the address to be accessed must be explicated. If an address out of the flash memory area is accessed, system operation is not guaranteed

7.3.1.4 Configure Code Flash Memory Sectors

Code flash memory on the A33G53x is a sector configuration of 2 KB units. The addresses for each sector are shown in the table below.

Table 42. Sector List of Code Flash Memory

Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address
0	2 KB	0x0000_0000	96	2 KB	0x0003_0000	192	2 KB	0x0006_0000	288	2 KB	0x0009_0000
1	2 KB	0x0000_0800	97	2 KB	0x0003_0800	193	2 KB	0x0006_0800	289	2 KB	0x0009_0800
2	2 KB	0x0000_1000	98	2 KB	0x0003_1000	194	2 KB	0x0006_1000	290	2 KB	0x0009_1000
3	2 KB	0x0000_1800	99	2 KB	0x0003_1800	195	2 KB	0x0006_1800	291	2 KB	0x0009_1800
4	2 KB	0x0000_2000	100	2 KB	0x0003_2000	196	2 KB	0x0006_2000	292	2 KB	0x0009_2000
5	2 KB	0x0000_2800	101	2 KB	0x0003_2800	197	2 KB	0x0006_2800	293	2 KB	0x0009_2800
6	2 KB	0x0000_3000	102	2 KB	0x0003_3000	198	2 KB	0x0006_3000	294	2 KB	0x0009_3000
7	2 KB	0x0000_3800	103	2 KB	0x0003_3800	199	2 KB	0x0006_3800	295	2 KB	0x0009_3800
8	2 KB	0x0000_4000	104	2 KB	0x0003_4000	200	2 KB	0x0006_4000	296	2 KB	0x0009_4000
9	2 KB	0x0000_4800	105	2 KB	0x0003_4800	201	2 KB	0x0006_4800	297	2 KB	0x0009_4800
10	2 KB	0x0000_5000	106	2 KB	0x0003_5000	202	2 KB	0x0006_5000	298	2 KB	0x0009_5000
11	2 KB	0x0000_5800	107	2 KB	0x0003_5800	203	2 KB	0x0006_5800	299	2 KB	0x0009_5800
12	2 KB	0x0000_6000	108	2 KB	0x0003_6000	204	2 KB	0x0006_6000	300	2 KB	0x0009_6000
13	2 KB	0x0000_6800	109	2 KB	0x0003_6800	205	2 KB	0x0006_6800	301	2 KB	0x0009_6800
14	2 KB	0x0000_7000	110	2 KB	0x0003_7000	206	2 KB	0x0006_7000	302	2 KB	0x0009_7000
15	2 KB	0x0000_7800	111	2 KB	0x0003_7800	207	2 KB	0x0006_7800	303	2 KB	0x0009_7800
16	2 KB	0x0000_8000	112	2 KB	0x0003_8000	208	2 KB	0x0006_8000	304	2 KB	0x0009_8000
17	2 KB	0x0000_8800	113	2 KB	0x0003_8800	209	2 KB	0x0006_8800	305	2 KB	0x0009_8800
18	2 KB	0x0000_9000	114	2 KB	0x0003_9000	210	2 KB	0x0006_9000	306	2 KB	0x0009_9000
19	2 KB	0x0000_9800	115	2 KB	0x0003_9800	211	2 KB	0x0006_9800	307	2 KB	0x0009_9800
20	2 KB	0x0000_A000	116	2 KB	0x0003_A000	212	2 KB	0x0006_A000	308	2 KB	0x0009_A000
21	2 KB	0x0000_A800	117	2 KB	0x0003_A800	213	2 KB	0x0006_A800	309	2 KB	0x0009_A800
22	2 KB	0x0000_B000	118	2 KB	0x0003_B000	214	2 KB	0x0006_B000	310	2 KB	0x0009_B000
23	2 KB	0x0000_B800	119	2 KB	0x0003_B800	215	2 KB	0x0006_B800	311	2 KB	0x0009_B800
24	2 KB	0x0000_C000	120	2 KB	0x0003_C000	216	2 KB	0x0006_C000	312	2 KB	0x0009_C000
25	2 KB	0x0000_C800	121	2 KB	0x0003_C800	217	2 KB	0x0006_C800	313	2 KB	0x0009_C800
26	2 KB	0x0000_D000	122	2 KB	0x0003_D000	218	2 KB	0x0006_D000	314	2 KB	0x0009_D000
27	2 KB	0x0000_D800	123	2 KB	0x0003_D800	219	2 KB	0x0006_D800	315	2 KB	0x0009_D800
28	2 KB	0x0000_E000	124	2 KB	0x0003_E000	220	2 KB	0x0006_E000	316	2 KB	0x0009_E000
29	2 KB	0x0000_E800	125	2 KB	0x0003_E800	221	2 KB	0x0006_E800	317	2 KB	0x0009_E800
30	2 KB	0x0000_F000	126	2 KB	0x0003_F000	222	2 KB	0x0006_F000	318	2 KB	0x0009_F000
31	2 KB	0x0000_F800	127	2 KB	0x0003_F800	223	2 KB	0x0006_F800	319	2 KB	0x0009_F800
32	2 KB	0x0001_0000	128	2 KB	0x0004_0000	224	2 KB	0x0007_0000	320	2 KB	0x000A_0000
33	2 KB	0x0001_0800	129	2 KB	0x0004_0800	225	2 KB	0x0007_0800	321	2 KB	0x000A_0800
34	2 KB	0x0001_1000	130	2 KB	0x0004_1000	226	2 KB	0x0007_1000	322	2 KB	0x000A_1000
35	2 KB	0x0001_1800	131	2 KB	0x0004_1800	227	2 KB	0x0007_1800	323	2 KB	0x000A_1800
36	2 KB	0x0001_2000	132	2 KB	0x0004_2000	228	2 KB	0x0007_2000	324	2 KB	0x000A_2000
37	2 KB	0x0001_2800	133	2 KB	0x0004_2800	229	2 KB	0x0007_2800	325	2 KB	0x000A_2800
38	2 KB	0x0001_3000	134	2 KB	0x0004_3000	230	2 KB	0x0007_3000	326	2 KB	0x000A_3000
39	2 KB	0x0001_3800	135	2 KB	0x0004_3800	231	2 KB	0x0007_3800	327	2 KB	0x000A_3800
40	2 KB	0x0001_4000	136	2 KB	0x0004_4000	232	2 KB	0x0007_4000	328	2 KB	0x000A_4000
41	2 KB	0x0001_4800	137	2 KB	0x0004_4800	233	2 KB	0x0007_4800	329	2 KB	0x000A_4800

Table 42. Sector List of Code Flash Memory (continued)

Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address
42	2 KB	0x0001_5000	138	2 KB	0x0004_5000	234	2 KB	0x0007_5000	330	2 KB	0x000A_5000
43	2 KB	0x0001_5800	139	2 KB	0x0004_5800	235	2 KB	0x0007_5800	331	2 KB	0x000A_5800
44	2 KB	0x0001_6000	140	2 KB	0x0004_6000	236	2 KB	0x0007_6000	332	2 KB	0x000A_6000
45	2 KB	0x0001_6800	141	2 KB	0x0004_6800	237	2 KB	0x0007_6800	333	2 KB	0x000A_6800
46	2 KB	0x0001_7000	142	2 KB	0x0004_7000	238	2 KB	0x0007_7000	334	2 KB	0x000A_7000
47	2 KB	0x0001_7800	143	2 KB	0x0004_7800	239	2 KB	0x0007_7800	335	2 KB	0x000A_7800
48	2 KB	0x0001_8000	144	2 KB	0x0004_8000	240	2 KB	0x0007_8000	336	2 KB	0x000A_8000
49	2 KB	0x0001_8800	145	2 KB	0x0004_8800	241	2 KB	0x0007_8800	337	2 KB	0x000A_8800
50	2 KB	0x0001_9000	146	2 KB	0x0004_9000	242	2 KB	0x0007_9000	338	2 KB	0x000A_9000
51	2 KB	0x0001_9800	147	2 KB	0x0004_9800	243	2 KB	0x0007_9800	339	2 KB	0x000A_9800
52	2 KB	0x0001_A000	148	2 KB	0x0004_A000	244	2 KB	0x0007_A000	340	2 KB	0x000A_A000
53	2 KB	0x0001_A800	149	2 KB	0x0004_A800	245	2 KB	0x0007_A800	341	2 KB	0x000A_A800
54	2 KB	0x0001_B000	150	2 KB	0x0004_B000	246	2 KB	0x0007_B000	342	2 KB	0x000A_B000
55	2 KB	0x0001_B800	151	2 KB	0x0004_B800	247	2 KB	0x0007_B800	343	2 KB	0x000A_B800
56	2 KB	0x0001_C000	152	2 KB	0x0004_C000	248	2 KB	0x0007_C000	344	2 KB	0x000A_C000
57	2 KB	0x0001_C800	153	2 KB	0x0004_C800	249	2 KB	0x0007_C800	345	2 KB	0x000A_C800
58	2 KB	0x0001_D000	154	2 KB	0x0004_D000	250	2 KB	0x0007_D000	346	2 KB	0x000A_D000
59	2 KB	0x0001_D800	155	2 KB	0x0004_D800	251	2 KB	0x0007_D800	347	2 KB	0x000A_D800
60	2 KB	0x0001_E000	156	2 KB	0x0004_E000	252	2 KB	0x0007_E000	348	2 KB	0x000A_E000
61	2 KB	0x0001_E800	157	2 KB	0x0004_E800	253	2 KB	0x0007_E800	349	2 KB	0x000A_E800
62	2 KB	0x0001_F000	158	2 KB	0x0004_F000	254	2 KB	0x0007_F000	350	2 KB	0x000A_F000
63	2 KB	0x0001_F800	159	2 KB	0x0004_F800	255	2 KB	0x0007_F800	351	2 KB	0x000A_F800
64	2 KB	0x0002_0000	160	2 KB	0x0005_0000	256	2 KB	0x0008_0000	352	2 KB	0x000B_0000
65	2 KB	0x0002_0800	161	2 KB	0x0005_0800	257	2 KB	0x0008_0800	353	2 KB	0x000B_0800
66	2 KB	0x0002_1000	162	2 KB	0x0005_1000	258	2 KB	0x0008_1000	354	2 KB	0x000B_1000
67	2 KB	0x0002_1800	163	2 KB	0x0005_1800	259	2 KB	0x0008_1800	355	2 KB	0x000B_1800
68	2 KB	0x0002_2000	164	2 KB	0x0005_2000	260	2 KB	0x0008_2000	356	2 KB	0x000B_2000
69	2 KB	0x0002_2800	165	2 KB	0x0005_2800	261	2 KB	0x0008_2800	357	2 KB	0x000B_2800
70	2 KB	0x0002_3000	166	2 KB	0x0005_3000	262	2 KB	0x0008_3000	358	2 KB	0x000B_3000
71	2 KB	0x0002_3800	167	2 KB	0x0005_3800	263	2 KB	0x0008_3800	359	2 KB	0x000B_3800
72	2 KB	0x0002_4000	168	2 KB	0x0005_4000	264	2 KB	0x0008_4000	360	2 KB	0x000B_4000
73	2 KB	0x0002_4800	169	2 KB	0x0005_4800	265	2 KB	0x0008_4800	361	2 KB	0x000B_4800
74	2 KB	0x0002_5000	170	2 KB	0x0005_5000	266	2 KB	0x0008_5000	362	2 KB	0x000B_5000
75	2 KB	0x0002_5800	171	2 KB	0x0005_5800	267	2 KB	0x0008_5800	363	2 KB	0x000B_5800
76	2 KB	0x0002_6000	172	2 KB	0x0005_6000	268	2 KB	0x0008_6000	364	2 KB	0x000B_6000
77	2 KB	0x0002_6800	173	2 KB	0x0005_6800	269	2 KB	0x0008_6800	365	2 KB	0x000B_6800
78	2 KB	0x0002_7000	174	2 KB	0x0005_7000	270	2 KB	0x0008_7000	366	2 KB	0x000B_7000
79	2 KB	0x0002_7800	175	2 KB	0x0005_7800	271	2 KB	0x0008_7800	367	2 KB	0x000B_7800
80	2 KB	0x0002_8000	176	2 KB	0x0005_8000	272	2 KB	0x0008_8000	368	2 KB	0x000B_8000
81	2 KB	0x0002_8800	177	2 KB	0x0005_8800	273	2 KB	0x0008_8800	369	2 KB	0x000B_8800
82	2 KB	0x0002_9000	178	2 KB	0x0005_9000	274	2 KB	0x0008_9000	370	2 KB	0x000B_9000
83	2 KB	0x0002_9800	179	2 KB	0x0005_9800	275	2 KB	0x0008_9800	371	2 KB	0x000B_9800
84	2 KB	0x0002_A000	180	2 KB	0x0005_A000	276	2 KB	0x0008_A000	372	2 KB	0x000B_A000
85	2 KB	0x0002_A800	181	2 KB	0x0005_A800	277	2 KB	0x0008_A800	373	2 KB	0x000B_A800
86	2 KB	0x0002_B000	182	2 KB	0x0005_B000	278	2 KB	0x0008_B000	374	2 KB	0x000B_B000
87	2 KB	0x0002_B800	183	2 KB	0x0005_B800	279	2 KB	0x0008_B800	375	2 KB	0x000B_B800

Table 42. Sector List of Code Flash Memory (continued)

Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address
88	2 KB	0x0002_C000	184	2 KB	0x0005_C000	280	2 KB	0x0008_C000	376	2 KB	0x000B_C000
89	2 KB	0x0002_C800	185	2 KB	0x0005_C800	281	2 KB	0x0008_C800	377	2 KB	0x000B_C800
90	2 KB	0x0002_D000	186	2 KB	0x0005_D000	282	2 KB	0x0008_D000	378	2 KB	0x000B_D000
91	2 KB	0x0002_D800	187	2 KB	0x0005_D800	283	2 KB	0x0008_D800	379	2 KB	0x000B_D800
92	2 KB	0x0002_E000	188	2 KB	0x0005_E000	284	2 KB	0x0008_E000	380	2 KB	0x000B_E000
93	2 KB	0x0002_E800	189	2 KB	0x0005_E800	285	2 KB	0x0008_E800	381	2 KB	0x000B_E800
94	2 KB	0x0002_F000	190	2 KB	0x0005_F000	286	2 KB	0x0008_F000	382	2 KB	0x000B_F000
95	2 KB	0x0002_F800	191	2 KB	0x0005_F800	287	2 KB	0x0008_F800	383	2 KB	0x000B_F800

7.3.1.5 Configure Data Flash Memory Sectors

Data flash memory on the A33G53x is a sector configuration of 2 KB units. The addresses for each sector are shown in the table below.

Table 43. Sector List of Data Flash Memory

Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address	Sector No.	Sector Size	Address
0	2 KB	0x0F00_0000	4	2 KB	0x0F00_2000	8	2 KB	0x0F00_4000	12	2 KB	0x0F00_6000
1	2 KB	0x0F00_0800	5	2 KB	0x0F00_2800	9	2 KB	0x0F00_4800	13	2 KB	0x0F00_6800
2	2 KB	0x0F00_1000	6	2 KB	0x0F00_3000	10	2 KB	0x0F00_5000	14	2 KB	0x0F00_7000
3	2 KB	0x0F00_1800	7	2 KB	0x0F00_3800	11	2 KB	0x0F00_5800	15	2 KB	0x0F00_7800

7.3.2 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's main clock.

The following sections are explains how to adjust each latency of the code flash memory and data flash memory. (see Table 58)

7.3.3 Flash Program and Erase Operations

It is important to remember that any interrupt must not be triggered for other operations while erasing or programming the Flash. Two operations below must be set up before starting the Flash control.

- All Interrupt disable
- Interrupt can be used in the Read-While-Write (RWW) function. However, bank mode must be enabled and this function is available in Self-PGM mode only.

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

7.3.3.1 Flash Memory Init

Before Flash erase/write for the first time, FMC_CON register must be cleared to 0x0000_0000, and EX in FMC_TEST register must also be disabled to 0x0.

- SELF of FMC_CON register needs write key when disable/enable.
  (Write key : SELF_CODE in FMC_CON register)
- EX of FMC_TEST register needs write key when disable/enable.
  (Write key : WRITE_KEY in FMC_TEST register)

The Flash Erase operation must be proceeded before starting the Flash programming. The Flash Erase can be processed in pages (512 bytes), Sectors (2 KB) or bulk.

7.4 Flash Memory Protection

Read Protection is a feature provided in the internal Code Flash, and protects user code from external hacking. The Read Protection operates in two different protection mode levels such as Unprotection (UNPROT) and Level1 (LVL1).

Data Flash operates in association with the Code Flash's Read Protection. When the Read Protection is applied to the Code Flash, it is applied to the Data Flash too. To disable this Read Protection in the Data Flash, Read Protection OTP in the Code Flash must be removed.

7.4.1 Read Protection

Table 44 describes three different protection mode levels of the Read Protection, such as UNPROT and LVL1.

Table 44. Available Operating Modes by Protection Level

Protection level	Operation mode	Code main		OTP		Data main	
		read	write	read	write	read	write
UNPROT	Normal mode	O	O	O	O	O	O
	Debug mode	O	O	O	O	O	O
	Boot mode	O	O	O	O	O	O
LVL1 (0x39)	Normal mode (Flash)	O	O	O	O	O	O
	Normal mode (SRAM)	X	O	X	O	X	O
	Debug mode	X	O	X	O NOTE	X	O
	Boot mode	X	O	X	O NOTE	X	O

NOTES:

1. The read protection area can be checked in the memory window in Debug mode. Bank swap area not verifiable.
2. A higher protection level cannot transition to a lower level without the Mass erase and OTP erase executed.

Example) Procedure for transitioning from read protection LVL1 to UNPROT.

A. Mass erase (Erase all contents of BANK0 and BANK1.) See 7.4.2.4.
B. Check the CERS in FMC_RPROT register.
C. OTP erase
D. Check the RPROT in FMC_RPROT register.
E. Write 0xFF to the FMRPROT[7:0] in FMC_RPROT register. After that, read protection LVL1 transition to UNPROT mode.

7.4.1.1 Read Protection Settings

Users write data to the Read Protection area of the OTP, 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.

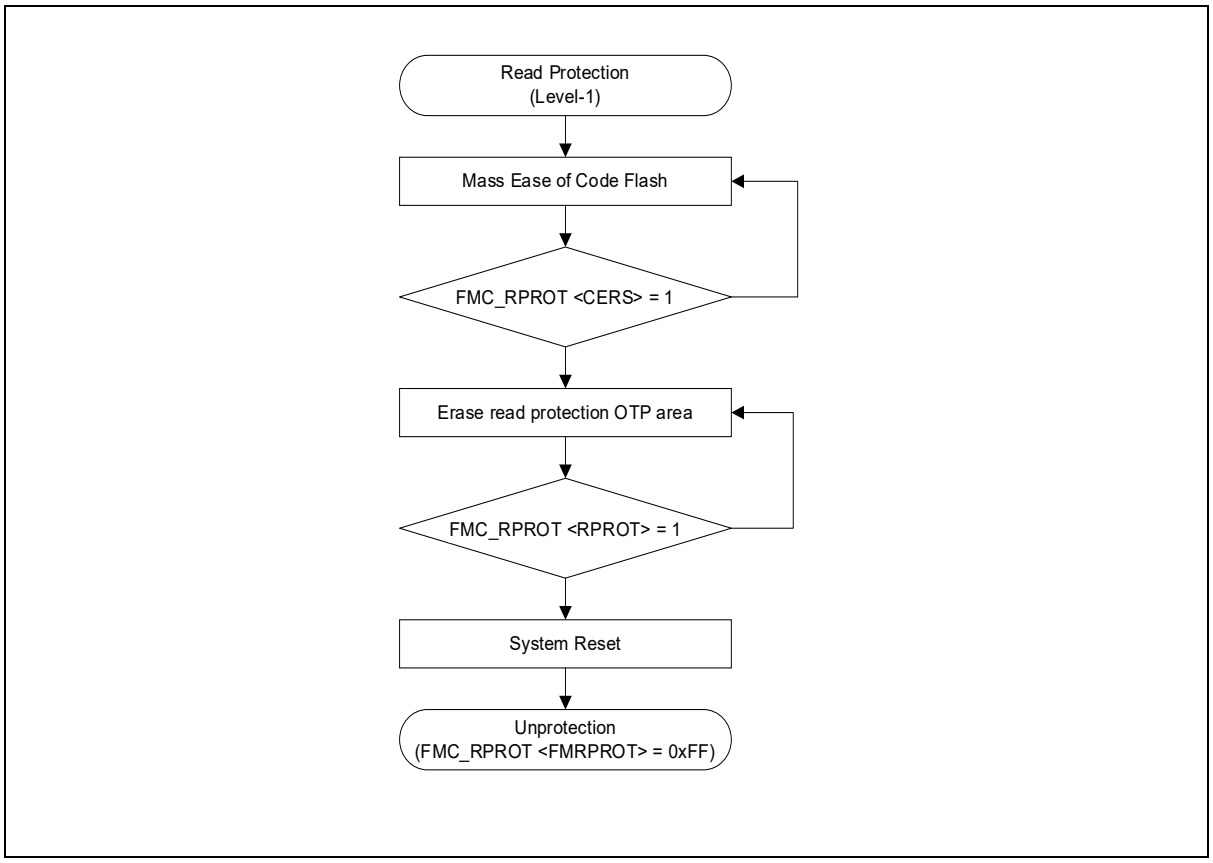
For the direct setup, users can write data in the Read Protection register to apply the settings immediately.

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.

7.4.1.2 Read Protection Removal

To remove the Read Protection, users must reset after deleting the related data in the OTP. To delete the data of the OTP in the Read Protection mode, the Mass Erase must be executed for the Code Flash. The state when the Mass Erase is complete for the Code Flash is stored and maintained until reset occurs. See chapter 7.4.2.4

Figure 45. Sequence Disabling Read Protection



7.4.2 Write Protection

Users can set up the Write / Erase Protection for a certain area of the Code Flash and Data Flash. Before setting the protections, remember the followings:

- If FMC_PROTECT register is '1' (Protection), the Mass Erase does not work because the protected area is included in the erase area.
- Similarly, each BBLK in FMC_CON Register does not erase 4 KB of each bank area during mass erase operation.
- Write Protection follows the logical address.

7.4.2.1 Write Protection for Code Flash

The FMC_PROTECT register is used to set the Protection and Unprotection, as described in Table 45 and Table 46.

Table 45. Code Flash Protection Area in 4 KB Units

Code Flash Size (512 KB)		Code Flash Size (768 KB)	
Name	**Address**	**Name**	**Address**
SP0	0x0000_0000 ~ 0x0000_0FFF	SP0	0x0000_0000 ~ 0x0000_0FFF
SP1	0x0000_1000 ~ 0x0000_1FFF	SP1	0x0000_1000 ~ 0x0000_1FFF
SP2	0x0000_2000 ~ 0x0000_2FFF	SP2	0x0000_2000 ~ 0x0000_2FFF
SP3	0x0000_3000 ~ 0x0000_3FFF	SP3	0x0000_3000 ~ 0x0000_3FFF
SP4	0x0004_0000 ~ 0x0004_0FFF	SP4	0x0006_0000 ~ 0x0006_0FFF
SP5	0x0004_1000 ~ 0x0004_1FFF	SP5	0x0006_1000 ~ 0x0006_1FFF
SP6	0x0004_2000 ~ 0x0004_2FFF	SP6	0x0006_2000 ~ 0x0006_2FFF
SP7	0x0004_3000 ~ 0x0004_3FFF	SP7	0x0006_3000 ~ 0x0006_3FFF

Table 46. Code Flash Protection Area in 48 KB Units

Code Flash Size (512 KB)		Code Flash Size (768 KB)	
Name	**Address**	**Name**	**Address**
BP0	0x0000_4000 ~ 0x0000_FFFF	BP0	0x0000_4000 ~ 0x0000_FFFF
BP4	0x0004_4000 ~ 0x0004_FFFF	BP6	0x0006_4000 ~ 0x0006_FFFF

Table 47. Code Flash Protection Area in 64 KB Units

Code Flash Size (512 KB)		Code Flash Size (768 KB)	
Name	**Address**	**Name**	**Address**
BP1	0x0001_0000 ~ 0x0001_FFFF	BP1	0x0001_0000 ~ 0x0001_FFFF
BP2	0x0002_0000 ~ 0x0002_FFFF	BP2	0x0002_0000 ~ 0x0002_FFFF
BP3	0x0003_0000 ~ 0x0003_FFFF	BP3	0x0003_0000 ~ 0x0003_FFFF
		BP4	0x0004_0000 ~ 0x0004_FFFF
BP5	0x0005_0000 ~ 0x0005_FFFF	BP5	0x0005_0000 ~ 0x0005_FFFF
BP6	0x0006_0000 ~ 0x0006_FFFF		
BP7	0x0007_0000 ~ 0x0007_FFFF	BP7	0x0007_0000 ~ 0x0007_FFFF
		BP8	0x0008_0000 ~ 0x0008_FFFF
		BP9	0x0009_0000 ~ 0x0009_FFFF
		BP10	0x000A_0000 ~ 0x000A_FFFF
		BP11	0x000B_0000 ~ 0x000B_FFFF

7.4.2.2 Write Protection for Data Flash

The FMC_PROTECT register is used to set the Protection and Unprotection as described in Table 48.

Table 48. Data Flash Protection Area in 8 KB Units

Name	Address	Name	Address
DP0	0x0F00_0000 ~ 0x0F00_1FFF	DP2	0x0F00_4000 ~ 0x0F00_5FFF
DP1	0x0F00_2000 ~ 0x0F01_3FFF	DP3	0x0F00_6000 ~ 0x0F00_7FFF

7.4.2.3 Write Protection for OTP

The FMC_PROTECT register is used to set the Protection and Unprotection as described in 7.9.8.

- BSOTP in the FMC_PROTECT register determines whether to protect the Bank swap area 0x0040_0000 ~ 0x0040_0200.
- CIP in the FMC_PROTECT register determines whether to protect the Read Protection area 0x0020_0000 ~ 0x0020_0200.

7.4.2.4 Write Protection for Mass Erase

The mass erase area varies depending on the FMC_PROTECT register as shown in Table 49 and Table 50. When Read Protection is activated, only the blocks written with “Support” in the Read Protection Release item can be mass erased.

Table 49. Write Protection for Mass Erase in Code Flash (768 KB)

PROTECT	Erase Address	Read Protection Release
0x0000_30FF	0x0000_0000 ~ 0x0005_FFFF	No Support
0x30FF_0000	0x0006_0000 ~ 0x000B_FFFF	No Support
0x30FF_30FF	0x0000_0000 ~ 0x000B_FFFF	Support
Other	No effect mass erase	No Support

Table 50. Write Protection for Mass Erase in Code Flash (512 KB)

PROTECT	Erase Address	Read Protection Release
0x0000_00FF	0x0000_0000 ~ 0x0003_FFFF	No Support
0x00FF_0000	0x0004_0000 ~ 0x0007_FFFF	No Support
0x00FF_00FF	0x0000_0000 ~ 0x0007_FFFF	Support
Other	No effect mass erase	No Support

Table 51. Write Protection for Mass Erase in Data Flash (32 KB)

PROTECT	Erase Address	Read Protection Release
0x0000_0F00	0x0F00_0000 ~ 0x0F00_7FFF	No Support
Other	No effect mass erase	No Support

7.4.2.5 Write Protection with Memory Bank Swap

The write protection provided by the FMC_PROTECT register has the same scope regardless of the memory bank's swap status. 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 logical address.

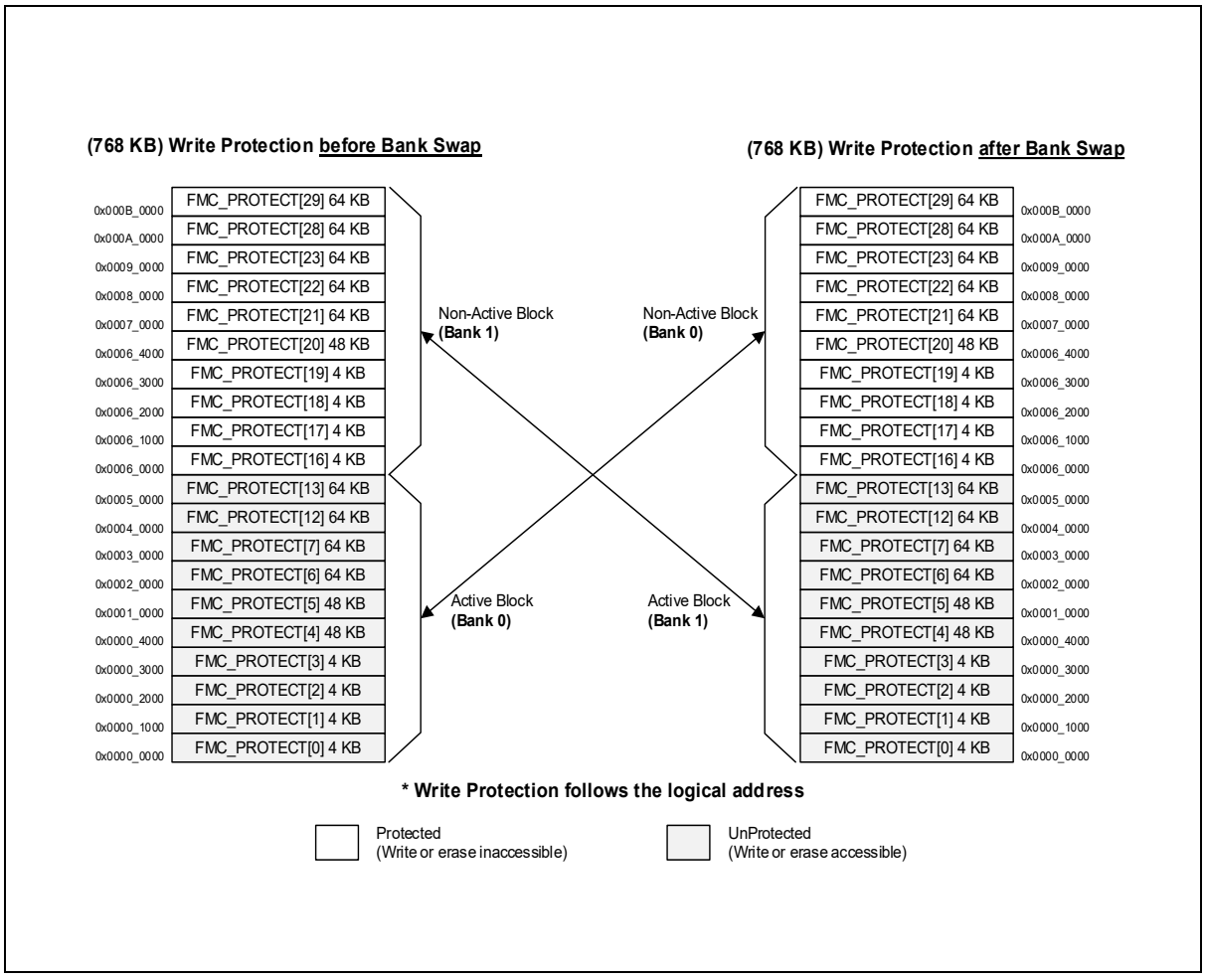
Write protected coverage of Non-active block before Memory Swap

- FMC_PROTECT = 0x000030FF in A33G538
  - active block (0x00000 to 0x5FFFF) is write/erase accessible.
  - Non-active block (0x60000 to 0xBFFFF) is write/erase inaccessible.

Write protected coverage of Non-active block after Memory Swap

- FMC_PROTECT = 0x000030FF in A33G538
  - active block (0x00000 to 0x5FFFF) is write/erase accessible.
  - Non-active block (0x60000 to 0xBFFFF) is write/erase inaccessible.

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



7.5 Bank Swap of Code Flash Memory

7.5.1 Bank Swap and Memory Structure

A33G53x series provide Flash Memory Bank Selection function for user applications and stable update and execution of user bootloader.

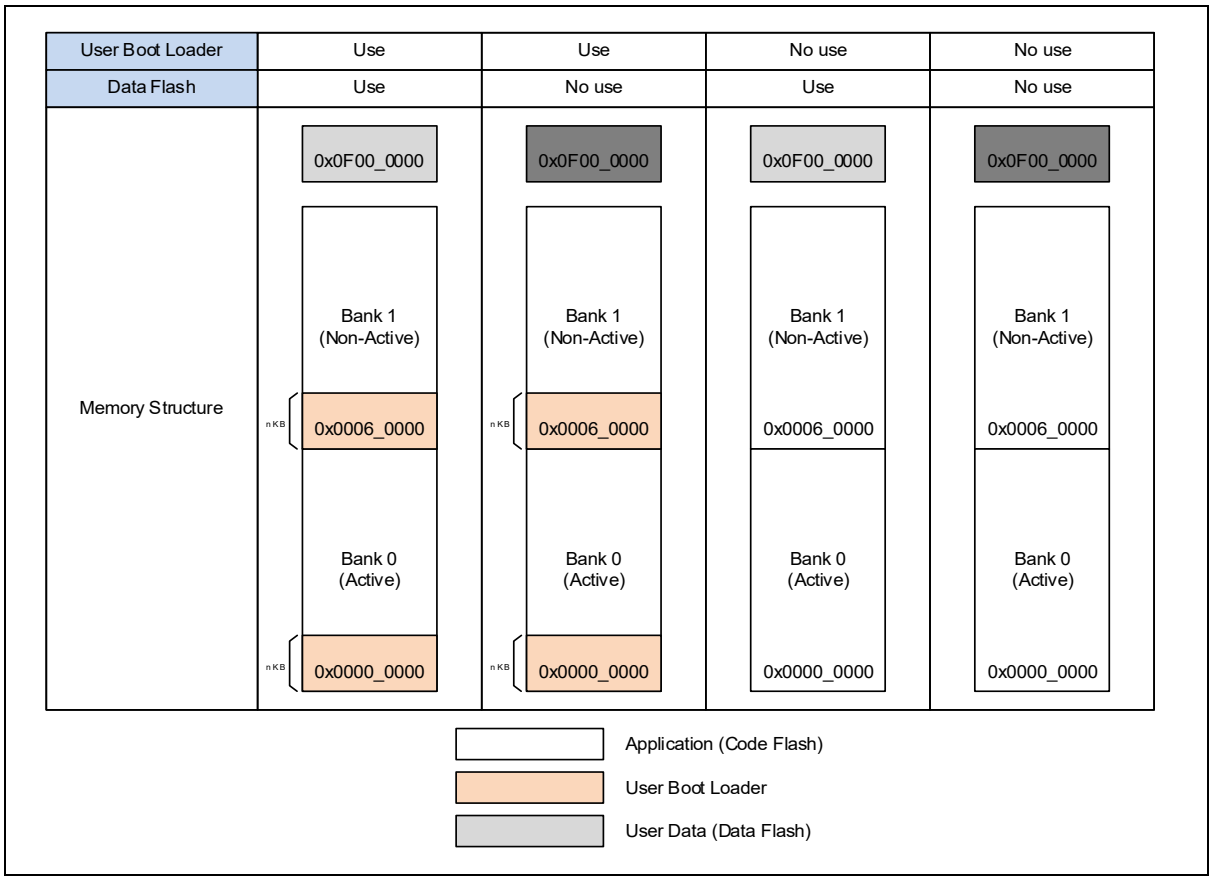
To enable Memory Swap function in A33G539, a user needs to divide 768 KB flash memory into two of 384 KB and develop them for programming. Similar to A33G539, it is recommended to divide 512 KB flash memory into two of 256 KB to utilize Memory Swap function in A33G538.

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

Table 52. Code Flash Memory Bank Status

FMC_BSR <BSST>	FMC_BSR <BSWST>	Bank Status	Active block	Non-Active block
0	0	Disable	Single	
1	0	Enable	Bank-0	Bank-1
1	1	Swap	Bank-1	Bank-0

Figure 47. A33G539 Memory Structure with Bank Swap available

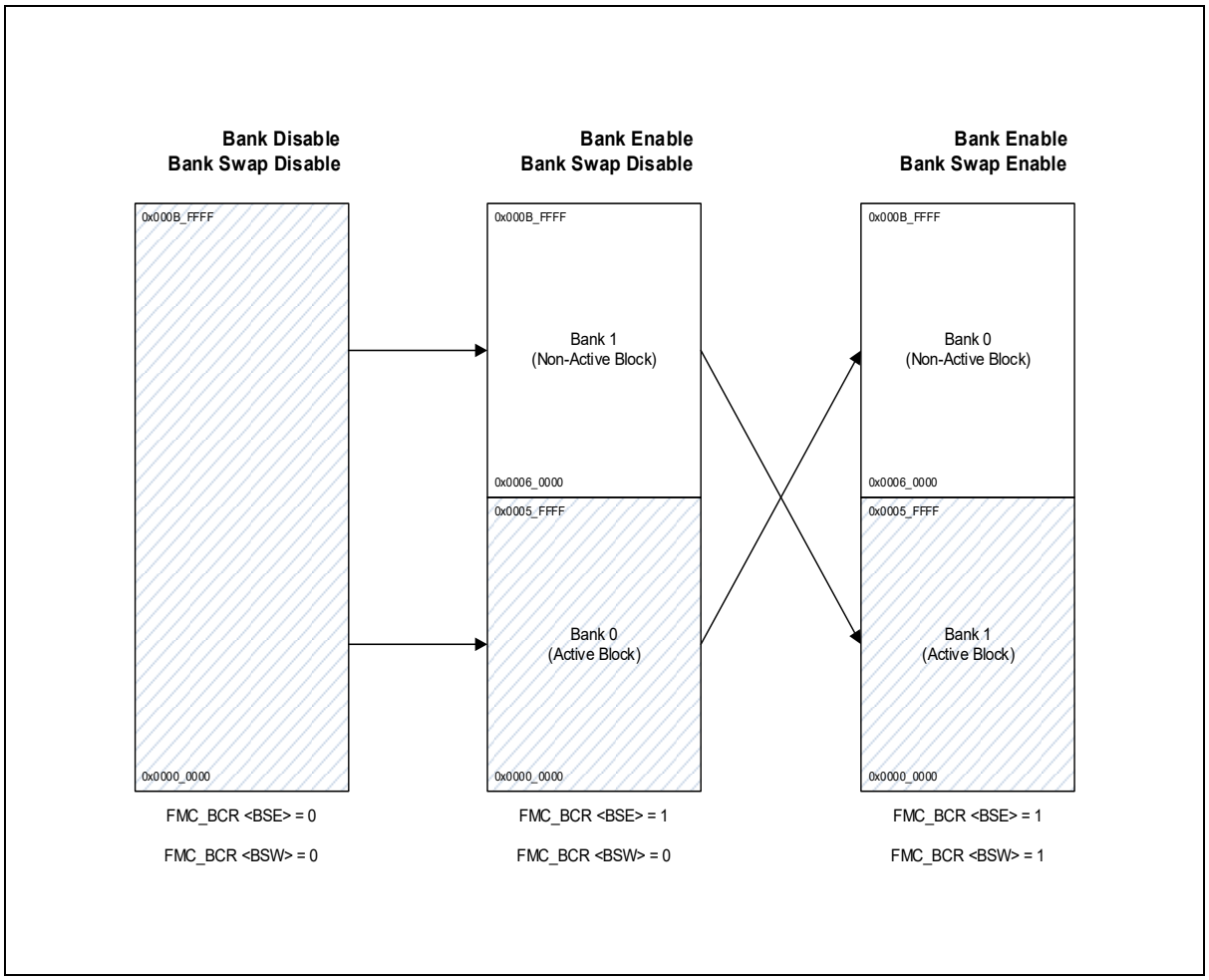


7.5.2 Bank Swap Operation

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 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

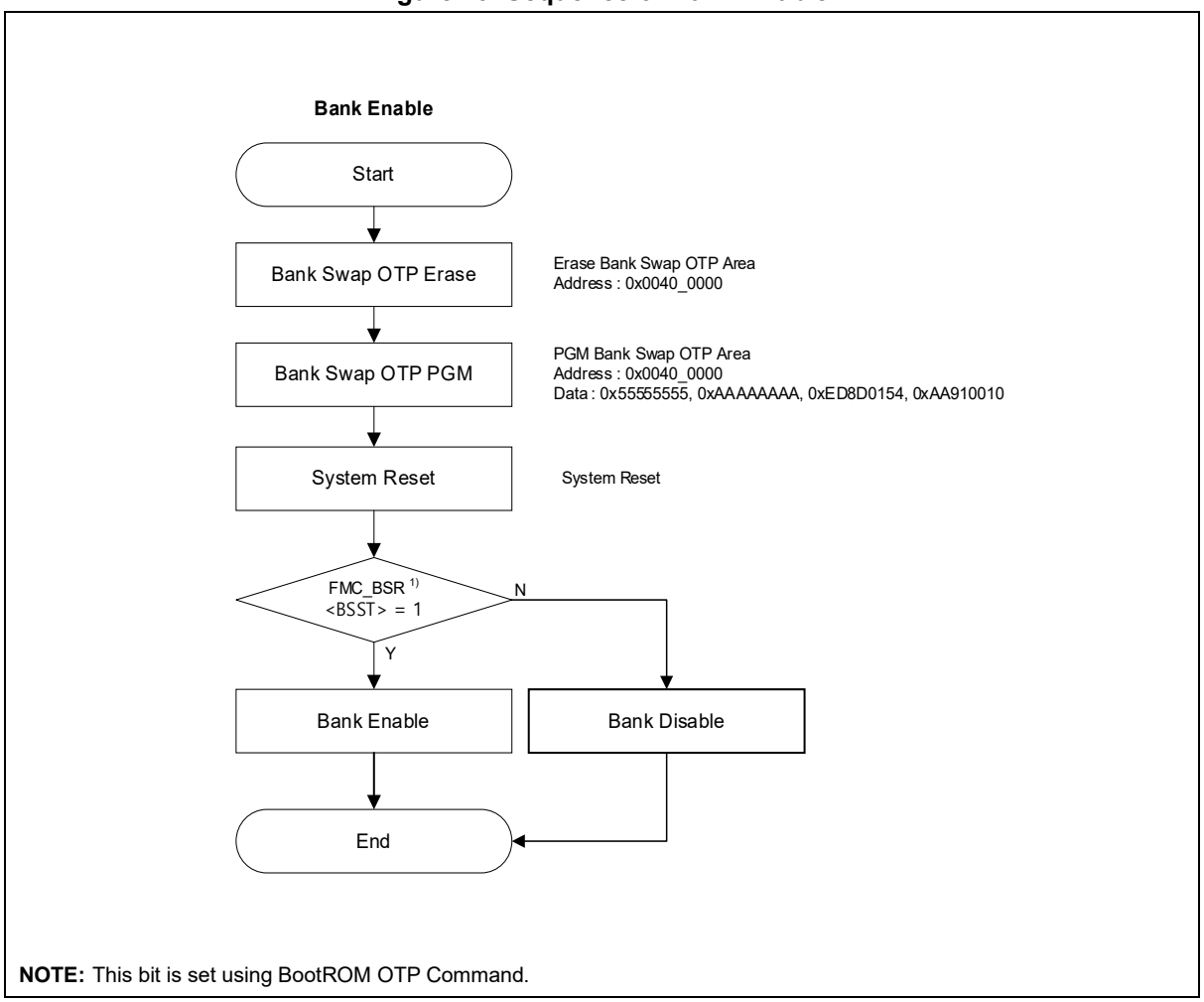


7.5.3 Limitations and Recommendations

- When Bank Swap is enabled, a user must use Self-PGM or Self-Erase to access the internal code flash memory. Sector Erase (Page Erase) and Sector Program in SRAM are also available.
- When a main program is running in Bank1 area, where the read protection is applied, if Chip (Mass) Erase is executed to disable only the read protection permanently, user data in whole area of Bank0 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.

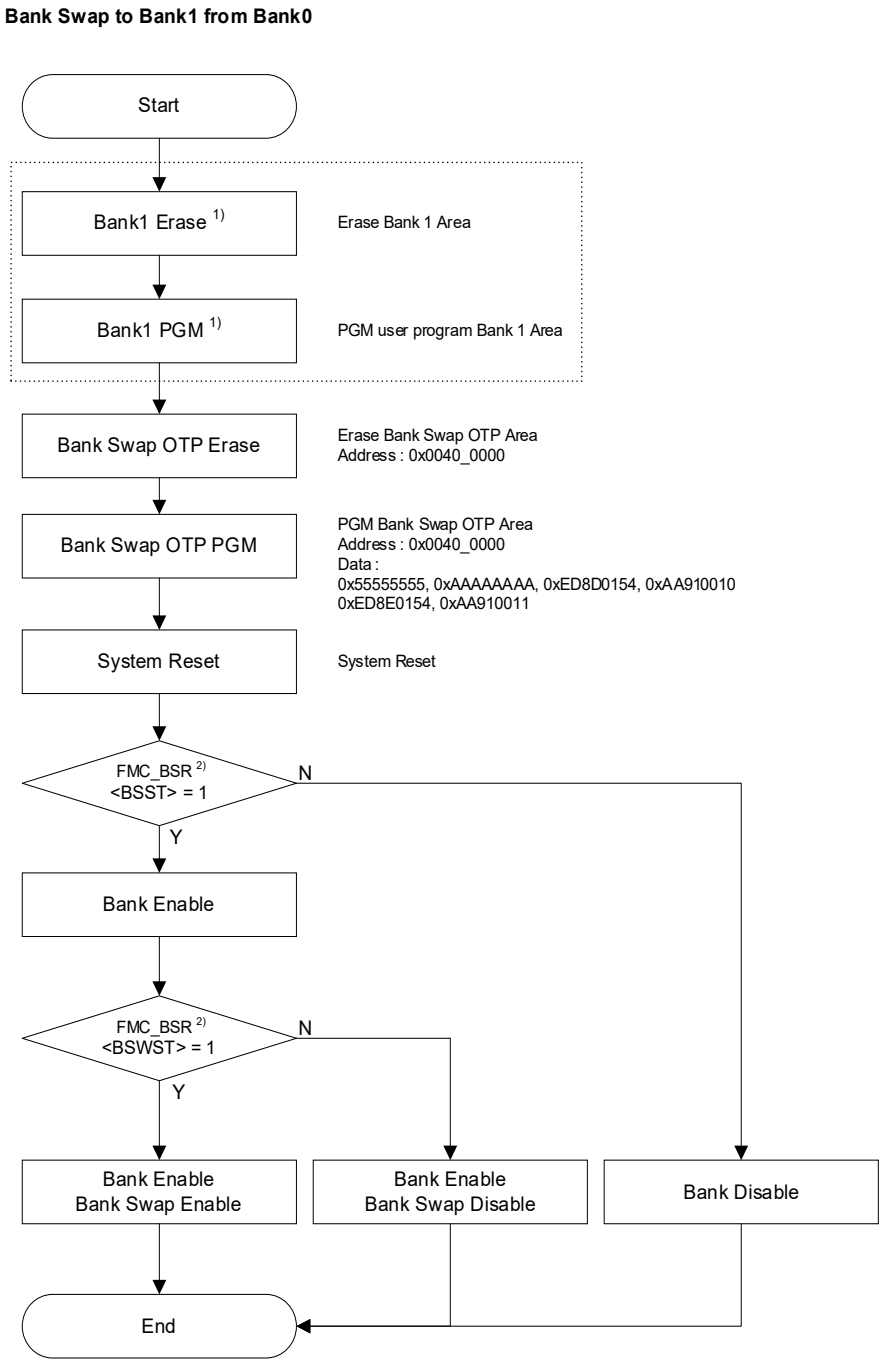
7.5.4 Sequence of Bank Enable or Swap

Figure 49. Sequence of Bank Enable



NOTE: This bit is set using BootROM OTP Command.

