S3FM02G

32-Bit CMOS MICROCONTROLLERS

Revision 1.00

January 2011

User's Manual

SAMSUNG ELECTRONICS RESERVES THE RIGHT TO CHANGE PRODUCTS, INFORMATION AND SPECIFICATIONS WITHOUT NOTICE.

Products and specifications discussed herein are for reference purposes only. All information discussed herein is provided on an "AS IS" basis, without warranties of any kind.

This document and all information discussed herein remain the sole and exclusive property of Samsung Electronics. No license of any patent, copyright, mask work, trademark or any other intellectual property right is granted by one party to the other party under this document, by implication, estoppel or otherwise.

Samsung products are not intended for use in life support, critical care, medical, safety equipment, or similar applications where product failure could result in loss of life or personal or physical harm, or any military or defense application, or any governmental procurement to which special terms or provisions may apply.

For updates or additional information about Samsung products, contact your nearest Samsung office.

All brand names, trademarks and registered trademarks belong to their respective owners.

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. Samsung assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained herein.

Samsung reserves the right to make changes in its products or product specifications with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

This publication does not convey to a purchaser of semiconductor devices described herein any license under the patent rights of Samsung or others.

Samsung makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Samsung assume any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

"Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by the customer's technical experts.

Samsung products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, for other applications intended to support or sustain life, or for any other application in which the failure of the Samsung product could create a situation where personal injury or death may occur.

Should the Buyer purchase or use a Samsung product for any such unintended or unauthorized application, the Buyer shall indemnify and hold Samsung and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorized use, even if such claim alleges that Samsung was negligent regarding the design or manufacture of said product.

Samsung Electronics Co., Ltd.
San #24 Nongseo-Dong, Giheung-Gu
Yongin-City, Gyeonggi-Do, Korea 446-711

Contact Us: prodakim @samsung.com
TEL: (82)-(31)-209-4956
FAX: (82)-(31)-209-6494

Home Page: http://www.samsungsemi.com

Revision History

Revision No.	Date	Description	Author(s)
1.00	Jan. 10, 2011	•	Juil. Kim

Table of Contents

List of Figures

Figure Number | Title | Page Number

List of Tables

List of Conventions

Register RW Access Type Conventions

Type	Definition	Description
R	Read Only	The application has permission to read the Register field. Writes to read-only fields have no effect.
W	Write Only	The application has permission to write in the Register field.
RW	Read & Write	The application has permission to read and writes in the Register field. The application sets this field by writing 1’b1 and clears it by writing 1’b0.

Register Value Conventions

Expression	Description
x	Undefined bit
X	Undefined multiple bits
?	Undefined, but depends on the device or pin status
Device dependent	The value depends on the device
Pin value	The value depends on the pin status

Reset Value Conventions

Expression	Description
0	
1	
x	

Warning: Some bits of control registers are driven by hardware or write only. As a result the indicated reset value and the read value after reset might be different.

1 Overview

1.1 Purpose This Document

The purpose of this document is to provide a complete reference specification of S3FM02G.

1.2 Instruction to S3FM02G

S3FM02G is a family of cost-effective and high-performance microcontrollers with Cortex^{TM}-M3 designed by Advanced RISC Machines (ARM). This Microcontroller unit (MCU) applies to inverter motor control within the home appliance applications.

- ARM Cortex^{TM}-M3 Core
- Built-in up to 384 Kbytes Program Flash Memory
- Built-in up to 16 Kbytes Data Flash Memory
- Internal up to 24 Kbytes SRAM for stack, data memory, or code memory
- Operating temperature: -40 ~ 85 °C
- Operating voltage range: 2.7 ~ 5.5 V
- Interrupt controller: Dynamically reconfigurable Nested Vectored Interrupt Controller (NVIC)
- Clock and Power Controller (CM)
- 10ch x DMA Controller (DMAC)
- Watch-Dog Timer (WDT)
- 8ch x 16-bit Timer/Counters (TC)
- 32-bit Free-Running Timer (FRT)
- 8ch x 16-bit PWM
- 2ch x 16bit Encoder Counter (ENC)
- 2ch x 6-Phase Inverter Motor Controller (IMC)
- 2ch x I2C, 2ch x SSP, 2ch x CAN and 4ch x USART
- 12-bit ADC(4 channels with OP-AMP)
- 10-bit ADC
- 5ch OP-AMP
- 4com x 40seg LCD Controller (LCDC)
- Support Normal, High-speed, IDLE, and STOP mode

1.3 Features

- CPU
  - 32-bit RISC ARM Cortex^{TM}-M3 Core
  - ETM function embedded with ARM Cortex^{TM}-M3
  - SWD(Serial Wire Debug) and JTAG Debugging Solution

- Memory
  - Up to 384 Kbytes Internal Program Full Flash
  - Up to 16 Kbytes Internal Data Flash
  - Up to 24 Kbytes Internal SRAM
  - Only little-endian support

- Interrupt Controller
  - Supports Nested Vectored Interrupt Controller of Cortex^{TM}-M3
  - Dynamically reconfigurable Interrupt Priority (16 priority levels)