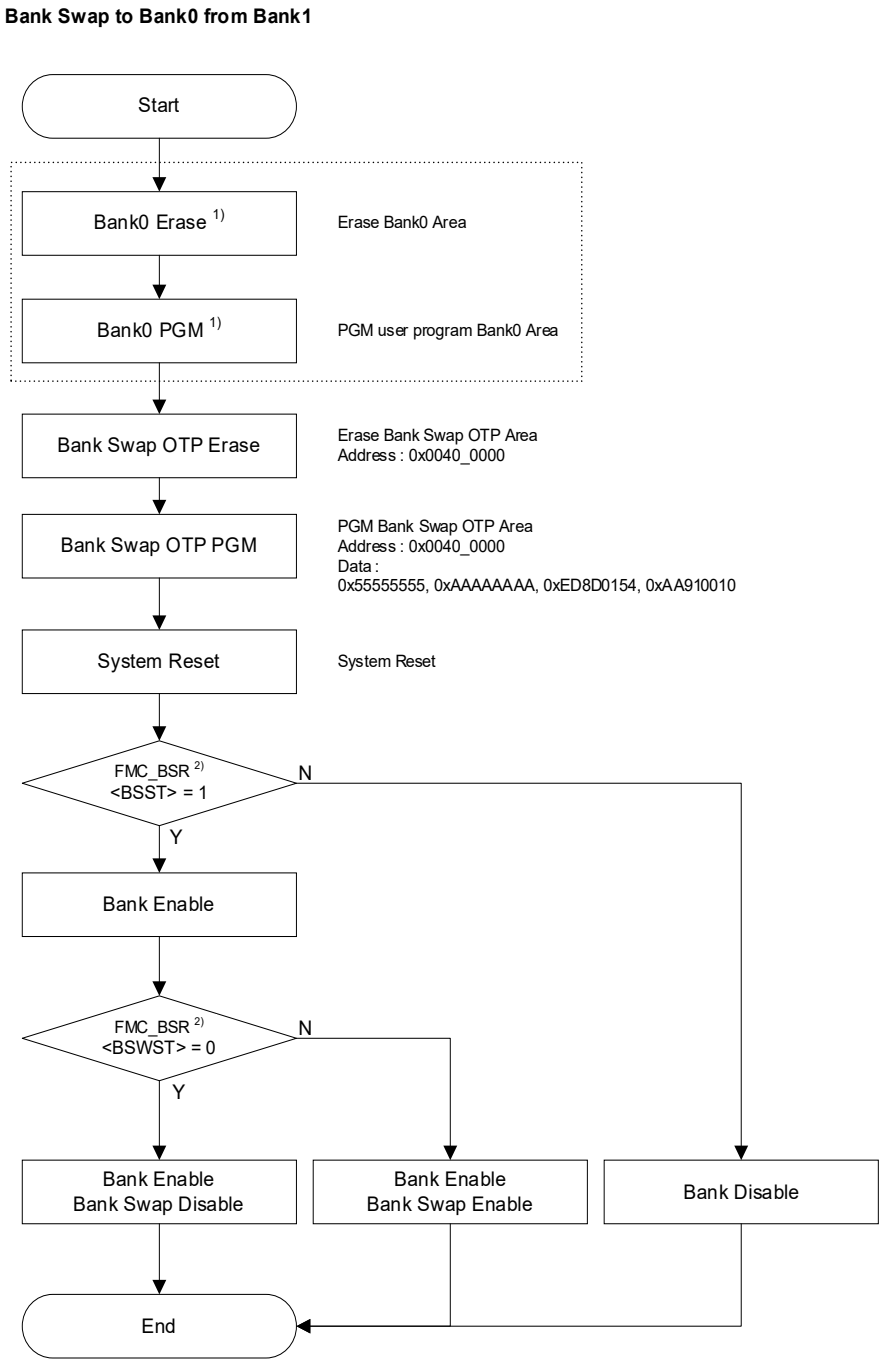
Figure 50. Sequence of Bank Swap: Bank0 → Bank1



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. This bit is set using BootROM OTP Command.

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. This bit is set using BootROM OTP Command.

7.6 RWW (Read-While-Write)

The A33G53x supports RWW(Read-While-Write) function that allows the system to be programmed while the code flash memory continues to operate. However, RWW(Read-While-Write) is available only under certain conditions, so refer to the following.

- 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
- PGM or Erase must use Self-PGM.
- Bank Disable doesn't support RWW. Refer to the 7.9.12

Table 53. Code Flash Memory RWW Features (Bank Disable)

Bank Status: Enable	Bank-0 (Active block)		Bank-1 (Non-Active block)	
	Erase	Write	Erase	Write
Bank0 (Active block)	X	X	X	X
Bank1 (Non-Active block)	X	X	X	X

Table 54. Code Flash Memory RWW Features (Bank Enable)

Bank Status: Enable	Bank-0 (Active block)		Bank-1 (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)]

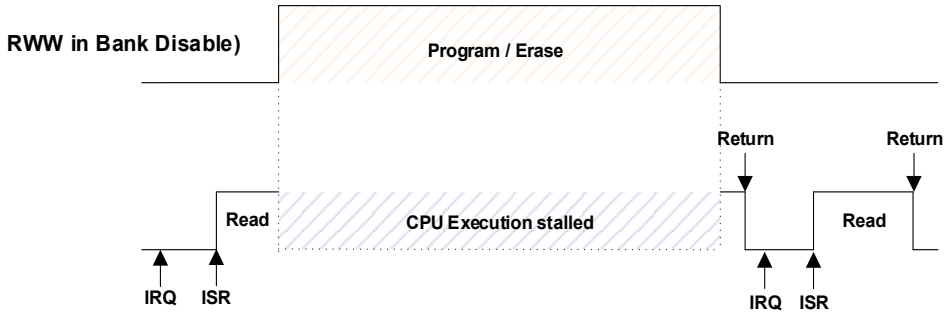
Table 55. Code Flash Memory RWW Features (Bank Swap)

Bank Status: Swap	Bank-1 (Active block)		Bank-0 (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.

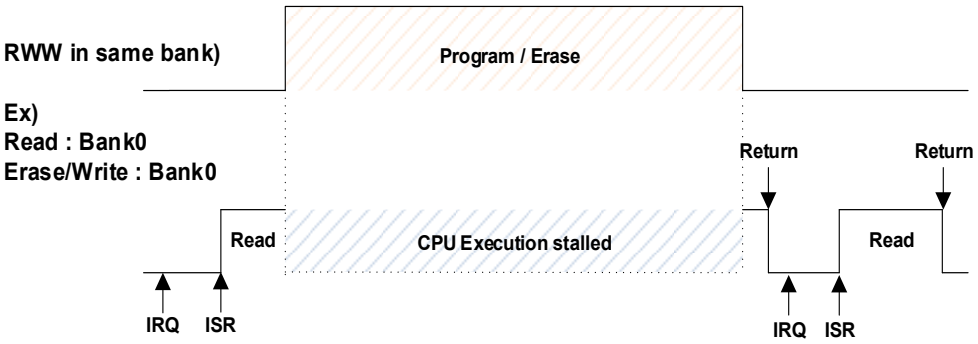
Figure 52. RWW in Code Flash Bank Disable



NOTES:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

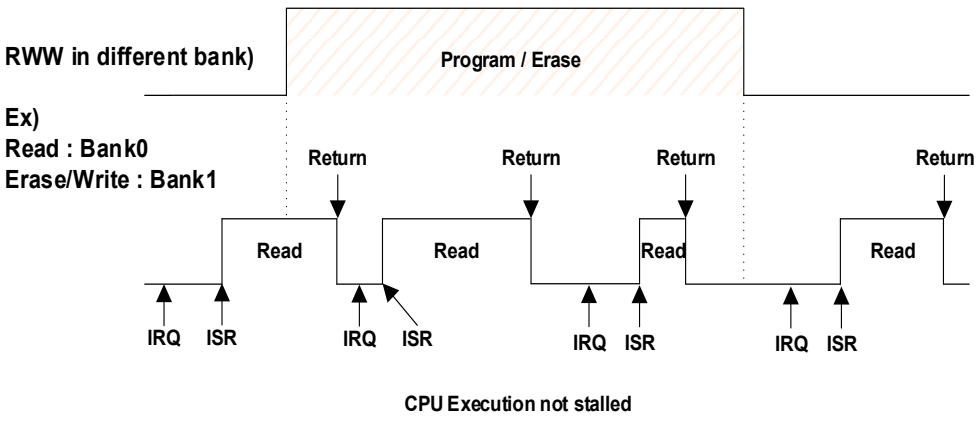
Figure 53. RWW in Code Flash Bank Enable and Bank Swap (Same Bank)



NOTES:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

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



NOTES:

- IRQ: Interrupt Request
- ISR: Interrupt Service Routine

7.7 ECC (Error Correction Code)

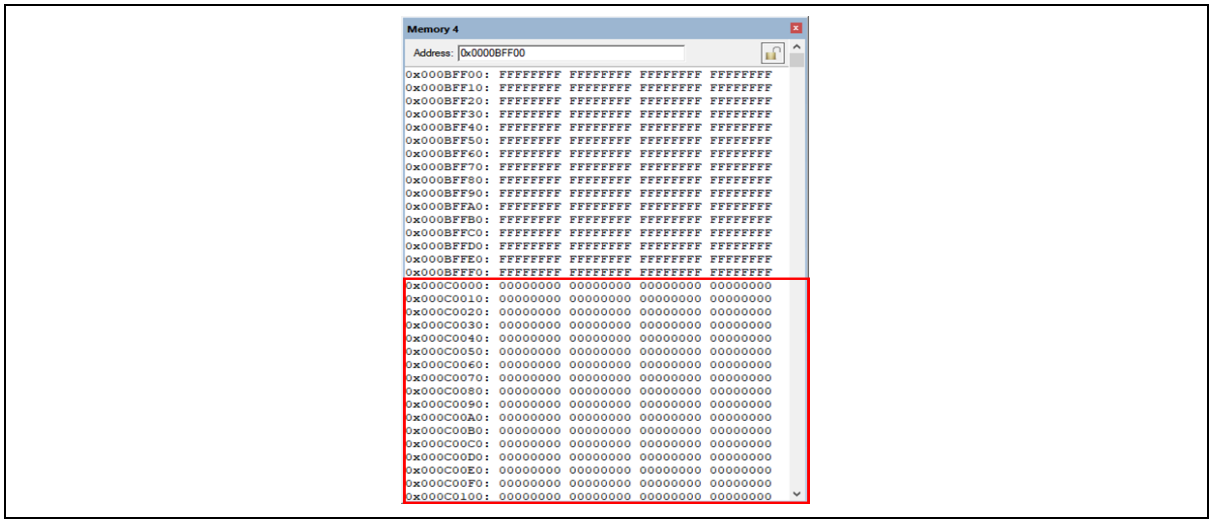
ECC has the function of correcting the flash to its original value when 1-bit is changed. If more than 2-bits are changed, no correction will be made and only an ECC error flag will be generated.

In the case of data flash, 32 bits must be written in the ecc enabled state, and 8 bit units can be written in the ecc disabled state.

Please keep the following in mind when using ECC.

- Since the code flash ECC flag occurs when an area outside the code flash area is accessed, the corresponding area should not be accessed from the debug memory window.
  - Code ECC Flag: 0xC0000 ~ 0xDFFFF **(@ A33G539 – 768 KB)**

Figure 55. Example of Code flash ECC flag occurring in debug memory window



- If the code flash memory is erased or written in the ECC enabled state and then read with ECC disabled, an ECC error will not occur.
- Conversely, if the code flash memory is erased or written while the ECC of the FMC ECCMR register is disabled, ECC function must not be enabled. In the ECC disable state, code flash memory erase/write action does not record the ECC calculation value, so if the code flash memory is erased or written while the ECC is disabled, and then enable ECC function, the calculation may be done with an wrong ECC value, which may result in incorrect correction or an ECC error flag.

7.8 Flash Interrupts

For the Code Flash and Data Flash control, A33G53x series supports some interrupt. This interrupt can be activated by setting the FMC_IER registers. To clear the error flag bit, Write “1” to this bit.

- WDIEN (Write done interrupt)
  - When Bank Swap is enabled (FMC_BCR[4] = ’1’) in Self PGM mode, WDF flag in FMC_ISR register will be set to ‘1’ when 1-word write is completed.
- WEIEN (Write error interrupt)
  - When Bank Swap is enabled (FMC_BCR[4] = ’1’) in Self PGM mode, WEF flag in FMC_ISR register will be set to ‘1’ when page or sector erase is completed.
- BSEN (Bank swap error interrupt)
  - The bank control register must be written in a running code from SRAM. Otherwise, the BSESR flag is raised.
  - If BSE in FMC_BCR register is set to ‘1’ and error occurs during SWAP operation, BSERR flag will be set to ‘1’, and the SWAP operation will be cancelled.
- DECCIEN (Data flash ECC error Interrupt)
  - If a changed value is read due to an error in the data flash memory, a data flash ECC error occurs.
  - DECCDERR (Data flash ECC double error) and DECCSERR (Data flash ECC single error) in FMC_ISR Register.
- CECCIEN (Code flash ECC error Interrupt)
  - If a changed value is read due to an error in the code flash memory, a data flash ECC error occurs.
  - CECCDERR (Code flash ECC double error) and CECCSERR (Code flash ECC single error) in FMC_ISR Register.

7.9 Flash Registers

The base address and register map of the Flash Memory Controller (FMC) are as follows.

Table 56. Base Addresses of flash Memory Controller

Name	Base address
FMC	0x4000_0100

Table 57. FMC Register Map

Name	Offset	Type	Description	Reset value	Reference
FMC_CFG	0x0000	RW	Flash Memory Configuration Register	0x00000303	7.9.1
FMC_CON	0x0004	RW	Flash Memory Control Register	0x00000000	7.9.2
FMC_ODR	0x0008	RW	Flash Memory Output Data Register	0x00000000	7.9.3
FMC_IDR	0x000C	RO	Flash Memory Input Data Register	0x00000000	7.9.4
FMC_AR	0x0010	RW	Flash Memory Address Register	0x00000000	7.9.5
FMC_TEST	0x0014	RW	Flash Memory Extension Mode Control Register	0x00000000	7.9.6
FMC_CRC	0x0018	WO	Flash CRC Register	0x0000FFFF	7.9.7
FMC_PROTECT	0x001C	RW	Flash memory protection register (768 KB)	0x00000000	7.9.8
FMC_PROTECT	0x001C	RW	Flash memory protection register (512 KB)	0x00000000	7.9.9
FMC_RPROT	0x0020	RO	Flash CRC Register	0x000000FF	7.9.10
FMC_BOOT	0x002C	RW	Flash Memory Protection Register	0x00000000	7.9.11
FMC_BCR	0x54	RW	Flash Memory Bank Control Register	0x00000000	7.9.12
FMC_BSR	0x58	RW	Flash Memory Bank Status Register	0x00000000	7.9.13
FMC_ECCMR	0x60	RW	Flash Memory ECC Mode Register	0x00000000	7.9.14
FMC_CECCEAR	0x64	RW	Code Flash Memory ECC Error Address Register	0x00000000	7.9.15
FMC_CECCELSBDR	0x68	RW	Code Flash Memory ECC Error LSB Data Register	0x00000001	7.9.16
FMC_CECCEMSBDR	0x6C	RW	Code Flash Memory ECC Error MSB Data Register	0x00000000	7.9.17
FMC_DECCEAR	0x70	RW	Data Flash Memory ECC Error Address Register	0x0F000000	7.9.18
FMC_DECCELSBDR	0x74	RW	Data Flash Memory ECC Error LSB Data Register	0x00000000	7.9.19
FMC_DECCEMSBDR	0x7C	RW	Data Flash Memory ECC Error MSB Data Register	0x00000000	7.9.20
FMC_IER	0x80	RW	Flash Memory Interrupt Enable Register	0x00000000	7.9.21
FMC_ISR	0x84	RW	Flash Memory Interrupt Status Register	0x00000000	7.9.22

7.9.1 FMC_CFG: Flash Memory Configuration Register

The FMC_CFG register is a timing setting register that adjusts the speed of the internal flash memory bus according to the internal bus speed of system.

FMC_CFG=0x4000_0100

31 30 29 28 27 26 25 24	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		DWAIT[4:0]	CZWAIT	Reserved	CWAIT[4:0]
-		00011	0	-	00011
-		RW	RW	-	RW

12 8	DWAIT	Wait value of data flash memory (Access timing is DWAIT + 1.5 clocks)
7	CZWAIT	Selection of zero-wait code flash memory (Do not use this function - Only for test)
	0	Disable zero-wait
	1	Enable zero-wait (for test only)
4 0	CWAIT	Wait value of Code Flash memory (Access timing is CWAIT + 1.5 clocks)

If CZWAIT is 1 (zero wait for code flash), the operating cycle of code flash's bus clock is 0.5 cycles and a maximum access timing of 16 MHz regardless a CWAIT value will be ignored. When CZWAIT is set to 0 (zero wait prohibition), the operating cycle that 1.5 cycles (Action 0.5 cycles + basic 1 cycle) is applied to the CWAIT setting value is applied for the flash access, and the maximum access timing is 25 MHz.

Then, the maximum access timing of code or data flash memory can be obtained following formula:

$$\text{Code or Data Flash Access Timing} = \frac{\text{HCLK}}{(0.5)} \leq 16\ \text{MHz}\ (\text{CZWAIT} = 1)$$

Otherwise,

$$\text{Code or Data Flash Access Timing} = \frac{\text{HCLK}}{(1.5\ + \text{CWAIT or DWAIT})} \leq 25\ \text{MHz}\ (\text{CZWAIT} = 0)$$

The table below shows flash access timing for each setting.

Table 58. Access Time Setting of Flash Memory depends on Bus Clock

Bus Clock (HCLK)	CZWAIT	CWAIT or DWAIT	Operating Cycle	Flash Access Timing	Max. Access Timing
8 MHz	0	0	1.5	5.3 MHz	25 MHz
8 MHz	1	X (Don’t Care)	0.5	16 MHz	16 MHz
16 MHz	0	0	1.5	10.6 MHz	25 MHz
16 MHz	1	X (Don’t Care)	-	Do not use	Do not use
20 MHz	0	0	1.5	13.3 MHz	25 MHz
20 MHz	0	1	2.5	8 MHz	25 MHz
20 MHz	1	X (Don’t Care)	-	Do not use	Do not use
25 MHz	0	0	1.5	16.6 MHz	25 MHz
25 MHz	0	1	2.5	10 MHz	25 MHz
25 MHz	1	X (Don’t Care)	-	Do not use	Do not use
32 MHz	0	0	1.5	21.3 MHz	25 MHz
32 MHz	0	1	2.5	12.8 MHz	25 MHz
32 MHz	0	2	3.5	9.14 MHz	25 MHz
74 MHz	0	2	3.5	21.1 MHz	25 MHz
74 MHz	0	3	4.5	16.4 MHz	25 MHz
75 MHz	0	2	3.5	21.4 MHz	25 MHz
75 MHz	0	3	4.5	16.6 MHz	25 MHz

7.9.2 FMC_CON: Flash Memory Control Register

FMC_CON register controls overall write / erase of flash memory and related operation control.

FMC_CON=0x4000_0104

31	30	29	28	27 26 25 24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST	STOP	Reserved	SELF	SELF_CODE [3:0]	Reserved	BBLK1	BBLK0	FSRD	Reserved	PAGE	TRSL	TSMD	DSEL	CSEL	AE	OE	CS	Reserved	DTBIT	CTBIT	NVSTR	Reserved	DTBIT_SYNC	CTBIT_SYNC	Reserved	MASE	SERA	PROG
0	0	-	0	0000	-	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	-	RO	RO	RW	-	RO	RO	-	RW	RW	RW

31	RST	Control flash reset signal
		0 Disable flash reset signal
		1 Enable flash reset signal
30	STOP	Control flash status (When the microcontroller enters the PWDN state, the flash automatically sets the PWDN state even if this bit is set to '0')
		0 Set flash normal state (Disable Flash Stop)
		1 Set flash power down state (Enable Flash Stop)
28	SELF	Code flash self program mode can be written with SELF_CODE [27:24] = 0110 (Self mode is SERA and PROG possible operating) Self-PGM only supports code flash.
		0 Disable flash self program
		1 Enable flash self program
27 24	SELF_CODE	Code value for self program of code flash (When performing self-programming function in code flash memory, 0110b value should be written in this field.)
22	BBLK1	Logical BOOT block1 protection - Boot block protection during chip (mass) erase 768 KB: (0x0006_0000 ~ 0x0006_0FFF) 512 KB: (0x0003_0000 ~ 0x0003_0FFF)
		0 Disable
		1 Enable
21	BBLK0	Logical BOOT block0 protection - Boot block protection during chip (mass) erase 768 KB: (0x0000_0000 ~ 0x0000_0FFF) 512 KB: (0x0000_0000 ~ 0x0000_0FFF)
		0 Disable
		1 Enable
20	FSRD	Fuse Area (For internal flash test)
		0 Disable
		1 Enable
18	PAGE	Code/Data flash erase per page (512 bytes)
		0 Disable
		1 Enable
17	TRSL	Test registers access (For internal flash test)
		0 Disable
		1 Enable
16	TSMD	TEST mode access (For internal flash test)
		0 Disable
		1 Enable

15	DSEL	Data flash selection	
		0	NO select
		1	Select
14	CSEL	Code flash selection	
		0	NO select
		1	Select
13	AE	Address latch enable	
		0	Disable the output the address
		1	Enable the output the address (Read access at the rising edge)
12	OE	Output enable	
		0	No action
		1	Code flash program mode
11	CS	Chip select	
		0	No action
		1	Select flashROM
9	DTBIT	The program/erase status flag of data flash	
		0	Data flash is idle (Completed)
		1	Data flash is busy
8	CTBIT	The program/erase status flag of code flash	
		0	Code flash is idle (Completed)
		1	Code flash is busy
7	NVSTR	Program/erase operation of flash memory	
		0	Stop
		1	Start
2	MASE	Mass (Full chip) erase of code or data flash memory. (See chapter 7.4.2.4) This bit is not supported in Self PGM mode.	
		0	Disable
		1	Enable
1	SERA	Select Code or Data Flash Sector (2KB) erase	
		0	Disable
		1	Enable
0	PROG	Select Code or Data Flash program	
		0	Disable
		1	Enable

7.9.3 FMC_ODR: Flash Memory Output Data Register

This register is used to write data to flash memory. In program mode, the data written to this register is stored in the flash memory.

FMC_ODR=0x4000_0108

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

31 0	FMODR	The output data to the flash memory

7.9.4 FMC_IDR: Flash Memory Input Data Register

This register is used to read data stored in the flash memory in flash memory program mode.

FMC_IDR=0x4000_010C

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

31 0	FMIDR	Input data from the flash memory

7.9.5 FMC_AR: Flash Memory Address Register

This register sets the output address of the flash memory and is used in the flash memory program mode.

When setting the code flash memory address value, you can use a width of up to 18 bits.

When setting the data flash memory address value, use a width of up to 15 bits.

Since the memory address can be set in 1-word (4 bytes) in Self Program in Code Flash memory, the bit value of A0 and A1 is always '0'. In contrast, the memory address can be set in 1-byte units in the program in the Data Flash memory, so that the values of A0 and A1 can be set to '1' or '0'.

FMC_AR=0x4000_0110

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2	1	0
Reserved	ADDR[17:0]			A1	A0
-	0x0000			0	0
-	RW			RW	RW

19 2	ADDR	Address output to the flash memory
1	A1	16-bit mode address A1 (for data flash memory)
0	A0	byte mode address A0 (for data flash memory)

NOTE:

To write or erase data in the flash memory area, the address to be accessed must be explicated. If an address out of the flash memory area is accessed, system operation is not guaranteed.

7.9.6 FMC_TEST: Flash Memory Extended Mode Control Register

This register is the Extended mode control register of the flash memory. When Extended mode is enabled, the FMC_CON register bits [31:16] are enabled. When Extended mode is enabled, you can use Self-Program, Self-Erase, 512-byte Page Erase, and Boot Block Protection.

The NVSTR signal controlled by the AUTODIS bit is an internal signal, not the same as NVSTR in bit 7 of the FMC_CON register. The NVSTR bit in FMC_CON is the user interface bit, which must be processed by the user after the flash program.

FMC_TEST=0x4000_0114

31 30 29 28 27 26 25 24	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		WRITE_KEY[7:0]	Reserved	AUTODIS	Reserved	EX
-		0x00	-	0	-	0
-		WO	-	RW	-	RW

15 8	WRITE_KEY	To set the AUTODIS or EX bit, this field must be written with 0x4A. (WRITE KEY: 0x4A)
4	AUTODIS	NVSTR auto disable function
	0	Enable auto clear
	1	Disable auto clear
0	EX	Extended mode selection
	0	Disable
	1	Enable

7.9.7 FMC_CRC: Flash Memory CRC Register

FMC_CRC is a register used to calculate the 16-bit CRC of the value stored in flash memory. Setting the CE bit to 0 initializes the CRC16 field to 0xFFFF. Then, after activating the CRC function with the CE bit set to 1, the address of the flash is designated as a pointer, and when the value stored in the flash is read, it is automatically calculated using the CRC-CCITT polynomial and the result is displayed in the CRC16 bit field. If the debugger is connected, close the memory window to calculate the correct CRC value.

FMC_CRC=0x4000_0118

31 30 29 28 27 26 25 24	23 22 21 20	19 18 17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	CE	Reserved	SEL	CRC16[15:0]	
-	0	-	0	0xFFFF	
-	RW	-	RW	RW	

20	CE	Enable/Disable CRC16 function	
		0	Disable (CRC16 result Initialized to 0xFFFF)
		1	Enable
16	SEL	Selection of CRC calculation input data	
		0	Code flash
		1	Data flash
15 0	CRC16	The result of CRC16 (Cyclic Redundancy Check 16) CRC-CCITT Polynomial : G1(x) = x16 + x12 + x5 + 1	

NOTE:

When CRC-16 is read more than 16 times, the value read at that time is calculated.

7.9.8 FMC_PROTECT: Flash Memory Protection Register (768 KB size)

This register protects the internal flash memories from unintended write and erase operation If written '0' to a bit, the relevant sectors are protected to be programmed or erased. If written '1' to a bit, the relevant sectors can be programmed or erased. (In case of writing '0', the NVSTR signal to the relevant address or sector is masked.)

FMC_PROTECT=0x4000_011C

31	30	29	28	27 26 25 24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BSOTP	CIPBANK1	BP11	BP10	Reserved	BP9	BP8	BP7	BP6	SP7	SP6	SP5	SP4	DIP	CIP	BP5	BP4	DP3	DP2	DP1	DP0	BP3	BP2	BP1	BP0	SP3	SP2	SP1	SP0
0	0	0	0	-	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

31	BSOTP	Always '0' (Bank Swap OTP)
		0 Disable (Protection)
		1 Enable (Unprotection)
30	CIPBANK1	Always '0'
29	BP11	Protection bit from 0x000B_0000 to 0x000B_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
28	BP10	Protection bit from 0x000A_0000 to 0x000A_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
23	BP9	Protection bit from 0x0009_0000 to 0x0009_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
22	BP8	Protection bit from 0x0008_0000 to 0x0008_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
21	BP7	Protection bit from 0x0007_0000 to 0x0007_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
20	BP6	Protection bit from 0x0006_4000 to 0x0006_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
19	SP7	Protection bit from 0x0006_3000 to 0x0006_3FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
18	SP6	Protection bit from 0x0006_2000 to 0x0006_2FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
17	SP5	Protection bit from 0x0006_1000 to 0x0006_1FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
16	SP4	Protection bit from 0x0006_0000 to 0x0006_0FFF
		0 Disable (Protection)
		1 Enable (Unprotection)

15	DIP	Always ‘0’	
14	CIP	Always ‘0’ (Read protection OTP)	
13	BP5	Protection bit from 0x0005_0000 to 0x0005_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
12	BP4	Protection bit from 0x0004_0000 to 0x0004_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
11	DP3	Protection bit of the Data Flash 8KB #3	
		0	Disable (Protection)
		1	Enable (Unprotection)
10	DP2	Protection bit of the Data Flash 8KB #2	
		0	Disable (Protection)
		1	Enable (Unprotection)
9	DP1	Protection bit of the Data Flash 8KB #1	
		0	Disable (Protection)
		1	Enable (Unprotection)
8	DP0	Protection bit of the Data Flash 8KB #0	
		0	Disable (Protection)
		1	Enable (Unprotection)
7	BP3	Protection bit from 0x0003_0000 to 0x0003_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
6	BP2	Protection bit from 0x0002_0000 to 0x0002_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
5	BP1	Protection bit from 0x0001_0000 to 0x0001_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
4	BP0	Protection bit from 0x0000_4000 to 0x0000_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
3	SP3	Protection bit from 0x0000_3000 to 0x0000_3FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
2	SP2	Protection bit from 0x0000_2000 to 0x0000_2FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
1	SP1	Protection bit from 0x0000_1000 to 0x0000_1FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
0	SP0	Protection bit from 0x0000_0000 to 0x0000_0FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)

7.9.9 FMC_PROTECT: Flash Memory Protection Register (512 KB size)

This register protects the internal flash memories from unintended write and erase operation If written '0' to a bit, the relevant sectors are protected to be programmed or erased. If written '1' to a bit, the relevant sectors can be programmed or erased. (In case of writing '0', the NVSTR signal to the relevant address or sector is masked.)

FMC_PROTECT=0x4000_011C

31	30	29 28 27 26 25 24	23	22	21	20	19	18	17	16	15	14	13 12	11	10	9	8	7	6	5	4	3	2	1	0
BSOTP	CIPBANK1	Reserved	BP9	BP8	BP7	BP6	SP7	SP6	SP5	SP4	DIP	CIP	Reserved	DP3	DP2	DP1	DP0	BP3	BP2	BP1	BP0	SP3	SP2	SP1	SP0
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

31	BSOTP	Always '0' (Bank Swap OTP)
		0 Disable (Protection)
		1 Enable (Unprotection)
30	CIPBANK1	Always '0'
23	BP9	Protection bit from 0x0007_0000 to 0x0007_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
22	BP8	Protection bit from 0x0006_0000 to 0x0006_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
21	BP7	Protection bit from 0x0005_0000 to 0x0005_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
20	BP6	Protection bit from 0x0004_4000 to 0x0004_FFFF
		0 Disable (Protection)
		1 Enable (Unprotection)
19	SP7	Protection bit from 0x0004_3000 to 0x0004_3FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
18	SP6	Protection bit from 0x0004_2000 to 0x0004_2FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
17	SP5	Protection bit from 0x0004_1000 to 0x0004_1FFF
		0 Disable (Protection)
		1 Enable (Unprotection)
16	SP4	Protection bit from 0x0004_0000 to 0x0004_0FFF
		0 Disable (Protection)
		1 Enable (Unprotection)

15	DIP	Always ‘0’	
14	CIP	Always ‘0’ (Read protection OTP)	
11	DP3	Protection bit of the Data Flash 8 KB #3	
		0	Disable (Protection)
		1	Enable (Unprotection)
10	DP2	Protection bit of the Data Flash 8 KB #2	
		0	Disable (Protection)
		1	Enable (Unprotection)
9	DP1	Protection bit of the Data Flash 8 KB #1	
		0	Disable (Protection)
		1	Enable (Unprotection)
8	DP0	Protection bit of the Data Flash 8 KB #0	
		0	Disable (Protection)
		1	Enable (Unprotection)
7	BP3	Protection bit from 0x0003_0000 to 0x0003_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
6	BP2	Protection bit from 0x0002_0000 to 0x0002_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
5	BP1	Protection bit from 0x0001_0000 to 0x0001_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
4	BP0	Protection bit from 0x0000_4000 to 0x0000_FFFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
3	SP3	Protection bit from 0x0000_3000 to 0x0000_3FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
2	SP2	Protection bit from 0x0000_2000 to 0x0000_2FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
1	SP1	Protection bit from 0x0000_1000 to 0x0000_1FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)
0	SP0	Protection bit from 0x0000_0000 to 0x0000_0FFF	
		0	Disable (Protection)
		1	Enable (Unprotection)

7.9.10 FMC_RPROT: Flash Memory Read Protection Register

This register is used to set the protection to the read, access to the flash memory.

FMC_RPROT=0x4000_0120

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

30	RPROT	This bit indicates the read protection status	
		0	Disabled (Allow read access)
		1	Enabled (Prevent read access)
28	CERS	This bit indicates the status of flash erase (This bit is set to '1' when the 'Mass (Chip) Erase' command is executed during the erase command). See chapter 7.4.2.4	
		0	Mass (Chip) erase NOT done
		1	Mass (Chip) erase done
7 0	FMRPROT	Setting read access protection (If read protection is enabled and a debugger is connected, The flash value is scrambled in the memory window and read as 0xAA55AA55.)	
		0xFF	No read protection (Permitted read access)
		0x39	Protection Level 1 (Protected from read access, permitted debugger access)

7.9.11 FMC_BOOT: Flash Memory Boot Register

This register identifies the boot mode and indicates the operating status of BootRom. The BOOT bit field can only be accessed in boot mode that boot code located in BootRom is executed. In normal user code execution mode, BOOT bit field value is always displayed 0.

FMC_BOOT=0x4000_012C

31	30	29	28	27	26	25	24	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																															BOOT
-																															0
-																															RW

0	BOOT	The operation status of BootROM. This field is '0' in normal user mode.

7.9.12 FMC_BCR: Flash Memory Bank Control Register

FMC_BCR register is used to control Banks. To utilize Bank SWAP function, a user needs to use Active Bank and Non-active Bank by enabling BSE function first. Then BSW bit needs to be set. To change values in this register, WTIDKY key must be set together. This register FMC_BCR must be accessed from BootROM area. Access from other area than the BootROM generates FMC_BSR<BSERR> error flag and maintains the previous state. Refer to the 7.5.4

FMC_RPROT=0x4000_0154

31 30 29 28 27 26 25 24	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		Reserved		BSE	Reserved	BSW
0x0000		-		0	-	0
WO		-		RW	-	RW

31 16	WTIDKY	Write identification key Write WTIDKY in this field to set the bank control register bits. If a value other than WTIDKY is entered, it is ignored.
	0xAA91	Write identification key to set the bank control register
4	BSE	Memory Bank Selection Enable bit for swap operation. When this bit is set to 1, Code flash is used as active and non-active banks of consecutive memory addresses. **NOTE:** This bit should be set with WTIDKY If BSW is 0, BSE of FMC BCR register can be changed in the main code, but do not change if BSW is 1. When BSE is 1, Read While Write (RWW) is supported when using Self PGM.
	0	Disable memory bank control bit to swap [A33G539] Active Bank = 768 KB (0x0000_0000～0x000B_FFFF) [A33G538] Active Bank = 512 KB (0x0000_0000～0x0007_FFFF)
	1	Enable memory bank control bit to swap [A33G539] Active Bank = 384 KB (0x0000_0000～0x0005_FFFF) No active Bank = 384KB (0x0006_0000～0x000B_FFFF) [A33G538] Active Bank = 256 KB (0x0000_0000～0x0003_FFFF) No Active Bank = 256 KB (0x0004_0000～0x0007_FFFF)
0	BSW	Bank Swap (Switching active and non-active bank) **NOTE:** This bit must be set with WTIDKY and the BSE bit set to '1'
	0	The logical address '0x0000_0000' starts from physical address 0x0000_0000.
	1	[A33G539] The logical address '0x0000_0000' starts from physical address 0x0006_0000. [A33G538] The logical address '0x0000_0000' starts from physical address 0x0004_0000.

7.9.13 FMC_BSR: Flash Memory Bank Status Register

FMC_BSR enables to generate a flag which is used to check status of memory bank operation. By checking the flag provided by this register, a user can recognize specific memory area to execute code at system initialization stage. This register CFMC_BSR must be accessed from BootROM area. Access from other area than the BootROM generates CFMC_BSR error flag and maintains the previous state.

FMC_BSR=0x4000_0158

31 30 29 28 27 26 25 24	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			CBF	Reserved	BSE	Reserved	BSW
-			0	-	0	-	0
-			RW	-	RO	-	RO

8	CBF	Current boot bank flag bit **NOTE:** This bit is used to set or release a state flag of Bank Swap operation. This bit can be used in the form of OTP Command and is set to '0' if the flag is not used.	
		0	Current boot area is bank0
		1	Current boot area is bank1
0	BSST	Bank selection status bit **NOTE:** BSST bit is activated when BSWST bit is enabled that indicates current active area is bank1	
		0	Disable memory bank selection. [A33G539] Active Bank = 768 KB (0x0000_0000～0x000B_FFFF) [A33G538] Active Bank = 512 KB (0x0000_0000～0x0007_FFFF)
		1	Enable memory bank selection. [A33G539] Active Bank = 384 KB (0x0000_0000～0x0005_FFFF) Non-active Bank = 384 KB (0x0006_0000～0x000B_FFFF) [A33G538] Active Bank = 256 KB (0x0000_0000～0x0003_FFFF) Non-Active Bank = 256 KB (0x0004_0000～0x0007_FFFF)
0	BSWST	Bank Swap status bit	
		0	Not occurred bank swap
		1	Occurred bank swap

7.9.14 FMC_ECCMR: Flash Memory ECC Mode Register

This register turns on or off the ECC function of the flash memories.

FMC_ECCMR=0x4000_0160

31 30 29 28 27 26 25 24	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				DECCEN	CECCEN
-				0	1
-				RW	RW

1	DECCEN	0	Disable data Flash ECC (Default)
		1	Enable data Flash ECC Enable
0	CECCEN	0	Disable code Flash ECC
		1	Enable code Flash ECC (Default)

7.9.15 FMC_CECCEAR: Code Flash Memory ECC Error Address Register

This register records the adderess of the first code flash that ECC errors occurred when an ECC error in the code flash memory occurs.

FMC_CECCEAR=0x4000_0164

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

31 0	CECCEAR	The address of the code flash where the ECC ERROR first occurred in the code flash memory. **NOTE:** After clearing the code flash ECC error status of FMC_ISR and reading the code flash area where the ECC error occurred, the ECC error address will be saved.

7.9.16 FMC_CECCELSBDR: Code Flash Memory ECC Error LSB Data Register

This register records the data at the first code flash address that ECC errors occurred when an ECC error in the code flash memory occurs.

FMC_CECCELSBDR=0x4000_0168

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

31 0	CECCELSBDR	The data at the code flash address where the ECC error first occurred in the code flash memory. **NOTE:** After clearing the code flash ECC error status of FMC_ISR and reading the code flash area where the ECC error occurred, the ECC error data will be saved.

7.9.17 FMC_CECCEMSBDR: Code Flash memory ECC error MSB data register

This register records the parity value at the first code flash address that ECC errors occurred when an ECC error in the code flash memory occurs.

FMC_ CECCEMSBDR=0x4000_016C

31 30 29 28 27 26 25 24	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			CECCEMSBDR
-			0x00
-			RO

31 0	CECCEMSBDR	The parity value at the code flash address where the ECC error first occurred in the code flash memory. **NOTE:** After clearing the code flash ECC error status of FMC_ISR and reading the code flash area where the ECC error occurred, the ECC error parity value will be saved.

7.9.18 FMC_DECCEAR: Data Flash Memory ECC Error Address Register

This register records the address of the first data flash that ECC errors occurred when an ECC error in the data flash memory occurs.

FMC_DECCEAR=0x4000_0170

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DECCEAR																															
0x0F000000																															
RO																															

31 0	DECCEAR	The address of the data flash where the ECC ERROR first occurred in the data flash memory **NOTE:** After clearing the data flash ECC error status of FMC_ISR and reading the data flash area where the ECC error occurred, the ECC error address will be saved.

7.9.19 FMC_DECCELSBDR: Data Flash Memory ECC Error LSB Data Register

This register records the data at the first data flash address that ECC errors occurred when an ECC error in the data flash memory occurs.

FMC_DECCELSBDR=0x4000_0174

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

31 0	DECCELSBDR	The data at the data flash address where the ECC error first occurred in the data flash memory. **NOTE:** After clearing the data flash ECC error status of FMC_ISR and reading the data flash area where the ECC error occurred, the ECC error data will be saved.

7.9.20 FMC_DECCEMSBDR: Data Flash memory ECC error MSB data register

This register records the parity value at the first data flash address that ECC errors occurred when an ECC error in the data flash memory occurs.

FMC_ DECCEMSBDR=0x4000_017C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			DECCEMSBDR
-			0x00
-			RO

31 0	DECCEMSBDR	The parity value at the data flash address where the ECC error first occurred in the code flash memory. **NOTE:** After clearing the data flash ECC error status of FMC_ISR and reading the data flash area where the ECC error occurred, the ECC error parity value will be saved.

7.9.21 FMC_IER: Flash Memory Interrupt Enable Register

The FMC_IER register is the internal flash memory's interrupt enable register. It is a 32-bit register.

FMC_IER=0x4000_0180

31 30 29 28 27 26 25 24	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			WDIEN	WEIEN	Reserved	BSEN	Reserved	DECCIEN	Reserved	CECCIEN
-			0	0	-	0	-	0	-	1
-			RW	RW	-	RW	-	RW	-	RW

13	WDIEN	Write done interrupt	
		0	Disable
		1	Enable
12	WEIEN	Write error Interrupt	
		0	Disable
		1	Enable
8	BSEN	Bank swap error interrupt	
		0	Disable
		1	Enable
4	DECCIEN	Data flash ECC error interrupt	
		0	Disable
		1	Enable
0	CECCIEN	Code flash ECC error interrupt	
		0	Disable
		1	Enable

NOTE:

Write Done and Write Error Interrupt can be checked by using Self PGM mode with Bank swap enabled (FMC_BCR[4] ='1').

7.9.22 FMC_ISR: Flash Memory Interrupt Status Register

The FMC_ISR register is the internal flash memory's interrupt status register. It is a 32-bit register.

FMC_ISR=0x4000_0184

31 30 29 28 27 26 25 24	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			WDF	WEF	Reserved	BSERR	Reserved	DECCDERR	DECCSERR	Reserved	CECCDERR	CECCSERR
-			0	0	-	0	-	0	0	-	0	1
-			WC	WC	-	WC	-	WC	WC	-	WC	WC

13	WDF	Write done flag
	0	No occur
	1	Write done. (To clear the error flag bit, Write "1" to this bit.)
12	WEF	Write error flag
	0	Disable
	1	Write error. (To clear the error flag bit, Write "1" to this bit.)
8	BSERR	Error handling flag bit for bank selection If the bank selection malfunctions, BSERR bit is set to "1". To clear the error flag bit, Write "1" to this bit. **NOTES:** 1. The bank control register must be controlled at SRAM or BootROM. Otherwise, the SWAPERR flag is raised. 2. If CFMC_BCR<BSE> bit is set to '1' and error occurs during SWAP operation, BSERR flag is generated, and the previous state is maintained. 3. If Swap operation error occurs (BSERR = 1), a user must enable SWAPFAILEN bit of SCU_NMIR register to process corresponding interrupts.
	0	No occurred bank selection Error
	1	Occurred bank selection Error (To clear the error flag bit, Write "1" to this bit.)
5	DECCDERR	Data flash ECC double error status
	0	No Error
	1	Error (To clear the error flag bit, Write "1" to this bit.)
4	DECCSERR	Data flash ECC single Error status
	0	No Error
	1	Error (To clear the error flag bit, Write "1" to this bit.)
1	CECCDERR	Code flash ECC double Error status
	0	No Error
	1	Error (To clear the error flag bit, Write "1" to this bit.)
0	CECCSERR	Code flash ECC single Error status
	0	No Error
	1	Error (To clear the error flag bit, Write "1" to this bit.)

NOTE:

Write Done and Write Error Interrupt can be checked by using Self PGM mode with Bank swap enabled (FMC_BCR[4] ='1').

7.9.23 FMC Register Map Summary

Table 59. FMC 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	FMC_CFG	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DWAIT[4:0]					CZWAIT	Res.	Res.	CWAIT[4:0]				
	Reset value																				0	0	0	1	1	0			0	0	0	1	1
0x04	FMC_CON	RST	STOP	Res.	SELF	SELF_CODE [3:0]				Res.	BBLK1	BBLK0	FSRD	Res.	PAGE	TRSL	TSMD	DSEL	CSEL	AE	OE	CS	Res.	DTBIT	CTBIT	NVSTR	Res.	DTBIT_SYNC	CTBIT_SYNC	Res.	MASE	SERA	PROG
	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
0x08	FMC_ODR	FMODR[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	FMC_IDR	FMIDR[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
0x10	FMC_AR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDR[16:0]																	A1	A0
	Reset value														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	FMC_TEST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRITE_KEY[7:0]								Res.	Res.	Res.	AUTODIS	Res.	Res.	Res.	EX
	Reset value																	0	0	0	0	0	0	0	0				0				0
0x18	FMC_CRC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CE	Res.	Res.	Res.	SEL	CRC16[15:0]															
	Reset value												0				0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x1C	FMC_PROTECT (768 KB)	BSOTP	CIPBNAK1	BP11	BP10	Res.	Res.	Res.	Res.	BP9	BP8	BP7	BP6	SP7	SP6	SP5	SP4	DIP	CIP	BP5	BP4	DP3	DP2	DP1	DP0	BP3	BP2	BP1	BP0	SP3	SP2	SP1	SP0
	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
0x1C	FMC_PROTECT (512 KB)	BSOTP	CIPBNAK1	Res.	Res.	Res.	Res.	Res.	Res.	BP9	BP8	BP7	BP6	SP7	SP6	SP5	SP4	DIP	CIP	Res.	Res.	DP3	DP2	DP1	DP0	BP3	BP2	BP1	BP0	SP3	SP2	SP1	SP0
	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
0x20	FMC_RPROT	Res.	RPROT	Res.	CERS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FMRPROT[7:0]							
	Reset value		0		0																					1	1	1	1	1	1	1	1
0x2C	FMC_BOOT	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.	BOOT
	Reset value																																0

Table 59. FMC 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	FMC_BCR	WTIDKY[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSE	Res.	Res.	Res.	BSW
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												0				0
0x58	FMC_BSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CBF	Res.	Res.	Res.	BSST	Res.	Res.	Res.	BSWST
	Reset value																								0				0				0
0x60	FMC_ECCMR	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.	DECCEN	CECCEN
	Reset value																															0	1
0x64	FMC_CECCEAR	CECCEAR[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	FMC_CECCELSBDR	CECCELSBDR[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	FMC_CECCEMSBDR	CECCEMSBDR[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	FMC_DECCEAR	DECCEAR[31:0]																															
	Reset value	0	0	0	0	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
0x74	FMC_DECCELSBDR	DECCELSBDR[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
0x7C	FMC_DECCEMSBDR	DECCEMSBDR[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
0x80	FMC_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDIEN	WEIEN	Res.	Res.	Res.	BSEN	Res.	Res.	Res.	DECCIEN	Res.	Res.	Res.	CECCIEN
	Reset value																			0	0				0				0				0
0x84	FMC_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDF	WEF	Res.	Res.	Res.	BSERR	Res.	Res.	DECCDERR	DECCSERR	Res.	Res.	CECCDERR	CECCSERR
	Reset value																			0	0				0				0				0

8. 32-bit Watchdog Timer (32-bit WDT)

8.1 WDT Introduction

Watchdog Timer (WDT) is used to detect microcontroller errors due to the 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 A33G53x series is a 32-bit down counter. If the WDT is set as a reset source, the microcontroller restarts when the down counter reaches zero value.

When it is not used to monitor the microcontroller, the WDT can be used as a cycle timer along with an interrupt.

8.2 WDT Main Features

The WDT module has main 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 WDTCS[1:0] in PMU_PCSR register : Main XTAL, IOSC16, SXOSC, RINGOSC
- Eight levels prescaler for the WDT input clock
- In debug mode, the WDT counter can be set not to operate.

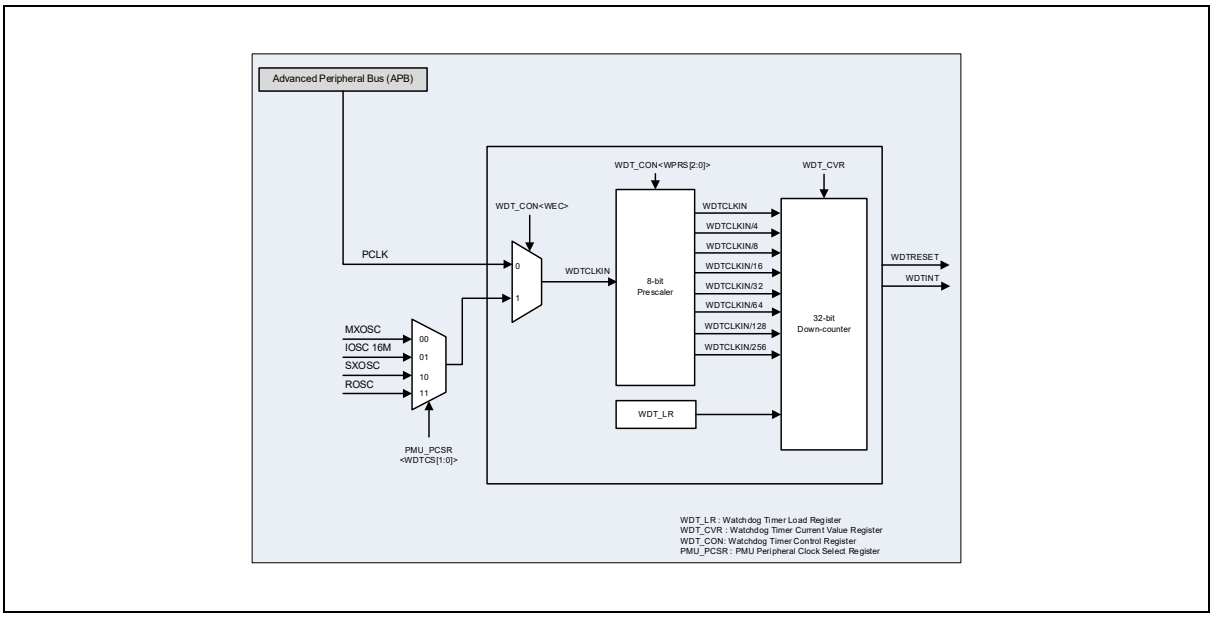
8.3 WDT Functional Description

Once the WDT module is enabled and the WEN in WDT_CON register is set to ‘1’, the 32-bit down counter is decremented. 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.

8.3.1 WDT Block Diagram

Figure 56 shows a block diagram for the WDT.

Figure 56. WDT Block Diagram



8.3.2 Enabling the WDT

The WDT is always enabled after power 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 WEC in WDT_CON register is set to ‘1’ and the PMU_PCSR register is set to select the external clock source. When changing the clock source, set it after disabling WEN in WDT_CON register.

8.3.3 Controlling the Down Counter

When the WEN in WDT_CON register is set to '1', the 32-bit down counter that is the WDT_CVR register starts counting down. To prevent an instant reset or an interrupt during the down counter operation, the WDT_CVR 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 WEN in 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 '0'

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] in WDT_CON register, and the minimum frequency value of the prescaler is 1/256 of the clock source frequency.

Table 60 describes the prescaled frequencies of various clock sources in the WDT.

Table 60. Prescaled WDT Counter Clock Frequency

Clock source	WDTCLKIN	WDTCLKIN / 4	WDTCLKIN / 16	WDTCLKIN / 64	WDTCLKIN / 256
RINGOSC	1 MHz	250 kHz	62.5 kHz	15.625 kHz	3.90625 kHz
Sub XTAL	32.768 kHz	8.192 kHz	2.048 kHz	512Hz	128Hz
PCLK	PCLK	PCLK / 4	PCLK / 16	PCLK / 64	PCLK / 256
IOSC16	16 MHz	6 MHz	1 MHz	250 kHz	62.5 kHz
Main XTAL	XTAL	XTAL / 4	XTAL / 16	XTAL / 64	XTAL / 256
PLL	PLL	PLL / 4	PLL / 16	PLL / 64	PLL / 256

NOTES:

1. Time-out period = (Load Value + 1) × (1 / pre-scaled WDT counter clock frequency) at Interrupt operation
2. Time-out period = (Load Value) × (1 / pre-scaled WDT counter clock frequency) at reset operation
3. At the Load Value reaches zero, 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 WIE in WDT_CON register to '1'. At the moment when the value of the WDT_CVR down counter changes from '1' to '0', the interrupt is triggered and the WOF in WDT_CON register is set to '1'. To clear the WOF bit, users must write a value other than zero in the WDT_LR register.

8.3.5 How to Program the WatchDog Timeout

8.3.5.1 Basic Reset Operation

To reset the WDT, users must set the WRE in WDT_CON register and WDTRSTE in PMU_RSER register to '1'. At the moment when the down counter reaches '0' (The moment the value of the down counter changes from '1' to '0'), the WDT generates the reset signal to reset microcontroller.

8.3.6 Debug Mode

If the WDH in 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 WDH in 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.

8.3.7 Setting Example

<Example 1> WDT (Watchdog Timer) periodic interrupt mode - PCLK 74 MHz, 1 ms Period

WDT_CON<WDH> = “0” WDT_CON<WOF> = “0” WDT_CON<WIE> = “0” WDT_CON<WRE> = “0” WDT_CON<WEN> = “0” WDT_CON<WEC> = “0” WDT_CON<WPRS[2:0]> = “000”	: Initializing Control Register of Watchdog Timer
WDT_LR<WDTLR[31:0]> = “00000000 00000000 00000000 00000000”	: Initializing Load Register of Watchdog Timer
WDT_CON<WDH> = “1” WDT_CON<WIE> = “1” WDT_CON<WRE> = “1” WDT_CON<WEC> = “0” WDT_CON<WPRS[2:0]>=”000” WDT_LR<WDTLR[31:0]> = ”00000000_00000001_00100001_00010000”	: Activation Debug of Watchdog Timer : Activation Interrupt of Watchdog Timer : Deactivation Reset of Watchdog Timer : Set clock source as PCLK(74 MHz) : Set Prescaler Value for Clock : when 1 tick = (1/74)us, set period = 1 ms → 1 ms / (1/74) us = 74000
WDT_CON<WEN> = ”1”	: Run Watchdog Timer
NVICICER[0]<CLRENA[31:0]> = “00000000_00000000_00000000_00001000”	: Clear Interrupt Register of NVIC Watchdog Timer
NVICICPR[0]<CLRPEND[31:0]> = “00000000_00000000_00000000_00001000”	: Clear Pending Bits of NVIC Watchdog Timer
NVICIP[3]<PRI_3[31:24]> = “11100000”	: Set Priority of NVIC Watchdog Timer
NVICISER[0]<SETPEND[31:0]> = “00000000_00000000_00000000_00001000”	: Activation Interrupt of NVIC Watchdog Timer

8.4 WDT Registers

The base address and register map of the WDT are as follows:

Table 61. Base Address of WDT

Name	Base address
WDT	0x4000_0400

Table 62. WDT Register Map

Name	Offset	Type	Description	Reset value	Reference
WDT_LR	0x0000	RW	WDT load register	0x0000_0000	8.4.1
WDT_CVR	0x0004	RO	WDT current value register	0x0000_FFFF	8.4.2
WDT_CON	0x0008	RW	WDT control register	0x0000_0047	8.4.3

8.4.1 WDT_LR: Watchdog Timer Load Register

The WDT_LR is used to update the value of the WDT_CVR register. To change the WDT_CVR's value, two conditions must be satisfied: the WEN in WDT_CON register must be set to '1', and the WDT_LR must be written.

If the WEN in WDT_CON register is set to '0', the value written to the WDT_LR will remain unrepresented in the WDT_CVR until the WEN bit is changed to '1'.

If the WDT is being used as a reset source, the WDT_LR must be written before the WDT_CVR value becomes '0' to prevent a reset.

The WDT triggers an event at the moment when the WDT_CVR 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_CVR with the WDT_LR value.

WDT_LR=0x4000_0400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTLR[31:0]																															
0x0000_0000																															
RW																															

31 0	LR[31:0]	WDT load value register If the WDTEN remains at '1', the WDT_LR register will be updated with the WDT_CNT value.

NOTE: Since the initial value of WDT_LR is zero, it must be set to the desired value before use.

8.4.2 WDT_CVR: Watchdog Current Value Register

The WDT_CVR 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 while the WEN in WDT_CON register is '1'.

Thus, when the WDT's count value reaches 0, an interrupt or reset is triggered.

To use the WDT as a reset source, both WRE in WDT_CON register and WDTRSTE in PMU_RSER register must be set to '1'.

WDT_CNT=0x4000_0404

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTCVR[31:0]																															
0x0000_FFFF																															
RO																															

31 0	WDTCVR[31:0]	WDT current count register The 32-bit down-counter counts down from the value written to WDT_LR.

8.4.3 WDT_CON: WatchDog 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 '0'.

WDT_CON=0x4000_0408

31 30 29 28 27 26 25 24	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		WDH	Reserved	WOF	WIE	WRE	Reserved	WEN	WEC	WPRS[2:0]
-		1	-	0	0	1	-	1	1	100
-		RW	-	RW	RW	RW	RW	RW	RW	RW

15	WDH	Whether to enable or disable the WDT in debug mode
	0	Enables the WDT in STOP mode debugging.
	1	Disables the WDT in STOP mode debugging.
8	WOF	WDT underflow flag (The bit is cleared when WDT_LR is written.)
	0	There is no underflow.
	1	Underflow is pending.
7	WIE	Whether to enable or disable the WDT counter underflow interrupt
	0	Disables the interrupt.
	1	Enables the interrupt.
6	WRE	Whether to enable or disable the WDT counter underflow reset
	0	Disables the WDT counter underflow reset.
	1	Enables the WDT counter underflow reset.
4	WEN	Whether to enable or disable the WDT counter
	0	Disables the WDT counter.
	1	Enables the WDT counter.
3	WEC	WDTCLKIN clock source selection
	0	PCLK
	1	External clock (PMU_PCSR)
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

8.4.4 WDT Register Map Summary

Table 63. 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	WDTLR[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_CVR	WDTCVR[31:0]																															
	Reset value	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	1
0x08	WDT_CON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDH	Res.	Res.	Res.	Res.	Res.	Res.	WOF	WIE	WRE	WEN	Res.	WEC	WPRS[2:0]		
	Reset value																	0							0	0	1	0		0	1	1	1

9. 16-bit Timer

9.1 16-bit Timer Introduction

A33G53x series has a TIMER module consisting of ten 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

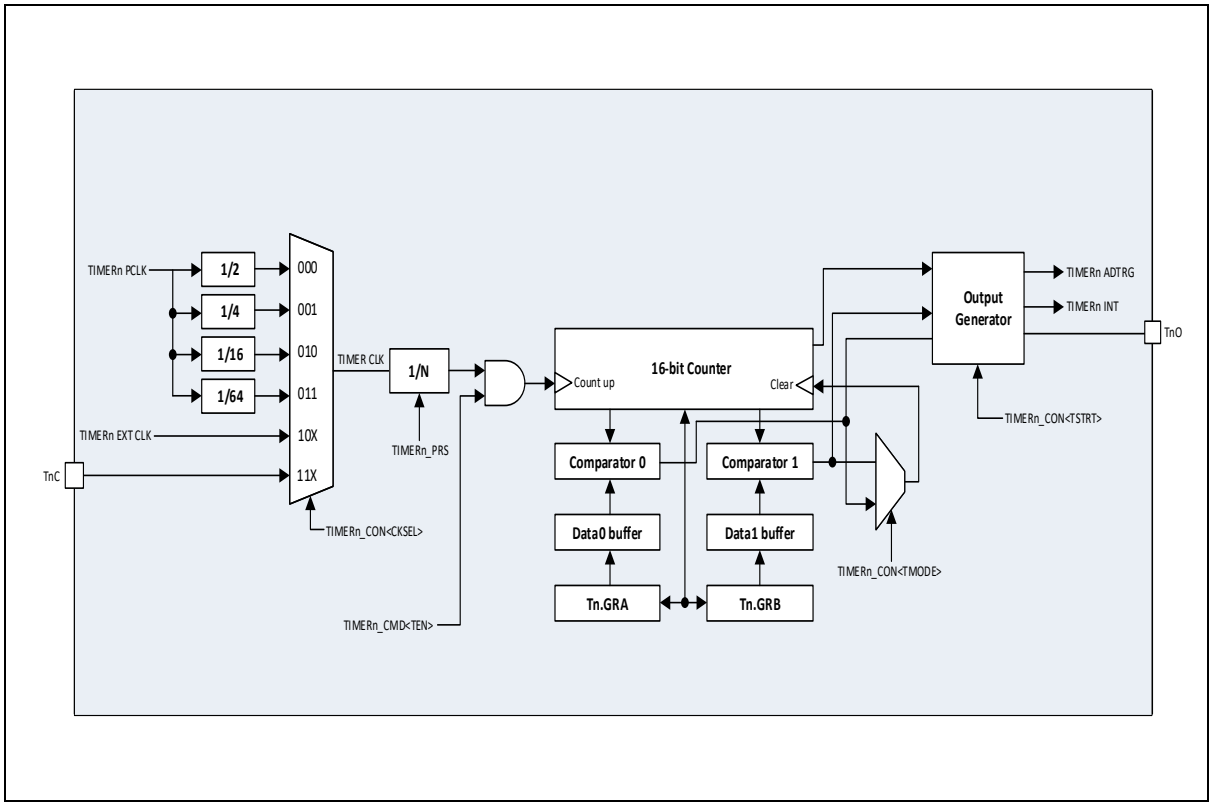
TIMERn includes the features below (where n is 0 to 9):

- 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 PMU_PCSR register: Main XTAL, IOSC16, Sub XTAL, RINGOSC
  - Timer clock source on port TnC used as input. (x = 0 to 9)
- A built-in 10-bit prescaler supporting the timer input clock

9.3 16-bit TIMER Functional Description

9.3.1 Block Diagram

Figure 57. 16-bit Timer Block Diagram (n = 0 to 9)



9.3.2 Pins and Internal Signals

Table 64 summarizes the I/O information of the TIMERn (n = 0 to 9).

Table 64. Input and Output Pins for TIMERn (n = 0 to 9)

Pin name	Type	Description	Supported packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G539MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
T0O	O	TIMER0 periodic / PWM / one-shot output	O	O	O	O
T0C	I	TIMER0 capture input signal external clock input	O	O	O	O
T1O	O	TIMER1 periodic / PWM / one-shot output	O	O	O	O
T1C	I	TIMER1 capture input signal external clock input	O	O	O	O
T2O	O	TIMER2 periodic / PWM / one-shot output	O	O	O	O
T2C	I	TIMER2 capture input signal external clock input	O	O	O	O
T3O	O	TIMER3 periodic / PWM / one-shot output	O	O	O	O
T3C	I	TIMER3 capture input signal external clock input	O	O	O	O
T4O	O	TIMER4 periodic / PWM / one-shot output	O	O	O	-
T4C	I	TIMER4 capture input signal external clock input	O	O	O	O
T5O	O	TIMER5 periodic / PWM / one-shot output	O	O	O	O
T5C	I	TIMER5 capture input signal external clock input	O	O	O	O
T6O	O	TIMER6 periodic / PWM / one-shot output	O	O	O	O
T6C	I	TIMER6 capture input signal external clock input	O	O	O	-
T7O	O	TIMER7 periodic / PWM / one-shot output	O	O	O	O
T7C	I	TIMER7 capture input signal external clock input	O	O	-	-
T8O	O	TIMER8 periodic / PWM / one-shot output	O	O	O	-
T8C	I	TIMER8 capture input signal external clock input	O	O	O	O
T9O	O	TIMER9 periodic / PWM / one-shot output	O	O	O	-
T9C	I	TIMER9 capture input signal external clock input	O	O	O	O

Table 65. 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 PMU_PCSR)
TnC	Input	Timer n input to TnC for external clock or capture input
TCS[2:0] in TIMERn_CON	Input	Counter clock source selection bit in the timer / counter control register
TIMERn_PRS	Input	Prescaler value for the counter clock
TEN in TIMERn_CMD	Input	Enable timer signal in the timer / counter control register
TSTRT in TIMERn_CON	Input	Starting output value signal in the timer / counter control register
TIE0, TIE1 in TIMERn_CON	Output	Timer n interrupt signal
TnO	Output	Timer n output signal to TnO

NOTE: 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 (PMU_PCSR register)
- Input to pin TnO (n = 0 to 9) for timer clock source

Users can select a clock source by configuring the TCS[2:0] in TIMERn_CON (n = 0 to 9).

9.3.3.1 Peripheral System Clock (PCLK)

Users can select a timer input clock source using 4 PCLK divide levels (PCLK / 2, PCLK / 4, PCLK / 16, PCLK / 64). If the TCS[2:0] in TIMERn_CON is set to a number between 0 and 3, the timer clock is provided by the PCLK divided level.

Table 66. PCLK Divide Levels and TCS[2:0] in TIMERn_CON

TCS[2:0] in TIMERn_CON	Counter clock source selection
000	PCLK / 2
001	PCLK / 4
010	PCLK / 16
011	PCLK / 64

9.3.3.2 Timer External Clock (PMU_PCSR Register)

Users can select a timer input clock source by configuring the PMU_PCSR register. The selectable clock sources include Main XTAL, IOSC16, Sub XTAL and RingOSC. If the TCS[2:0] in TIMERn_CON is set to ‘10X’, the timer clock is provided by the PMU_PCSR register.

Table 67. PMU_PCSR Register and TCS[2:0] in TIMERn_CON

TCS[2:0] in TIMERn_CON	Counter clock source selection
10X	EXT_CLK (PMU_PCSR register)

9.3.3.3 Input to Pin TnC for Timer Clock Source

Users can select a timer input clock source using input on the port TnC (n = 0 to 9).

If the TCS[2:0] in TIMERn_CON is set to ‘11X’, the timer clock is provided by input on the port TnC.

Table 68. TnC and CKSEL[2:0] in TIMERn_CR1

TCS[2:0] in TIMERn_CON	Counter clock source selection
11X	Input to pin TnC (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 is incremented 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 purpose register A : TIMERn_GRA (n = 0 to 9)
- Timer general purpose register B : TIMERn_GRB (n = 0 to 9)
- Peripheral clock source register : PMU_PCSR

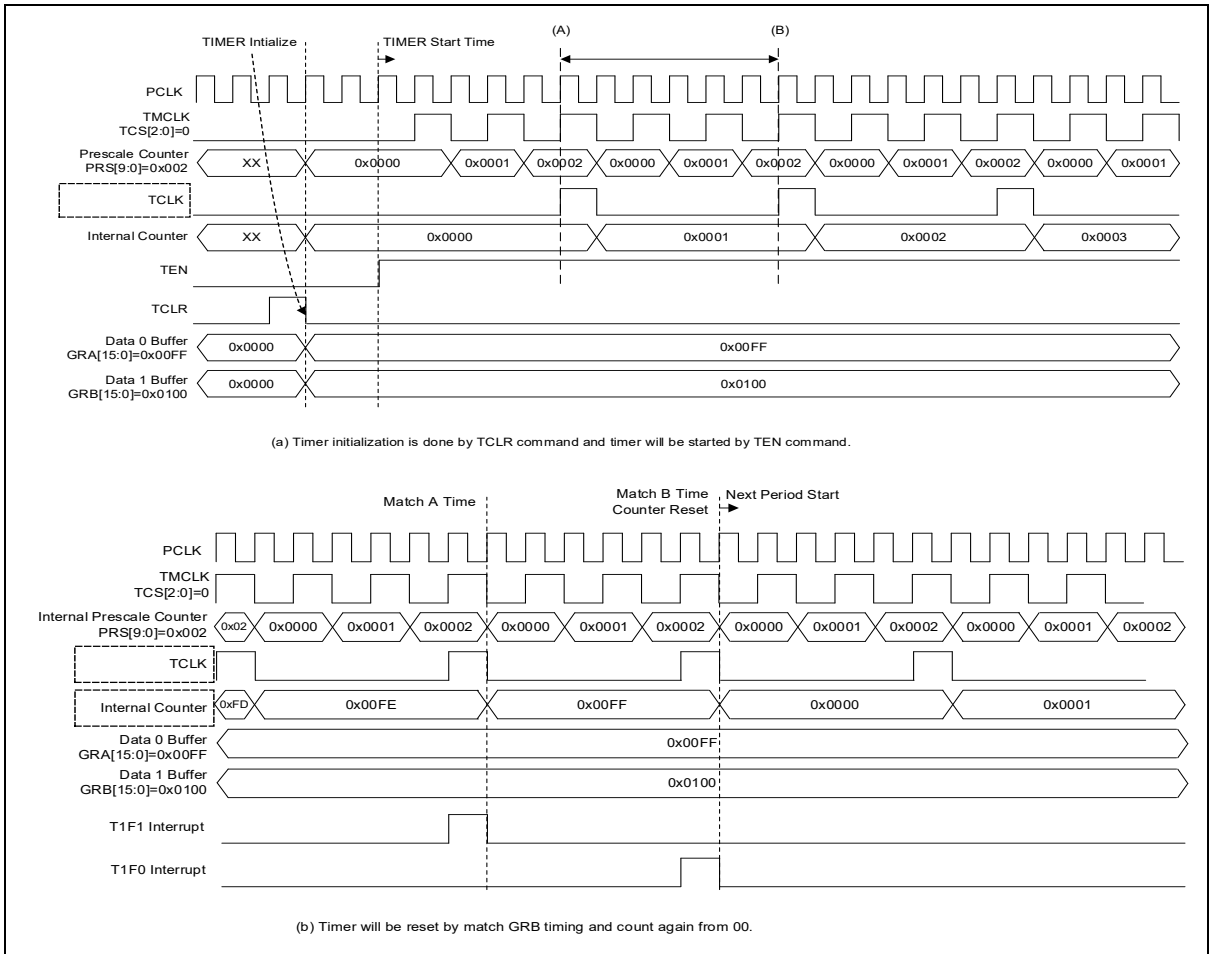
9.3.4.1 Prescaler Description

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_CON register.

The TMCLK shown in Figure 58 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 in TIMERn_CMD before setting the TEN in TIMERn_CMD.

Figure 58. Counter Timing Diagram with Basic Start and Match Operations



The timer count period can be obtained by the following formula:

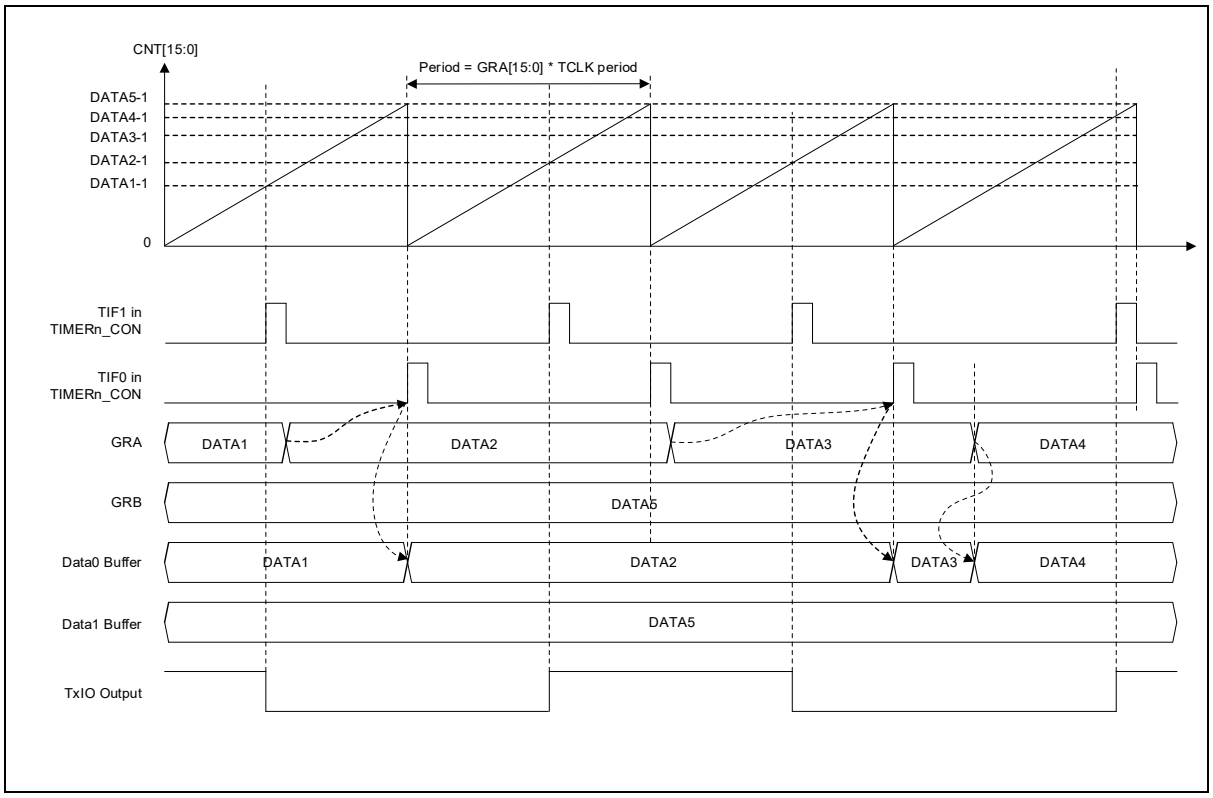
- Period = TMCLK Period × TIMERn_GRA Value
- Match A Interrupt Timing = TMCLK Period × TIMERn_GRA Value

It is recommended to write the value first in the TCLR of TIMERn_CMD before changing the TEN bit value of TIMERn_CMD when changing timer setting or restarting with new setting value

9.3.5 Periodic Mode

Figure 59 shows a timing diagram for the normal periodic mode operation. The TIMERn_GRA value determines the timer period. The TIMERn_GRA value defines the period of the timer, but the TIMERn_GRB value does not affect the period of the timer.

Figure 59. 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 TIMERn_GRA value is '0', it means that the timer period is '0'. In this case, the TIMERn_CMD <TEN> bit is set to '1' to execute the timer, but the timer does not work.

When a load condition occurs, the TIMERn_GRA and TIMERn_GRB values are loaded into the internal comparison data 0 buffer. In periodic mode, the TCLR behavior of the TIMERn_CMD register is loaded into the data buffer and the next GRA match event is loaded into the data buffer.

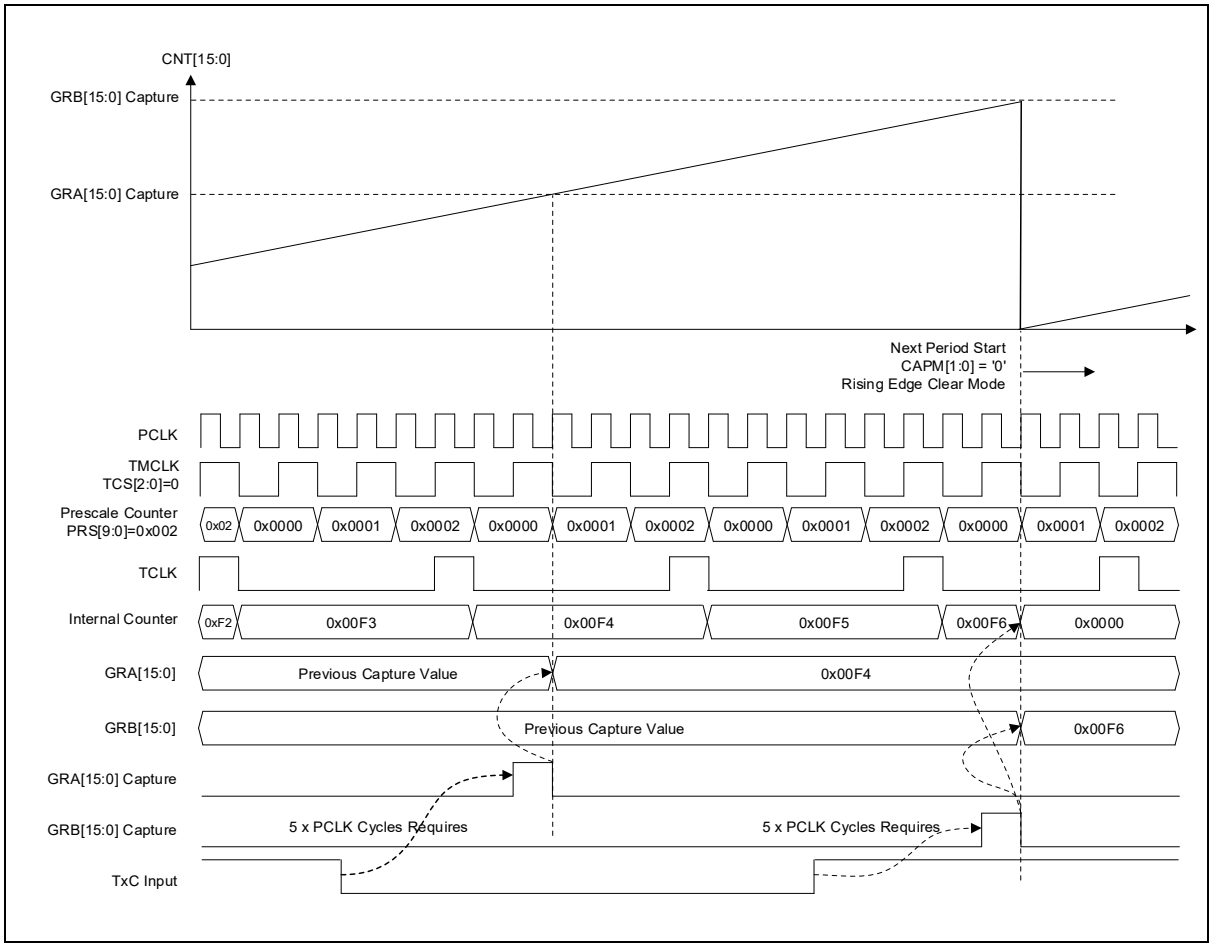
9.3.5.1 Counter Mode Register Setting Sequence

1. In order to enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register (PMU_PER1) and the peripheral clock control register (PMU_PCCR).
2. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
3. To set the timer mode, enter the corresponding mode set value in the TMODE[1:0] in TIMERn_CON.
4. To set the timer clock source, enter the corresponding set value in the TCS[2:0] in TIMERn_CON.
5. 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.
6. Enter the set value in the TIMERn_GRA and TIMERn_GRB registers for period timer operation. TIMERn_GRB value does not affect the period of the timer.
7. Write ‘1’ to the corresponding bit of the TIMERn_CON register to clear the pending interrupt.
8. Write ‘1’ to the TIE0, TIE1 in TIMERn_CON to set the timer match interrupt.
9. Write ‘1’ to the TCLR in TIMERn_CMD to apply current timer setting.
10. Write ‘1’ to the TEN in TIMERn_CMD to start the timer operation.

9.3.6 Capture Mode

Figure 60 shows a timing diagram for the capture mode operation. The TnC input signal is used for capturing the pulse. Rising and falling edges can capture the counter value in each capture condition.

Figure 60. Timing Diagram of Capture Mode Operation



The actual capture point is five PCLK clock cycles after the rising or falling edge of the TnC input signal. The internal counter can be cleared in multiple modes. The CAPM[0] in TIMERn_CON controls counter clearing in capture mode. The supported modes include rising-edge clear mode, falling-edge clear mode.

Example timing diagram in Figure 60 describes rising-edge clear mode. On the falling edge of the input signal on the TnC, the TIMERn_GRA register captures the CNT value; on the rising edge, the TIMERn_GRB register captures the CNT value.

To use the capture mode, set the GPIO port to function pin mode (TnC, n = 0 to 9) and then set logic input.

Table 69. Operation Status by Capture Mode

CAPM[1:0] in TIMERn_CON	Clear mode	TIMERn_GRA	TIMERn_GRB
0	Rising edge clear	Falling edge capture	Rising edge capture
1	Falling edge clear	Rising edge capture	Falling edge capture

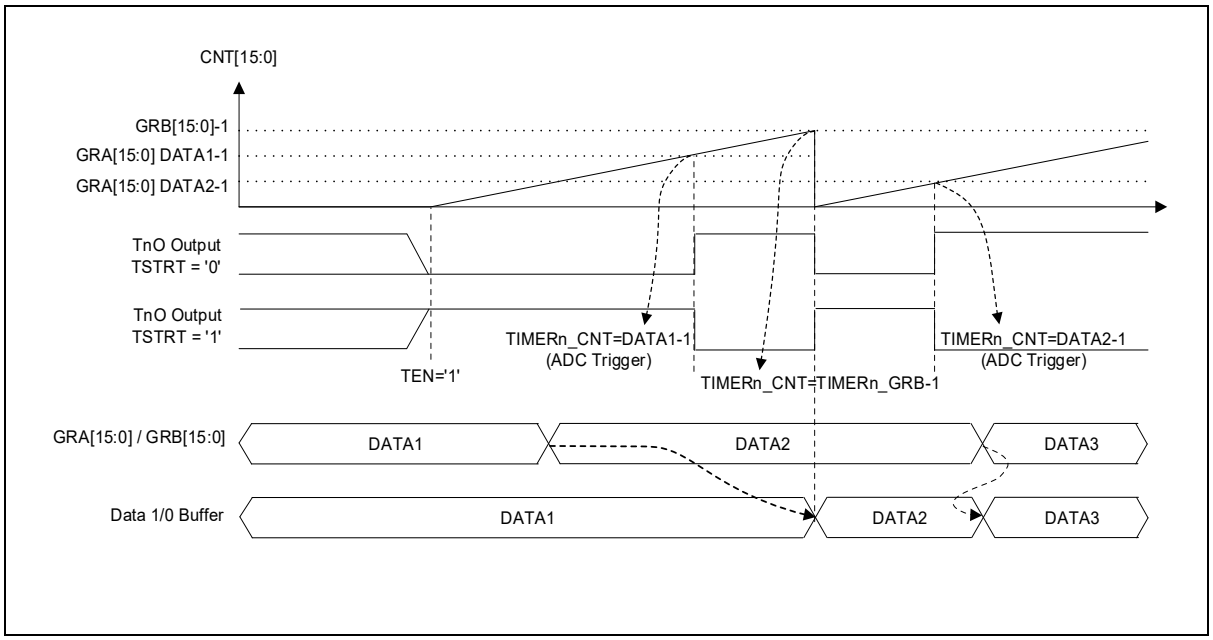
9.3.6.1 Capture Mode Register Ssetting Sequence

1. In order to enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register (PMU_PER1) and the peripheral clock control register (PMU_PCCR).
2. Initialize the TIMERn_GRA and TIMERn_GRB registers to 0.
3. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the TMODE[1:0] in TIMERn_CON.
5. To set the timer clock source, enter the corresponding set value in the TCS[2:0] in TIMERn_CON.
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. For the clear selection in capture mode, enter the corresponding set value in the CAPM[0] in TIMERn_CON.
8. Write ‘1’ to the TCLR in TIMERn_CMD to apply current timer setting.
9. Write ‘1’ to the TEN in TIMERn_CMD to start the timer operation.

9.3.7 PWM Mode

Figure 61 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 61. 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 0, the timer cannot be started even if the TEN in TIMERn_CMD is set to ‘1’ because the period is 0.

When a load condition occurs, the TIMERn_GRA and TIMERn_GRB values are loaded into the internal compare data 0 buffer and data 1 buffer. In one-shot mode, the TCLR write behavior of the TIMERn_CMD register is loaded into the data buffer and the next GRB match event is loaded into the data buffer.

The TnO 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 TSTRT in TIMERn_CON.

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 the TIMERn_GRB value.

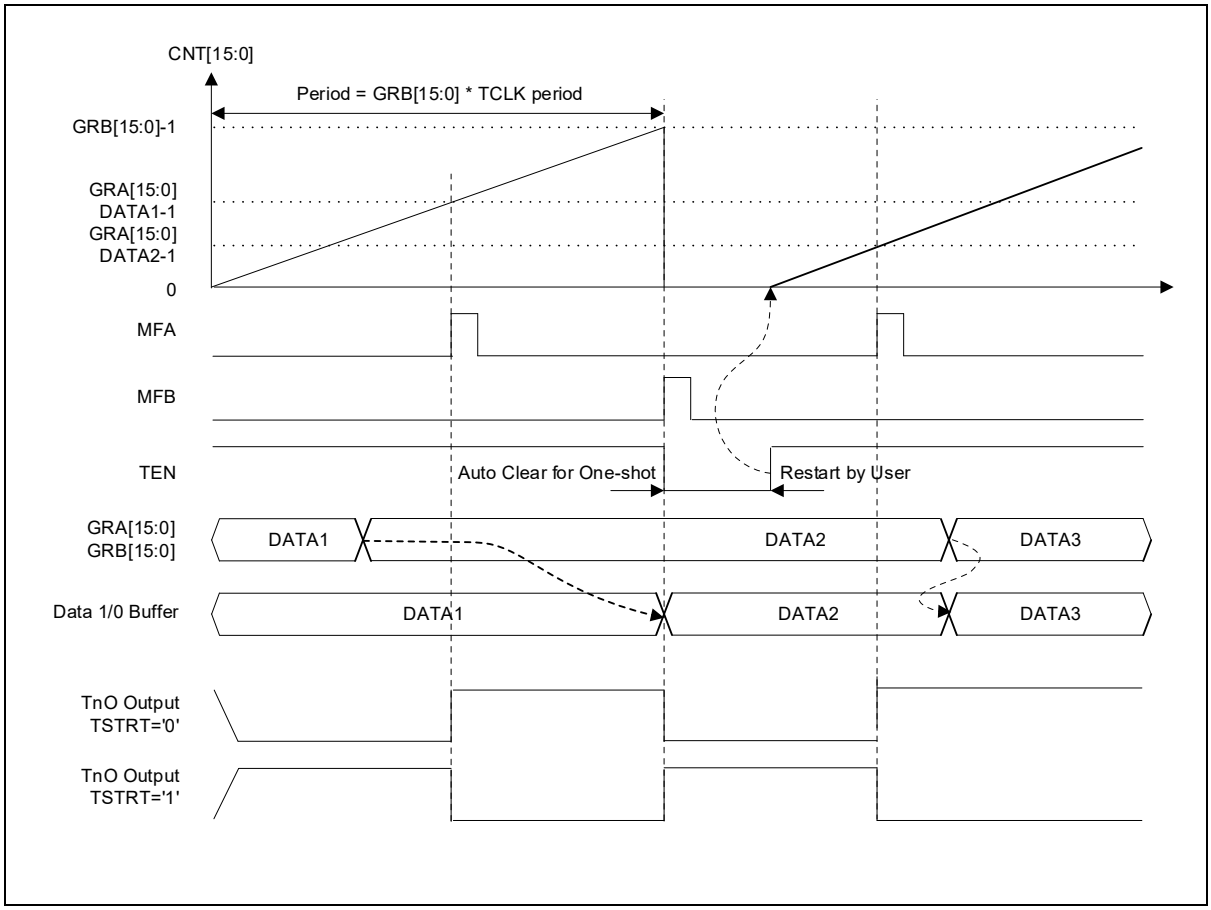
9.3.7.2 PWM Mode Register Setting Sequence

1. In order to enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register (PMU_PER1) and the peripheral clock control register (PMU_PCCR).
2. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
3. To set the timer mode, enter the corresponding mode set value in the TMODE[1:0] in TIMERn_CON.
4. To set the timer clock source, enter the corresponding set value in the TCS[2:0] in TIMERn_CON.
5. 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.
6. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
7. Write ‘1’ to the TCLR in TIMERn_CMD to apply current timer setting.
8. Write ‘1’ to the TEN in TIMERn_CMD to start the timer operation.

9.3.8 One-Shot Mode

Figure 62 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 62. 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 0, the timer cannot be started even if the TEN in TIMERn_CMD is set to ‘1’ because the period is 0.

When a load condition occurs, the TIMERn_GRA and TIMERn_GRB values are loaded into the internal compare data 0 buffer and data 1 buffer. In one-shot mode, the TCLR write behavior of the TIMERn_CMD register is loaded into the data buffer and the next GRB match event is loaded into the data buffer.

The TnO 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 for setting the TIMERn_GRA register are shown below:

- The TIMERn_GRA value must be greater than ‘0’ and less than the TIMERn_GRB value.

9.3.8.2 One-Shot Mode Register Setting Sequence

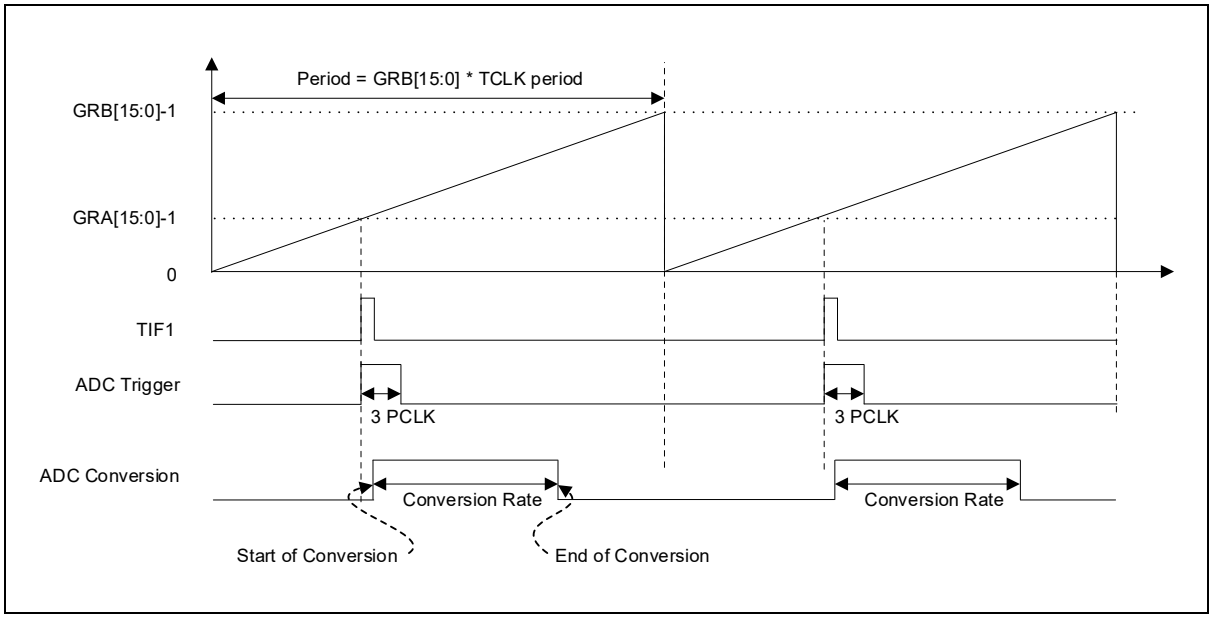
1. In order to enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register (PMU_PER1) and the peripheral clock control register (PMU_PCCR).
2. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
3. To set the timer mode, enter the corresponding mode set value in the TMODE[1:0] in TIMERn_CON.
4. To set the timer clock source, enter the corresponding set value in the TCS[2:0] in TIMERn_CON.
5. 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.
6. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
7. Write ‘1’ to the TCLR in TIMERn_CMD to apply current timer setting.
8. Write ‘1’ to the TEN in TIMERn_CMD to start the timer operation.

9.3.9 ADC Trigger

A33G53x's 16-bit timer can generate the ADC start trigger signal and can be the trigger source for a ADC block. The ADC trigger source control of timer can be set in ADC control register (ADC_CR)

Figure 63 is shown the timing of 16-bit timer's ADC trigger mode. To use a timer as the ADC trigger source, the ADC conversion rate must be less than the timer period. Otherwise, an overrun situation will occur. The ADC ACK is not needed because the trigger signal is automatically cleared after three PCLK clock pulses.

Figure 63. Timing Diagram of ADC Triggering



9.3.10 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 TSTRT in TIMERn_CON register. Setting the TSTRT allows users to select the initial output value and select whether to set the start output value to Low or High.

9.3.11 Debug Mode

When the microcontroller enters debug mode, the timer counter continues to work. Therefore, the outputs (TnO, n = 0 to 9) are enabled, although the core halted, when the TEN in TIMERn_CMD is set. When the TEN in TIMERn_CMD is cleared, the outputs (TnO, n = 0 to 9) are disabled.

9.3.12 Interrupts

The TIMERn can generate interrupts, as shown in Table 70.

Table 70. Interrupt Requests

Interrupt event	Event flag	Enable control bit	Interrupt clear method
Overflow interrupt	TOVF in TIMERn_CON	TOVE in TIMERn_CON	Writing a '1' to TOVF in TIMERn_CON
GRB match interrupt	TIF0 in TIMERn_CON	TIE0 in TIMERn_CON	Writing a '1' to TIE0 in TIMERn_CON
GRA match interrupt	TIF1 in TIMERn_CON	TIE1 in TIMERn_CON	Writing a '1' to TIE1 in TIMERn_CON

NOTE: The overflow flag only occurs in capture mode.

9.4 TIMER Registers

The base addresses and register map of 16-bit timers are as follows:

Table 71. Base Addresses of 16-bit Timer

Name	Base address
TIMER0	0x4000_0C00
TIMER1	0x4000_0C20
TIMER2	0x4000_0C40
TIMER3	0x4000_0C60
TIMER4	0x4000_0C80
TIMER5	0x4000_0CA0
TIMER6	0x4000_0CC0
TIMER7	0x4000_0CE0
TIMER8	0x4000_0D00
TIMER9	0x4000_0D20

Table 72. TIMER Register Map

Name	Offset	Type	Description	Reset value	Reference
TIMERn_CON	0x00	RW	Timer n Control Register	0x0000_0000	9.4.1
TIMERn_CMD	0x04	RW	Timer n Command Register	0x0000_0000	9.4.2
TIMERn_GRA	0x08	RW	Timer n General Purpose Register A	0x0000_0000	9.4.3
TIMERn_GRB	0x0C	RW	Timer n General Purpose Register B	0x0000_0000	9.4.4
TIMERn_PRS	0x10	RW	Timer n Prescaler Register	0x0000_0000	9.4.5
TIMERn_CNT	0x14	RW	Timer n Counter Register	0x0000_0000	9.4.6

NOTE: n = 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9.

9.4.1 TIMERn_CON: Timer n Control Register

It consists of a 32-bit register that controls all timer functions such as operating mode, timer clock source, and interrupts. The timer's operation mode and interrupts can be set according to the purpose of the user application.

TIMER0_CON=0x4000_0C00, TIMER1_CON=0x4000_0C20, TIMER2_CON=0x4000_0C40, TIMER3_CON=0x4000_0C60, TIMER4_CON=0x4000_0C80, TIMER5_CON=0x4000_0CA0, TIMER6_CON=0x4000_0CC0, TIMER7_CON=0x4000_0CE0, TIMER8_CON=0x4000_0D00, TIMER9_CON=0x4000_0D20

31 30 29 28 27 26 25 24	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		Reserved	TOVF	TIF1	TIF0	Reserved	TOVE	TIE1	TIE0	TSTRT	TCS[2:0]	CAPM	Reserved	TMODE[1:0]
-		-	0	0	0	-	0	0	0	0	000	00	-	00
-		-	RC	RC	RC	-	RW	RW	RW	RW	RW	RW	-	RW

13	TOVF	The flag of timer overflow interrupt	
		0	No overflow occurred
		1	Overflow occurred (Write ‘1’ to clear this flag)
13	TIF1	The flag of match 1 interrupt	
		0	No match1 interrupt occurred
		1	Match 0 interrupt occurred (Write ‘1’ to clear this flag)
12	TIF0	The flag of match 0 interrupt	
		0	No match 0 interrupt occurred
		1	Match 0 interrupt occurred (Write ‘1’ to clear this flag)
10	TOVE	Enable / disable overflow interrupt	
		0	Disable overflow interrupt
		1	Enable overflow interrupt
9	TIE1	Enable / Disable match 1 interrupt	
		0	Disable match 1 interrupt
		1	Enable match 1 interrupt
8	TIE0	Enable / Disable match 0 interrupt	
		0	Disable match 0 interrupt
		1	Enable match 0 interrupt
7	TSTRT	Setting Initial output in Periodic/PWM/One-shot mode	
		0	Initial output “L” upon counter being cleared
		1	Initial output “H” upon counter being cleared
6 4	TCS[2:0]	Select the clock source of timer	
		000	TCLK/2
		001	TCLK/4
		010	TCLK/16
		011	TCLK/64
		10X	Select the clock source by PMU_PCSR
		11X	Select external TnC input as timer clock source
3	CAPM	Select clear timing of internal counter	
		00	Rising-edge clear
		01	Falling-edge clear
1 0	TMODE[1:0]	Timer operation mode control	
		00	Periodic mode
		01	PWM mode

	10	One-shot mode
	11	Capture mode

9.4.2 TIMERn_CMD: Timer n Command Register

This register controls timer operation such as timer running or stop. The timer counter can also be initialized by writing '1' to the TCLR in TIMERn_CMD Register.

TIMER0_CMD=0x4000_0C04, TIMER1_CMD=0x4000_0C24, TIMER2_CMD=0x4000_0C44
TIMER3_CMD=0x4000_0C64, TIMER4_CMD=0x4000_0C84, TIMER5_CMD=0x4000_0CA4
TIMER6_CMD=0x4000_0CC4, TIMER7_CMD=0x4000_0CE4, TIMER8_CMD=0x4000_0D04
TIMER9_CMD=0x4000_0D24

31 30 29 28 27 26 25 24	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		TOUT	Reserved		TCLR	TEN
-		0	-		0	0
-		RO	-		W	RW

15	TOUT	The status of external output pin
0	TEN	Clearing of the timer count register
	0	No effect
	1	Initialize Timer. - Clear counter register - Set output TOUT as TSTRT value - Set internal buffer with TnGRA/B - Initialize capture signal
0	TEN	Select the operation of timer
	0	Stop the timer.
	1	Start the timer.

9.4.3 TIMERn_GRA: Timer n General Purpose Register A

TIMERn_GRA is a 32-bit register.

TIMER0_GRA=0x4000_0C08, TIMER1_GRA=0x4000_0C28, TIMER2_GRA=0x4000_0C48
TIMER3_GRA=0x4000_0C68, TIMER4_GRA=0x4000_0C88, TIMER5_GRA=0x4000_0CA8
TIMER6_GRA=0x4000_0CC8, TIMER7_GRA=0x4000_0CE8, TIMER8_GRA=0x4000_0D08
TIMER9_GRA=0x4000_0D28

31 30 29 28 27 26 25 24	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 n General Register A This register is used for different purposes depending on the operating mode.
		Period / PWM / One-shot Modes: This register is used as target count comparator. When the counter value is equal to this register, the counter will be cleared and the timer will restart in periodic mode and PWM mode, or will stop in one-shot mode. When the timer restarts from '0', the output of TOUT becomes TSTRT. If the counter reaches this register value, then TIF0 interrupt will occur. On the TIF0 interrupt or the clear operation. This register value is copied onto internal data buffer 0. $\text{Period} = \frac{\text{TCLKIN}}{\text{TnGRA}}$
		Capture mode: - In rising-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnC. - In falling-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnC.

9.4.4 TIMERn_GRB: Timer n General Purpose Register B

TIMERn_GRB is a 32-bit register.

TIMER0_GRB=0x4000_0C0C, TIMER1_GRB=0x4000_0C2C, TIMER2_GRB=0x4000_0C4C
TIMER3_GRB=0x4000_0C6C, TIMER4_GRB=0x4000_0C8C, TIMER5_GRB=0x4000_0CAC
TIMER6_GRB=0x4000_0CCC, TIMER7_GRB=0x4000_0CEC, TIMER8_GRB=0x4000_0D0C
TIMER9_GRB=0x4000_0D2C

31 30 29 28 27 26 25 24	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: Not used
		PWM / One-shot mode: This register is used as target count value. When the counter value is equal to this register, then TIF1 interrupt will occur. On the TIF1 interrupt or the clear operation. This register value is copied onto internal data buffer 1.
		Capture mode: - In rising-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnC. (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 TnC. (The opposite edge to that of TIMERn_GRA)

9.4.5 TIMERn_PRS: Timer n Prescaler Register

TIMERn_PRS 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 the TIMERn_CON register.

TIMER0_PRS=0x4000_0C10, TIMER1_PRS=0x4000_0C30, TIMER2_PRS=0x4000_0C50
TIMER3_PRS=0x4000_0C70, TIMER4_PRS=0x4000_0C90, TIMER5_PRS=0x4000_0CB0
TIMER6_PRS=0x4000_0CD0, TIMER7_PRS=0x4000_0CF0, TIMER8_PRS=0x4000_0D10
TIMER9_PRS=0x4000_0D30

31 30 29 28 27 26 25 24	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]
-			0000000000
-			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 TCS[2:0] in TIMERn_CON)

9.4.6 TIMERn_CNT: Timer / Counter n Ccount Register

TIMERn_CNT is a 32-bit register. The count is incremented based on the specified input clock. This register can be both read and written to.

TIMER0_CNT=0x4000_0C14, TIMER1_CNT=0x4000_0C34, TIMER2_CNT=0x4000_0C54
TIMER3_CNT=0x4000_0C74, TIMER4_CNT=0x4000_0C94, TIMER5_CNT=0x4000_0CB4
TIMER6_CNT=0x4000_0CD4, TIMER7_CNT=0x4000_0CF4, TIMER8_CNT=0x4000_0D14
TIMER9_CNT=0x4000_0D34

31 30 29 28 27 26 25 24	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 TIMER Register Map Summary

Table 73. 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_CON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOVF	TIF1	TIF0	Res.	TOVE	TIE1	TIE0	TSTRT	TCS[2:0]			CAPM	Res.	TMODE[1:0]	
	Reset value																		0	0	0		0	0	0	0	0	0	0	0		0	0
0x04	TIMERn_CMD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCLR	TEN
	Reset value																	0														0	0
0x08	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
0x0C	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
0x10	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
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

10. Free-Run Timer (FRT)

10.1 FRT Introduction

The A33G53x has a built-in 32-bit up-count timer for FRT (Free-run Timer). This FRT can perform overflow interrupts or match interrupts according to the setting period of user application.

10.2 FRT Main Features

The FRT of the A33G53x series has the main features shown below:

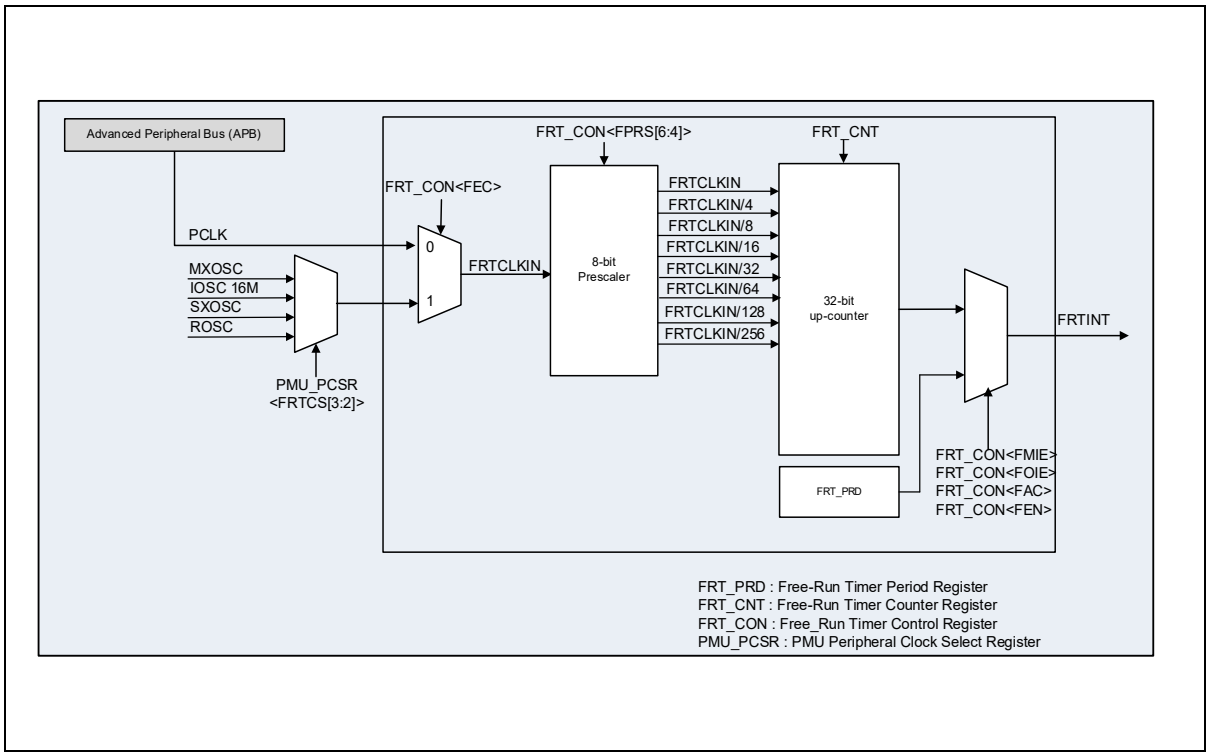
- 32-bit up count timer’s two operations
  - Free-run Timer Mode operation
  - Periodic Timer Mode operation based on the setup period
- FRT overflow interrupt and match interrupt functions
- Selectable FRT input clocks
  - PCLK
  - Input clock source can be selected using the PMU_PCSR register: Main XTAL, IOSC16, SXOSC, RINGOSC
- 8-steps FRT prescaler support for FRT (Free-run Timer) input clock

10.3 FRT Functional Description

10.3.1 FRT Bblock Diagram

Figure 64 shows a block diagram of the FRT.

Figure 64. FRT Block Diagram



10.3.2 FRT Clock Selection

The operation mode of A33G53x FRT (Free-run Timer) is set by FAC in FRT_CON.

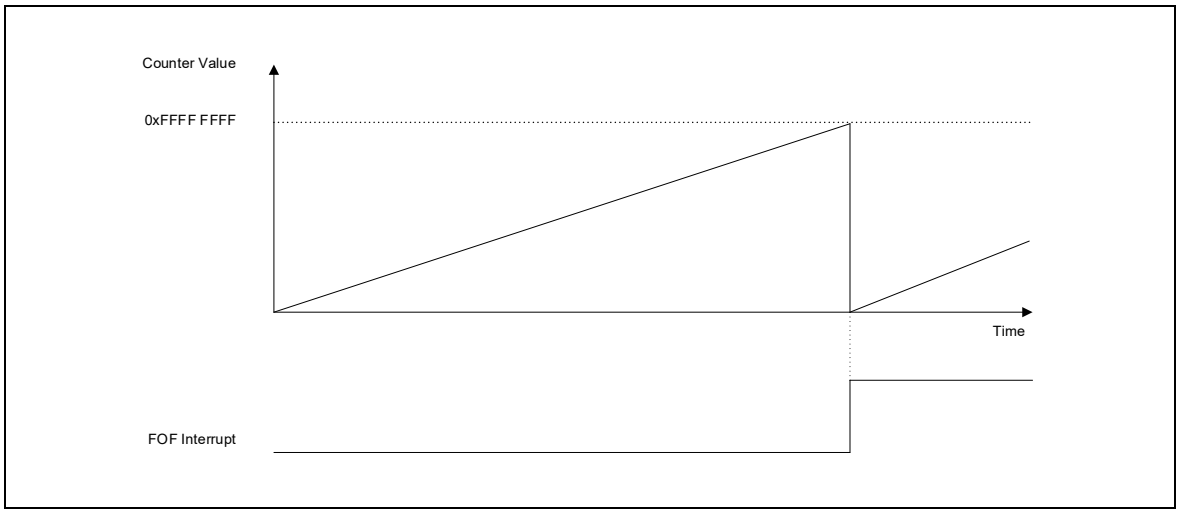
When FAC in FRT_CON register is set to zero, FRT runs in free-runs mode and when FAC in FRT_CON value is set to ‘1’, FRT (Free-run Timer) runs in periodic mode.

When FRT runs in free-run timer mode, FOIE in FRT_CON determines whether overflow interrupt occurs. When FRT runs in periodic timer, FMIE in FRT_CON determines whether match interrupt occurs.

10.3.3 FRT Run Mode

When FRT operates free-run mode, FRT_CNT value increases from 0 to 0xFFFF_FFFF regardless of FRT_PRD value. If FRT_CNT value reaches to 0xFFFF_FFFF, the next value is set to zero. In this case, FRT_CON<FOIE> bit was set to '1', an overflow interrupt occurs when FRT_CNT value changes from 0xFFFF_FFFF to zero.

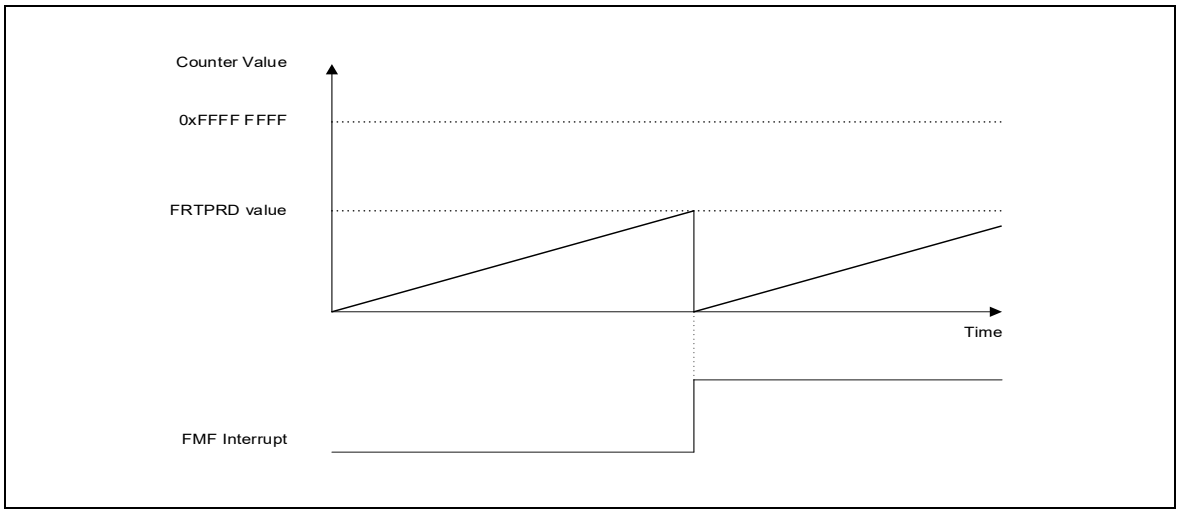
Figure 65. A Operating Timing in Free run mode



10.3.3.1 Periodic Timer Mode

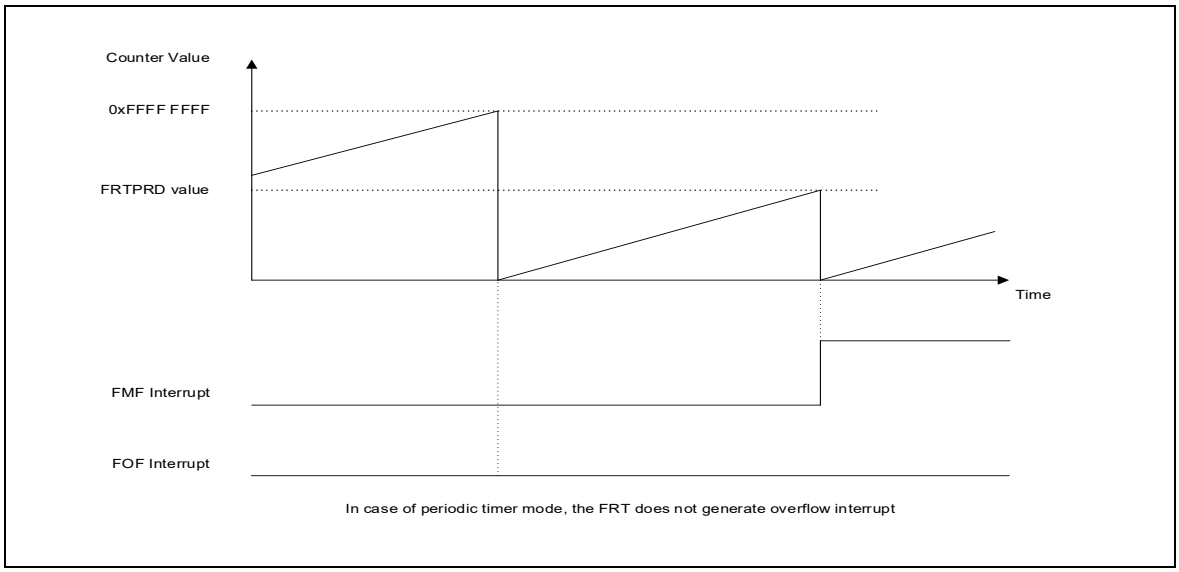
When the FRT is executed in the periodic timer mode, the FRT_CNT value is initialized to zero and counted up again when the FRT_CNT value becomes equal to the FRT_PRD value. At this moment, If the FRT_CON<FMIE> bit was set to '1', a match interrupt occurs.

Figure 66. A Operating Example in Periodic mode - 1



If the FRT_CNT value is set to a value larger than FRT_PRD, FRT_CNT initializes 0xFFFF_FFFF, and the next operation is the same as the periodic timer mode. In this case, Although FRT_CON<FOIE> bit was set to '1', match interrupt does not occur when FRT_CNT value changes from 0xFFFF_FFFF to zero. In this case, if the FRT_CNT value changes from 0xFFFF_FFFF to zero, even though the FRT_CON<FOIE> bit is set to '1', no match interrupt occurs.

Figure 67. A Operating Example in Periodic mode - 2



10.3.3.2 Periodic Timer Mode

FRT interval timer mode, the timer is assumed to calculate the timing. Clock source PCLK 16 MHz, FRTPRD value is 0x4000 (16,384), FPRS = 000 is assumed.

- 1-clock period : 1/16 MHz = 0.0625 μsec
- Interrupt interval : 0.0625 μsec x 16384 = 1.024 msec

The table below shows the interval of interrupt depending on the FPRS values.

Table 74. Interrupt Intervals according to the FPRS Value(PCLK=16 MHz, FRTPRD=0x4000)

FRT_CON<FPRS[6:4]>	Clock Source	Interval of Interrupt
000	FRTCLKIN	1.024 msec
001	FRTCLKIN/4	4.096 msec
010	FRTCLKIN/8	8.192 msec
011	FRTCLKIN/16	16.384 msec
100	FRTCLKIN/32	32.768 msec
101	FRTCLKIN/64	65.536 msec
110	FRTCLKIN/128	131.072 msec
111	FRTCLKIN/256	262.144 msec

10.3.4 Setting Example

<Example 1> Setting the free-run mode of FRT (8 MHz XTAL, Period = 1 s)

```
PMU_PCSR<FRTCS[3:2]> = “00”                  : Selects FRT Clock source as an 8 MHz external crystal.
FRT_CON<FEC> = “1”                           : Uses FRT clock source set by PMU_PCSR (FRTCLKIN)
FRT_CON<FPRS[6:4]> = “010”                   : Uses FRT Clock(FRTCLKIN) is divided 8.
FRT_CON<FAC> = “0”                           : Sest FRT operation mode
FRT_CNT<FRTCNT[31:0]>                        : Initializes FRT Counter Value to zero
=”00000000_00000000_00000000_00000000”
FRT_PRD<FRTPDR[31:0]>                        : Set FRT operating clock (FRTCLKIN/8 = 1 MHz)
=”00000000_00001111_01000010_00111111”        1 period : 1 s -> 1 s / 1 us = 1,000,000
FRT_CON<FMIE> = “1”                          : Enables FRT match interrupt event
FRT_CON<FOIE> = “1”                          : Enables FRT overflow interrupt event

NVICICER[0]<CLRENA[31:0]>                    : Clears NVIC FRT interrupt bit
= “00000000_00000000_00000000_00010000”
NVICICPR[0]<CLRPEND[31:0]>                   : Clears NVIC FRT pending bit
= “00000000_00000000_00000000_00010000”
NVICIP[4]<PRI_4[7:0]> = “11100000”           : Sest priority of NVIC FRT
NVICISER[0]<SETPEND[31:0]>                   : Enables interrupt of NVIC FRT
= “00000000_00000000_00000000_00010000”


FRT_CON<FEN> = “1”                           : Starts the operation of FRT counter
```

<Example 2> Setting the periodic timer mode of FRT (8 MHz XTAL, Period = 1 s)

```
PF_MR<PF11[23:22]> = “10”                    : Set PF11 pin MUX to FRT match flag output.
PF_CR<P11[23:22]> = “00”                     : Set PF11 pin direction to push-pull output
PF_PCR<P11> = “0”                            : disable the pull-up/pull-down resistor of PF11 pin

PMU_PCSR<FRTCS[3:2]> = “00                   : Use 8 MHz MXOSC as FRT clock source
FRT_CON<FEC> = “1”                           : Use the clock defined by PMU_PCSR as the clock source
FRT_CON<FPRS[6:4]> = “010”                   of FRT.
FRT_CON<FAC> = “1”                           : Set prescaler value of FRT input clock
                                             : Use the FRT as Periodic timer mode


FRT_PRD<FRTPDR[31:0]>                        : 1 tick : (1/1) us
=”00000000_00001111_01000010_00111111”       period: 1 s -> 1 s / 1 us = 1,000,000

FRT_CON<FMIE> = “1”                          : Enable FRT match interrupt event

NVICICER[0]<CLRENA[31:0]>                    : Set NVIC FRT interrupt.
= “00000000_00000000_00000000_00010000”      - IRQ number: Free-run Timer IRQ(4)
NVICICPR[0]<CLRPEND[31:0]>                   - Priority: 1
= “00000000_00000000_00000000_00010000”      - Lower priority: 3
NVICIP[4]<PRI_4[7:0]> = “11100000”           - Enable interrupt of NVIC FRT
NVICISER[0]<SETPEND[31:0]>
= “00000000_00000000_00000000_00010000”


FRT_CON<FEN> = “1”                           : Starts the operation of FRT counter
FRT_CNT<FRTCNT[31:0]>                        : Initializes FRT Counter Value to zero
=”00000000_00000000_00000000_00000000”
```

10.4 FRT Registers

The base address and register map of the FRT module are as follows:

Table 75. Base Address of FRT Interface

Name	Base address
FRT	0x4000_0500

Table 76. FRT Register Map

Name	Offset	Type	Description	Reset value	Reference
FRT_PRD	0x00	RW	FRT Period Register	0x00000000	10.4.1
FRT_CNT	0x04	RW	FRT Counter Register	0x00000000	10.4.2
FRT_CON	0x08	RW	FRT Control Register	0x00000000	10.4.3

10.4.1 FRT_PRD: FRT Period Register

This register is for setting the period register when FRT(Free-run Timer) operates in periodic timer mode.

In the free-run mode of the A33G53x, when the value of the FRT_CNT register increases and becomes equal to the FRT_PRD value, a match interrupt occurs when the match interrupt bit is enabled. And match interrupt was occurred, FRTCNT will be '0' and the FRT will restart. And match interrupt was occurred, FRT_CNT will be zero and the FRT will be restart.

The value calculated by subtracting 1 from the FRT period value must be entered to FRT_PRD.

FRT_PRD=0x4000_0500

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
FRTPRD[31:0]			
0x0000_0000			
RW			

31 0	FRTPRD	Set the period value of FRT (Free-Run Timer)

10.4.2 FRT_CNT: FRT n Counter Register

FRT_CNT represents the current value of the timer as a free-running timer and provides a 32-bit register that can access with write or read. This FRT_CNT register is an up-count timer in which the count value increases according to the FRT clock.

If FEN bit in the FRT_CON register is set to '1', FRT_CNT register can write or read access. When FEN = '0', the count value is not reflected in FRT_CNT but is reflected immediately in FRT_CNT when FEN = '1'.

FRT_CNT=0x4000_0504

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
FRTCNT[31:0]			
0x0000_0000			
RW			

31 0	FRTCNT[31:0]	Current count value of FRT (Free-run Timer) When FEN in FRT_CON register is '0', the counter value is not reflected. When FEN in FRT_CON register is '1', the counter value is reflected.

10.4.3 FRT_CON: FRT Control Register

This register controls overall operations of FRT. FRT_CON register offers match and overflow interrupt of free-run timer, and the function to set the operating mode and speed of free-run timer. If match interrupt is used, FRTO_FMF

When using FRT match interrupts, you can set PF11 pin to FRTO_FMF by setting pin mux. this function enables match interrupts by activating FMIE in FRT_CON register and then executes FRT. Next, when the FRT is counted and a match interrupt event occurs, the output signal of FRTO_FMF pin outputs 'H' when FMF in FRT_CON register = ‘1’. On the contrary, if FMF in FRT_CON register = ‘0’ is written to clear the match interrupt event, the output signal of the FRTO_FMF pin becomes 'L'.

FRT_CON=0x4000_0508

31 30 29 28 27 26 25 24	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			FMF	FOF	FEC	FPRS[2:0]	FMIE	FOIE	FAC	FEN
-			0	0	0	000	0	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW

9	FMF	FRT counter match interrupt flag	
		0	The match interrupt event has not occurred.
		1	The match interrupt event has occurred. (Writing a ‘0’ to the bit clears this flag)
8	FOF	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)
7	FEC	FRT clock source selection	
		0	Select PCLK as a clock source
		1	Select a clock source with PMUPCSR
6 4	FPRS	FRTCLKIN prescaler	
		000	FRTCLKIN
		001	FRTCLKIN / 4
		010	FRTCLKIN / 8
		011	FRTCLKIN / 16
		100	FRTCLKIN / 32
		101	FRTCLKIN / 64
		110	FRTCLKIN / 128
		111	FRTCLKIN / 256
3	FMIE	FRT counter match interrupt flag	
		0	Disable match interrupt
		1	Enable match interrupt
2	FOIE	FRT counter overflow interrupt flag	
		0	Disable overflow interrupt
		1	Enable overflow interrupt
1	FAC	FRT mode	
		0	Free run mode
		1	Periodic mode
0	FEN	FRT counter operation	

0	Stop FRT counter
1	Enable FRT counter

10.4.4 FRT Register Map

Table 77. 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	FRT_PRD	FRTPRD[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	FRT_CNT	FRTCNT[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	FRT_CON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FMF	FOF	FEC	FPRS[2:0]			FMIE	FOIE	FAC	FEN
	Reset value																							0	0	0	0	0	0	0	0	0	0

11. Universal Asynchronous Receiver / Transmitter (UART)

11.1 Introduction

The A33G53x has four channels of UART (Universal Asynchronous Receiver / Transmitter) compatible with 16C550/16C450. Among these, UART0 and UART1 channels are 16550 with FIFO type, and UART2 and UART3 channels are 16450 with double buffer type. All UART channels operate in the same data mode (no FIFO mode) as 16450 when they are initialized. After initialization, FIFO mode setting is possible only for UART0 and UART1. In the FIFO mode, up to 16 bytes of data can be stored in the transmit/receive FIFO.

The built-in UART (Universal Asynchronous Receiver / Transmitter) can read out the data of the set configuration and the received data or the current UART status. UART status information can be checked not only for the type and conditions of transmission and reception by UART (Universal Asynchronous Receiver / Transmitter) but also for errors (parity, overrun, framing, break interrupt) that occur when data is received.

Each Universal Asynchronous Receiver / Transmitter (UART) has a programmable baud rate generator to divide the prescaled clock into values from 1 to 65,535. This divided clock is again divided into 16 clocks to create a clock that drives the UART's internal transmit / receive blocks.

Communication with the UART can also be controlled by an interrupt using a user-programmable interrupt function. However, the general purpose 16450/16550 has a modem control signal and associated registers, but not the A33G53x.

11.2 Main Features

The UART of A33G53x series features the followings:

- 16550/16450 compatible asynchronous serial communication port 4 channels.
  - 16550 compatible device with FIFO type 2 channels: UART0, UART1
  - 16450 compatible devices in double buffer type 2 channels: UART2, UART3
- 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 detection
  - 1, 1.5, or 2 stop bit generation and detection
- A 16-bit baud-rate generator and an 8-bit fractional compensator are included.
- Independently configurable transmit, receive, line status, interrupt
- Single or multi-sampling for start and data bits
- Built-in decimal point divider to improve baud rate accuracy.
- Support boot program using UART0 channel when entering boot mode
- 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
- Loop back control
- A priority-based interrupt system

11.2.1 UART Implementation

Table 78 describes the UART implementation on A33G53x series.

Table 78. UART Features

UART modes / features	UART
Continuous communication using DMA	Not supported
UART data length	5, 6, 7 and 8 bits

11.3 UART Functional Description

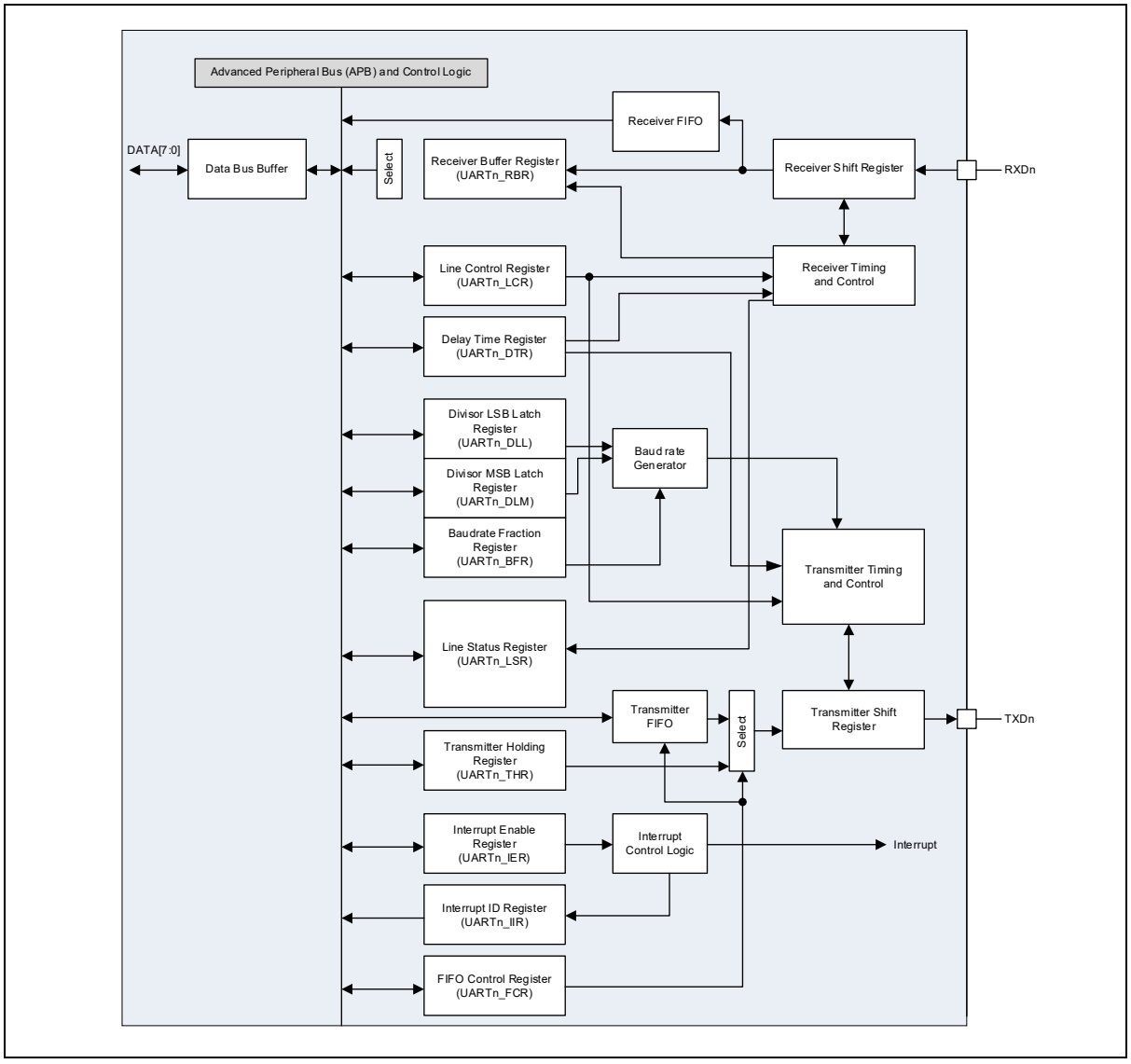
The A33G53x has a built-in UART that supports single-sampling and multisampling modes when the start bit or data bit is detected by the SMS in UARTn_DTR register and DMS in UARTn_DTR register bits. The user can select the sampling method according to the operating environment. This sampling method supports noise-resistant UART communication environment.

In addition, to reduce the communication error caused by line noise introduced into the received data of the UART, it is necessary to activate the debounce function of the PMC (Port Map Control).

11.3.1 UART Block Diagram

Figure 68 shows a block diagram of the UART.

Figure 68. UART Block Diagram



11.3.2 UART Pins and Signals

The external pins assigned for UART are described in Table 79.

Table 79. Pin Assignment of UART: External Pins

Pin Name	Type	Description	Supported Packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G539MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
TXD0	O	UART channel 0 transmit output	O	O	O	O
RXD0	I	UART channel 0 receive input	O	O	O	O
TXD1	O	UART channel 1 transmit output	O	O	O	O
RXD1	I	UART channel 1 receive input	O	O	O	O
TXD2	O	UART channel 2 transmit output	O	O	O	O
RXD2	I	UART channel 2 receive input	O	O	O	O
TXD3	O	UART channel 3 transmit output	O	O	O	O
RXD3	I	UART channel 3 receive input	O	O	O	O

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 UARTn_LCR register as described below:

- For 5-bit character length: DLEN[1:0] = ‘00’
- For 6-bit character length: DLEN[1:0] = ‘01’
- For 7-bit character length: DLEN[1:0] = ‘10’
- For 8-bit character length: DLEN[1:0] = ‘11’

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 69 shows a timing diagram of data length programming. A Break condition is interpreted 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 69. Timing Diagram of 8-bit Data Length Programming

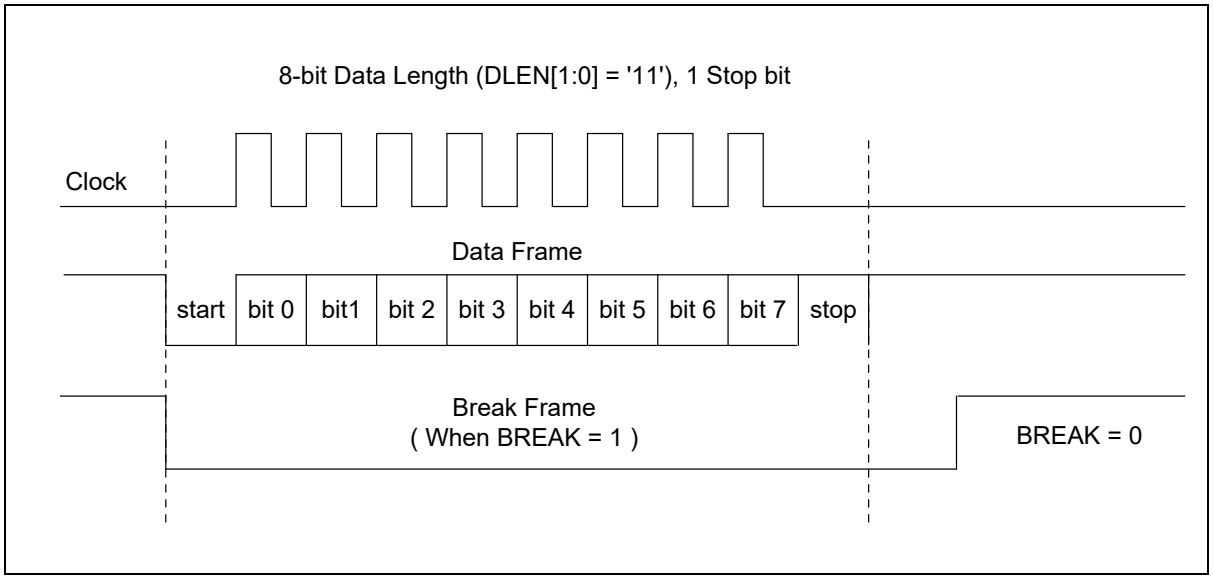


Figure 70. Timing Diagram of 7-bit Data Length Programming

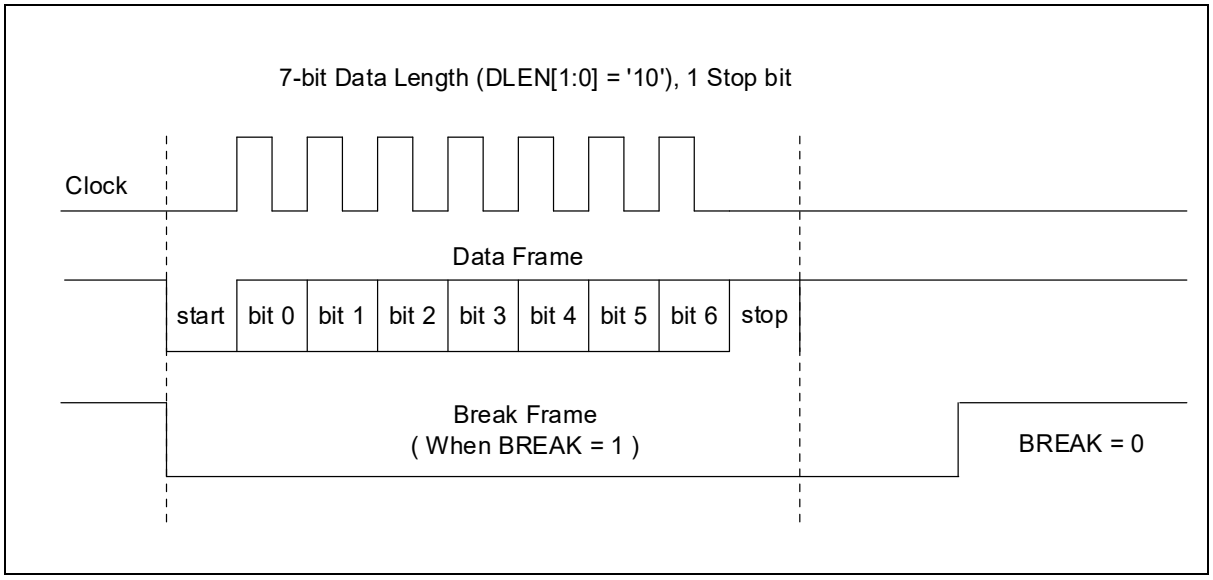


Figure 71. Timing Diagram of 6-bit Data Length Programming

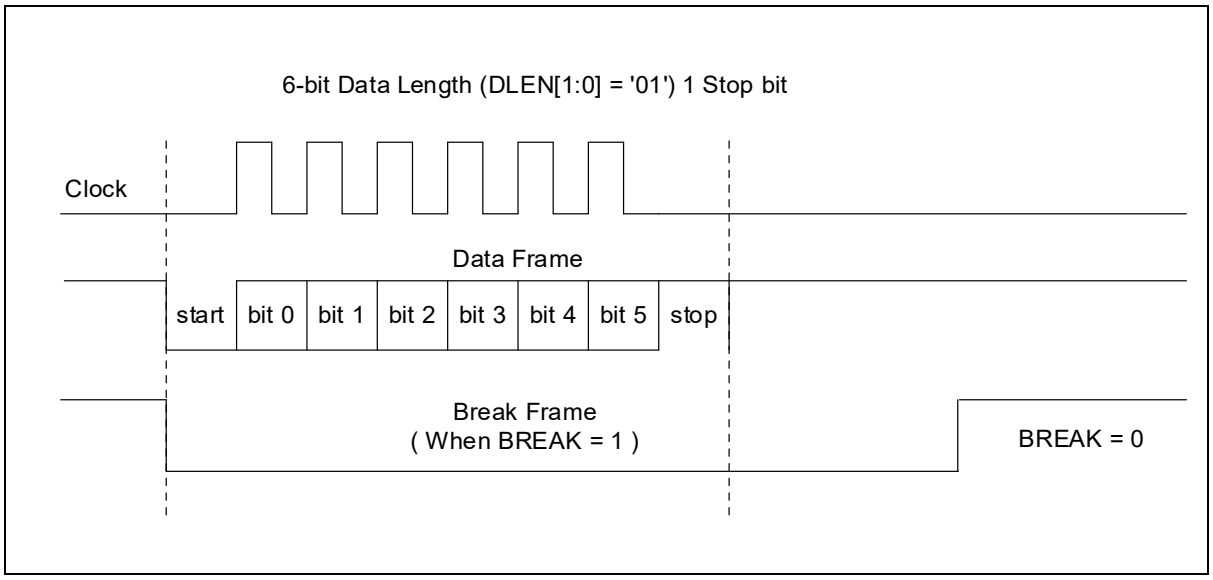
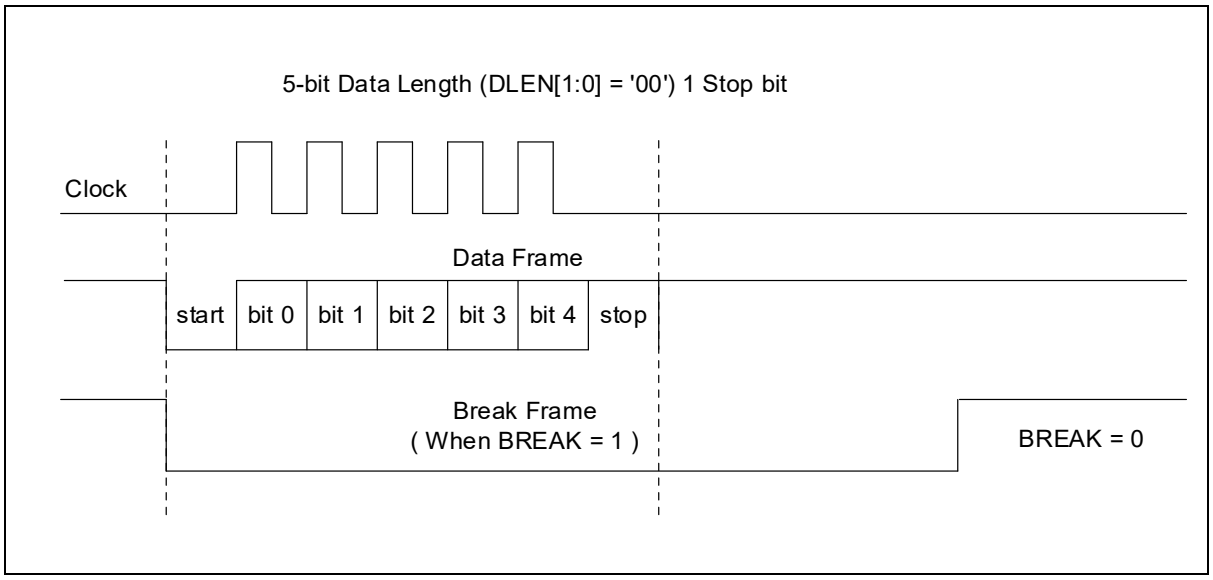


Figure 72. Timing Diagram of 5-bit Data Length Programming



11.3.4 UART Transmitter

11.3.4.1 Data Format

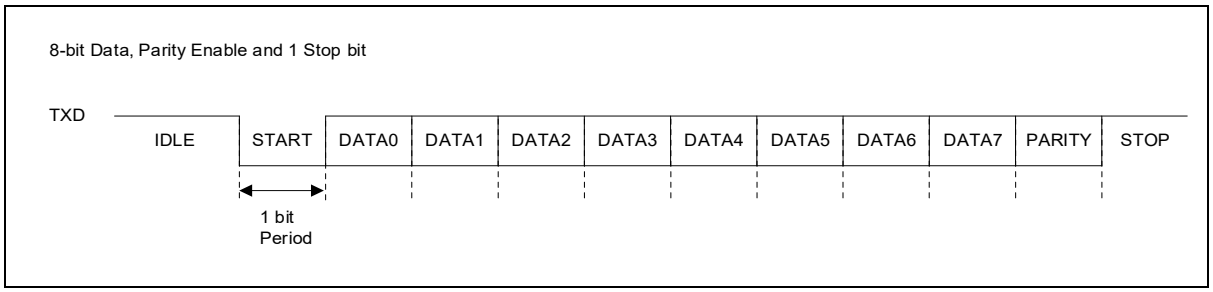
The UART transmitter is in charge of transmitting data. For each data word, the start bit, data bits, optional bit, and stop bit are shifted serially in and out of the corresponding register, one bit at a time at the Least Significant Bit (LSB).

The number of data bits can be determined by setting the DLEN[1:0] in 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] in UARTn_LCR register.

The parity bit type can be determined by setting the PARITY and PEN in 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 in UARTn_LCR register.

Figure 73 shows an example format of transmit data.

Figure 73. Example Format of Transmit Data



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.

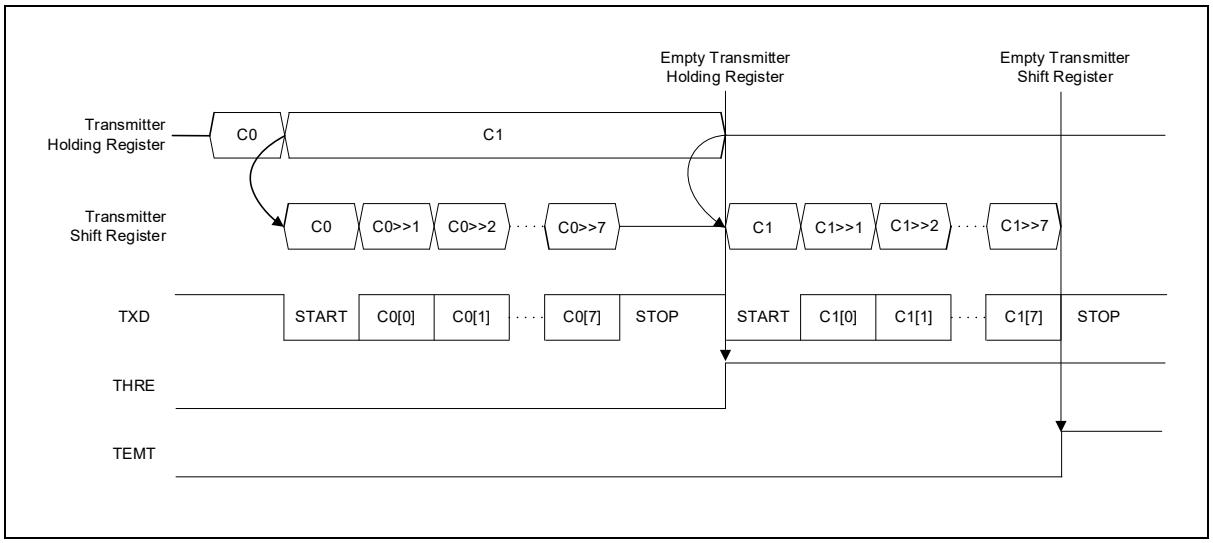
11.3.4.2 Transmit Interrupt

During UART transmission, some types of interrupt flags are used.

When the Transmit data Hold Resistor (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 74.

Figure 74. 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.

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: The UART peripheral and clock enable bits must be set in the PMU_PER and PMU_PCCR 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 in 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 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. Interrupts, and Polling can be used to update the UARTn_THR register after confirming the communication is completed.
   A. Checking if the verifying that the FID[0] + IID[1:0] value is ‘001’,
      - If the FID[0] + IID[1:0] in 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.
   B. Polling the TEMT and THRE in UARTn_LSR register,
      - The TEMT bit implies that the Transmit Shift Register is empty.
      - The THRE bit implies that the UARTn_THR register is empty.
4. The UARTn_THR register can be updated only when these two 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] in 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 outputted data will be updated with new data to start next communication.

The Transmitter, which output a stop bit, restarts communication outputting a start bit.

11.3.4.7 Break Condition

Setting the BREAK in 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: 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.

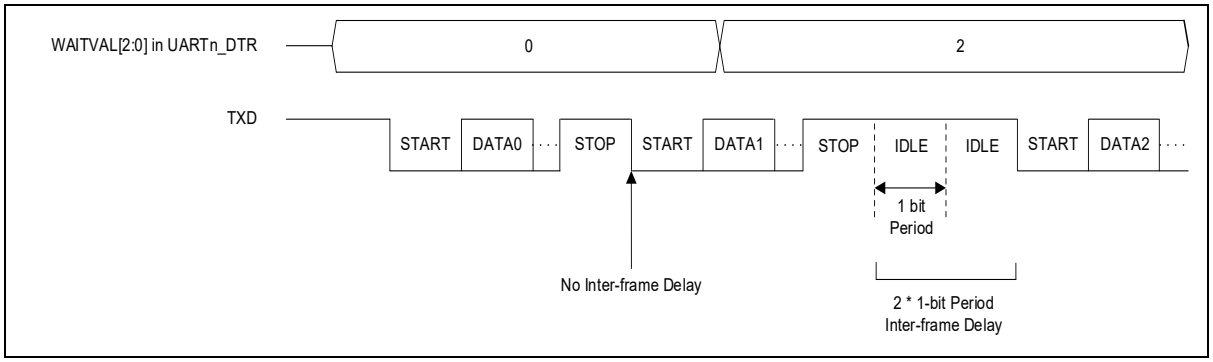
11.3.4.8 Inter-Frame Delay for Data Transmission

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] in UARTn_DTR register.

If the WAITVAL[2:0] 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 75 shows an example format of the transmit data.

Figure 75. Example Format of Transmit Data



11.3.5 UART Receiver

11.3.5.1 Receive Data Single Sampling Timing

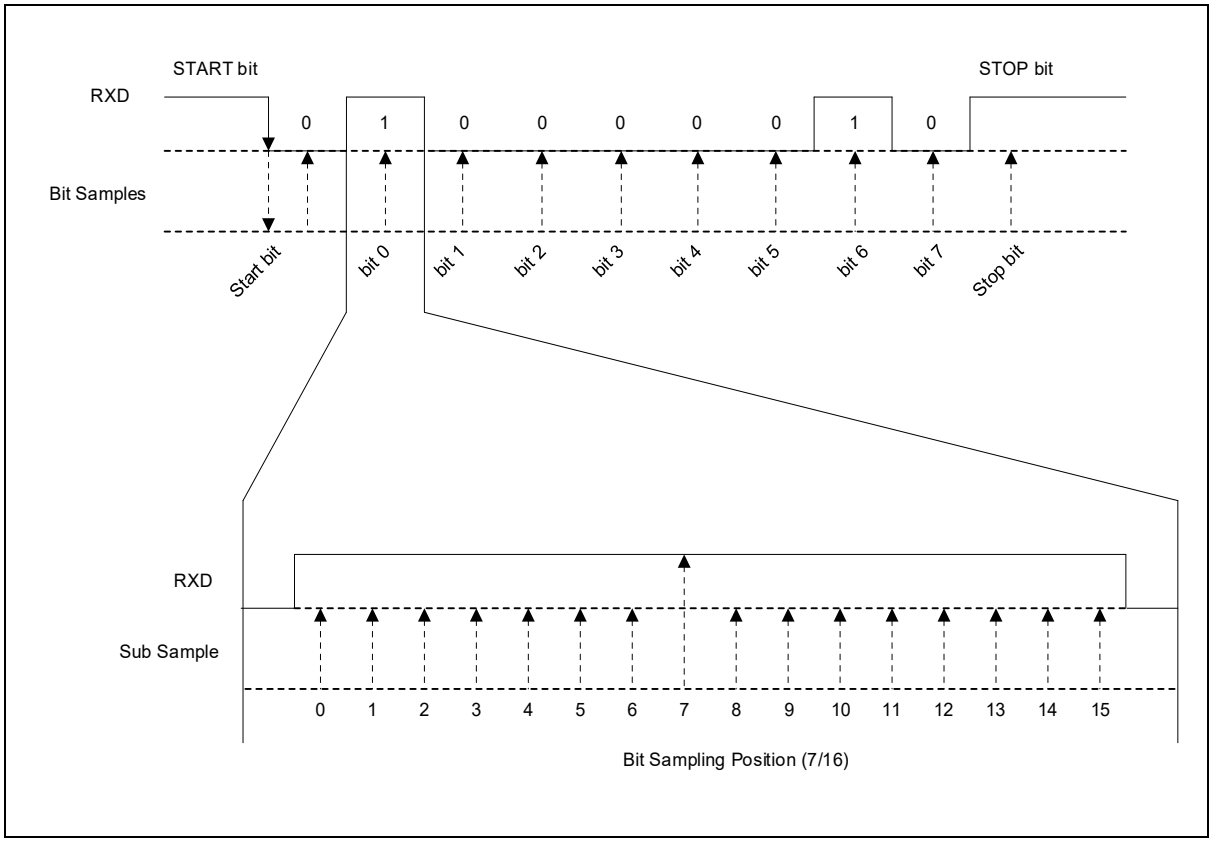
Timing of the UART operation is described below:

On detecting a falling edge of input signal, UART recognizes the falling edge as a START bit signal. From this edge, all bits of a character transmission are sampled by 16-multiple clock signal.

If the SMS or DMS bit is '0' in UARTn_DTR register, Among the 16 samples, the value at the 7th 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] in UARTn_LCR register.

Figure 76. Single Sampling Timing of a UART Receiver



To enhance protection against external glitch noise, it is recommended to enable port debouncing in the PMC module.

NOTE: To use the port debounce function, the debounce clock in the PCn_DER and PCn_DPR registers must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction. (n = A, B, C, D, E, F)

11.3.5.2 Receive Data Multi Sampling Timing

Timing of the UART operation is described below:

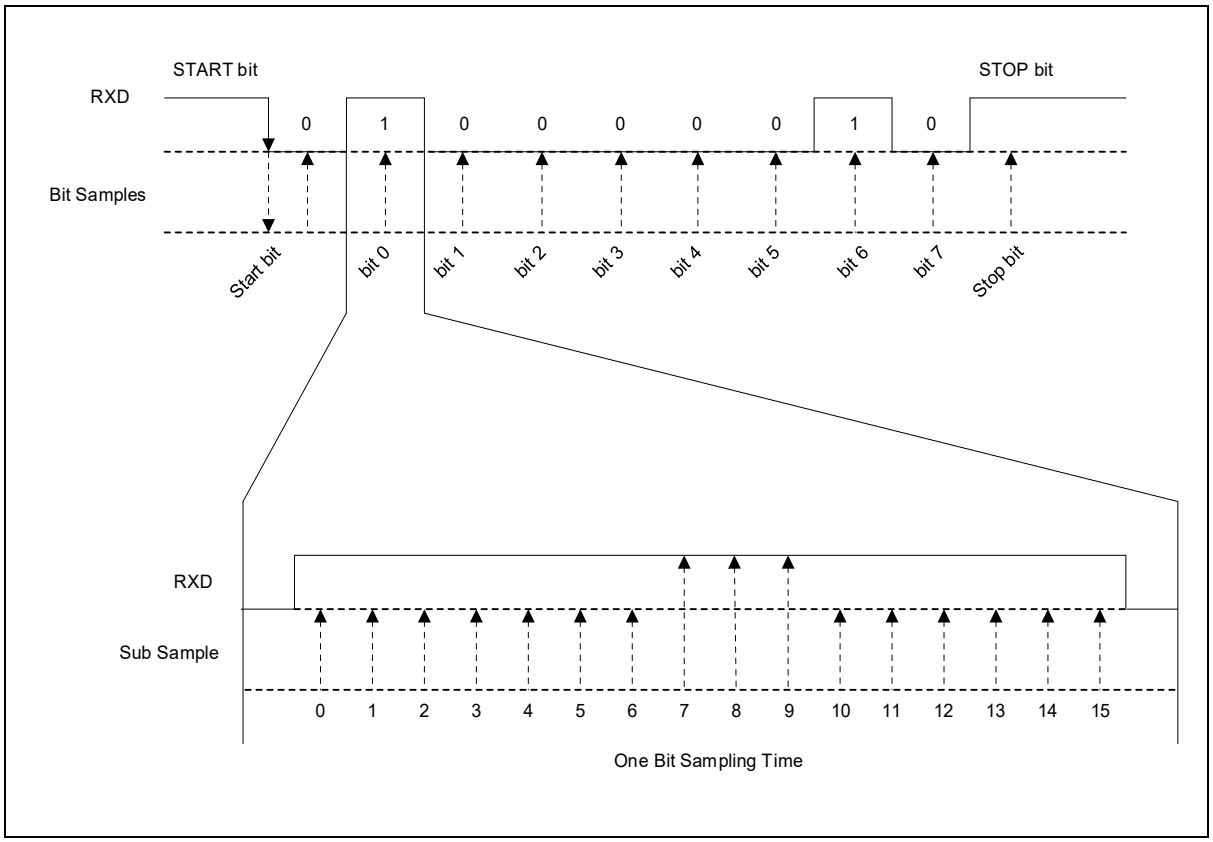
On detecting a falling edge of input signal, UART recognizes the falling edge as a START bit signal. From this edge, all bits of a character transmission are sampled by 16-multiple clock signal.

If the SMS or DMS bit is '1' in UARTn_DTR register, most occurrence value of signals on 7th, 8th, 9th clocks will be the bit value.

To avoid noise errors, oversampling techniques are used (except in synchronous mode) for data recovery by discriminating between valid incoming data and noise.

The UART receiver can receive data words of either 5, 6, 7, or 8 bits, depending on the DLEN[1:0] in UARTn_LCR register.

Figure 77. Multi Sampling Timing of a UART Receiver



To enhance protection against external glitch noise, it is recommended to enable port debouncing in the PMC module.

NOTE: To use the port debounce function, the debounce clock in the PCn_DER and PCn_DPR registers must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction. (n = A, B, C, D, E, F)

11.3.5.3 Start Bit Detection

In the UART receiver block, there are two types of determination strategy of START bit.. One is by single-clock sampling and the other is by multi-clock sampling.

The single-clock sampling feature is the general sampling method, this feature acquires start bit value in the center of the bit period.

The multi-clock sampling feature has noise-rejection function for accurate communication against line noise. The timing diagrams of the single sampling and multi sampling features are shown as followings.

Figure 78. Start Bit Detection of Single-sampling Timing

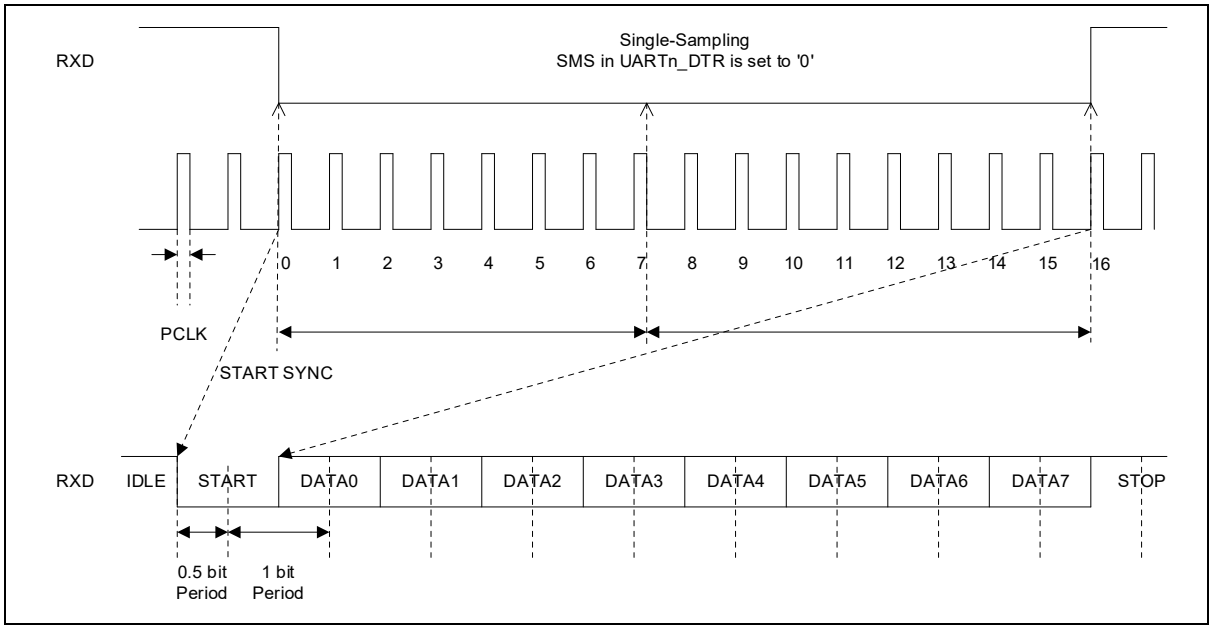
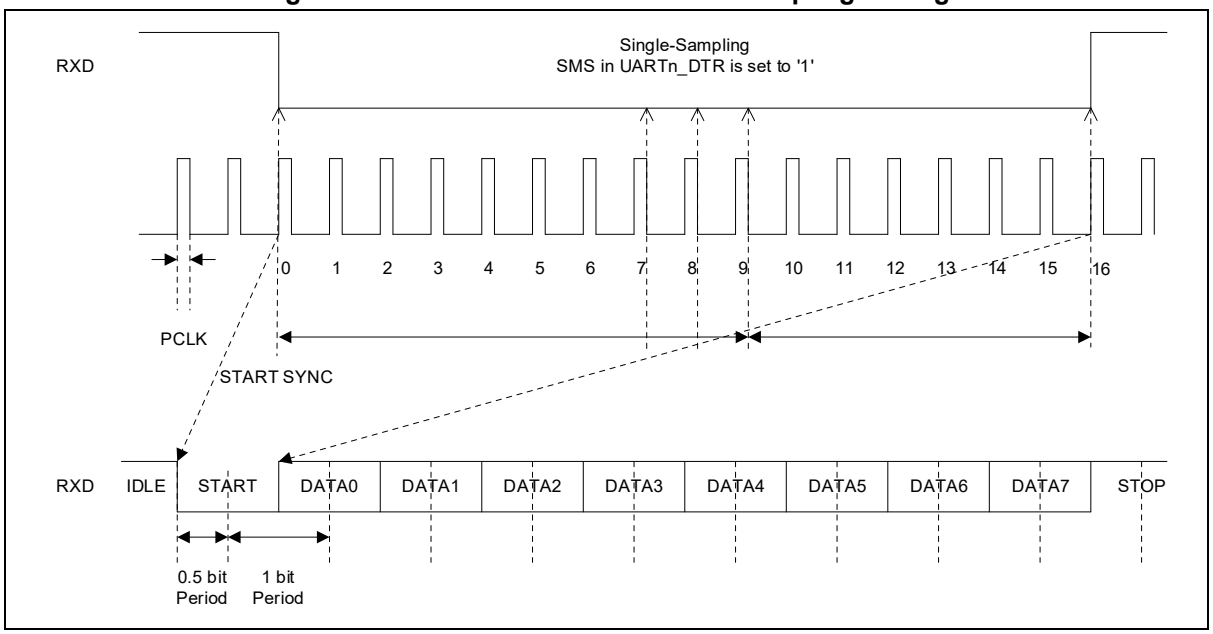


Figure 79. Start Bit Detection of Multi-sampling Timing



11.3.5.4 Data Bit Sampling

If the DMS in UARTn_DTR register is set to ‘1’, the values sampled at the 7th, 8th, and 9th 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 80 shows the single-sampling of data bits when the DMS is set to ‘0’.

Figure 80. Single-sampling Timing of Data Bit

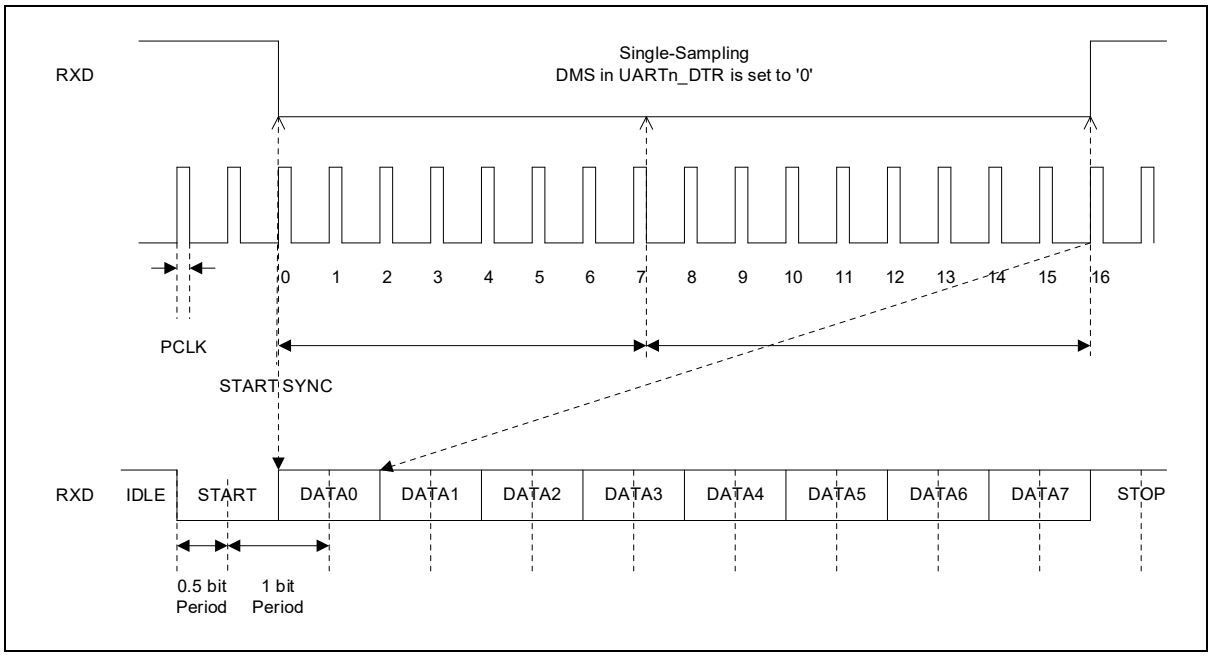
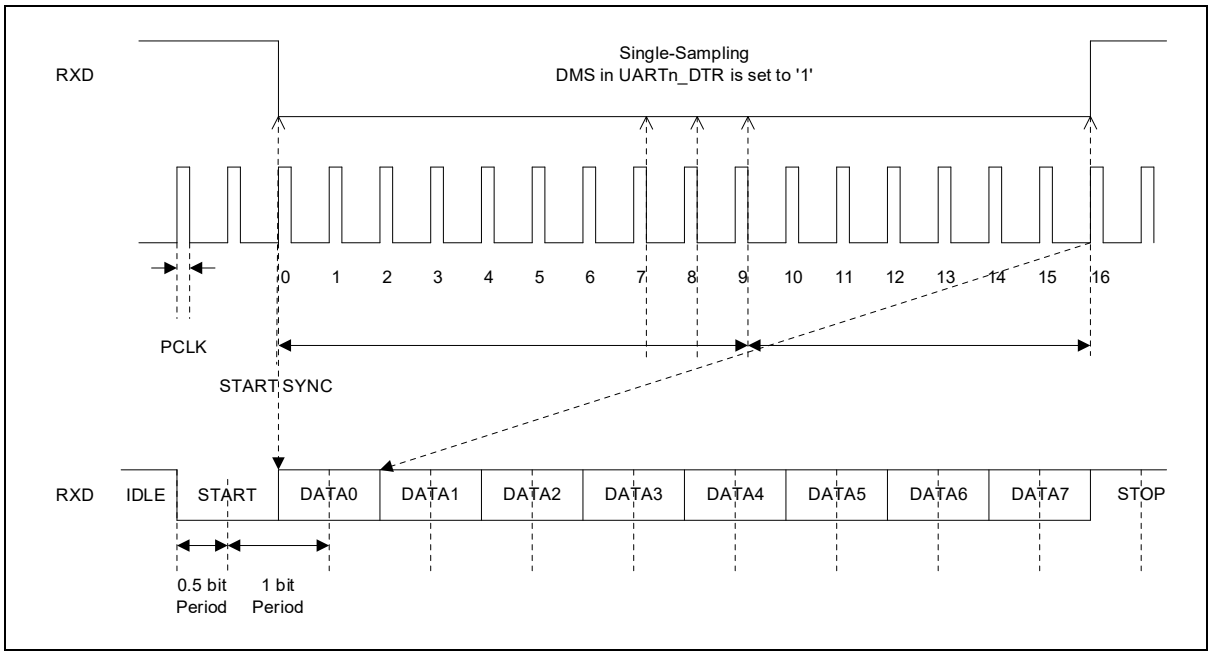


Figure 81 shows the Multi-sampling of data bits when the DMS is set to ‘1’.

Figure 81. Multi-sampling Timing of Data Bit



11.3.5.5 Data Sampling Strategy

An internal synchronous circuit is used in the receiver block to prevent abnormal noises in the RXD signal line. The start bit and data bits can be either single-sampled or multi-sampled. The SMS in UARTn_DTR register defines the sampling strategy for the start bit, and the DMS in UARTn_DTR register defines the sampling strategy for the data bits.

Figure 82. Data Sampling Timing

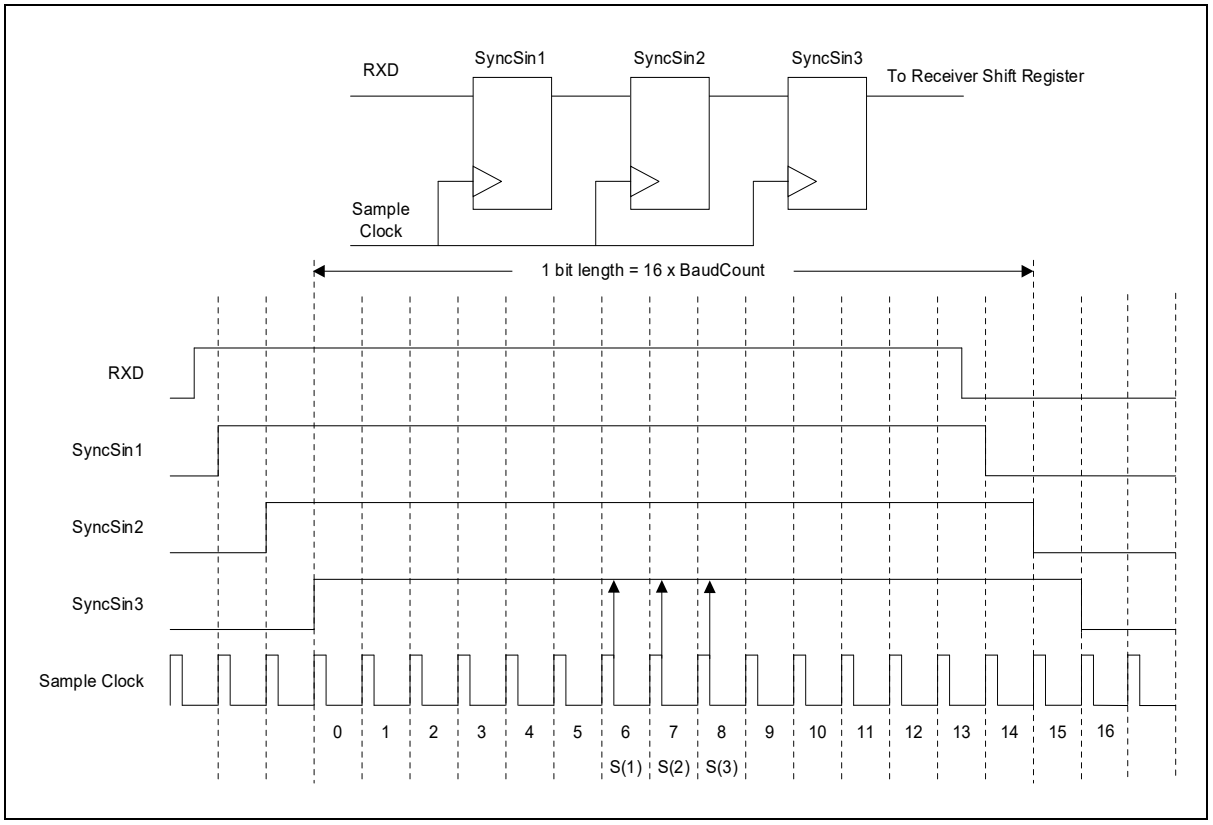


Figure 82 shows a block diagram and a sampling timing diagram of the anti-noise synchronous circuit.

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 for the Start bit.

Table 80. 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

11.3.5.6 Break Condition

When a break condition is received, the UART generates break interrupt.

11.3.5.7 Overrun Error

The OE in UARTn_LSR register indicates whether or not the Overrun Error has been detected. The Overrun Error occurs if the Receiver completes the new Reception operation when the UARTn_RBR data was not read (The DR in 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 in 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] in 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 in 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 in UARTn_LCR register value. Users can change the STOPBIT value while the Receiver is operating.

11.3.6 UART Baud-Rate Generation

Each channel of the UART has a programmable baud rate generator that can receive the PCLK clock and divide it by a divide value from 2 to 65,535. (If dividing value is '1', it is the highest baud rate.). The output clock frequency of the baud rate generate is 16 x Baud rate. In order for UART communication to work properly, the UARTn_DLL and UARTn_DLM registers value must be written to appropriate values before UART operation.

Below is the formula for calculating the baud-rate:

$$\mathrm{BDR(UARTn\_DLM : UARTn\_DLL)} = \frac{PCLK\ /\ 2}{16 \times\ BaudRate}$$

Table 81 shows the baud rate generator baud rate and transmit/receive error rate at each baud rate for PCLK = 37.5 MHz, and the error rate is minimized at baud rates below 19200 bps. The frequency of the input clock affects the accuracy of the baud rate. It is not recommended to use '0' as the division value.

Table 81. Divisor value for each baud rate (PCLK=37.5 MHz)

PCLK = 37.5 MHz (HCLK / 2)		
Baud-rate	**Decimal Divisor Value**	**Error Rate (%)**
1,200	976	0.06 %
2,400	488	0.06 %
4,800	244	0.06 %
9,600	122	0.06 %
19,200	61	0.06 %
38,400	30	1.70 %
57,600	20	1.70 %
115,200	10	1.70 %

Table 82. Divisor value for each baud rate (PCLK=8 MHz)

PCLK = 8 MHz (HCLK / 2)		
Baud-rate	**Decimal Divisor Value**	**Error Rate (%)**
1,200	208	0.16%
2,400	104	0.16%
4,800	52	0.16%
9,600	26	0.16%
19,200	13	0.16%
38,400	X	X
57,600	X	X
115,200	X	X

Table 83 below shows the BFR values for each baud rate of the UART channel based on the 75 MHz peripheral clock. You can set the baud rate more accurately by referring to the BFR value.

For example, when the baud-rate is 2,400 bps:

$$\frac{UART\ clock\ /\ 2}{16 \times\ BaudRate} = \frac{75000000\ /\ 2}{16 \times\ 2400} = 976.56\ \ Divisor = 976, Float = 0.56$$

$$\text{FCNT} = \text{Float} \times 256\ = 0.56 \times 256 = 144$$

$$\text{BDR (UARTn\_DLM : UARTn\_DLL)} = 976, \text{UARTn\_BFR} = 144$$

FCNT = (Decimal point value) × 256.

In this case, the decimal point value is derived from decimal point of the BDR that calculated the UARTn_DLM, UARTn_DLL, PCLK, and baud rate values.

For example, FCNT value can calculated above equation. For example, the target baud rate is 2,400 bps and UART_PCLK is 75 MHz case, the BDR value is 976.56. The integer number 976 should be the BDR value and the floating number 0.56 will make the FNCT value as below.

FCNT = 0.56 × 256 = 144, So the FCNT value is 144.

8-bit fractional counter will count up by FCNT value every (baud rate)/16 period and whenever fractional counter overflow is happen, the divisor value will increment by 1. So this period will be compensated. Then next period, the divisor value will return to original set value.

Table 83. Examples Calculation of Baud-rate Using a UART Fraction Counter

UART clock = 75 MHz					
Baud-rate (bps)	**Divisor (BDR) (DLM : DLL)**	**Decimal Point Value**	**FCNT**	**UARTn_BFR**	**Error (%)**
1,200	1953.13	0.13	32.00	32	0.00%
2,400	976.56	0.56	144.00	144	0.00%
4,800	488.28	0.28	72.00	72	0.00%
9,600	244.14	0.14	36.00	36	0.00%
19,200	122.07	0.07	18.00	18	0.00%
38,400	61.04	0.04	9.00	9	0.00%
57,600	40.69	0.69	176.67	176	0.38%
115,200	20.35	0.35	88.33	88	0.38%
230,400	10.17	0.17	44.17	44	0.38%
460,800	5.090	0.09	22.08	22	0.38%

11.3.7 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 UARTn_LCR register, the possible UART frame formats are as listed in Table 84.

Table 84. UART Frame Formats

DLEN[1:0] bits	PEN bits	UART frame (1)
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 |

NOTES:

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).

11.3.7.1 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 in UARTn_LCR register = ‘1’ and PEN in UARTn_LCR register = ‘1’).

11.3.7.2 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 in UARTn_LCR register = ‘0’ and PEN in UARTn_LCR register = ‘1’).

11.3.7.3 Stick Parity

The STICKP in UARTn_LCR register determines whether or not 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.

11.3.7.4 Various Configurations of Parity Bit

A parity bit is generated based on the settings of the PEN, PARITY and STICKP in UARTn_LCR register.

Table 85 describes various configurations to create a parity bit.

Table 85. Various Configurations to Create a Parity Bit

Name	Offset	Type	Description
STICKP	PARITY	PEN	Parity
X	X	0	No parity
0	0	1	Odd parity
0	1	1	Even parity
1	0	1	Forced 1 stick parity
1	1	1	Forced 0 stick parity

11.3.7.5 Parity Checking in Reception

If the parity check fails, the PE flag is set by configuring the PE in UARTn_LSR register, and the corresponding interrupt is generated if the RLSIE in UARTn_IER register is set to '1'.

Reading the UARTn_LSR register clears the PE bit to '0'.

11.3.7.6 Parity Generation in Transmission

If the PEN in 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.

11.4 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 FID[0] + IID[1:0] in UARTn_IIR register.

The UART module uses a total of three interrupts with different priorities. These interrupts include the followings:

- Receive line status interrupt
- Receive data ready interrupt / Character time-out interrupt
- Transmit data hold register empty interrupt

11.5 Setting Example

<Example 1> Initialization of UART0 channel (baud rate = 19,200 bps, PCLK = 8 MHz)

PCMR<P8[17:16]> = “01”	: Selects the function of PC8 pin as RXD0
PCMR<P9[19:18]> = “01”	: Selects the function of PC9 pin as TXD0
PCCR<P8[17:16]> = “10”	: Selects the direction of PC8(RXD0) pin as logic input
PCCR<P9[19:18]> = “00”	: Selects the direction of PC9(TXD0) pin as push-pull output
UARTn_DTR< RXINV> = “0”	: Disables the inverted output of RXD0
UARTn_DTR< TXINV> = “0”	: Disables the inverted output of TXD0
UARTn_LCR<DLEN[0:1]> = “11”	: Set the data length to 8 bits.
UARTn_LCR<STOPBIT> = “0”	: Sets 1 stop bit
UARTn_LCR<PEN> = “0”	: Disables parity bit
UARTn_LCR<PARITY> = “0”	: Sets odd parity mode
UARTn_LCR<DLAB> = “0”	: Sets divisor latch access bit to normal transmission mode
UARTn_DLL <DLL[7:0]> = “00000000”	: Sets the value of divisor latch low byte to 0
UARTn_DLM< DLM[7:0]> = “00001101”	: Sets the value of divisor latch high byte to 13
UARTn_BFR<BFR[7:0]>= “0”	: Sets the value of divisor decimal point calculation register to 0
UARTn_FCR< FIFOEN> = “0”	: Disables transmit/receive FIFO type
UARTn_FCR< FCR1> = “0”	: Initializes the receive FIFO
UARTn_FCR< FCR2> = “0”	: Initializes the transmit FIFO
UARTn_FCR< FIFODEPTH > = “0”	: Sets the 1 byte receive FIFO trigger level
UARTn_SCR<SCR[7:0] =“00000000”	: Initialies the value of scratch byte data buffer
UARTn_RBR<RO[7:0]> READ	: Read the received byte data
UARTn_THR<THR[7:0]> READ	: Read the transmit byte data

11.6 UART Registers

The base addresses and register map of the UART are as follows:

Table 86. Base Addresses of UART

Name	Base address
UART0 (16-depth FIFO)	0x4000_0B00
UART1 (16-depth FIFO)	0x4000_0B40
UART2 (Double buffer)	0x4000_0B80
UART3 (Double buffer)	0x4000_0BC0

Table 87. UART Register Map

Name	Offset	Type	Description	Reset value	Reference
UARTn_RBR	0x0000	RO	UART n Rx Buffer Register (DLAB=0)	0x0000_0000	11.6.1
UARTn_THR	0x0000	WO	UART n Tx Data Hold Register (DLAB=0)	0x0000_0000	11.6.2
UARTn_DLL	0x0000	RW	UART n Divisor Latch Register LSB (DLAB=1)	0x0000_0000	11.6.3
UARTn_DLM	0x0004	RW	UART n Divisor Latch Register MSB (DLAB=1)	0x0000_0000	11.6.4
UARTn_IER	0x0004	RW	UART n Interrupt Enable Register	0x0000_0000	11.6.5
UARTn_IIR	0x0008	RO	UART n Interrupt ID Register	0x0000_0001	11.6.6
UARTn_FCR	0x0008	WO	UART n FIFO Control Register	0x0000_0000	11.6.7
UARTn_LCR	0x000C	RW	UART n Line Control Register	0x0000_0000	11.6.8
UARTn_LSR	0x0014	RO	UART n Line Status Register	0x0000_0060	11.6.9
UARTn_SCR	0x001C	RW	UART n Scratch pad Register	0x0000_0000	11.6.10
UARTn_BFR	0x0024	RW	UART n Divisor Decimal Point Calculation Register	0x0000_0000	11.6.11
UARTn_DTR	0x0028	RW	UART n Delay Time Register	0x0000_0000	11.6.12

NOTE: n = 0, 1, 2, and 3.

11.6.1 UARTn_RBR: UART n Receive Buffer Register

The UART receive data buffer register can be read with a maximum length of 8 bits. Serial input data is stored in the receive buffer register. The last receive data is stored in this register until a new bit is received.

This register is a read-only register and can be accessed when DLAB (the seventh bit in the UARTn_LCR register) is set to '0'.

UART0_RBR=0x4000_0B00, UART1_RBR=0x4000_0B40
UART2_RBR=0x4000_0B80, UART3_RBR=0x4000_0BC0

31 30 29 28 27 26 25 24	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]	Received data byte

11.6.2 UARTn_THR: UART n Transmit Data Hold Register

The data to be output to the serial communication port of the UART transmit data buffer register is written with a maximum length of 8 bits. This register is a write-only register and can be accessed when DLAB (7th bit in UARTn_LCR register) is '0'.

UART0_THR=0x4000_0B00, UART1_THR=0x4000_0B40
UART2_THR=0x4000_0B80, UART3_THR=0x4000_0BC0

31 30 29 28 27 26 25 24	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]
-			0x00
-			WO

7 0	THR[7:0]	Data byte to be transmitted

11.6.3 UARTn_DLL: UART n Divisor Latch Register LSB

UART0_DLL=0x4000_0B00, UART1_DLL=0x4000_0B40
UART2_DLL=0x4000_0B80, UART3_DLL=0x4000_0BC0

31 30 29 28 27 26 25 24	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			DLL[7:0]
-			0x00
-			RW

7 0	DLL[7:0]	Divider latch low byte

11.6.4 UARTn_DLM: UART n Divisor Latch Register MSB

UART0_DLM=0x4000_0B04, UART1_DLM=0x4000_0B44
UART2_DLM=0x4000_0B84, UART3_DLM=0x4000_0BC4

31 30 29 28 27 26 25 24	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			DLM[7:0]
-			0x00
-			RW

7 0	DLM[7:0]	Divider latch high byte

Each channel of the UART has a programmable baud rate generator that can receive the PCLK clock and divide it by a divide value from 2 to 65,535. (If dividing value is '1', it is the highest baud rate.). The output clock frequency of the baud rate generate is 16 × Baud rate. In order for UART communication to work properly, the DLL and DLM registers value must be written to appropriate values before UART operation.

For more information on the baud rate calculation formula, see chapter 11.3.6.

11.6.5 UARTn_IER: UART n Interrupt Enable Register

This register enables four types of UART interrupts. Each interrupt generates an interrupt output signal. Setting all bits in the interrupt enable register (IER) to 0 disables all interrupts in the UART and sets the corresponding bit in UARTn_IER to 1 to enable only certain interrupts. If the interrupt is disabled, it is not written to the UARTn_IIR register. Even if the corresponding interrupt condition occurs, the interrupt output signal is not generated. All other UART functions, such as the status display of the UARTn_LSR register, operate identically regardless of the UARTn_IER setting.

UART0_IER=0x4000_0B04, UART1_IER=0x4000_0B44
UART2_IER=0x4000_0B84, UART3_IER=0x4000_0BC4

31 30 29 28 27 26 25 24	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				TEMTIE	RLSIE	THREIE	DRIE
-				0	0	0	0
-				RW	RW	RW	RW

3	TEMTIE	Whether to enable or disable the transmission empty interrupt	
		0	Disables the transmission empty interrupt.
		1	Enables the transmission empty interrupt.
2	RLSIE	Whether to enable or disable the receive line status interrupt	
		0	Disables the receive line status interrupt.
		1	Enables the receive line status interrupt.
1	THREIE	Whether to enable or disable the transmit data hold register empty (THRE) interrupt	
		0	Disables the THRE interrupt.
		1	Enables the THRE interrupt.
0	DRIE	Whether to enable or disable the data receive ready interrupt [in double buffer mode] If this bit is enabled in double buffer mode, data receive ready interrupt is enabled. [in FIFO (First in, first out) mode] If this bit is enabled in FIFO mode, data receive ready interrupt and character time-out interrupt are enabled.	
		0	Disables the data receive ready interrupt.
		1	Enables the data receive ready interrupt.

11.6.6 UARTn_IIR: UART n Interrupt ID Register

This register is an 8-bit read-only register that is displayed as an ID value for each interrupt operation set in the UART interrupt enable register (UARTn_IER) register. When CORE accesses the UARTn_IIR register, the UART locks all interrupts and the highest priority interrupt is read. Interrupts occur while CORE is accessed, but do not change state until the current access is completed.

UART0_IIR=0x4000_0B08, UART1_IIR=0x4000_0B48
UART2_IIR=0x4000_0B88, UART3_IIR=0x4000_0BC8

31 30 29 28 27 26 25 24	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			FIFO[1:0]	Reserved	XMITE	FID	IID[1:0]	IPEN
-			00	-	0	0	00	1
-			RO	-	RO	RO	RO	RO

7 6	FIFO[1:0]	Display the current UART operation mode	
		00	None-FIFO (Double buffer mode)
		11	FIFO mode
4	XMITE	Transmit empty interrupts	
		0	NOT occurred transmit empty interrupt
		1	Occurred transmit empty interrupt
3 1	IID[1:0]/FID	The ID value of interrupt (Table 88. Interrupt IDs and Control)	
0	IPEN	Display the status of interrupt	
		0	Pending a interrupt to be processed
		1	None of interrupt

The UART supports interrupts with three levels of priority and displays the current highest priority interrupt ID in the current UARTn_IIR. Three levels of interrupts are configured with the following priorities:

1) Receive Line Status Interrupt

2) Receive data ready interrupt / Character time-out interrupt

3) Transmit holding register empty register

Detail description is below:

- **IPEN**: This bit indicates whether there are currently unprocessed interrupts. If IPEN = '0', it means that a specific interrupt condition has occurred. IID and FID of UARTn_IIR can be used as a pointer to the interrupt processing routine. If IPEN = '1', it indicates that an unhandled interrupt is not present.
- **IID**: This 2-bit value indicates the ID of the highest priority interrupt currently occurring. This is covered in detail in "Figure 11.5 Interrupt list and way for handling interrupt".
- **FID**: 16450 mode, this bit is always 0, and in FIFO mode, along with IID, indicates that the time-out interrupt is the highest priority interrupt currently generated.
- **FIFO**:These two bits become '1' when FIFOEN in UARTn_FCR register is ‘1’.

Table 88. Interrupt IDs and Control

Priority level	FIFO mode only	IPEN			Interrupt source		
	Bit 3	Bit 2	Bit 1	Bit 0	Interrupt name	Interrupt condition	Interrupt clear
-	0	0	0	1	N/A	-	-
1	0	1	1	0	Receiver	Over-run, Parity, Framing, Break, etc.	Read UARTn_LSR register
2	0	1	0	0	Line Status	Received data exists Or Received FIFO level is reached	Read receive buffer data Or Receive data is read until it is below FIFO level
2	1	1	0	0	Receiver	Continuous data is NOT received for 4 character time after a data below FIFO level is received.	Read receive buffer data
3	0	0	1	0	Data Available	THR (Transmitter Holding Register) is empty.	Read UARTn_IIR register or Write a value to UARTn_THR register
4	0	0	0	0	N/A	-	-

11.6.7 UARTn_FCR: UART n FIFO Control Register

This register is a write-only 8-bit register that has the same location as UARTn_IIR, and is a register that can be used by the UART0 and UART1 channels only. (Conversely, UARTn_IIR is a read-only register). The UARTn_FCR register is used to enable the FIFO, clear the transmit/receive FIFO, and set the trigger level of the receive FIFO.

UART0_FCR=0x4000_0B08, UART1_FCR=0x4000_0B48
UART2_FCR=0x4000_0B88, UART3_FCR=0x4000_0BC8

31 30 29 28 27 26 25 24	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			FIFODEPTH [1:0]	Reserved	FCR2	FCR1	FIFOEN
-			0	-	0	0	0
-			WO	-	WO	WO	WO

7 6	FIFODEPTH[1:0]	Receiving FIFO trigger level	
		00	1 byte
		01	4 bytes
		10	8 bytes
		11	14 bytes
2	FCR2	0	Initial value
		1	Initialize transmit FIFO
1	FCR1	0	Initial value
		1	Initialize receive FIFO
0	FIFOEN	0	Disable FIFO
		1	Enable transmit/receive FIFO

The functions supported in the UARTn_FCR register are described below.

- **FIFOEN**: As writing “1” to this field, both transmit and receive FIFO are enabled and UART enters the FIFO mode. If FIFOEN in UARTn_FCR register is set to “0”, all data in the FIFO is erased while mode is changed from the FIFO to 16450. Although FIFOEN is set to “1” when FIFOEN in UARTn_FCR register is “0”, all data in FIFO is erased, too. If FIFOEN in UARTn_FCR register is not “1”, the others of the UARTn_FCR cannot be written.
- **FCR1**: Writing “1” to the FCR1 in UARTn_FCR register clears all data in the RCVR FIFO and reset its counter logic to “0”. (The receiver shift register is not cleared). Written data “1” is automatically cleared to “0”.
- **FCR2**: Writing “1” to the FCR2 in UARTn_FCR register clears all data in the XMIT FIFO and reset its counter logic to “0”. (Transmitter shift register is not cleared) Written data “1” is automatically cleared to “0”.
- **FIFODEPTH**: This bit is used to specify level for interrupt generation of the RCV FIFO.

11.6.8 UARTn_LCR: UART n Line Control Register

This register can set the asynchronous data transmit / receive format of the UART line or set the divisor latch access bit (DLAB).

The UARTn_LCR register is able to read / write access. It is not necessary to separately store the current format of the asynchronous data transmission / reception line in the system memory.

UART0_LCR=0x4000_0B0C, UART1_LCR=0x4000_0B4C
UART2_LCR=0x4000_0B8C, UART3_LCR=0x4000_0BCC

31 30 29 28 27 26 25 24	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			DLAB	BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]
-			0	0	0	0	0	0	00
-			RW	RW	RW	RW	RW	RW	RW

7	DLAB	0	Normal transmission
		1	Divisor latch access mode
6	BREAK	0	Normal transmission
		1	Send break
5	STICKP	0	Stick parity as ‘L’
		1	Stick parity as ‘H’
4	PARITY	0	Odd parity
		1	Even parity
3	PEN	0	Disable parity
		1	Enable parity
2	STOPBIT	0	1 stop bit
		1	1.5 / 2 stop bits
1 0	DLEN[1:0]	00	5-bit data
		01	6-bit data
		10	7-bit data
		11	8-bit data

It is possible to make an asynchronous data transmit/receive configuration and set DLAB to write data on divisor latch register. LCR is r/w register, as using this register, it is not need to store the configuration of current asynchronous data transmit/receive. See the next description for details.

- **DLEN** : DLEN is used to specify the number of bit for a transmit/receive data word
- **STOPBIT** : This bit represents a type of stop bit which is inserted at the end of transmit/receive data. If this bit is “0”, 1-stop bit is inserted at the end of the transmit data. When this bit is”1”, if the word length is set to 5bit by DLEN, 1.5-stop bit is inserted at the end of the transmit data and stop bit will be 2-bit in the case of 6~8 bit of word length. Receiver checks the first bit of stop bit regardless of the number of stop bit selected by DLEN and STOP.

- **PEN** : This bit is parity enable bit. When PEN is “1”, parity bit is generated between last data bit and stop bit in transmitter, then receiver checks parity bit between last data bit and stop bit. (parity bit is generated to make the number of bits with value of “1” in a given set of data and parity to even or odd according to the configuration of PARITY and STICKP bits.)
- **PARITY** : This is a parity selection bit. If PEN is “1” and PARITY is “0”, the number of “1” should be odd. If PEN is “1” and PARITY is “1”, the number of “1” should be even. The number of “1” includes all “1” in data and parity.
- **STICKP** : This bit is Stick parity. If PEN, PARITY and STICKP are all “1”, parity bit always should be “0”. If PEN and STICKP is “1” and PARITY is “0”, parity bit always should be “1”. When STICKP is “0”, stick parity is disabled.
- **DLAB** : This bit is a divisor latch access bit to control an access to DLL/DLM register. In order to access divisor latch register to control baud generator, this bit should be “1”. To access receive buffer, Transmit hold register and interrupt enable register, DLAB should be “0”.
- **BREAK** : This is break control bit. If BREAK set to “1”, serial output (TxD) become”0” (spacing state). Break condition is cleared by setting BREAK bit to “0”

  NOTE: This function is used to warn on the opposite side in communication system. In the case that break function is used, transfer can stop by “break”

11.6.9 UARTn_LSR: UART n Line Status Register

UARTn_LSR is a read-only register that shows information on the status of data transmission and reception.

UART0_LSR=0x4000_0B14, UART1_LSR=0x4000_0B54
UART2_LSR=0x4000_0B94, UART3_LSR=0x4000_0BD4

31 30 29 28 27 26 25 24	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			FIFOE	TEMT	THRE	BI	FE	PE	OE	DR
-			0	1	1	0	0	0	0	0
-			RO	RO	RO	RO	RO	RO	RO	RO

7	FIFOE	0	FIFO is valid
		1	FIFO has invalid data
6	TEMT	0	Transmitter NOT empty
		1	Transmitter empty
5	THRE	0	THR NOT empty
		1	THR empty
4	BI	0	No break detection
		1	Break interrupted
3	FE	0	No framing error
		1	Framing error
2	PE	0	No parity error
		1	Parity error
1	OE	0	No overrun error
		1	Overrun error
0	DR	0	No data received
		1	Received data ready

This register represents transmit/receive status. The next describes each field in details:

- **DR** : This bit represent that data is completely received and moved to receive FIFO. If receive buffer register or FIFO is read, DR becomes to “0”
- **OE** : This bit represents overrun error. OE will be set if next receive data is moved to receive buffer register before previous receive data is read. OE is cleared to “0” when LSR is read by CPU. In FIFO mode, if the next receive data is completely received to the shift register after 16 bytes of FIFO is filled to the full with received data, OE is set to “1”. At this time, the data in receive register is damaged but it is not moved to FIFO.
- **PE** : This is parity error indicator. PE is set to “1” when received data did not satisfy parity condition in LCR. PE is cleared to “0” when LSR is read by CPU. In FIFO mode, PE becomes “1” when parity error data is placed on the top of the FIFO.

- **FE** : This is framing error indicator that represents stop bit of received data is not correct. FE is set when stop bit that follows after the last data bit or parity bit is detected to “0”. FE is cleared to “0” as reading LSR by ARM7TDMI core. In FIFO mode, PE set to “1” when framing error data is placed on the top of the FIFO. In order to resynchronize the receiver again, UART starts next word receive with assumption that UART framing error is generated by start bit of next receive data.
- **BI** : BI is break interrupt bit. BI is set when received data input keeps “0” for more time than total data receive time(the total receive time for start bit, data bit, parity bit, and stop bit). BI is cleared as reading by ARM7TDMI core. In FIFO mode, PE is set to “1” when break error data is placed on the top of the FIFO. If Break condition occurs, one zero data is sent to the FIFO. Next input of data starts when new start bit is received after receiver becomes “1”.

  NOTE: Bit 1, 2, 3, and 4 of LSR causes the receive line status interrupt when corresponding interrupt is enabled.
- **THRE** : This bit represents transmit holding register empty (THRE). THRE is a cause of interrupt when transmitter holding register empty interrupt (THREIE) is enabled and UART is ready to receive new data from CPU. THRE is set to “1” when transmitter holding register is possible to be written because transfer data is moved from transmitter holding register to transmitter shift register. If new data is written to transmitter holding register, it is cleared to “0”. In FIFO mode, THRE is set to “1” when there are not any data in XMIT FIFO, and THRE is set to “0” when there are one or more data in XMIT FIFO.
- **TEMT** : This bit represents transmit data empty. TEMP is set to “1” when transmit holding register (THR) and transmit shift register (TSR) are empty and TEMP is set to “0” when there are data in THR or TSR. In FIFO mode, TEMP is set to “1” when XMIT FIFO and TSR are all empty.
- **FIFOE** : In 16450 mode , this bit is always “0”. In FIFO mode, FIFOE is set to “1” when there is one or more event among parity error, framing error, or break detection. FIFOE is cleared to “0” when all data with error are read and LSR is read by Core.

  NOTE: Line status register is read-only register.

11.6.10 UARTn_SCR: UART n Scratch Register

This 8-bit register is independent of the UART operation and settings. This register is prepared for the software to be used as temporary space for storing data.

UART0_SCR=0x4000_0B1C, UART1_SCR=0x4000_0B5C
UART2_SCR=0x4000_0B9C, UART3_SCR=0x4000_0BDC

31 30 29 28 27 26 25 24	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			SCR[7:0]
-			0
-			RW

7 0	SCR	Scratch byte data

11.6.11 UARTn_BFR: UART n Divisor Decimal Point Calculation Register

This 8-bit register is used to increase the baud rate accuracy.

Calculate the UART baud rate using UARTn_DLM and UARTn_DLL registers, and if there is a decimal point value, multiply the decimal point part by '256' and set the integer value in UARTn_BFR register.

UART0_BFR=0x4000_0B24, UART1_BFR=0x4000_0B64
UART2_BFR=0x4000_0BA4, UART3_BFR=0x4000_0BE4

31 30 29 28 27 26 25 24	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]	Divisor Decimal Point Calculation Register **NOTE:** For more information on the setting of BFR, see chapter 11.3.6

11.6.12 UARTn_DTR: UART n Delay Time Register

This UARTn_DTR register adjusts the delay time between UART start bit or data bit frame to prevent start error due to noise during sampling or set inverted input or output of UART Rx/Tx signal.

UART0_DTR=0x4000_0B28, UART1_DTR=0x4000_0B68
UART2_DTR=0x4000_0BA8, UART3_DTR=0x4000_0BE8

31 30 29 28 27 26 25 24	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	RXINV	TXINV	Reserved	WAITVAL
-			0	0	0	0	-	0
-			RW	RW	RW	RW	-	RW

7	SMS	0	Start bit sampling single time
		1	Start bit sampling three times (Start error prevention due to noise)
6	DMS	0	Data bit sampling single time
		1	Data bit sampling three times (Error prevention due to noise)
5	RXINV	0	No external RX input change
		1	External RX input inversion
4	TXINV	0	No TX output change
		1	TX output inversion
1	WAITVAL	To set the delay time between the output of the continuous data. $\text{Wait time } (W_T) = \frac{\text{WAITVAL}}{\text{Baud rate}}$	

- **SMS** : DTR register adjusts the delay time between the UART start bit or data bit to prevent start-up errors due to noise during sampling or to set the inverted input / output of the UART Rx/Tx signal. This bit is '1'. At this time, the value is output when all the acquired sample values are 1 or 0.

  NOTE: If the channel is disabled in the PMU_PER register after powering on the microcontroller, the SMS bit value is automatically set to '1' when the UART n channel is enabled.

- **DMS** : If DMS value is '1', Sampling of received data bit is performed 3 times. At this time, the most frequent value of each sample value is output.
- **RXINV** : If the RXINV bit is '1', the external RX input signal is inverted and input to the inside.
- **TXINV** : When the TXINV bit is '1', the internal TX output signal is inverted and output to the outside.
- **WAITVAL** : By entering a value from 0 to 7 in this WAITVAL field, the delay time between successive data outputs can be adjusted. At this time, the delay time (WT) can be obtained by the following formula.

NOTES:

1. Sysem Parameter 1(0x0020_0000~0x0020_01FF) area contains commands related to the UARTn_DTR register.
2. If this area is cleared, the UARTn_DTR initial value will be ‘0x80’,
3. if this area is not cleared, the UARTn_DTR initial value will be ‘0x00’.
4. It needs to be careful when using.

11.6.13 UART Registers Map Summary

Table 89. 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
0x00	UARTn_DLL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DLL[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x04	UARTn_DLM	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DLM[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	Res	Res	TEMTIE	RLSIE	THREIE	DRIE
	Reset value																													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	FIFO[1:0]		Res	XMITE	FID	IID[1:0]		IPEN
	Reset value																									0	0		0	0	0	0	1
0x08	UARTn_FCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FIFODEPTH [1:0]		Res	Res	Res	FCR2	FCR1	FIFOEN
	Reset value																									0	0				0	0	0
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	DLAB	BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]	
	Reset value																									0	0	0	0	0	0	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	FIFOE	TEMT	THRE	BI	FE	PE	OE	DR
	Reset value																									0	1	1	0	0	0	0	0
0x1C	UARTn_SCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SCR[7:0]							
	Reset value																									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
0x28	UARTn_DTR	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	RXINV	TXINV	Res	WAITVAL[2:0]		
	Reset value																									0	0	0	0		0	0	0

12. Serial Peripheral Interface (SPI)

12.1 SPI Introduction

The A33G53x 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 three-wire or 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.

12.2 SPI Main Features

The SPI of the A33G53x series features the followings:

- Selectable between the master and slave operations
- Full-duplex, three-wire and four-wire synchronous transfers supported
  - 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 bits
- Configurable interrupts triggered by the transmit status and SS signal
- Loop-back mode for internal checkups
- User-programmable start, burst, and stop delay times
- Selectable between MSB first transfer or LSB first transfer
- The transmission status check function
- Support boot program function using SPI0 channel when entering boot mode

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 SPIn_RDR register while subsequent data is being received, or by writing subsequent data to the SPIn_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 pin MOSI, the slave responds through the pin MISO.

The built-in SPI features are described in Table 90.

Table 90. SPI Implementation

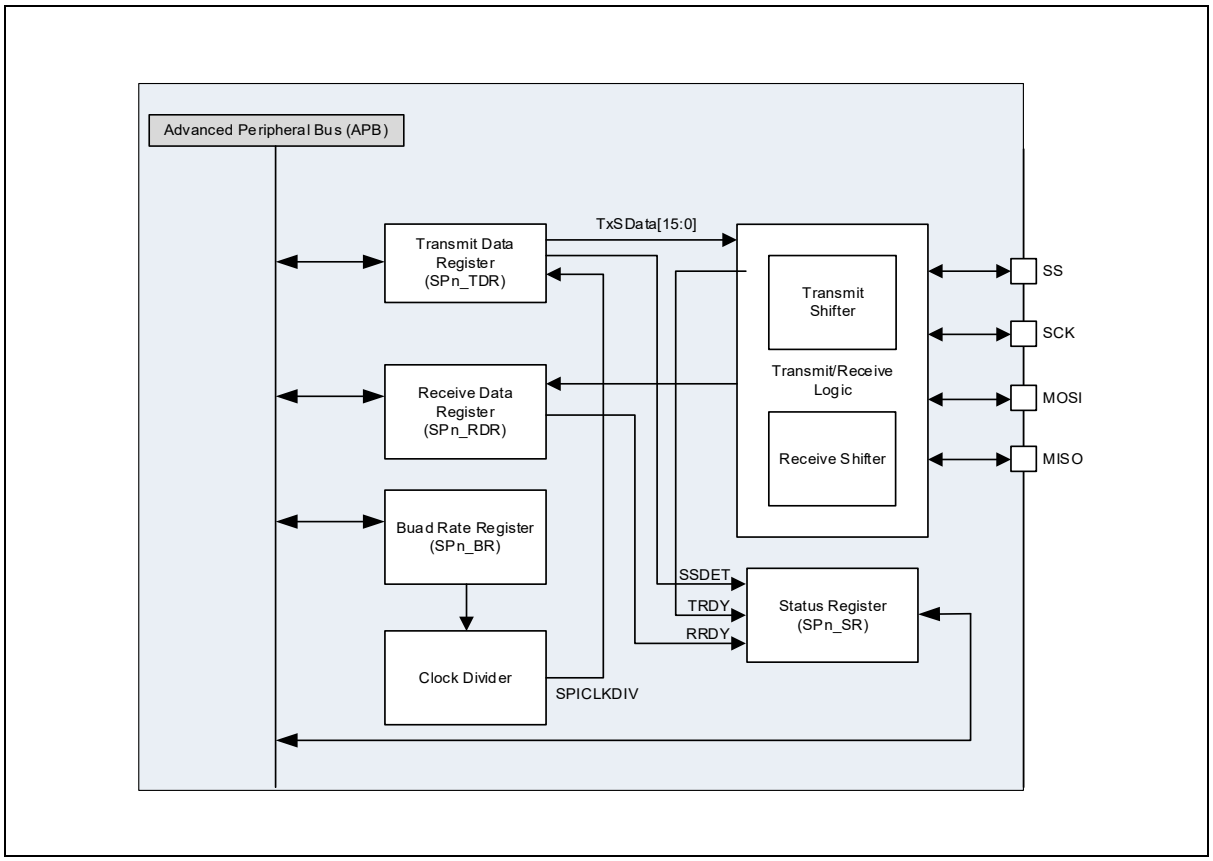
SPI Feature	Description	SPI0	SPI1
Data size	Data size selection for transmission and reception (8-bit, 9-bit, 16-bit, 17-bit)	O	O
MSB / LSB selection	Transfer format selection (MSB or LSB)	O	O
Error flag	Error flags (Overrun, Underrun)	O	O
Latency	Programmable latency (Start, Burst, Stop)	O	O

12.4 SPI Functional Description

12.4.1 SPI Block Diagram

Figure 83 shows the block diagram for the SPI of the A33G53x.

Figure 83. SPI Block Diagram



12.4.2 SPI Pins and Internal Signals

Table 91 describes the pins assigned for the SPI.

Table 91. Pin Assignment of SPI: External Pins

Pin name	Type	Description	Supported packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G539MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
SS0	I/O	Slave Select signal of SPI0	O	O	O	O
SCK0	I/O	Serial Clock signal of SPI0	O	O	O	O
MOSI0	I/O	MOSI Data signal of SPI0 (Master-Out Slave-In)	O	O	O	O
MISO0	I/O	MISO Data signal of SPI0 (Master-In Slave-Out)	O	O	O	O
SS1	I/O	Slave Select signal of SPI1	O	O	O	O
SCK1	I/O	Serial Clock signal of SPI1	O	O	O	O
MOSI1	I/O	MOSI Data signal of SPI1 (Master-Out Slave-In)	O	O	O	O
MISO1	I/O	MISO Data signal of SPI1 (Master-In Slave-Out)	O	O	O	O

12.4.3 Communications between One Master and One Slave

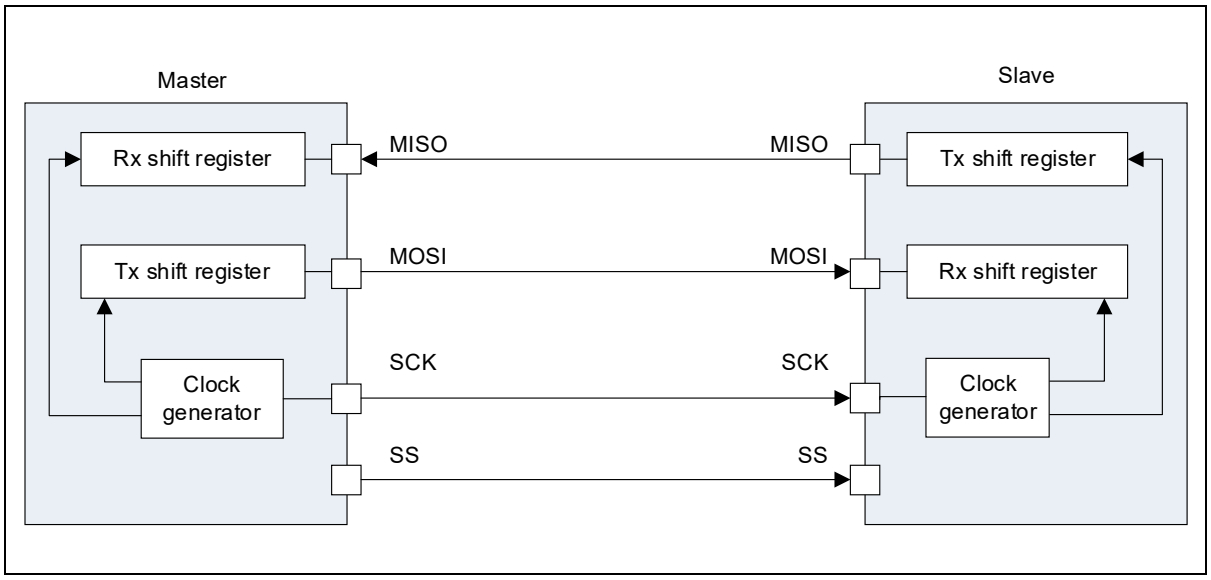
12.4.3.1 Full-Duplex Communication

Figure 84 shows a normal connections 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 84.

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 84. Block Diagram of SPI-Master to Slave Connections: Full-duplex



12.4.3.2 Simplex Communication

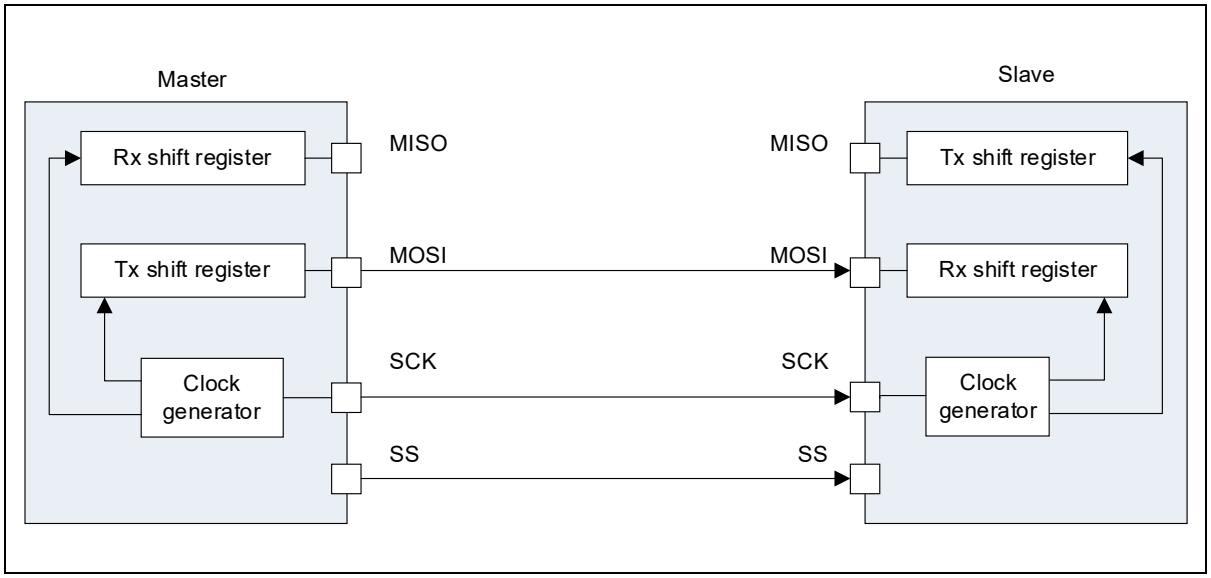
If only the Master's transmission and the Slave's reception are required, the Master and Slave need to be connected as shown in Figure 85.

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 85. Block Diagram of SPI-Master to Slave Connections: 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 SSMODE in SPIn_CR register.

If the SSMODE is set to ‘1’, the pin is driven internally according to the settings of the SSOUT in SPIn_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 SPIn_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 in 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 86. SPI Transfer Timing 1 of 4 (CPHA = '0', CPOL = '0', MSBF = '0')

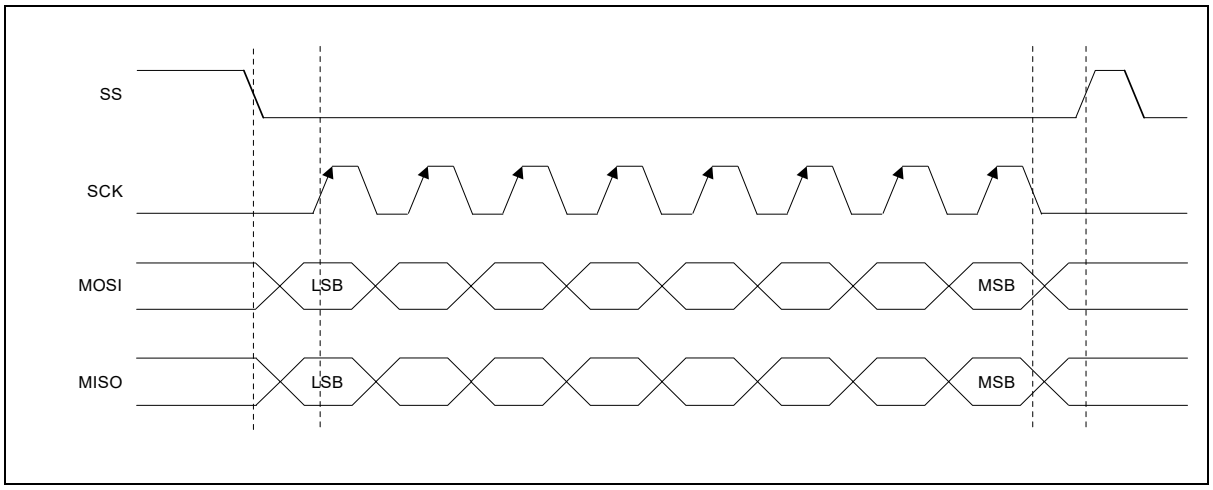


Figure 87. SPI Transfer Timing 2 of 4 (CPHA = '0', CPOL = '1', MSBF = '1')

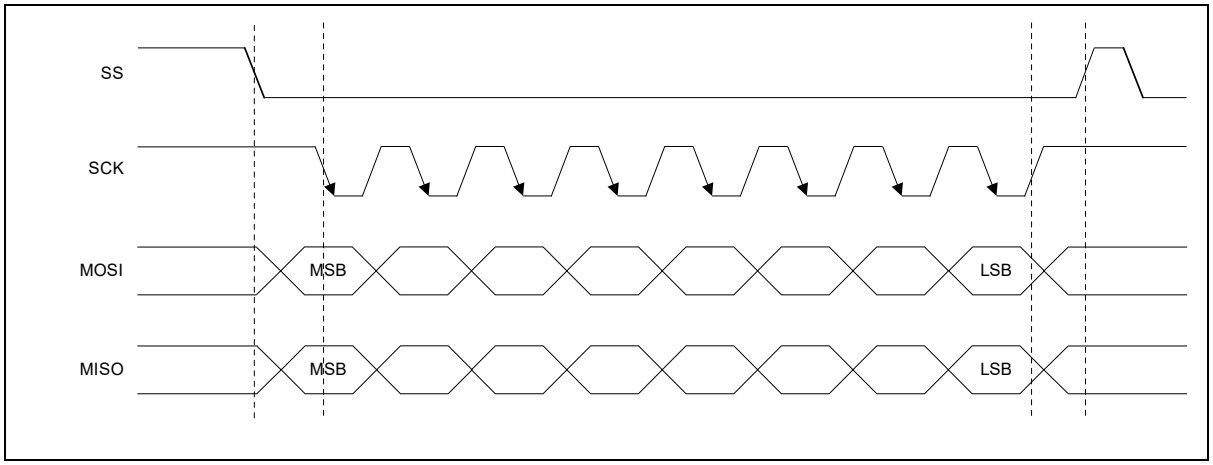


Figure 86 and Figure 87 illustrate SPI transfer timings when the CPHA is set to '0'.

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 88. SPI Transfer Timing 3 of 4 (CPHA = ‘1’, CPOL = ‘0’, MSBF = ‘0’)

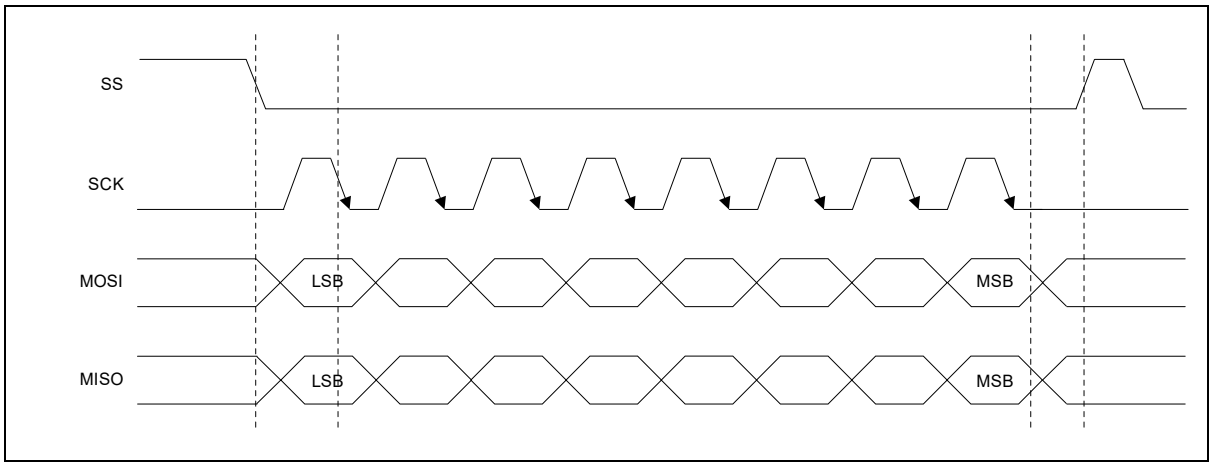


Figure 89. SPI Transfer Timing 4 of 4 (CPHA = ‘1’, CPOL = ‘1’, MSBF = ‘1’)

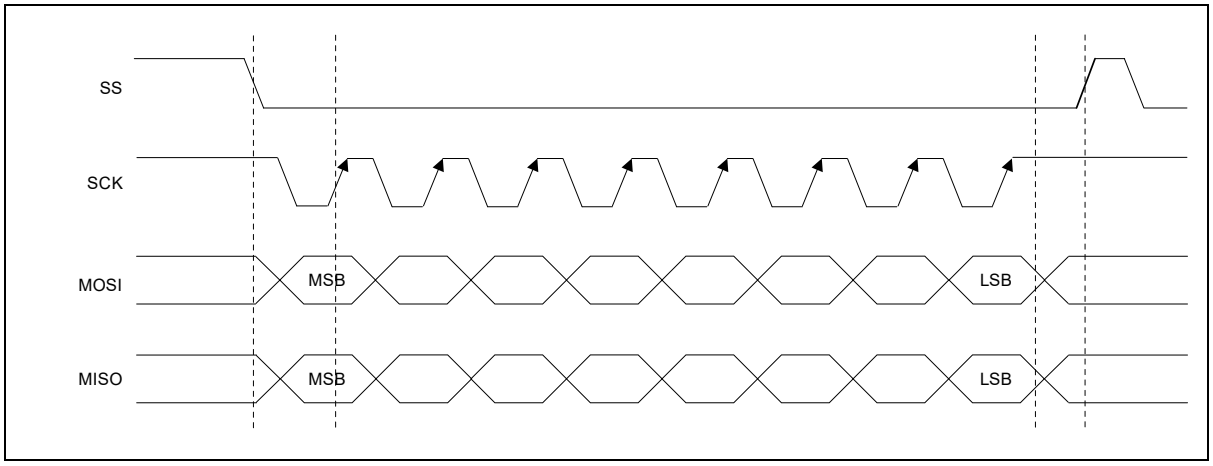


Figure 88 and Figure 89 illustrate SPI transfer timings when the CPHA is set to ‘1’.

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.

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 at bit position 0.
- When the MSB is transferred first, data is transferred starting at bit position 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:

- Set the GPIO pin to fit the alternative function. (MOSI, MISO, SCK, and SS pins)
- Enable the SPI related bits of the PMU_PER and SCU_PCCR registers.
- The SPIn_CR register setting method is as follows and can be set in any order.
- Configure a combination of the CPOL and CPHA bits.
- Enable the TXBC and RXBC bits of the SPIn_CR register to clear the Transmit Buffer and Receive Buffer, respectively.
- Set the MS in SPIn_CR register depending on whether the SPI is in Master mode or Slave mode.
- Set the MSBF in SPIn_CR register to determine whether to transmit the MSB first or the LSB first.
- Set the BITSEL[1:0] in SPIn_CR register to determine the data transfer size.
- In Master mode, set the SSMO in SPIn_CR register to ‘1’ to enable the SS output signal.
- Write the baud rate to the SPIn_BR register to determine the SPI communication rate.

When all of the above steps are complete, the basic initial setup is concluded. Additional setups such as SPI latency settings are discussed in the later sections in this chapter.

For more information on the data communication, see chapter 12.4.8 Data Transmission and Reception Procedures.

12.4.7 Procedure for Enabling SPI

The SPIn_EN register is used to enable or disable the SPI. When the ENABLE in 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.

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.

Tx transmission methods in Master 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. Set the TXIE in SPIn_CR register to ‘1’ to enable the Transmit Interrupt, when the transmission is required.
4. Check the TRDY in SPIn_SR register to know whether the Transmit Buffer is busy or ready, if using SPI Interrupt Handler.
   - If the Transmit Buffer is ready, write the transfer data to the SPIn_TDR register

Polling method

1. When the SPI initial setup is complete, set the ENABLE in SPIn_EN register to ‘1’.
2. Check the TRDY in SPIn_SR register to know whether the Transmit Buffer is busy or ready, when the transmission is required.
   - If the Transmit Buffer is ready, write the transfer data to the SPIn_TDR register.

Rx reception methods in Master 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. Set the RXIE in SPIn_CR register to ‘1’ to enable the Receive Interrupt, when the reception is required.
4. Check the TRDY in SPIn_SR register to know whether the Transmit Buffer is busy or ready, if using SPI Interrupt Handler.
   - If the Transmit Buffer is ready, write dummy data to the SPIn_TDR register
5. At the same time, check the RRDY in SPIn_SR register to confirm whether the Receive Buffer is empty or not, using the SPI Interrupt Handler.
   - If the Receive Buffer has data, the RRDY in SPIn_SR register is set to ‘1’ and the SPIn_RDR register is read.

Polling method

1. When the SPI initial setup is complete, set the ENABLE in SPIn_EN register to ‘1’.
2. Check the TRDY in SPIn_SR register to know whether the Transmit Buffer is busy or ready, when the reception is required.
   - If the Transmit Buffer is ready, write dummy data to the SPIn_TDR register
3. And then, check the RRDY in SPIn_SR register to confirm whether the Receive Buffer is empty or not.
   - If the Receive Buffer has data, the RRDY in SPIn_SR register is set to ‘1’ and the SPIn_RDR register is read.

Tx / Rx transfer 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. Since the Slave mode is determined by the Master device, the Slave device can transmit data at the same time it receives data.
   - To be ready for receiving data, set the RXIE in SPIn_CR register to ‘1’ to enable the Receive Interrupt.
4. Since the Slave device can transmit data at the same time it receives data, write the transfer data to the SPIn_TDR register before receiving data.
5. Check the RRDY in SPIn_SR register to confirm whether the Receive Buffer is empty or not, using the SPI Interrupt Handler.

Polling method

1. When the SPI initial setup is complete, set the ENABLE in SPIn_EN register to ‘1’.
2. Since the Slave device can transmit data at the same time it receives data, write the transfer data to the SPIn_TDR register before receiving data.
3. The Slave device waits for receiving data that the Master transmits. Check the RRDY in SPIn_SR register to confirm whether the Receive Buffer is empty or not.

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 in 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

Users can designate the SPI transmission / reception data size. By setting BITSEL[1:0] in SPIn_CR register, users can select 17, 16, 9, 8-bit.

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)

TThe 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'.

12.4.10 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 SPIn_RDR register is not yet read. Reading the SPIn_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 Setting Example

<Example 1> SPI0 Master Initial Setting Example

```
PB_MR<21:20> = “01”                 : Select the function of PB10 pin as SS0
PB_MR<23:22> = “01”                 : Select the function of PB11 pin as SCK0
PB_MR<25:24> = “01”                 : Select the function of PB12 pin as MOSI0
PB_MR<27:26> = “01”                 : Select the function of PB13 as MISO0
PB_CR<21:20> = “00”                 : Set the direction of PB10 (SS0) pin as push-pull output
PB_CR<23:22> = “00”                 : Set the direction of PB11 (SCK0) pin as push-pull output
PB_CR<25:24> = “00”                 : Set the direction of PB12 (MOSI0) pin as push-pull output
PB_CR<27:26> = “10”                 : Set the direction of PB13 (MISO0) pin as logic input
PB_PCR<10> = “0”                    : Disable pull-up/pull-down resistor of PB10 (SS0) pin
PB_PCR<11> = “0”                    : Disable pull-up/pull-down resistor of PB11 (SCK0) pin
PB_PCR<12> = “0”                    : Disable pull-up/pull-down resistor of PB12 (MOSI0) pin
PB_PCR<13> = “0”                    : Disable pull-up/pull-down resistor of PB13 (MISO0) pin

SPI0_CR<BITSEL[1:0]> = “00”         : Select the size of transmit/receive data as 8-bit
SPI0_CR<MSBF> = “1”                 : Select the SPI0 serial data type as MSB-first
SPI0_CR<MS> =”1”                    : Select SPI0 channel as master mode
SPI0_CR<SSPOL> = “1”                : Select the start clock polarity on the SS0 pin
SPI0_CR<CPHA> = “1”                 : Set the clock phase to output data at the start phase
SPI0_CR<CPOL> = “1”                 : Set the start clock of SPI0 serial for data sample to high
SPI0_BR<BR[7:0]> = “0100 1111”      : Select SPI0 transfer rate as 1 MHz
SPI0_EN<ENABLE> = “1”               : Reset the transmit buffer when transmitting the first data.
```

<Example 2> SPI1 Slave Initial Setting Example

PD_MR<17:16> = “01”	: Select the function of PD8 pin as SS1
PD_MR<19:18> = “01”	: Select the function of PD9 pin as SCK1
PD_MR<21:20> = “01”	: Select the function of PD10 pin as MOSI1
PD_MR<23:26> = “01”	: Select the function of PD11 pin as MISO1
PD_CR<17:16> > = “10”	: Select the direction of PD8 (SS1) pin as logic input
PD_CR<19:18> = “10”	: Select the direction of PD9 (SCK1) pin as logic input
PD_CR<21:20> = “10”	: Select the direction of PD10 (MOSI1) pin as logic input
PD_CR<23:26> > = “00”	: Select the direction of PD11 (MISO1) pin as logic input
PD_PCR<8> = “0”	: Disable pull-up/pull-down resistor of PD8 (SS1) pin
PD_PCR<9> = “0”	: Disable pull-up/pull-down resistor of PD9 (SCK1) pin
PD_PCR<10> = “0”	: Disable pull-up/pull-down resistor of PD10 (MOSI1) pin
PD_PCR<11> = “0”	: Disable pull-up/pull-down resistor of PD11 (MISO1) pin
SPI1_CR<BITSEL[1:0]> = “00”	: Select the size of transmit/receive data as 8-bit
SPI1_CR<MSBF> = “1”	: Select the SPI0 serial data type as MSB-first
SPI1_CR<MS> =”0”	: Select SPI0 channel as master mode
SPI1_CR<SSPOL> = “1”	: Select the start clock polarity of SS1 pin as 'H'.
SPI1_CR<CPHA> = “1”	: Set the clock phase to output data at the start phase
SPI1_CR<CPOL> = “1”	: Set the start clock of SPI1 serial for data sample to high
SPI1_BR<BR[7:0]> = “0100 1111”	: Select SPI1 transfer rate as 1 MHz
SPI1_EN<ENABLE> = “1”	: Reset the transmit buffer when transmitting the first data.

<Example 3> Data Output of SPI0 Master Example

SPI0_CR<SSOUT> = “0”	: Select the start clock polarity of SS1 pin as 'L'.
SPI0_SR<TRDY> READ	: Read SPI0 transmit/receive buffer - Check if transfer is ready
SPI0_EN<ENABLE> =”0”	: Disable SPI0 to insert transfer data into the SPI0 buffer
SPI0_TDR<16:0> = “VALUE”	: Input transfer value to SPI0 transmit data register.
SPI0_En<ENABLE> = “1”	: Enable SPI0 master channel and start to output data
SPI0_SR<TRDY> READ	: Read SPI0 transmit/receive buffer - Check if receiving is ready
SPIn_CR<SSOUT> = “1”	: Select the start clock polarity of SS0 pin as 'H'.
SPI0_EN<ENABLE> =”0”	:. Disable SPI0 master channel

<Example 4> Data Output of SPI1 Slave Example

SPI1_En<ENABLE> = “1”	: Enable SPI1 Slave channel
SPI1_SR<RRDY> READ	: Check whether the SPI1 channel is receiving data
SPI1_RDR<RDR> READ	: Read the value of the SPI1 channel's receive data register

12.6 SPI Registers

The base addresses and register map of the SPI are as follows:

Table 92. Base Addresses of SPI

Name	Base address
SPI0	0x4000_0800
SPI1	0x4000_0820

Table 93. SPI Register Map

Name	Offset	Type	Description	Reset value	Reference
SPIn_TDR	0x0000	WO	SPI n Transmit Data Buffer Register	-	12.6.1
SPIn_RDR	0x0000	RO	SPI n Receive Data Buffer Register	0x0000_0000	12.6.2
SPIn_CR	0x0004	RW	SPI n Control Register	0x0000_0820	12.6.3
SPIn_SR	0x0008	RC	SPI n Statue Register	0x0000_0002	12.6.4
SPIn_BR	0x000C	RW	SPI n Baudrate Register	0x0000_00FF	12.6.5
SPIn_EN	0x0010	RW	SPI n Enable Register	0x0000_0000	12.6.6
SPIn_LR	0x0014	RW	SPI n Timing Register	0x0000_1866	12.6.7

NOTE: n = 0 and 1.

12.6.1 SPIn_TDR: SPI n Transmit Data Buffer Register

The SPIn_TDR register is a 17-bit sized read / write register. It contains serial transmit data.

SPI0_TDR=0x4000_0800, SPI1_TDR=0x4000_0820

31 30 29 28 27 26 25 24	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]	
-		-	
-		WO	

16 0	TDR[16:0]	Transmit Data Value

12.6.2 SPIn_RDR: SPI n Receive Data Buffer Register

The SPIn_RDR register is a 17-bit sized read / write register. It contains serial receive data.

SPI0_RDR=0x4000_0800, SPI1_RDR=0x4000_0820

31 30 29 28 27 26 25 24	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]	
-		0x00000	
-		RO	

16 0	RDR[16:0]	Receive Data Value

12.6.3 SPIn_CR: SPI n Control Register

The SPIn_CR register is a 20-bit sized read / write register and can configure SPI operation mode.

SPI0_CR=0x4000_0804, SPI1_CR=0x4000_0824

31 30 29 28 27 26 25 24	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	Reserved	SSCIE	TXIE	RXIE	SSMODE	SSOUT	LBE	SSMARK	SSMO	SSPOL	Reserved	MS	MSBF	CPHA	CPOL	BITSEL[1:0]
-		0	0	-	0	0	0	0	1	0	0	0	0	-	1	0	0	0	00
-		WO	WO	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW

20	TXBC	Tx buffer clear
		0 No operation
		1 TX BUFFER empty
19	RXBC	Rx buffer clear
		0 No action
		1 Clear Rx buffer
15	SSCIE	Select the edge interrupt of SS signal
		0 Disable SS interrupt
		1 Enable the interrupt on the both edge (L→H, H→L) of SS signal
14	TXIE	Select Tx data interrupt
		0 Disable Tx data interrupt
		1 Enable Tx data interrupt
13	RXIE	Select Rx data interrupt
		0 Disable Rx data interrupt
		1 Disable Rx data interrupt
12	SSMODE	SS automatic or manual control of SS output.
		0 Automatic control of SS output
		1 Manual control of SS output using SSOUT bit
11	SSOUT	Select SS output level
		0 SS output 'L'
		1 SS output 'H'
10	LBE	Select the Loop-back mode in Master mode
		0 Normal communication mode
		1 Loop-back mode in Master mode
9	SSMASK	Select Ignorance of the SS input
		0 Enable Rx only when the SS signal is active (Masked SPI SCK)
		1 Ignore SS input (Always Rx enabled)
8	SSMO	Select SS signal output
		0 Disable the output of SS signal (SSMODE and SSOUT can not be output)
		1 Enable the output of SS signal
7	SSPOL	Select the polarity of SS signal.
		0 SS signal is Active-Low.
		1 SS signal is Active-High.
6	TEST	Do not use (This bit is used only for internal testing purposes.)
5	MS	Master / Slave select bit.
		0 SPI is in Slave mode.
		1 SPI is in Master mode.
4	MSBF	MSB / LSB Transmit select bit.

		0	LSB is transferred first.
		1	MSB is transferred first.
3	CPHA	SPI Clock Phase bit.	
		0	Output data at start phase
		1	Output data at next phase
2	CPOL	SPI Clock Polarity bit.	
		0	Start clock at ‘L’
		1	Start clock at ‘H’
1 0	BITSEL[1:0]	Transmit / Receive Data Bits select bit.	
		00	8 bits
		01	9 bits
		10	16 bits
		11	17 bits

NOTES:

1. CPOL = ‘0’, CPHA = ‘0’ : data sampling at rising edge, data changing at falling edge
2. CPOL = ‘0’, CPHA = ‘1’ : data sampling at falling edge, data changing at rising edge
3. CPOL = ‘1’, CPHA = ‘0’ : data sampling at falling edge, data changing at rising edge
4. CPOL = ‘1’, CPHA = ‘1’ : data sampling at rising edge, data changing at falling edge

12.6.4 SPIn_SR: SPI n Status Register

The SPIn_SR register is a 6-bit sized read / write register. It contains the status of SPI interface.

SPI0_SR=0x4000_0808, SPI1_SR=0x4000_0828

31 30 29 28 27 26 25 24 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		SSDET	SSON	OVRF	UDRF	SBUSY	TRDY	RRDY
-		0	0	0	0	0	1	0
-		RW C0	RC	RO	RO	RO	RO	RO

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 SPIn_RDR.
3	UDRF	Transmit Underrun Error flag.	
		0	Transmit Underrun is not occurred.
		1	Transmit Underrun is occurred. This bit is cleared by writing SPIn_TDR.
2	SBUSY	Busy (Transmit or receive) flag	
		0	Idle status
		1	Busy (Transmit or receive in progress) status
1	TRDY	Transmit Buffer Empty flag.	
		0	Transmit Buffer is busy.
		1	Transmit Buffer is ready. This bit is cleared by writing data to SPIn_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 SPIn_RDR.

NOTE: It is not able to know when the electrical signal of the data being transmitted will be final output.

12.6.5 SPIn_BR: SPI n Baud-Rate Register

The SPIn_BR register is a 16-bit sized read / write register. Baud rate can be set by writing the register.

SPI0_BR=0x4000_080C, SPI1_BR=0x4000_082C

31 30 29 28 27 26 25 24 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[7:0]
-	0xFF
-	RW

7 0	BR[7:0]	SPI Data transfer rate of SCK. The BR value must be more than 1. $Baud\ Rate = \frac{PCLK}{BR+1}$ (However BR ≥ 1)

12.6.6 SPIn_EN: SPI n Enable Register

he SPIn_EN register is a bit sized read / write register. It contains SPI enable bit.

SPI0_EN=0x4000_0810, SPI1_EN=0x4000_0830

31 30 29 28 27 26 25 24 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 SPIn_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 SPIn_TDR register before this bit is active.

NOTE:

When '1' is written to this bit, the data in the SPI transmit buffer is shifted to the transmit shifter. (When sending the first data, the ENABLE bit must be initialized.)

If the ENABLE bit in the SPIn_EN register is set to '1', the value set to '1' in the UDRF, OVRF, and SSDET bits in the SPIn_SR register is initialized to '0' by setting the ENABLE bit in the SPIn_EN register to '0'.

1) When the 3rd bit of SPIn_SR, UDRF, is set to '1' and a transmit underrun error occurs.

2) When OVRF, the fourth bit of SPIn_SR, is set to '1' and a receive overrun error occurs

3) When 7th bit of SPIn_SR, SSDET, is set to '1' and SS negate is detected.

The values of the remaining bits except the three flag bits are retained.

12.6.7 SPIn_LR: SPI n Timing Register

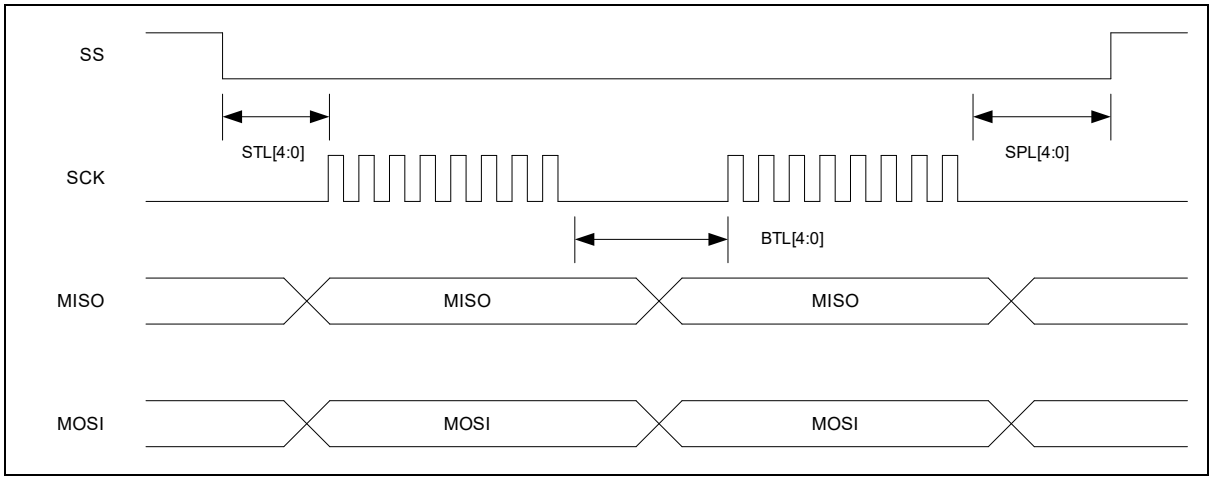
The SPIn_LR register sets the delay timing of the SPI signal. It operates based on the SCK clock set in the SPIn_BR register.

SPI0_LR=0x4000_0814, SPI1_LR=0x4000_0834

31 30 29 28 27 26 25 24	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[4:0]	BTL[4:0]	STL[4:0]
-		00110	00011	00110
-		RW	RW	RW

14 10	SPL[4:0]	Set Stop delay time
		Stop delay time = SPL[4:0] × (1/SCK)
9 5	BTL[4:0]	Set the packet interval for continuous output
		Burst delay time = BTL[4:0] × (1/SCK)
4 0	STL[4:0]	Set Start delay time
		Start delay time = STL[4:0] × (1/SCK)

Figure 90. SPI Waveforms (STL[4:0], BTL[4:0], and SPL[4:0])



12.6.8 SPI Registers Map Summary

Table 94. 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	Res.	Res.	Res.	SSCIE	TXIE	RXIE	SSMODE	SSOUT	LBE	SSMASK	SSMO	SSPOL	TEST	MS	MSBF	CPHA	CPOL	BITSEL[1:0]	
	Reset value												0	0				0	0	0	0	1	0	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.	Res.	Res.	Res.	SSDET	SSON	OVRF	UDRF	SBUSY	TRDY	RRDY
	Reset value																										0	0	0	0	0	1	0
0x0C	SPIn_BR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR[7:0]							
	Reset value																									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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPL[4:0]					BTL[4:0]			STL[4:0]						
	Reset value																		0	0	1	1	0	0	0	0	1	1	0	0	1	1	0

13. Inter-Integrated Circuit (I2C)

13.1 I2C Introduction

The I2C (Inter-integrated Circuit) interface built in A33G53x series satisfies the standard I2C communication protocol, and is used for the 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 A33G53x series 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 A33G53x series features the followings:

- Compliant with I2C protocol
  - Supports two units.
- 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 implemented in the A33G53x series have all features of an I2C peripheral as shown in Table 95.

Table 95. Features of I2Cn (n = 0 to 1)

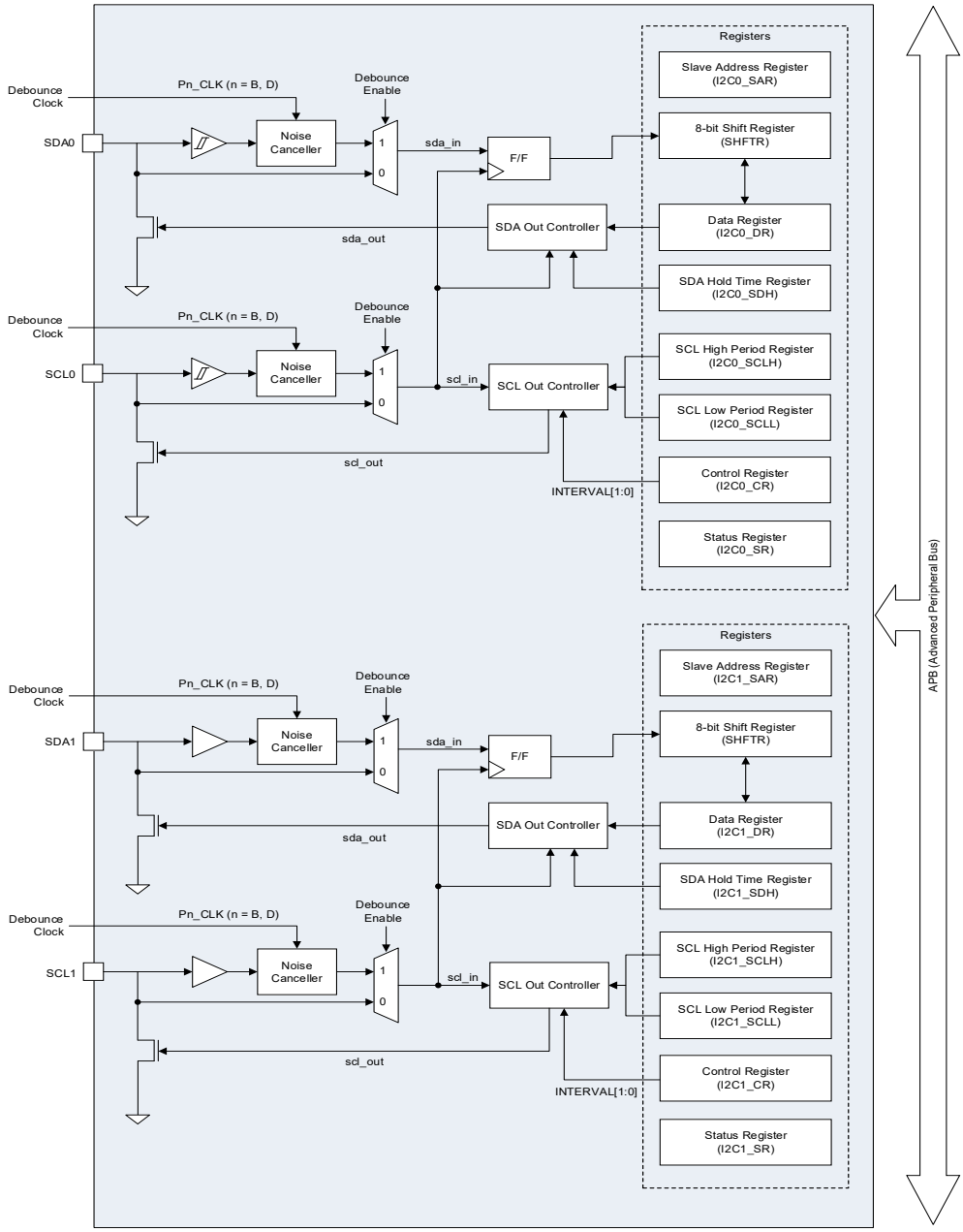
I2C Features	I2C0	I2C1
7-bit addressing mode	O	O
Standard-mode (up to 100 kbit/s)	O	O
Fast-mode (up to 400 kbit/s)	O	O
General call	O	O

13.4 I2C Functional Description

13.4.1 I2C Block Diagram

Figure 91 shows a block diagram of the I2C module.

Figure 91. I2C Block Diagram



NOTES:

1. SDA0, SCL0 of I2C0 channel is Schmitt-trigger input type.
2. SDA1, SCL1 of I2C1 channel is CMOS input type.
3. For more information on the Clock Source used by the Noise Canceler, refer to Table 35. The Examples of Debounce Filter Windows

13.4.2 I2C Pins and Signals

Table 96 describes pins assigned for the I2C interface.

The I2C (Inter Integrated Circuit) has two signals: SCL and SDA. For communication, these signals must be set to the internal or external pull-up state of the open-drain. However, if only internal pull-up is connected without external pull-up, the communication speed may be limited due to high resistance value.

Table 96. Pin Assignment of I2C: External Pins

Pin name	Type	Description	Supported packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G53MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
SCL0	I/O	I2C channel 0 serial clock bus line (Open-drain)	O	O	O	O
SDA0	I/O	I2C channel 0 serial data bus line (Open-drain)	O	O	O	O
SCL1	I/O	I2C channel 1 serial clock bus line (Open-drain)	O	O	O	O
SDA1	I/O	I2C channel 1 serial data bus line (Open-drain)	O	O	O	O

NOTES:

1. SDA0, SCL0 of I2C0 channel is Schmitt-trigger input type.
2. SDA1, SCL1 of I2C1 channel is CMOS input type.

13.4.3 I2C Mode Selection

The I2C module of the A33G53x series 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 in I2Cn_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 I2C0 or I2C1 in PMU_PER register to '1'. This enables the I2C module.
2. Set the I2C0 or I2C1 in PMU_PCCR register to '1'. This enables the I2C module clock.
3. Configure the I2Cn_CR register to control the I2C communication.

13.4.4.2 Noise Canceller (I2C Port Debounce)

I2C is one of the standard serial communication protocols, which is widely used in industry. The I2C uses 2 bus lines such as Serial Data Line (SDAn) and Serial Clock Line (SCLn) to exchange data.

I2C features the followings (n = 0 and 1):

Table 97. Pin Assignment of I2C: External Pins

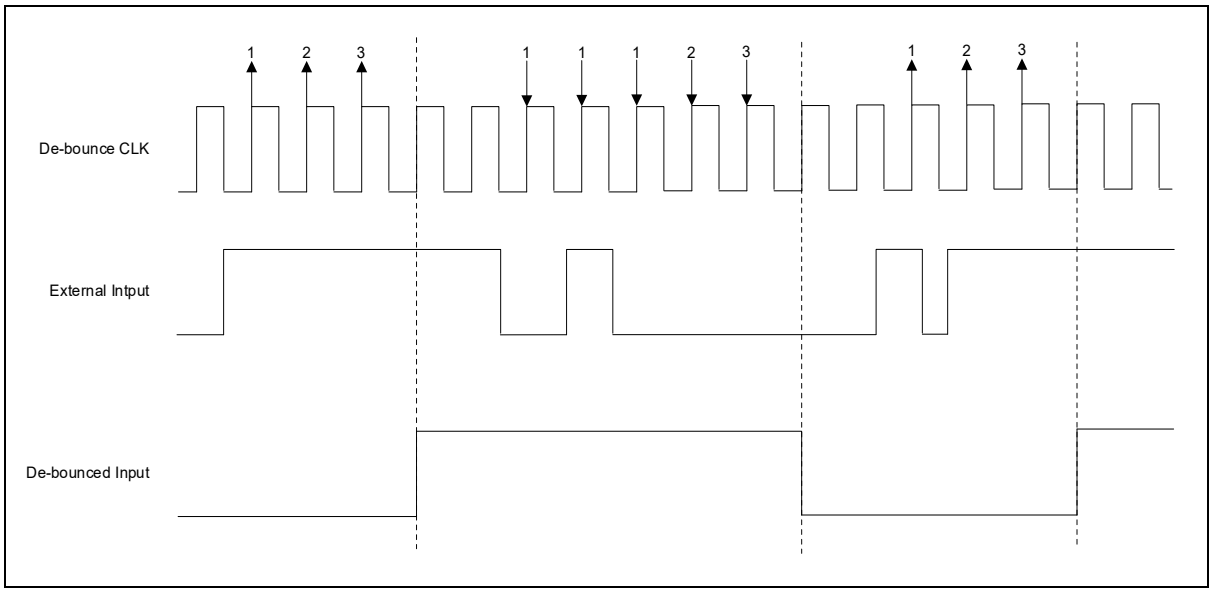
Channel	Pin name		Description
I2C0	PB14	SCL0	Output or Schmitt-trigger input
	PB15	SDA0	Output or Schmitt-trigger input
I2C1	PD14	SCL1	Output or CMOS input
	PD15	SDA1	Output or CMOS input

For the input pins of SCL1, SDA1 CMOS type, it is recommended to debounce. The debounce settings of I2C1 channel is described in Figure 92.

Each of SCD1 pin and SCA1 pin can use the debounce to prevent reset malfunction which is due to noise. This is configured in registers PD_DER (0x4000_026C) and the PD_DPR (0x4000_027C), and must be applied to the pin ports of SCL1 and SDA1.

First, a user needs to set the debounce clock (prescaler) and enable the debounce filter by configuring the PD_DPR. When the filter is enabled, input signals in less than 4 clocks is considered as noise and ignored.

Figure 92. Noise Canceller Timing (Debounce)



Assuming that input time of the PD_DPR is 16us and PCLK = 20 MHz, the debounce filtering time is calculated as shown below (f = debounce clock frequency, T = debounce clock period).

$4 \times T \geq 16\ \mu s$ $T \geq 4\ \mu s$ $\frac{1}{f} \geq 4\ \mu s$ $f = \frac{PCLK}{2^x} \leq \frac{1}{4\ \mu s} = \frac{1}{4}\mu s$ $2^x \geq \frac{PCLK}{\frac{1}{4}MHz} = \frac{20\ MHz}{\frac{1}{4}\ MHz} = 80$ $x \geq 7$	For Example, $f = \frac{PCLK}{2^7} = \frac{20MHz}{2^7} = \frac{20}{128}\ \mu s$ $T = \frac{128}{20}\ \mu s$ $\text{Filtering time} = 4T = 4 \times \frac{128}{20}\ \mu s = 25.6$

After the calculation, the result is ‘7’. Add the value to the PD_DPR to get 25.6 µs as the filtering time. For detailed information and relationship between the PD_DPR, PCLK, and debounce clock, please refer to spec. Table 35.

Example code shown below can be added during initialization process to complete software debounce processing for a PD pin.

- DPR in PD_DPR register is ‘7’.
- P14 in PD_DER register is ‘1’.

NOTE: 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] in I2Cn_SCLH register to the value less than the current setting.

13.4.5 I2C Software Reset

The I2C can be reset by software.

- Writing ‘1’ to the SOFTRESET in I2C0_CR or I2C1_CR register resets all related registers to the initialization state and SOFTRESET is auto clear.
- Users can reset the I2C by configuring the PMU_PER register. Writing ‘0’ to the I2C0 or I2C1 in PMU_PER register resets all related registers to the initialization state. To use the I2C module again, set the I2C0 or I2C1 in PMU_PER register to ‘1’ again.

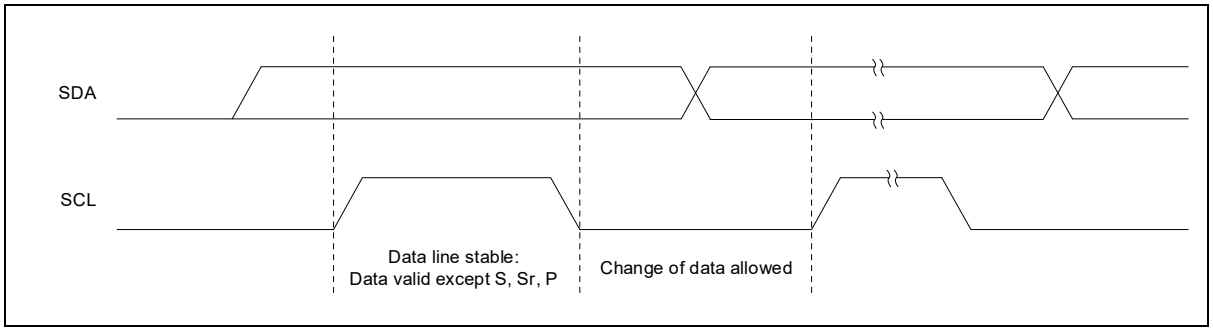
13.4.6 I2C Module Protocol

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 93 shows a timing diagram of the I2C Bus Bit Transfer.

Figure 93. I2C Bus Bit Transfer Timing Diagram



13.4.6.2 START / Repeated START / STOP

During the operation of the I2C bus, unique situations arise, which are defined as the START (S) and STOP (P) conditions. (See Figure 94. START and STOP Conditions):

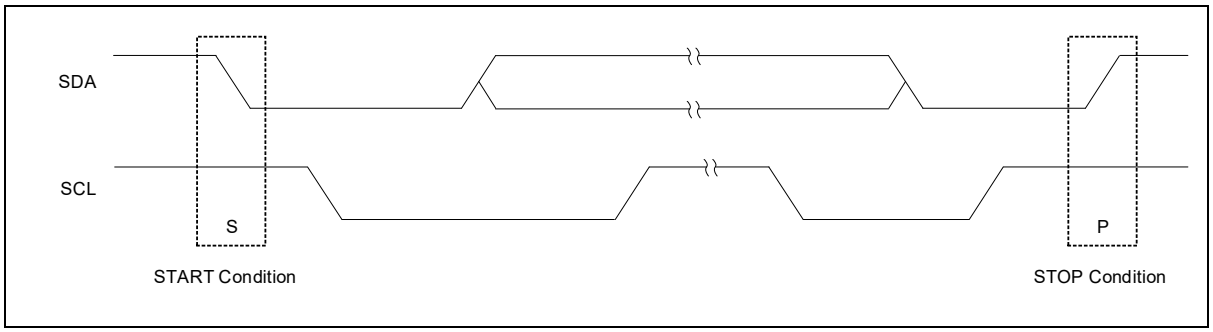
- START (S) condition: An “H” to “L” transition on the SDA line while SCL is “H”
- STOP (P) condition: An “L” to “H” transition on the SDA line while SCL is “H”

These START and STOP conditions are always generated by the master. The bus is considered as 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 94. START and STOP Conditions



13.4.6.3 Acknowledge

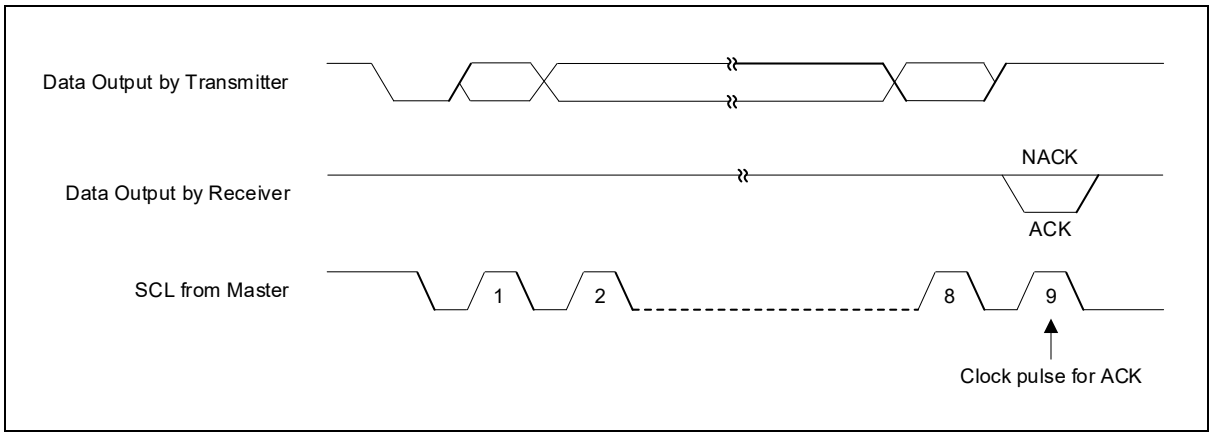
A data transfer with acknowledgement is necessary. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (“H”) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains at stable “L” during the “H” period of this clock pulse. (See Figure 95. I2C Bus Response)

Set-up and hold times must also be considered. When a slave doesn’t acknowledge the slave address (For example, it’s unable to receive or transmit because it’s performing some real-time function), the data line must be left “H” by the slave. 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 be generate the STOP or repeated START condition.

Figure 95. I2C Bus Response

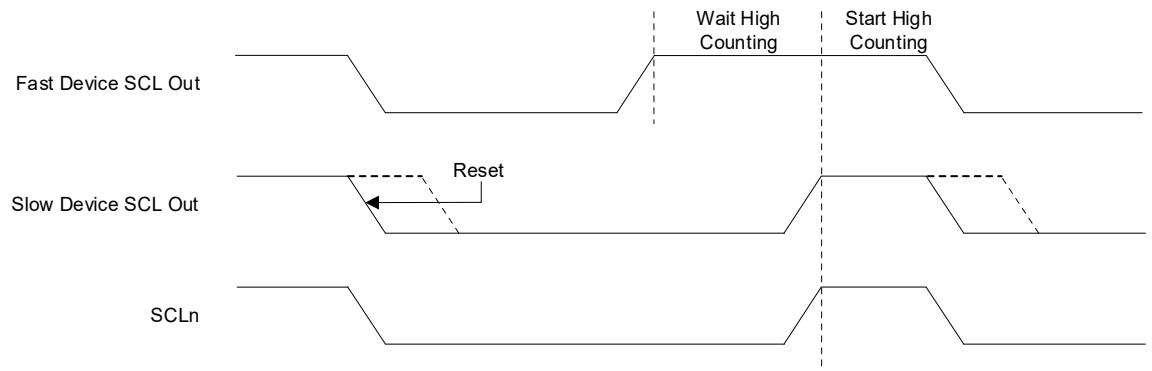


13.4.6.4 Synchronization

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 96. Clock Synchronization during Arbitration



NOTE: However, the “L” to “H” transition of this clock may not change the state of the SCL line if another clock is still within its “L” by the device with the longest “L” period. Devices with shorter “L” periods enter an “H” wait-state during this time.

When all devices 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.

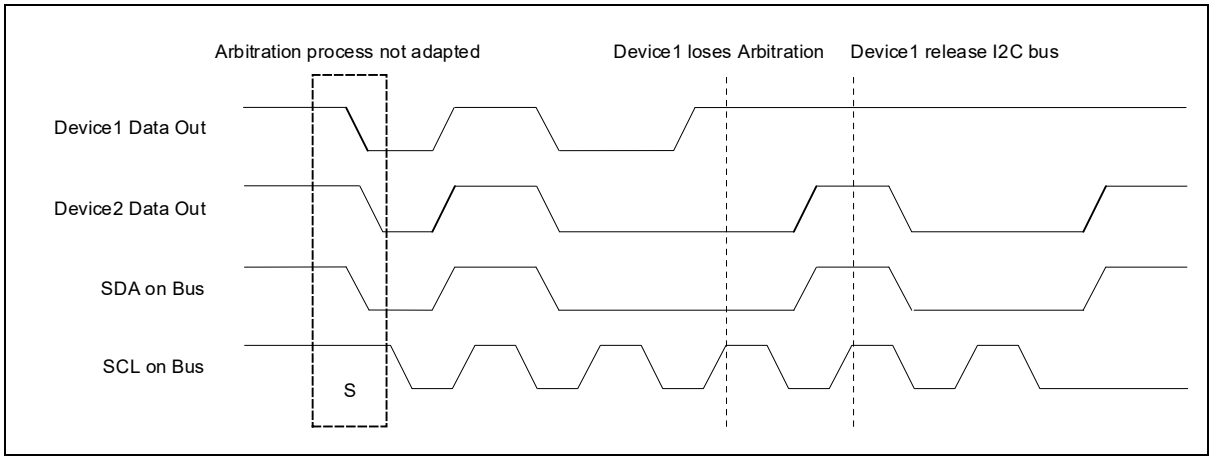
Arbitration can continue for many bits. Its first stage is the comparison of the address bits. If the masters are trying to address the same device, arbitration proceeds with comparing either the data-bits (If they are master-transmitters) or the acknowledge-bits (If they are master-receivers).

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 97 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 DataOut and the actual level on the SDA line, its data output is switched off, which means that an Device1 release I2C bus. This will not affect the data transfer initiated by the winning master.

Figure 97. Arbitration Process between Two Masters



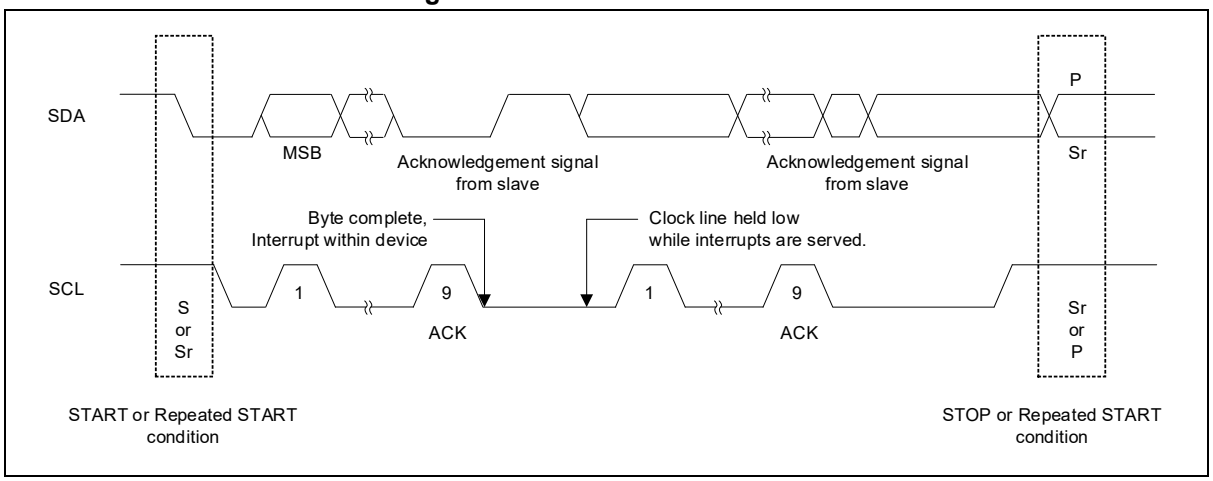
13.4.6.6 Data Transfer

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 98. I2C Bus Data Transfer)

If a slave can't 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 98. I2C Bus Data Transfer

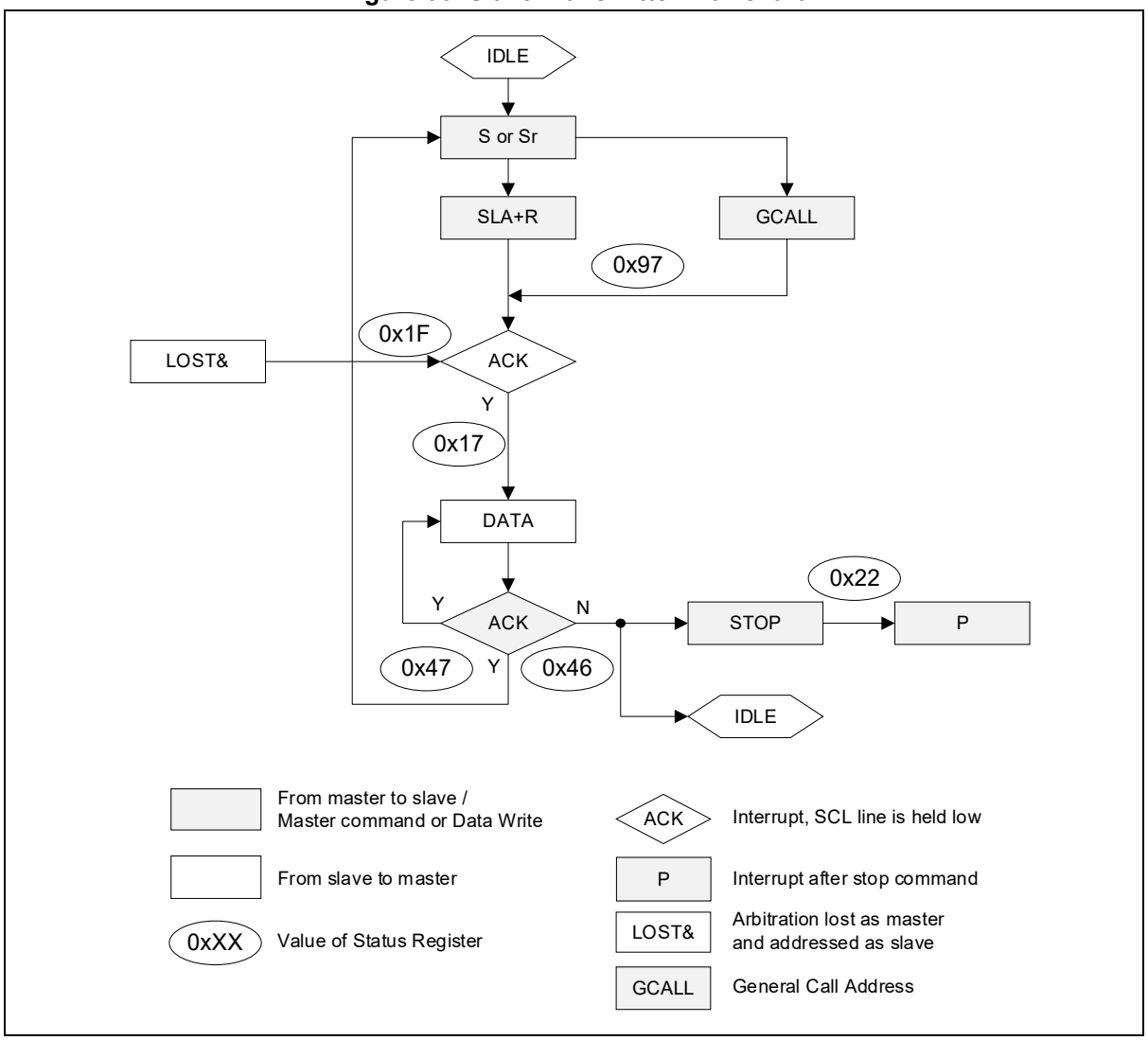


13.4.7 I2C Slave Mode

13.4.7.1 Slave Transmitter

Figure 99 shows a flowchart of the transmitter in slave mode.

Figure 99. Slave Transmitter Flowchart



To operate the I2C in slave transmitter mode, users must follow the steps below:

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, I2C (slave) cannot transmit serial data properly.
2. Enable I2C by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.

3. When a START condition is detected, I2C receives one byte of data and compares it with SLA bits in I2CSAR. If the GCALLEN bit in I2CSAR is enabled, I2C compares the received data with value 0x00, the general call address.

4. If the received address does not equal to SLA bits in I2CSAR, I2C enters idle state, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, I2C generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, I2C enters idle state. When SSEL interrupt occurs, load transmit data to I2CDR and write arbitrary value to I2CSR to release SCL line.

5. Byte of data is being transmitted.

6. In this step, I2C generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   Case 1. No ACK signal is detected and I2C waits STOP or repeated START condition.

   Case 2. ACK signal from master is detected. Load data to transmit into I2CDR.

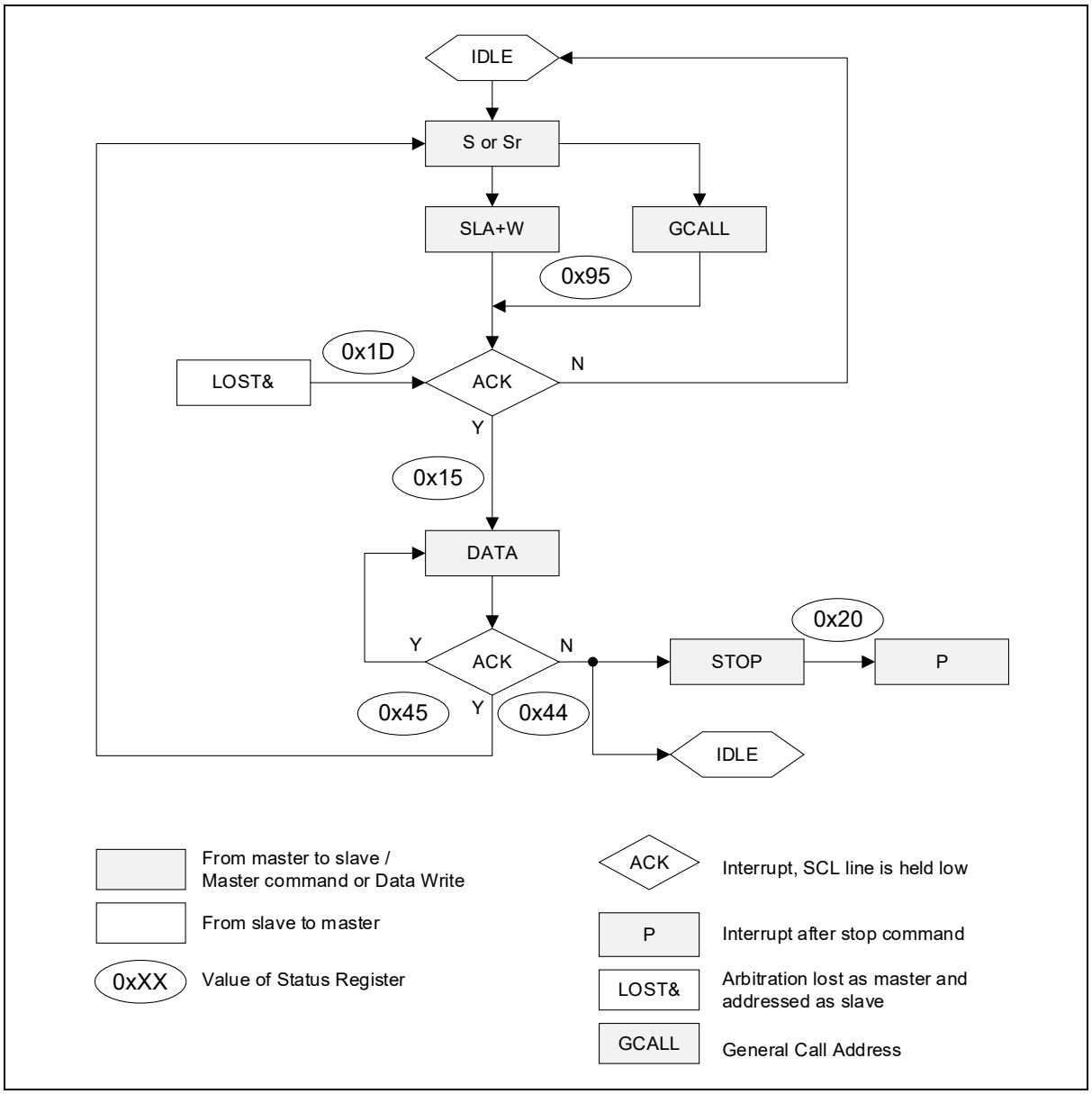
   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.

7. This is the final step for slave transmitter function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I2C enters idle state.

13.4.7.2 Slave receiver

Figure 100 shows a flowchart of the receiver in slave mode.

Figure 100. Slave Receiver Flowchart



To operate the I2C in slave receiver mode, users must follow the steps below:

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, I2C (slave) cannot transmit serial data properly.
2. Enable I2C by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.
3. When a START condition is detected, I2C receives one byte of data and compares it with SVAD in I2C_SAR. If the GCALLEN in I2C_SAR register is enabled, I2C compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLA bits in I2C_SAR, I2C enters idle state, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, I2C generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, I2C enters idle state. When SSEL interrupt occurs and I2C is ready to receive data, write arbitrary value to I2C_SR to release SCL line.
5. Byte of data is being received.
6. In this step, I2C generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   Case 1. No ACK signal is detected (ACKEN=0) and I2C waits STOP or repeated START condition.

   Case 2. ACK signal is detected (ACKEN=1) and I2C can continue to receive data from master.

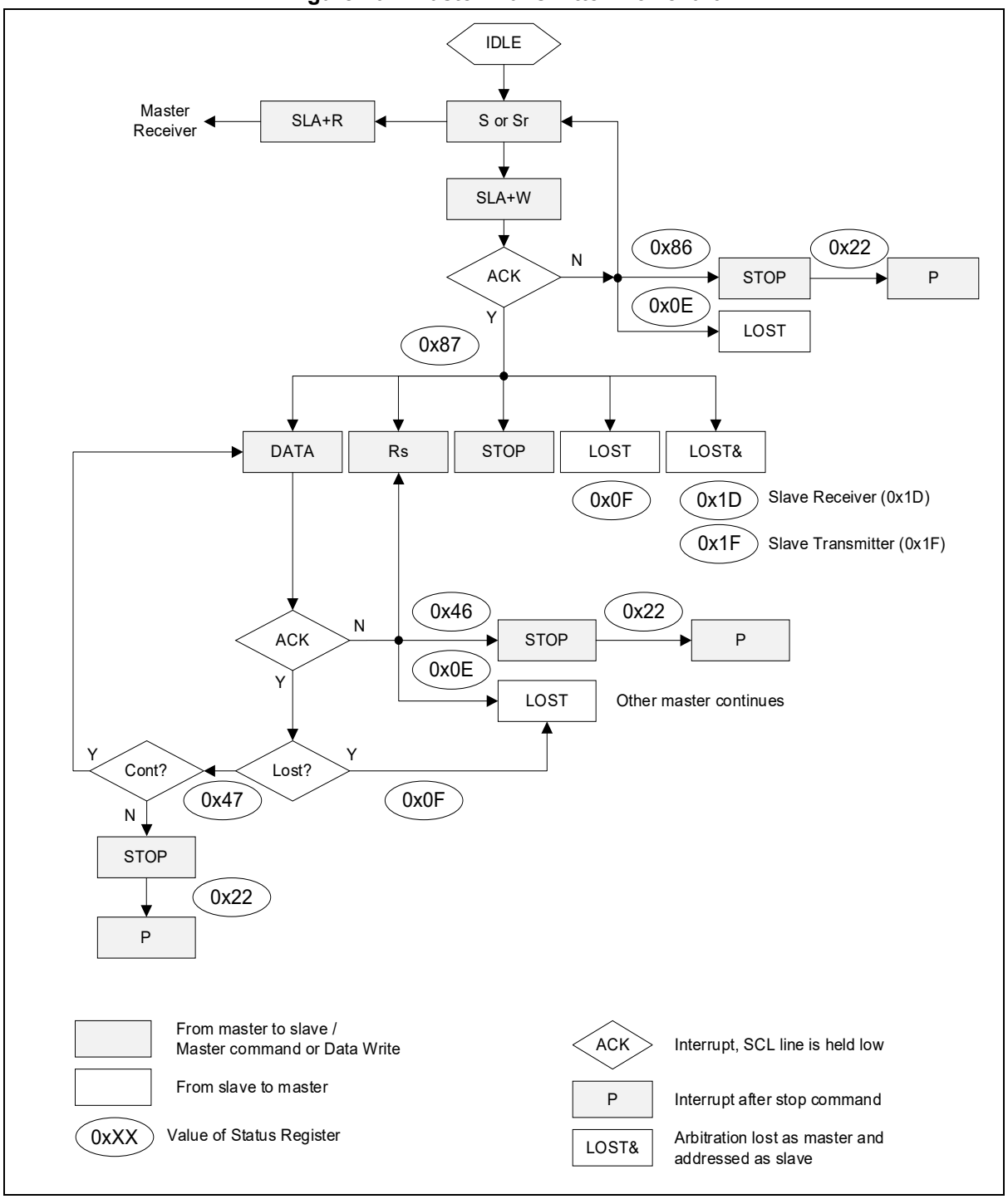
   After doing one of the actions above, write arbitrary value to I2C_SR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.
7. This is the final step for slave receiver function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2C_SR, write arbitrary value to I2C_SR. After this, I2C enters idle state.

13.4.8 I2C Master Mode

13.4.8.1 Master Transmitter

Figure 101 shows a flowchart of the transmitter in master mode.

Figure 101. Master Transmitter Flowchart



To operate the I2C in master transmitter mode, users must follow the steps below:

1. Enable I2C by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+W into the I2CDR where SLA is address of slave device and W is transfer direction from the viewpoint of the master. For master transmitter, W is '0'. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCL. If the master gains bus mastership, I2C generates GCALL interrupt regardless of the reception of ACK from the slave device. When I2C loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and I2C waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case I2C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I2C holds the SCL LOW. This is because to decide whether I2C continues serial transfer or stops communication. The following steps continue assuming that I2C does not lose mastership during first data transfer. I2C (Master) can choose one of the following cases regardless of the reception of ACK signal from slave.

   Case 1. Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.

   Case 2. Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.

   Case 3. Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is '1' go to master receiver section.

7. Byte of data is being transmitted. During data transfer, bus arbitration continues.

8. This is ACK signal processing stage for data packet transmitted by master. I2C holds the SCL LOW. When I2C loses bus mastership while transmitting data arbitrating other masters, the MLOST bit in I2CSR is set. If then, I2C waits in idle state. When the data in I2CDR is transmitted completely, I2C generates TEND interrupt. I2C can choose one of the following cases regardless of the reception of ACK signal from slave.

   Case A. Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.

   Case B. Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.

   Case C. Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set the START bit in I2CMR.

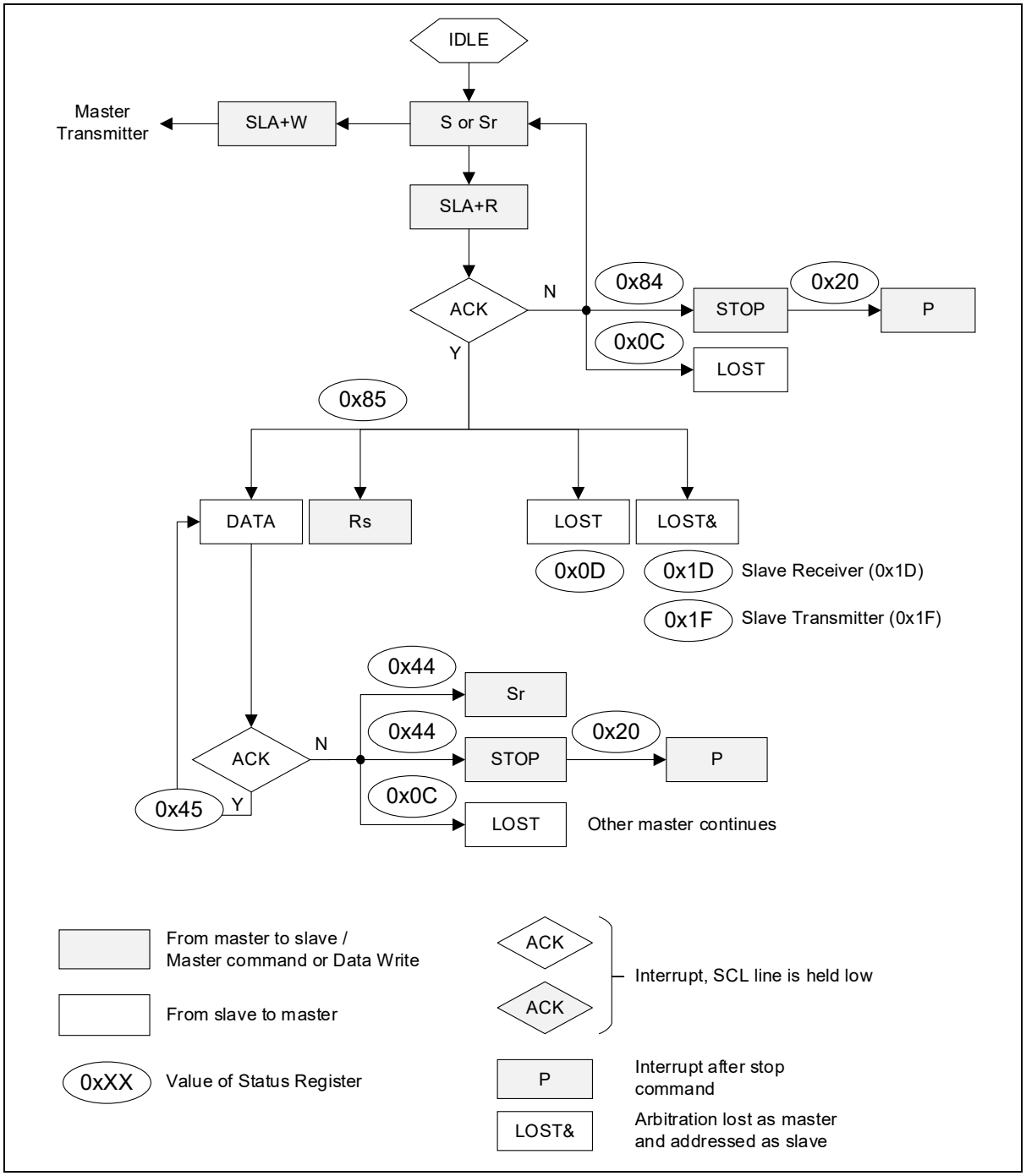
   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is ‘1’ go to master receiver section.

9. This is the final step for master transmitter function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I2C enters idle state.

13.4.8.2 Master Receiver

Figure 102 shows a flowchart of the receiver in master mode.

Figure 102. Master Receiver Flowchart



To operate the I2C in master receiver mode, users must follow the steps below:

1. Enable I2C by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+R into the I2CDR where SLA is address of slave device and R is transfer direction from the viewpoint of the master. For master receiver, R is '1'. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCL. If the master gains bus mastership, I2C generates GCALL interrupt regardless of the reception of ACK from the slave device. When I2C loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and I2C waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case I2C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I2C holds the SCL LOW. This is because to decide whether I2C continues serial transfer or stops communication. The following steps continue assuming that I2C does not lose mastership during first data transfer.
   I2C (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   Case 1. Master receives ACK signal from slave, so continues data transfer because slave can prepare and transmit more data to master. Configure ACKEN bit in I2CMR to decide whether I2C Acknowledges the next data to be received or not.

   Case 2. Master stops data transfer because it receives no ACK signal from slave. In this case, set the STOP bit in I2CMR.

   Case 3. Master transmits repeated START condition due to no ACK signal from slave. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is '0' go to master transmitter section.

7. Byte of data is being received.

8. This is ACK signal processing stage for data packet transmitted by slave. I2C holds the SCL LOW. When 1-Byte of data is received completely, I2C generates TEND interrupt.
I2C can choose one of the following cases according to the RXACK flag in I2CSR.

   Case A. Master continues receiving data from slave. To do this, set ACKEN bit in I2CMR to Acknowledge the next data to be received.

   Case B. Master wants to terminate data transfer when it receives next data by not generating ACK signal. This can be done by clearing ACKEN bit in I2CMR.

   Case C. Because no ACK signal is detected, master terminates data transfer. In this case, set the STOP bit in I2CMR.

   Case D. No ACK signal is detected, and master transmits repeated START condition. In this case, load SLA+R/W into the I2CDR and set the START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) and 2), move to step 7. In case of 3), move to step 9 to handle STOP interrupt. In case of 4), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is '0' go to master transmitter section.

9. This is the final step for master receiver function of I2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I2C enters idle state.

13.5 I2C Interrupts

Setting the IINTEN in I2Cn_CR register to ‘1’ can enable the I2C interrupt. With the IINTEN bit enabled, the interrupt is executed as the SCL pulse is output when the master sets the START 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, see 13.4.7 I2C Slave Mode and 13.4.8 I2C Master Mode

13.6 Setting Example

<Example 1> I2C0 Master Initial Setting Example

```
PMU_PER<I2C0> = “1”                 : Enables the I2C0 channel
PMU_PCCR<I2C0> = “1”                : Enables the clock supply of I2C0 channel

PB_MR<P15[31:30]> = “01”            : Selects the function of PB15 pin as SDA0
PB_MR<P14[29:28]> = “01”            : Selects the function of PB14 pin as SCL0
PB_CR<P15[31:30]> = “01”            : Selects SDA0(PB15) pin as open drain
PB_CR<P14[29:28]> = “01”            : Selects SCL0(PB14) pin as open drain
PB_PCR<P15> = “0”                   : Disables pull-up/pull-down function of SDA0(PB15) pin
PB_PCR<P14> = “0”                   : Disables pull-up/pull-down function of SCL0(PB14) pin

I2C0_CR<SOFTRESET> = “1”            : Initializes the internal registers of I2C0 serial device
I2C0_CR<INTERVAL[9:8]> = “01”       : Configures the internal delay option value of I2C0
I2C0_CR<IINTEN> = “1”               : Enables the interrupt of I2C0 channel
I2C0_SCLL<SCL[31:0]> = ”0”          : Initializes SCL0 low period register
I2C0_SCLH<SCLH[31:0]> = “0”         : Initializes SCL0 high period register
I2C0_SDH<SDH[31:0]> = “0”           : Initializes SDA0 hold time register
I2C0_CR<ACKEN> = “1”                : Sets ACK signal output function after receiving data

I2C0_SAR<SVAD[7:1]> = “010 0000”    : Sets 7 bits slave address '0x20'
```

13.7 I2C Registers

The base addresses and register map of the I2C module are as follows:

Table 98. Base Addresses of I2C Interface

Name	Base address
I2C0	0x4000_0A00
I2C1	0x4000_0A80

Table 99. I2C Register Map

Name	Offset	Type	Description	Reset value	Reference
I2Cn_DR	0x0000	RW	I2C Tx/Rx data register	0x0000_00FF	13.7.1
I2Cn_SR	0x0008	RW	I2C status register	0x0000_0000	13.7.2
I2Cn_SAR	0x000C	RW	I2C slave address register	0x0000_0000	13.7.3
I2Cn_CR	0x0014	RW	I2C control register	0x0000_0000	13.7.4
I2Cn_SCLL	0x0018	RW	I2C SCL low period register	0x0000_FFFF	13.7.5
I2Cn_SCLH	0x001C	RW	I2C SCL high period register	0x0000_FFFF	13.7.6
I2Cn_SDH	0x0020	RW	I2C SDA hold register	0x0000_7FFF	13.7.7

NOTE: n = 0 and 1.

13.7.1 I2Cn_DR: I2Cn Data Register

The I2Cn_DR register is an 8-bit register that has read / write access and stores data to be transferred and received data.

I2C0_DR=0x4000_0A00, I2C1_DR=0x4000_0A80

31 30 29 28 27 26 25 24	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]	[Transmission mode] The data stored in this register is output via the serial data line (SDA) and transmitted. [Receive mode] The value entered through SDA is saved and the received data can be read by reading this register.

13.7.2 I2Cn_SR: I2Cn Status Register

The I2Cn_SR register is a 8-bit R/W 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_0A08, I2C1_SR=0x4000_0A88

31 30 29 28 27 26 25 24	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 or not the ACK address (AACK) has been received from a slave. If the I2C module is a slave, it indicates whether or not 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: I2Cn Slave Address Register

The I2Cn_SAR register is an 8-bit readable and writable register. The upper 7 bits stores the address selected when the device operates as a slave, and the lower 1 bit can select whether to receive the general call address from the I2C master bus.

I2C0_SAR=0x4000_0A0C, I2C1_SAR=0x4000_0A8C

31 30 29 28 27 26 25 24	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]	GCALL
-			0x00	0
-			RW	RW

7 1	SVAD[6:0]	7-bit slave address	
0	GCALL	Whether to enable or disable general call	
		0	Dismisses the general call address.
		1	Receives the general call address.

13.7.4 I2Cn_CR: I2Cn Control Register

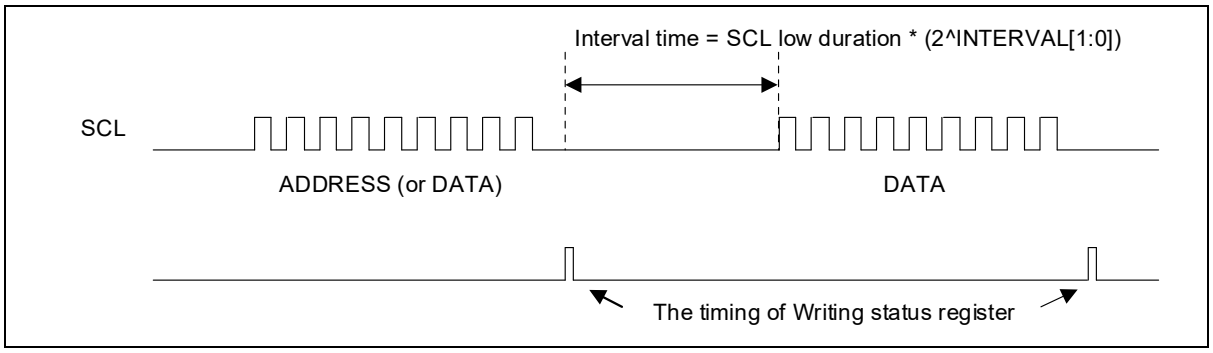
The I2Cn_CR register is a 10-bit R/W register. It sets the operating modes and enablement of the I2C module.

I2C0_CR=0x4000_0A14, I2C1_CR=0x4000_0A94

31 30 29 28 27 26 25 24	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			INTERVAL [1:0]	IIF	Reserved	SOFTRESET	IINTEN	ACKEN	Reserved	STOP	START
-			00	0	-	0	0	0	-	0	0
-			RW	RO	-	WO	RW	RW	-	RW	RW

9 8	INTERVAL[1:0]	Internal delay between address and data transfers (Or between two data transfers)
	0	(I2Cn_SCLL × 1) delay added
	1	(I2Cn_SCLL × 2) delay added
	2	(I2Cn_SCLL × 4) delay added
	3	(I2Cn_SCLL × 8) delay added
7	IIF	This flag indicates that an interrupt has occurred. To clear the IIF bit, the relevant interrupt flag must be cleared in the I2Cn_SR register.
	0	No interrupt has occurred or the flagged interrupt has been cleared.
	1	An interrupt has occurred.
5	SOFTRESET	Initializing the internal register of I2C device.
	0	No reset
	1	Reset (I2C register)
4	IINTEN	Whether to enable or disable interrupts
	0	Disables interrupts.
	1	Enables interrupts.
3	ACKEN	Whether to enable or disable ACK in receive mode
	0	No ACK signal is generated.
	1	ACK signal is generated.
1	STOP	This field is used to stop transmission. If this bit is set to '1' during data transmission/receive, forced termination is performed during transmission. Even if an ACK signal is received after data transmission in transmission mode, if this bit is set to '1', transmission is terminated without transmitting the next data.
	0	Continue communication
	1	Stop communication
0	START	Transmission start in master mode
	0	The I2C waits in slave mode.
	1	The I2C starts transmitting in master mode. (At the end of the address phase, this field is automatically cleared.)

Figure 103. INTERVAL[1:0] (Inter Delay) in Master Mode



13.7.5 I2Cn_SCLL: I2Cn SCL Low Period Register

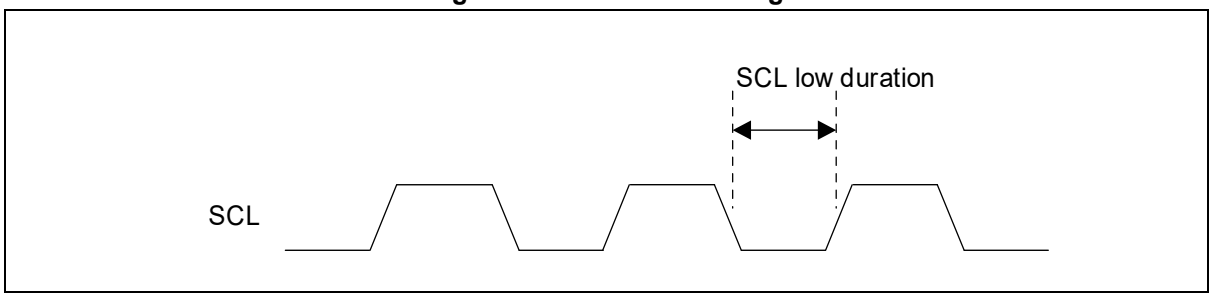
The I2Cn_SCLL register is a 16-bit register. The SCL low period time can be set in master mode.

I2C0_SCLL=0x4000_0A18, I2C1_SCLL=0x4000_0A98

31 30 29 28 27 26 25 24	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]	Determine the period of the SCL low signal for I2C transmit rate in master mode. $\text{Timimg} = \frac{(1/\text{PCLK})}{(\text{SCLL}+1)}$

Figure 104. SCL Low Timing



13.7.6 I2Cn_SCLH: I2Cn SCL High Period Register

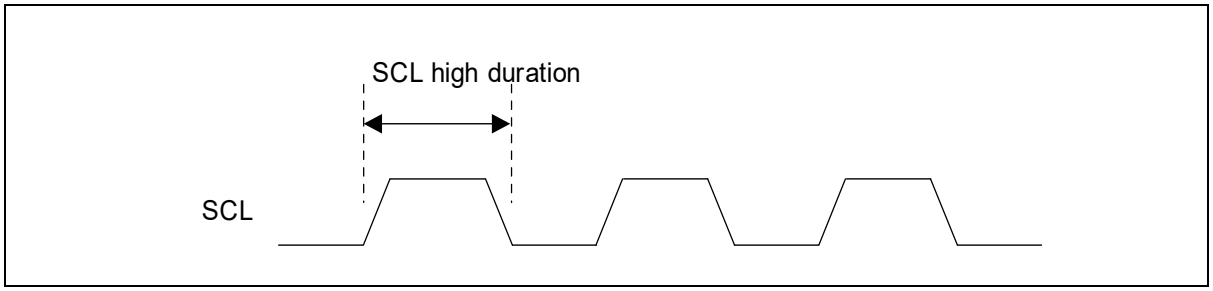
The I2Cn_SCLH register is a 16-bit register. The SCL high period time can be set in master mode.

I2C0_SCLH=0x4000_0A1C, I2C1_SCLH=0x4000_0A9C

31 30 29 28 27 26 25 24	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]	Determine the period of the SCL high signal for I2C transmit rate in master mode. $\text{Timimg} = \frac{(1/\text{PCLK})}{(\text{SCLH} + 3)}$

Figure 105. SCL High Timing



13.7.7 I2Cn_SDH: I2Cn SDA Hold Time Register

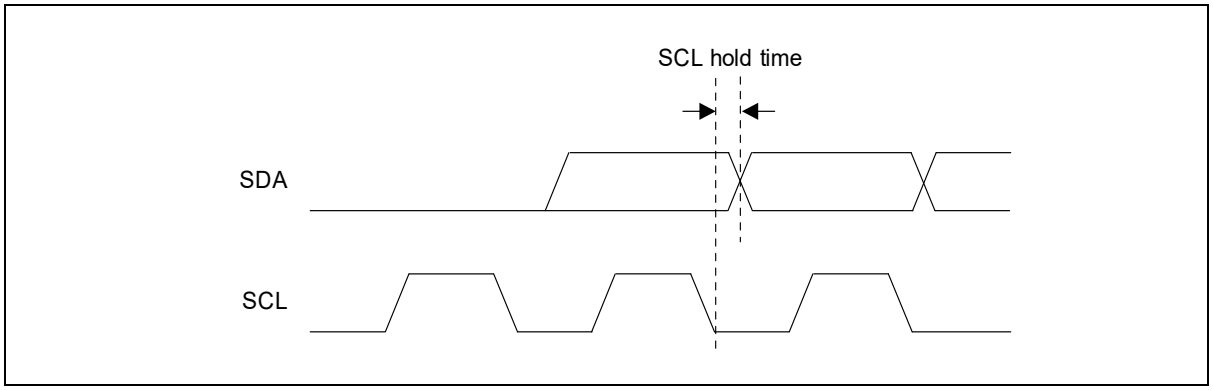
The I2Cn_SDH register is a 15-bit R/W register. The SCL hold time can be set in master mode.

I2C0_SDH=0x4000_0A20, I2C1_SDH=0x4000_0AA0

31 30 29 28 27 26 25 24	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]	Set SDA hold time	$\text{Timimg} = \frac{(1/\text{PCLK})}{(\text{SDH} + 4)}$

Figure 106. SDA Hold Timing



13.7.8 I2C Register Map Summary

Table 100. 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	TMODE	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]							GCALL
	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.	INTERVAL [1:0]		IIF	Res.	SOFTRESET	IINTEN	ACKEN	Res.	STOP	START
	Reset value																							0	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. PWM Generator

14.1 PWM Generator Introduction

The A33G53x has eight built-in PWM (Pulse-Width Modulation) generators that can output PWM (Pulse-Width Modulation) waveforms. This PWM (Pulse-Width Modulation) generator has a 16-bit resolution, four channels form a 1-unit, and each unit has a built-in prescaler. PWM (Pulse-Width Modulation) It can be used as square waveform signal required for user application such as LED, motor, inverter, etc. through PWM related register setting.

14.2 PWM Generator Main Features

The PWM Generator module of the A33G53x series has the following main features in PWM modes:

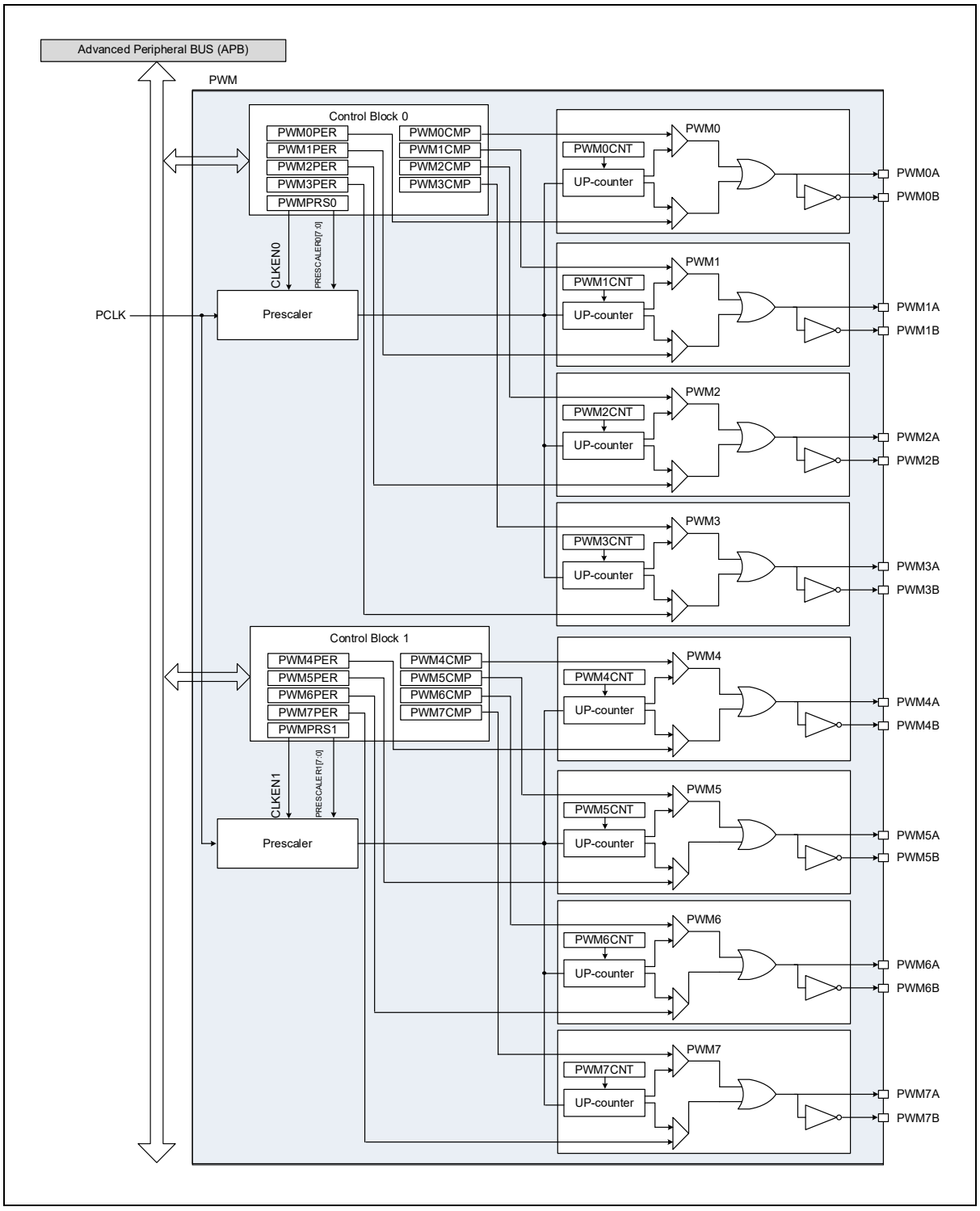
- Output with 16-bit resolution
- Outputs from up to eight channels
- PWM invert signal output channels
  - 100-pin (PWM0 to PWM7)
  - 80-pin (PWM1 to PWM7)
- PWM duty setting function by setting counter and period register.
- PWM input clocks
  - PCLK
  - Four steps (1/2, 1/4, 1/8, 1/16) prescaler
- Integrated 1-unit 8-bit prescaler for PWM input clock division
- In multichannel PWM operation, timing synchronization based on the PWM signal with the shortest cycle

14.3 PWM Generator Functional Description

14.3.1 Block Diagram

Figure 107 shows a block diagram of the PWM Generator.

Figure 107. PWM Block Diagram



14.3.2 Pins and Internal Signals

The external pins connected to the PWM module are described in Table 101.

Table 101. Pin Assignment of PWM: External Pins

Pin name	Type	Description	Supported packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G539MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
PWM0A	O	Output of PWM0	O	O	O	O
PWM0B	O	Reverse output of PWM0	O	O	-	-
PWM1A	O	Output of PWM1	O	O	O	O
PWM1B	O	Reverse output of PWM1	O	O	O	-
PWM2A	O	Output of PWM2	O	O	O	O
PWM2B	O	Reverse output of PWM2	O	O	O	-
PWM3A	O	Output of PWM3	O	O	O	O
PWM3B	O	Reverse output of PWM3	O	O	O	-
PWM4A	O	Output of PWM4	O	O	O	O
PWM4B	O	Reverse output of PWM4	O	O	O	-
PWM5A	O	Output of PWM5	O	O	O	O
PWM5B	O	Reverse output of PWM5	O	O	O	-
PWM6A	O	Output of PWM6	O	O	O	-
PWM6B	O	Reverse output of PWM6	O	O	O	O
PWM7A	O	Output of PWM7	O	O	O	-
PWM7B	O	Reverse output of PWM7	O	O	O	O

14.3.3 Setting the Input Clock of PWM (Pulse-Width Modulation) Generator

The source clock for PWM (Pulse-Width Modulation) is determined by the system peripheral clock and the prescaler in the PWM_PRSn register or the prescaler in the PWMn_CTRL register. PWM (Pulse-Width Modulation) source clock can be obtained by the formula below:

$$\text{PWMCLKn} = \frac{\text{PCLK}}{\text{PRESCALERn} + 1}$$

The first way to get the PWM source clock is to obtain the PWM input clock PWMCLKn by activating PWMCLKn with the PWMPRSn [15] bits in the PWM_PRSn register and setting the PRESCALERn [7: 0] value to the system peripheral input clock. PWMCLK0 is the clock applied to PWM0~3 group, and PWMCLK1 is the clock applied to PWM4~7 group.

PWMCLKn applied to each PWMn can be set to the input clock of each PWM (Pulse-Width Modulation) channel by dividing 2, 4, 8, or 16 by the value set in CKSEL [7: 5] through the clock divider.

14.3.4 Setting the Period and PWM Output Signal

Each PWMn channel has a PWMn_PER register for setting the PWM frequency.

PWM period can be obtained by below formula with 16-bit width value.

The PWM (Pulse-Width Modulation) period can be obtained by applying the following formula with 16-bit PWMn_PER value and the PWM input clock.

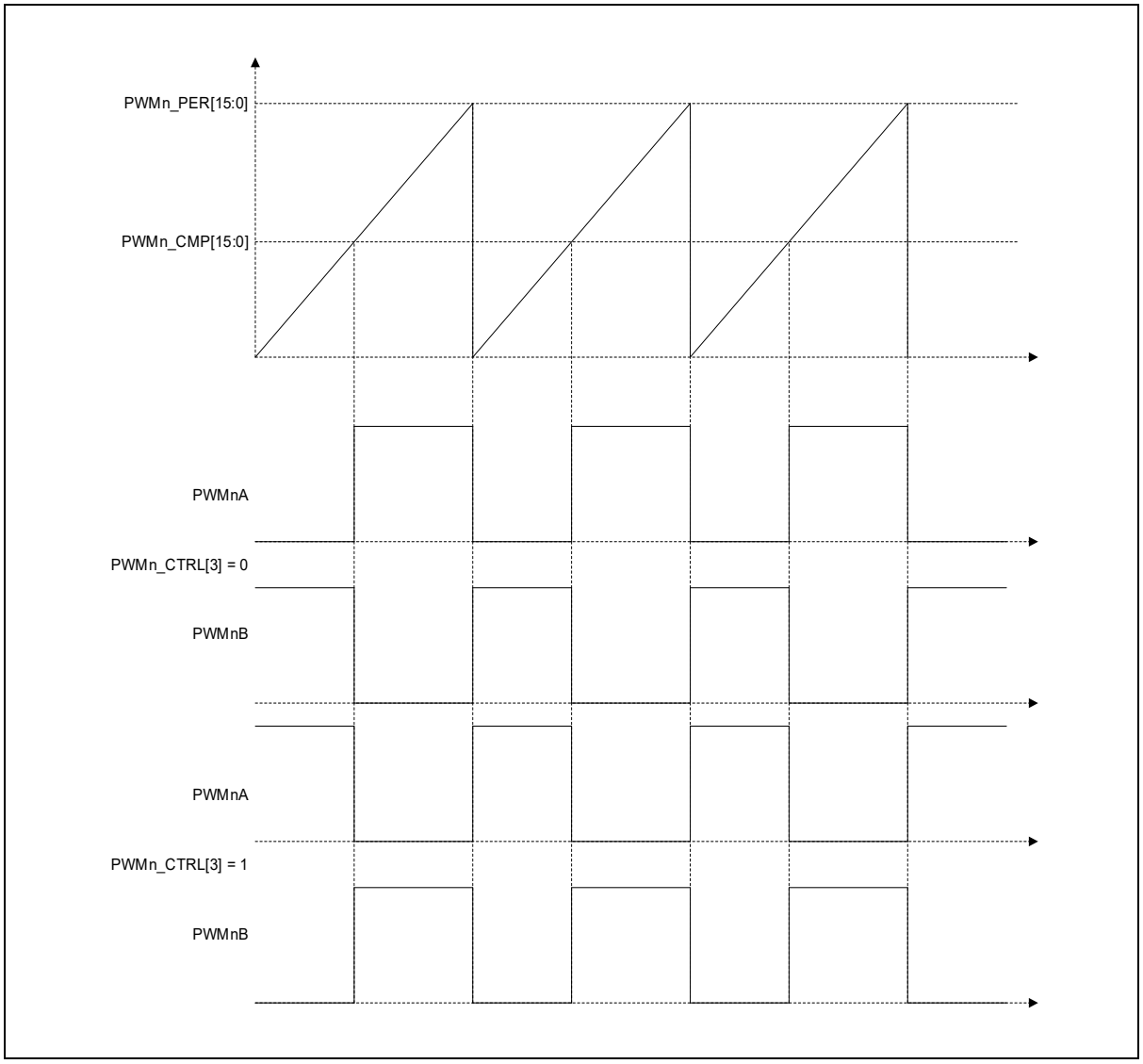
$$\text{PWM Period} = \frac{\text{PWMCLK}}{\text{PERIOD} + 1}$$

Each PWMn channel can set PWM duty as well as period. PWM duty can be set to a value in the range of 0 ~ PERIOD in PWMn_CMP[15:0].

The default output of PWMnA is 'L', and the PWMnA output changes to 'H' when the PWMn_CNT register value reaches the PWMn_CMP value.

You can also invert the starting output level of PWMnA by entering a value in the INVA bit in the PWMn_CTRL register. If you enter 0 as the default value, the start output value will be Low. If you input 1, High will be output. PWMnB is output when the PWMnA waveform is inverted.

Figure 108. PWM Timing Diagram



After each of the above functions are set, if 1 is input to the STRT bit in the PWMn_CTRL register, PWM is output. If 0 is input, PWM output is stopped.

14.3.5 Synchronization of PWM Channels

The PWM (Pulse-Width Modulation) block has a function to set synchronization between channels. if the synchronization function is enabled through the control registers of each channel in the PWM0~3 group synchronizes the counting of the active PWM channels. The timing of the synchronized PWM channels is set based on the period of the shortest PWM channel. Similarly, PWM4~7 groups operate in the same way as PWM0~3 groups. However, the synchronization function of PWM0~3 group and PWM4~7 group is independent and does not affect the operation between PWM0~3 and PWM4~7 groups.

NOTE: In a PWM group (PWM0~3 or PWM4~7), when the enabled synchronous (SYNC = 1) channel and asynchronous (SYNC = 0) channel are used at the same time, the period of synchronous PWM channels is effected by asynchronous PWM channel. Therefore, if the application requires independent PWM operation, it is recommended to use PWM channel of other PWM group or to use TIMER with PWM mode.

14.4 PWM Interrupts

Each of PWMn channels can be configured periodic interrupt. If the PRIE bit in the PWMn_CTRL register is set to '1', the PWM periodic interrupt is enabled. If the PRIE bit in the PWMn_CTRL register is set to '0', the periodic interrupt is disabled.

Interrupt vector is supported for each channel. If an interrupt occurs in each PWMn channel, it is possible to check whether an interrupt is generated by reading the PRF bit in the PWMn_CTRL register. If a periodic interrupt occurs, the PRF bit must be cleared by writing 1 to it.

Table 102. Interrupt on PWMn

Interrupt event	Event flag	Enable control bit	Interrupt clear method
PWM0 period	PRF in PWM0_CTRL	PRIE in PWM0_CTRL	Write 1 in PRF
PWM1 period	PRF in PWM1_CTRL	PRIE in PWM1_CTRL	Write 1 in PRF
PWM2 period	PRF in PWM2_CTRL	PRIE in PWM2_CTRL	Write 1 in PRF
PWM3 period	PRF in PWM3_CTRL	PRIE in PWM3_CTRL	Write 1 in PRF
PWM4 period	PRF in PWM4_CTRL	PRIE in PWM4_CTRL	Write 1 in PRF
PWM5 period	PRF in PWM5_CTRL	PRIE in PWM5_CTRL	Write 1 in PRF
PWM6 period	PRF in PWM6_CTRL	PRIE in PWM6_CTRL	Write 1 in PRF
PWM7 period	PRF in PWM7_CTRL	PRIE in PWM7_CTRL	Write 1 in PRF

14.4.1 Interrupt Generation

Table 103 describes the interrupts that the PWM module.

Table 103. Interrupt Vector of PWMn

Units	Interrupt priority	Vector address	Interrupt source
PWM0	24	0x0000_00A0	PWM0 periodic interrupt
PWM1	25	0x0000_00A4	PWM1 periodic interrupt
PWM2	26	0x0000_00A8	PWM2 periodic interrupt
PWM3	27	0x0000_00AC	PWM3 periodic interrupt
PWM4	28	0x0000_00B0	PWM4 periodic interrupt
PWM5	29	0x0000_00B4	PWM5 periodic interrupt
PWM6	30	0x0000_00B8	PWM6 periodic interrupt
PWM7	31	0x0000_00BC	PWM7 periodic interrupt

14.5 PWM Registers

The base addresses and register map of the PWM are as follows:

Table 104. Base Addresses of PWM Interface

Name	Base address
PWM0	0x4000_0700
PWM1	0x4000_0720
PWM2	0x4000_0740
PWM3	0x4000_0760
PWM4	0x4000_0780
PWM5	0x4000_07A0
PWM6	0x4000_07C0
PWM7	0x4000_07E0
PWMPRS0	0x4000_077C
PWMPRS1	0x4000_07FC

Table 105. PWM Register Map

Name	Offset	Type	Description	Reset value	Reference
PWM_PRS0	0x007C	RW	PWM0 to 3 Prescaler Register 0	0x0000_0000	14.5.1.1
PWM_PRS1	0x00FC	RW	PWM4 to 7 Prescaler Register 1	0x0000_0000	14.5.1.2
PWMn_CTRL	0x0000	RW WC	PWM Timer n Control Register	0x0000_0000	14.5.1.3
PWMn_CNT	0x0004	RW	PWM Timer n Counter Register	0x0000_0000	14.5.1.4
PWMn_PER	0x0008	RW	PWM Timer n Period Register	0x0000_0000	14.5.1.5
PWMn_CMP	0x000C	RW	PWM Timer n Compare Register	0x0000_0000	14.5.1.6

14.5.1 PWM Generator Register

14.5.1.1 PWM_PRS0: PWM 0 to 3 Prescaler Register

This register is used to set Unit 0 of the PWM (Pulse-Width Modulation) generator. This register provides a prescaler for dividing the PWM input clock of unit 0 and the input clock set in this register is applied to the PWM0 to PWM3 channels.

PWM_PRS0=0x4000_077C

31 30 29 28 27 26 25 24	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		CLKEN0	Reserved	PRESALER0[7:0]
-		0	-	0x00
-		RW	-	RW

15	CLKEN0	Operation control of prescaler 0 clock
		0 Stop prescaler 0 clock
		1 Run prescaler 0 clock
7 0	PRESCALER0 [7:0]	The prescaler value of unit 0 $\text{PWMCLK0} = \frac{\text{PCLK}}{\text{PRESCALER0} + 1}$

14.5.1.2 PWM_PRS1: PWM 4 to 7 Prescaler Register

This register is used to set Unit 1 of the PWM (Pulse-Width Modulation) generator. This register provides a prescaler for dividing the PWM input clock of unit 1 and the input clock set in this register is applied to the PWM4 to PWM7 channels.

PWM_PRS1=0x4000_07FC

31 30 29 28 27 26 25 24	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		CLKEN1	Reserved	PRESALER1[7:0]
-		0	-	0x00
-		RW	-	RW

15	CLKEN1	Operation control of prescaler 1 clock
		0 Stop prescaler 1 clock
		1 Run prescaler 1 clock
7 0	PRESCALER1 [7:0]	The prescaler value of unit 1 $\text{PWMCLK1} = \frac{\text{PCLK}}{\text{PRESCALER1} + 1}$

14.5.1.3 PWMn_CTRL: PWM Timer n Control Register

PWM (Pulse-Width Modulation) This register is used to control channel n. This register has a function to specify PWM clock and control settings for each PWM output channel.

PWM0_CTRL=0x4000_0700, PWM1_CTRL=0x4000_0720
PWM2_CTRL=0x4000_0740, PWM3_CTRL=0x4000_0760
PWM4_CTRL=0x4000_0780, PWM5_CTRL=0x4000_07A0
PWM6_CTRL=0x4000_07C0, PWM7_CTRL=0x4000_07E0

31 30 29 28 27 26 25 24	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		PRF	Reserved	PRIE	CKSEL[2:0]	Reserved	INVA	Reserved	SYNC	STRT
-		0	-	0	000	-	0	-	0	0
-		WC	-	RW	RW	-	RW	-	RW	RW

12	PRF	Display the PWM periodic interrupt flag	
		0	No periodic interrupt
		1	Periodic interrupt occurs (write '1' to clear)
8	PRIE	Enable/Disable PWM periodic interrupt	
		0	Disable PWM periodic interrupt
		1	Enable PWM periodic interrupt
7 5	CKSEL[2:0]	Select a input clock source of PWM	
		000	divided by 2
		001	divided by 4
		010	divided by 8
		011	divided by 16
		1xx	Disable input clock
3	INVA	Selection for PWM Inversion Output	
		0	Normal PWM output A (Not inverted)
		1	Inverted PWM output A
1	SYNC	Synchronize the current PWM channel to another channel's timing.	
		0	Do not synchronize this channel with the others
		1	Synchronize this channel to the other channels (Synchronized to the timing of the PWM channel with the smallest PRD value among the several PWM output channels.) (See chapter 14.3.5)
0	STRT	Control the operation of PWM	
		0	Stop counter
		1	Run counter

14.5.1.4 PWMn_CNT: PWM Timer n Counter Register

This register is the count value register of the PWM channel n. This register consists of 16 bits.

PWM0_CNT=0x4000_0704, PWM1_CNT=0x4000_0724
PWM2_CNT=0x4000_0744, PWM3_CNT=0x4000_0764
PWM4_CNT=0x4000_0784, PWM5_CNT=0x4000_07A4
PWM6_CNT=0x4000_07C4, PWM7_CNT=0x4000_07E4

31 30 29 28 27 26 25 24	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]	Current PWM counter value

14.5.1.5 PWMn_PER: PWM Timer n Counter Period Register

This register specifies the period of PWM channel n counter and consists of 16 bits. The value of the counter reaches the value specified in this register, the counter value is reset.

PWM0_PER=0x4000_0708, PWM1_PER=0x4000_0728
PWM2_PER=0x4000_0748, PWM3_PER=0x4000_0768
PWM4_PER=0x4000_0788, PWM5_PER=0x4000_07A8
PWM6_PER=0x4000_07C8, PWM7_PER=0x4000_07E8

31 30 29 28 27 26 25 24	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]	
-		0x0000	
-		RW	

15 0	PERIOD[15:0]	The period value of PWM channel n $\text{PWM Period} = \frac{\text{PWMCLK}}{\text{PERIOD} + 1}$

14.5.1.6 PWMn_CMP: PWM Timer n Compare Register

This register specifies the value of PWM (Pulse-Width Modulation) comparator A for output A of channel n and consists of 16 bits. When the counter is reset, the output A is output as 'L'. If the value of this register matches the value of the PWM (Pulse-Width Modulation) counter, the output A is set to 'H'.

PWM0_CMP=0x4000_070C, PWM1_CMP=0x4000_072C
PWM2_CMP=0x4000_074C, PWM3_CMP=0x4000_076C
PWM4_CMP=0x4000_078C, PWM5_CMP=0x4000_07AC
PWM6_CMP=0x4000_07CC, PWM7_CMP=0x4000_07EC

31 30 29 28 27 26 25 24	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		CMP[15:0]	
-		0x0000	
-		RW	

15 0	CMP[15:0]	The comparator value of PWM channel n

14.5.1.7 PWM Register Map Summary

Table 106. PWM 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
0x077C	PWM_PRS0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKEN0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER0[7:0]							
	Reset value																	0								0	0	0	0	0	0	0	0
0x07FC	PWM_PRS1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKEN1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER1[7:0]							
	Reset value																	0								0	0	0	0	0	0	0	0
0x00	PWMn_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.				PRF				PRIE	CKSEL[2:0]			Res.	INVA		SYNC	STRT
	Reset value																				0				0	0	0	0		0		0	0
0x04	PWMn_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
0x08	PWMn_PER	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	0	0
0x0C	PWMn_CMP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15. Analog-to-Digital Converters (ADC)

15.1 Introduction

The A33G53x has a built-in 12-bit SAR type analog-to-digital converter (ADC) 1-unit and samples the analog signal as a digital signal at 70ksps speed. The ADC (Analog to Digital Converter) built in the A33G53x is designed to receive up to 16 channels in time-divisional manner.

When converting an analog signal to a digital signal, convert the analog value input to the ADC (Analog to Digital Converter) channel to a 12-bit resolution digital value using the signal applied to the AVDD pin (analog voltage supply pin) as the reference voltage. In other words, the input reference voltage of the ADC can be set by varying the voltage of the AVDD pin, and the analog voltage ranging from GND to AVDD can be input and output as a digital signal.

15.2 ADC Main Features

Main features of the ADC module are listed below:

- 12-bit analog to digital resolution
- 3.0 V minimum to 5.5 V operating voltage range
- Up to 16 multiplexed input channels
  - 100-pin : 16 channels
  - 80-pin : 16 channels
  - 64-pin : 10 channels
- AVDD pin can be used to set the ADC input reference voltage
- Analog input voltage range: GND to AVDD
- Conversion Time: Max 15us per channel (at 4 MHz ADC clock)
- A / D conversion end interrupt flag support
- External ADC start trigger signal input selection
  - A/D conversion trigger source through timer match function
- ADC input range:
  - AGND ≤ ANx (x = 0 to 15) ≤ AVDD ≤ VDD

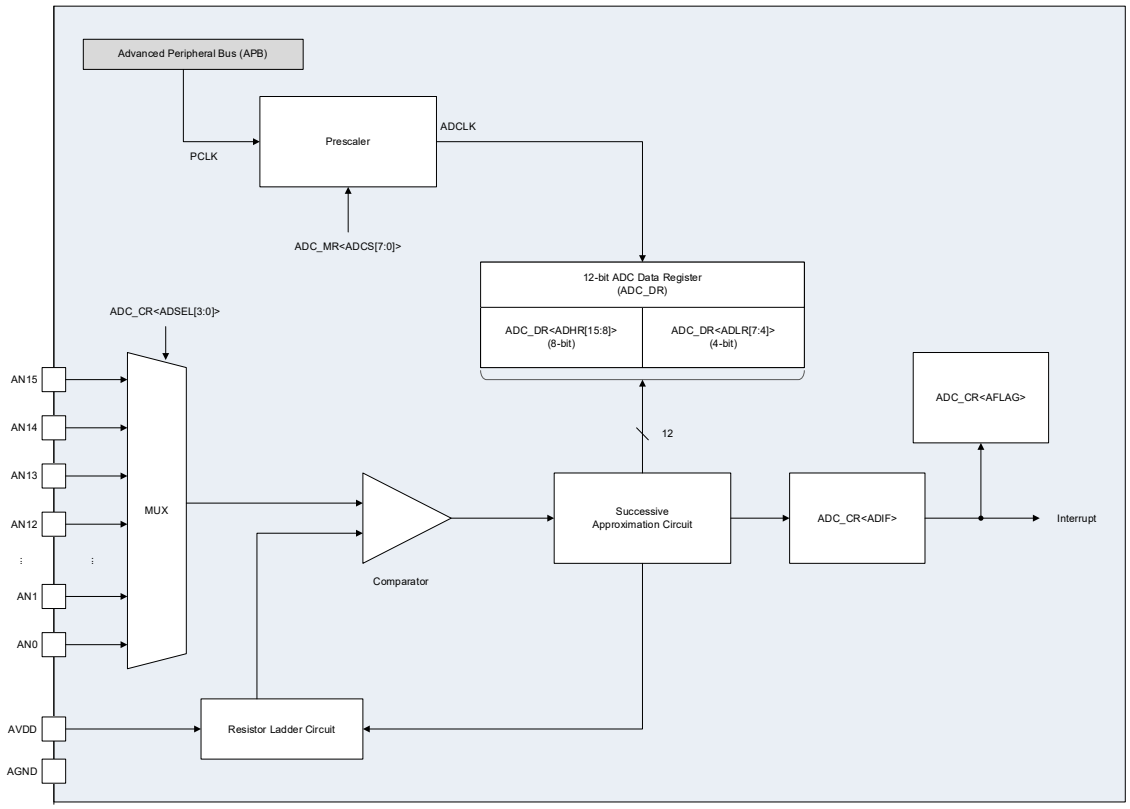
NOTE: ADC performance such as INL or DNL may be degraded if other analog modules use the AVDD reference voltage. Therefore, in applications where the precision of the ADC is important, it is recommended that the software be configured to use only the ADC analog module.

15.3 ADC Functional Description

15.3.1 ADC Block Diagram

Figure 109 shows a block diagram of the 12-bit ADC.

Figure 109. 12-bit ADC Block Diagram



NOTES:

1. The number of ADC input ports depends on the package type. Read the pin configuration of the ADC (Table 107)
2. The AGND port is not supported in the 64-pin package type and is connected to GND inside the chip.

15.3.2 ADC Pins and Signals

The pins assigned for the ADC are described in Table 107.

Table 107. Pin Assignment of ADC External Pins

Pin name	Type	Description	Supported packages			
			A33G539VQ A33G538VQ (MQFP-100)	A33G539VL A33G538VL (LQFP-100)	A33G539MM A33G538MM (LQFP-80)	A33G539RL A33G538RL (LQFP-64)
AVDD[1)]	Power	Analog Power Input	O	O	O	O
AGND[2)]	Power	Analog GND	O	O	O	-
AN0	Input	Analog Input Channel 0	O	O	O	O
AN1	Input	Analog Input Channel 1	O	O	O	O
AN2	Input	Analog Input Channel 2	O	O	O	O
AN3	Input	Analog Input Channel 3	O	O	O	O
AN4	Input	Analog Input Channel 4	O	O	O	O
AN5	Input	Analog Input Channel 5	O	O	O	O
AN6	Input	Analog Input Channel 6	O	O	O	O
AN7	Input	Analog Input Channel 7	O	O	O	O
AN8	Input	Analog Input Channel 8	O	O	-	-
AN9	Input	Analog Input Channel 9	O	O	-	-
AN10	Input	Analog Input Channel 10	O	O	-	-
AN11	Input	Analog Input Channel 11	O	O	-	-
AN12	Input	Analog Input Channel 12	O	O	-	-
AN13	Input	Analog Input Channel 13	O	O	-	-
AN14	Input	Analog Input Channel 14	O	O	O	O
AN15	Input	Analog Input Channel 15	O	O	O	O

NOTES:

1. Analog VDD voltage is equal to or lower than VDD voltage.
2. The AGND port is not supported in the 64-pin package type and is connected to GND inside the chip.

15.3.3 ADC Clocks (ADCLK)

15.3.3.1 ADC Module Control

Before enabling the ADC clock, users must select an ADC unit to be enabled by setting a bit of the ADC in PMU_PER register to '1'. Then, enable the clock of the selected ADC unit by setting a bit of the ADC in PMU_PCCR register to '1'.

To initialize the registers of ADC unit, set corresponding bit of the ADC in PMU_PER register and the ADC in PMU_PCCR register to '0' and then set back to '1'.

ADC in PMU_PER register controls ADC unit. To enable or disable ADC unit, set the ADC in PMU_PCCR register to the number shown below:

- If the ADC in PMU_PER register = '0', ADC unit is disabled.
- If the ADC in PMU_PER register = '1', ADC unit is enabled.

To provide the ADC unit with the ADC clock, configure the ADC in PMU_PCCR register referring the settings shown below:

- If the ADC in PMU_PCCR register = '0', ADC clock is disabled.
- If the ADC in PMU_PCCR register = '1', ADC clock is enabled.
-

15.3.3.2 ADC Clock Sources

The PCLK operates only as the ADC operation clock (ADCLK) based on the APB domain.

Internal clock source

The programmable divider coefficient can be selected to range from 0 to 255 by configuring the ADCS[7:0] in ADC_MR register, and the PCLK is divided by the coefficient to be used as the ADC operation clock.

NOTE: The ADC clock should be under 4MHz (in case of VDD = 5 V)

$$\mathrm{ADCLK} = \frac{\mathrm{PCLK}}{\mathrm{ADCS} + 1}$$

15.3.4 ADC On-Off Control (ADCEN)

Before starting the AD conversion, users must enable the ADC module using the ADCEN in ADC_MR register.

- Setting the ADCEN = '1' enables the ADC unit. When the ADC unit is enabled, the ADC functionality and the ASTART bit control are possible.
- Setting the ADCEN = '0' disables the ADC unit. When the ADC unit is disabled, the AD conversion cannot be performed.

15.3.4.1 Software Procedure to Enable the ADC

1. Clear the ADEOC in ADC_CR register by writing '0'.
2. Setting the ADCEN in ADC_MR register to '1' enables the ADC module.

15.3.4.2 Software Procedure to Disable the ADC

1. Confirm that the ADST in ADC_CR register is set to '0'. This ensures that no conversion is ongoing.
2. Disable the ADC module by setting the ADCEN in ADC_MR register to '0'

15.3.5 Constraints when Writing the ADC Control Bits

15.3.5.1 When Writing ADC Control Bit

While accessing the PMU_PER and PMU_PCCR registers to enable the ADC operation clock source and ADC module, the ADC module must be disabled. For this, set the ADCEN in ADC_MR register to '0'.

During the AD conversion operations, if users write to each ADC control bit of the PMU_PER and PMU_PCCR registers, the ADC clock bus and ADC registers are not controllable and become Unknown state.

Software can write to the control bits configuring the AD conversion, only when the ADC module is enabled (ADCEN = '1') and the AD conversion is not performed (e.g. ADST = '0').

- Set the ADCEN in ADC_MR register to '0'.
- Clear status bits in ADC_CR register.

15.3.6 Channel Selection

15.3.6.1 ADC Analog Input Channel

Each ADC has up to 16 multiplexed channels.

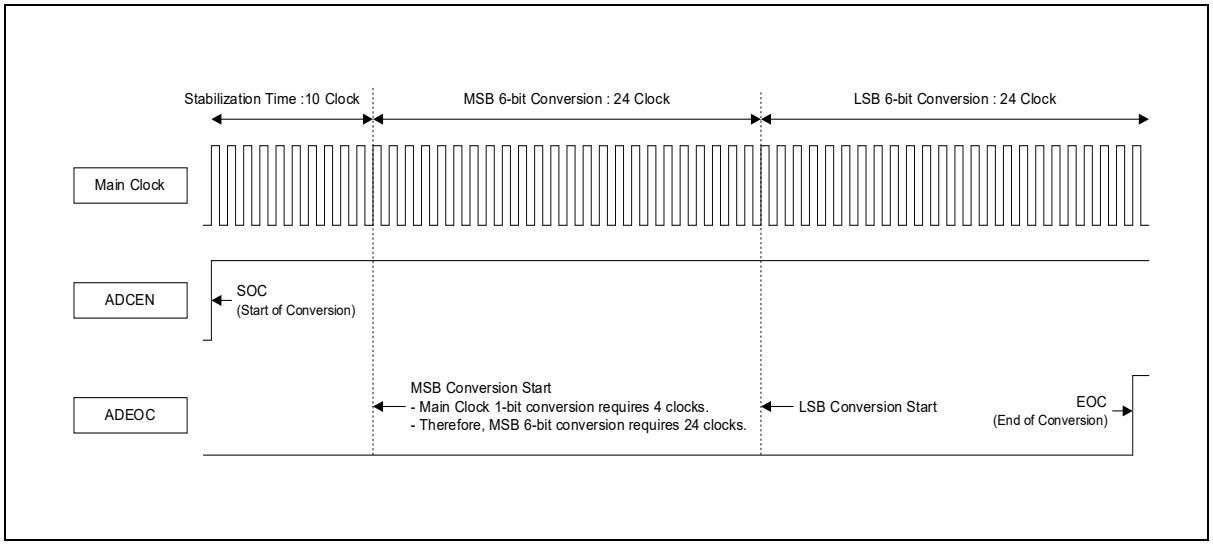
- When the analog inputs come from GPIO pads,
  A. For the analog inputs on the ADC channel, set mode of the channel you want to use to ADC using the Pn_MR registers (n = A to F).
  B. Set the pin direction to 'Analog input' using the Pn_CR register (n = A to F).
  C. Disable the internal pull-up or pull-down resistance functionality using the Pn_PCR register (n = A to F).

15.3.7 ADC Conversion Timing

The 12-bit ADC built into the A33G53x operates at the following timings.

After start-of-A/D conversion, 10 clocks for stabilization, MSB to LSB conversion, and one bit conversion takes four clocks. Therefore, the total conversion time is 10 clock + (12-bit data × 4 clock) = total 58 clocks.

Figure 110. Example of ADC Conversion Timing



15.3.8 ADC Operating Conditions

The ADC of the A33G53x microcontroller should be used by setting the A/D conversion time according to the reference input voltage AVDD. The following table specifies the ADC operating conditions.

Table 108. ADC Register Map

AVDD Voltage (V)	ADC Clock (MHz)	A/D Conversion Time (μs)
2.4	0.25	240
2.7	1.00	60
3.0	2.00	30
4.0	3.00	20
5.0	4.00	15

NOTE: AGND ≤ ANx (x = 0 to 15) ≤ AVDD ≤ VDD.

15.3.9 Starting Conversions (ADST, Trigger Source)

15.3.9.1 Start AD Conversion (ADST)

The AD conversion starts when the ADST in ADC_CR register is set to ‘1’ and the EXTRG in ADC_MR register is set to '0' by software as described below:

- If ADST = 0, ADC conversion does not start. (Software Trigger)
- If ADST = 1 and EXTRG = 0, ADC conversion starts. ADST is cleared at the next ADC clock cycle. (Software Trigger)

15.3.9.2 Start AD Conversion by a Trigger Source

If the ADC timer trigger source is used, software allows the AD conversion to start when the timer trigger source is occurred after timer operating.

15.3.10 Conversion on External Ttrigger (TSEL[2:0])

The AD conversion can be triggered by the software or the events of the 16-bit TIMER module (16-bit Timer GRA capture with input pin or 16-bit timer GRA interrupt, etc.). When setting the trigger source, it is recommended to stop the AD conversion by setting the ADST bit to ‘0’.

15.3.10.1 ADC Trigger Source Selection (TSEL[2:0], EXTRG)

The user can select a trigger source by configuring the TSEL[2:0] and EXTRG in ADC_MR register in Table 109.

Table 109. External Trigger Source Configuration

EXTRG	TSEL[2:0]	Trigger source	Description
0	xxx	ADST	Only soft trigger
1	000	TIMER0 GRA match	Enables the timer0 event trigger
1	001	TIMER1 GRA match	Enables the timer1 event trigger
1	010	TIMER2 GRA match	Enables the timer2 event trigger
1	011	TIMER3 GRA match	Enables the timer3 event trigger
1	100	TIMER4 GRA match	Enables the timer4 event trigger
1	101	TIMER5 GRA match	Enables the timer5 event trigger
1	110	TIMER6 GRA match	Enables the timer6 event trigger
1	111	TIMER7 GRA match	Enables the timer7 event trigger

To use the TIMER as the ADC trigger source, select the interrupt polarity of the TIMER channel.

15.3.10.2 ADC Trigger Start Control

If a trigger event occurs after the trigger source's timer operation starts, AD conversion is performed by trigger source.

Table 110. Start Control of AD Trigger Conversion

EXTRG	TSEL[2:0]	Trigger source	Start control bit (Register)
0	xxx	ADST	ADST = 1 in ADC_CR register
1	000	TIMER0 GRA match	TEN = 1 in TIMER0_CMD register
1	001	TIMER1 GRA match	TEN = 1 in TIMER1_CMD register
1	010	TIMER2 GRA match	TEN = 1 in TIMER2_CMD register
1	011	TIMER3 GRA match	TEN = 1 in TIMER3_CMD register
1	100	TIMER4 GRA match	TEN = 1 in TIMER4_CMD register
1	101	TIMER5 GRA match	TEN = 1 in TIMER5_CMD register
1	110	TIMER6 GRA match	TEN = 1 in TIMER6_CMD register
1	111	TIMER7 GRA match	TEN = 1 in TIMER7_CMD register

15.3.11 End of Conversion, End of Sampling Phase (EOC)

Interrupts for the End of ADC conversion can be enabled by configuring the ADIE in ADC_MR register as shown below:

- If ADIE = 0, trigger source event interrupt is disabled.
- If ADIE = 1, trigger source event interrupt is enabled.

When the ADC trigger interrupt is enabled, the ADIF flag is generated at the point of time when an ADC trigger source event of each AD conversion channel is generated. The ADIF flag is cleared by writing ‘1’ to this bit.

Although the trigger interrupt of TIMER is not enabled, ADIF flag of ADC is generated when trigger source event happens. At this time, If the ADIE bit was enabled, a trigger event interrupt is generated. If the trigger event occurs while ADIE bit disabled, the ADIF bit can be checked by polling with software.

NOTE: The initial state of the End of Conversion (EOC) of the ADC is 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 ADST in ADC_CR register to ‘1’ and perform two dummy transfers of the ADC_DR register to clear the EOC. And clear also the ADEOC in ADC_CR register. Users must consider this situation when initializing the ADC, since it cannot be processed by hardware.

15.4 ADC Interrupts

For each ADC unit, an interrupt can be generated by setting below:

- End of A/D conversion Interrupt (ADIE)

Each interrupt enable bits are available for flexibility.

Table 111. ADC Interrupts

Interrupt event	Event flag	Enable control bit
End of A/D conversion Interrupt	ADIF	ADIE

15.5 ADC Registers

The base address and register map of the ADC are as follows:

Table 112. Base Address of ADC

Name	Base address
ADC	0x4000_0E00

Table 113. ADC Register Map

Name	Offset	Type	Description	Reset value	Reference
ADC_CR	0x0000	RW	ADC Control Register	0x0000_0100	15.5.1
ADC_MR	0x0004	RW	ADC Mode Register	0x0000_40FF	15.5.2
ADC_DR	0x0008	RO	ADC Data Register	0x0000_0000	15.5.3

15.5.1 ADC_CR: ADC Control Register

This register has the function to control the operation of the ADC and to check the interrupt and status flags.

ADC_MR=0x4000_0E00

31 30 29 28 27 26 25 24	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			ADEOC	ADST	AFLAG	ADIF	Reserved	ADSEL[3:0]
-			1	0	0	0	-	0x0
-			RO	WO	RO	RC	-	RW

8	ADEOC	A/D conversion status (Read/Write)
	0	No conversion or converting state
	1	End-of-Conversion
7	ADST	Start/Stop the A/D conversion
	0	Stop A/D Conversion
	1	Start A/D Conversion
6	AFLAG	The status flag of A/D conversion (Read-only)
	0	Conversion Busy
	1	Conversion Idle
5	ADIF	A/D conversion
	0	No A/D interrupt
	1	Occurred A/D interrupt (Write '1' to clear this bit)
3 0	ADSEL	Select ADC input channel ANn input select (n = 0 ~ 15)

15.5.2 ADC_MR: ADC Mode Register

This register provides settings for ADC operating clock and operating mode, and interrupt.

ADC_MR=0x4000_0E04

31 30 29 28 27 26 25 24	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		ADCEN	ADSTBY	Reserved	ADIE	EXTRG	TSEL[2:0]	ADCS[7:0]
-		0	1	-	0	0	000	0xFF
-		RW	RW	-	RW	RW	RW	RW

15	ADCEN	Enable/Disable A/D converter	
		0	Disable A/D converter
		1	Enable A/D converter
14	ADSTBY	A/D converter stand-by mode	
		0	ADC normal operation
		1	ADC stand-by mode
13	Reserved	Always ‘0’	
12	ADIE	Enable/Disable end-of-A/D conversion interrupt	
		0	Disable ADC interrupt
		1	Enable ADC interrupt
11	EXTRG	Select external trigger condition for start conversion	
		0	Trigger by ADST bit
		1	Trigger by external trigger source
10 8	TSEL[2:0]	A/D conversion trigger source	
		000	Timer 0 match
		001	Timer 1 match
		010	Timer 2 match
		011	Timer 3 match
		100	Timer 4 match
		101	Timer 5 match
		110	Timer 6 match
		111	Timer 7 match
7 0	ADCS[7:0]	Select A/D conversion clock $\mathrm{ADCLK} = \frac{\mathrm{PCLK}}{\mathrm{ADCS} + 1}$ The ADC clock should be under 4MHz (in case of VDD=5V)	

15.5.3 ADC_DR: ADC Data Register

This register is a 12-bit read-only register that stores the converted data values from the ADC.

ADC_DR=0x4000_0E08

31 30 29 28 27 26 25 24	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		ADHR[7:0]	ADLR[3:0]	Reserved
-		0x00	0x0	-
-		RO	RO	-

15 8	ADHR[7:0]	Upper byte of A/D conversion data
7 4	ADLR[3:0]	Lower byte of A/D conversion data

15.5.4 ADC Register Map Summary

Table 114. 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	ADC_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADEOC	ADST	AFLAG	ADIF	Res.	ADSEL[3:0]			
	Reset value																								0	0	0	0		0	0	0	0
0x04	ADC_MR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADCEN	ADSTBY	Res.	ADIE	EXTRG	TSEL[2:0]			ADCS[7:0]							
	Reset value																	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	ADC_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADHR[7:0]								ADLR[3:0]				Res.	Res.	Res.	Res.
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0				

16. Internal SRAM

16.1 Introduction

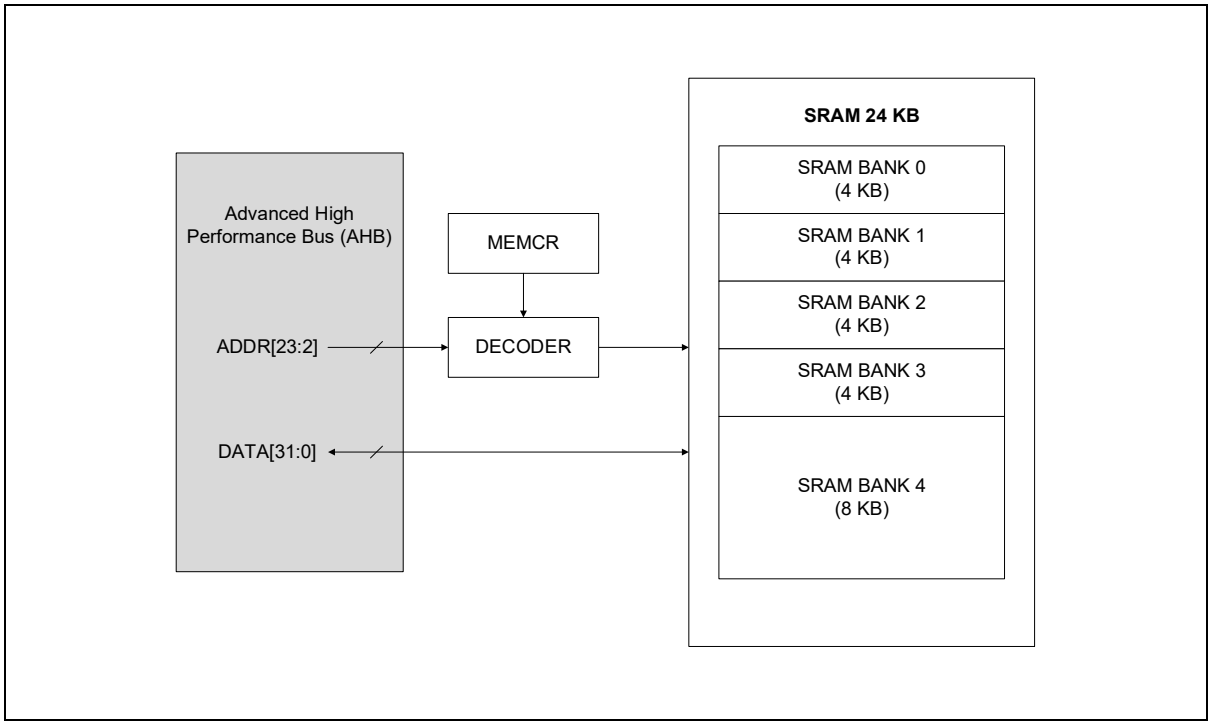
The A33G53x has a built-in 24-Kbyte SRAM capable of data and code area.

16.2 SRAM Functional Description

16.2.1 SRAM Block Diagram

Figure 109 shows a block diagram of the 24 KB SRAM.

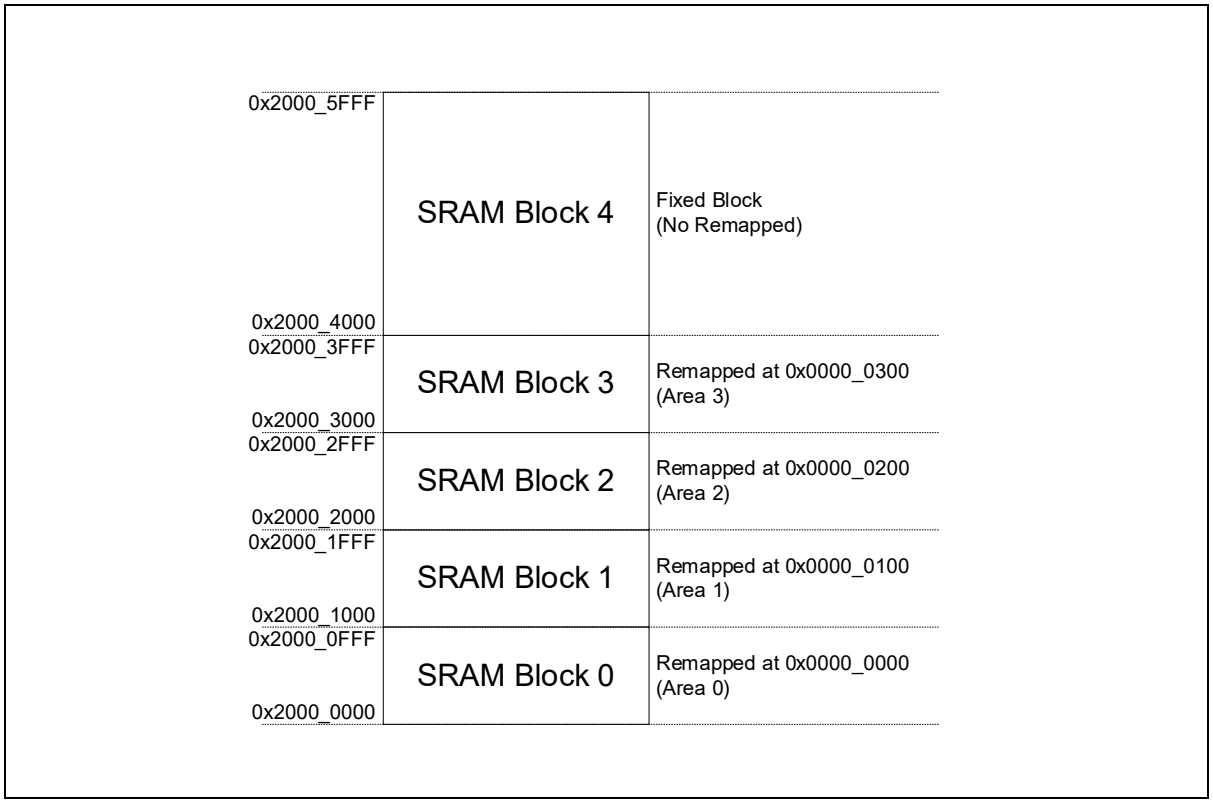
Figure 111. 24 KB SRAM Block Diagram



16.2.2 Bank Configuration of SRAM

The SRAM consists of several blocks as below.

Figure 112. Bank Configuration of SRAM



16.3 SRAM Registers

The base addresses and register map of the SRAM are as follows:

Table 115. Base Addresses of SRAM

Name	Base address
SRAM	0x4000_00E0

Table 116. ADC Register Map

Name	Offset	Type	Description	Reset value	Reference
SRAM_CR	0x0000	RW	SRAM memory Control Register	0x0000_0000	15.5.1

16.3.1 SRAM_CR: SRAM Memory Control Register

The SRAM_CR register provides the mapping control of the corresponding area in units of 4 KB of the maximum 16 KB SRAM area.

SRAM_CR=0x4000_00E0

31 30 29 28 27 26 25 24	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				REMAP3	REMAP2	REMAP1	REMAP0
-				0	0	0	0
-				RW	RW	RW	RW

3	REMAP3	Remap 0x00003000 ~ 0x00003FFF Area (Area 3)	
		0	Set Flash ROM at relevant area
		1	Remap SRAM at relevant area
2	REMAP2	Remap 0x00002000 ~ 0x00002FFF Area (Area 2)	
		0	Set Flash ROM at relevant area
		1	Remap SRAM at relevant area
1	REMAP1	Remap 0x00001000 ~ 0x00001FFF Area (Area 1)	
		0	Set Flash ROM at relevant area
		1	Remap SRAM at relevant area
0	REMAP0	Remap 0x00000000 ~ 0x00000FFF Area (Area 0)	
		0	Set Flash ROM at relevant area
		1	Remap SRAM at relevant area

16.3.2 SRAM Register Map Summary

Table 117. SRAM 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	SRAM_CR	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.	REMAP3	REMAP2	REMAP1	REMAP0
	Reset value																													0	0	0	0

Abbreviations for Registers

The following abbreviations are used in register descriptions:

read / write (RW)	Software can read and write to this bit.
read-only (RO)	Software can only read this bit.
write-only (WO)	Software can only write to this bit. Reading this bit returns the reset value.
read / clear (RC)	Software can read to clear this bit.
read / write 0 clear (RWC0)	Software can read and clear this bit by writing 0.
read / write 1 clear (RWC1)	Software can read and clear this bit by writing 1.
Reserved (Res.)	Reserved bit must be kept at reset value.

Glossary

This section gives a brief definition of acronyms and abbreviations used in this document:

1. **Word**: Data of 32-bit length.
2. **Half-word**: Data of 16-bit length.
3. **Byte**: Data of 8-bit length.
4. **PGM**: Flash programming.
5. **AHB**: Advanced high-performance bus.
6. **APB**: Advanced peripheral bus.

Revision History

Revision	Date	Description
1.00	Dec. 28. 2023	Initial 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
93, 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