- Clock Manager (CM)
  - External Oscillator 4 ~ 8MHz (EMCLK: External Main Clock) and 32.768KHz (ESCLK: External Sub-Clock)
  - Internal Oscillator 8/16/20MHz (IMCLK: Internal Main Clock) and 32.768KHz (ISCLK: Internal Sub-Clock)
  - Up to 75MHz by Phase-Locked Loop Control (PLL)
  - Clock Monitor to detect an external main and sub-oscillator failure
  - Support Low Power Mode (IDLE / STOP) by Clock Gating Control
  - Programmable Clock Dividers (SDIV, PDIV)
  - Reset Management
  - Include basic timer for reset generation and STOP wake-up

- DMAC: Direct Memory Access Controller
  - Up to 10 channels
  - Transfer from Memory to Memory
  - Transfer between Peripheral and Memory
  - Transfer between Peripheral and Peripheral

- WDT: Watchdog Timer
  - Configurable micro-controller reset event
  - Programmable 16-bit down counter

- TC: 16-bit Timer/Counter
  - Up to 8 channels (TC0 ~ TC7)
  - Operation in an interval, capture, match & overflow, or PWM mode
  - Match and overflow interrupt
  - Selectable an internal or external timer clock

- FRT: Free Running Timer
  - 32-bit Timer
  - Can operate in stop mode with ISCLK, as an independent timer

- PWM: Pulse Width Modulation
  - Up to 8 channels
  - 16-bit PWM signal generation
  - Interval Mode
  - Programmable Idle Level
  - Support extension PWM function

- ENC: Encoder Counter
  - Up to 2 channels
  - 3 input signals: PHASEA, PHASEB, and PHASEZ
  - Support position counter and speed counter
  - Up/Down counter
  - Support capture mode

- IMC: Inverter Motor Controller
  - UP to 2 channels
  - Support 3-Phase 16-bit PWM generation
  - Programmable dead time insertion
  - ADC conversion start signal generation

- CAN: Controller Area Network
  - CAN0/1 With 32 Buffers
  - Support CAN 2.0A and 2.0B Full Speed
  - Stampable Message

- USART: Universal Sync/Async Receiver Transmitter
  - Up to 4 channels
  - Support 5, 6, 7, and 8bit Data length
  - Programmable baud rate generator
  - Parity, framing and overrun error detection
  - Support loop-back mode
  - Support full duplex
  - Idle flag for J1587 protocol
  - Support LIN protocol: LIN1.2 or LIN 2.0 configurable release
  - Smart-card protocol: Error signaling and re-transmission
  - Dedicated DMA channel

- SSP: Serial Synchronous Peripheral Interface
  - Up to 2 channels
  - Programmable data frame from 4 to 16-bit
  - Support Master and Slave Mode
  - Programmable Clock Pre-scale
  - Separate 16 x 32-bit width Transmit/Receive FIFO
  - Dedicated DMA channel

- IIC: Inter-Integrated Circuit
  - Up to 2 channels
  - Multi-Master IIC-Bus
  - Serial, 8-bit Oriented and Bi-directional Data Transfers
  - 100Kbit/s in Standard Mode and up to 400Kbit/s in Fast mode
  - Dedicated DMA channel

- ADC: A/D Converters
  - Up to 16 channel’s Analog Inputs
  - 12-bit ADC0 x 2 and 10-bit ADC1
  - 4 x Op-amp for 12-bit ADC input level amplification
  - supports simultaneous sampling and conversion up to 2-input channels
  - Dedicated DMA channel

- OP amp
  - 5 ch OP amp
  - Can be operated with ADC
  - Edge detection function which is related with IMC

- LCDC: LCD Controller
  - 4 com x 40 segment
  - Static, 1/2 and 1/3 bias mode
  - External and internal resistor bias

- General Purpose IO (GPIO)
  - Disabling IO port enables the function of peripherals on pins
  - Output Open-drain / Push-pull configuration
  - Input Pull-up Resistor enable/disable configuration
  - GPIO Interrupt

- Two Low Power Modes
  - IDLE: Only CPU clock stops
  - STOP: Selected system clock and CPU clock stop
  - Fast wake-up with internal 8/16/20MHz oscillator (from STOP mode to normal mode)
  - Programmable external event/interrupt sources for Wake-up

- POR: Power-On Reset

- LVD: Low Voltage Detection
  - LVD for reset with configurable voltage levels
  - LVD for interrupt with configurable voltage levels

- PLL: Phase-Locked Loop
  - Input Frequency: 4 ~ 8 MHz
  - Output Frequency: 8 ~ 75MHz

- Operating Voltage Range
  - 2.7V ~ 5.5V

- Operating Frequency Range
  - 4 ~ 8 MHz by external main oscillator clock
  - 8/16/20MHz by internal main oscillator clock
  - 32.768KHz external/internal sub-oscillator clock
  - 8 ~ 75 MHz by PLL clock

- Operating Temperature Range
  - -40 ~ 85 °C

- Available in 128 ETQFP Package

1.4 Block Diagram

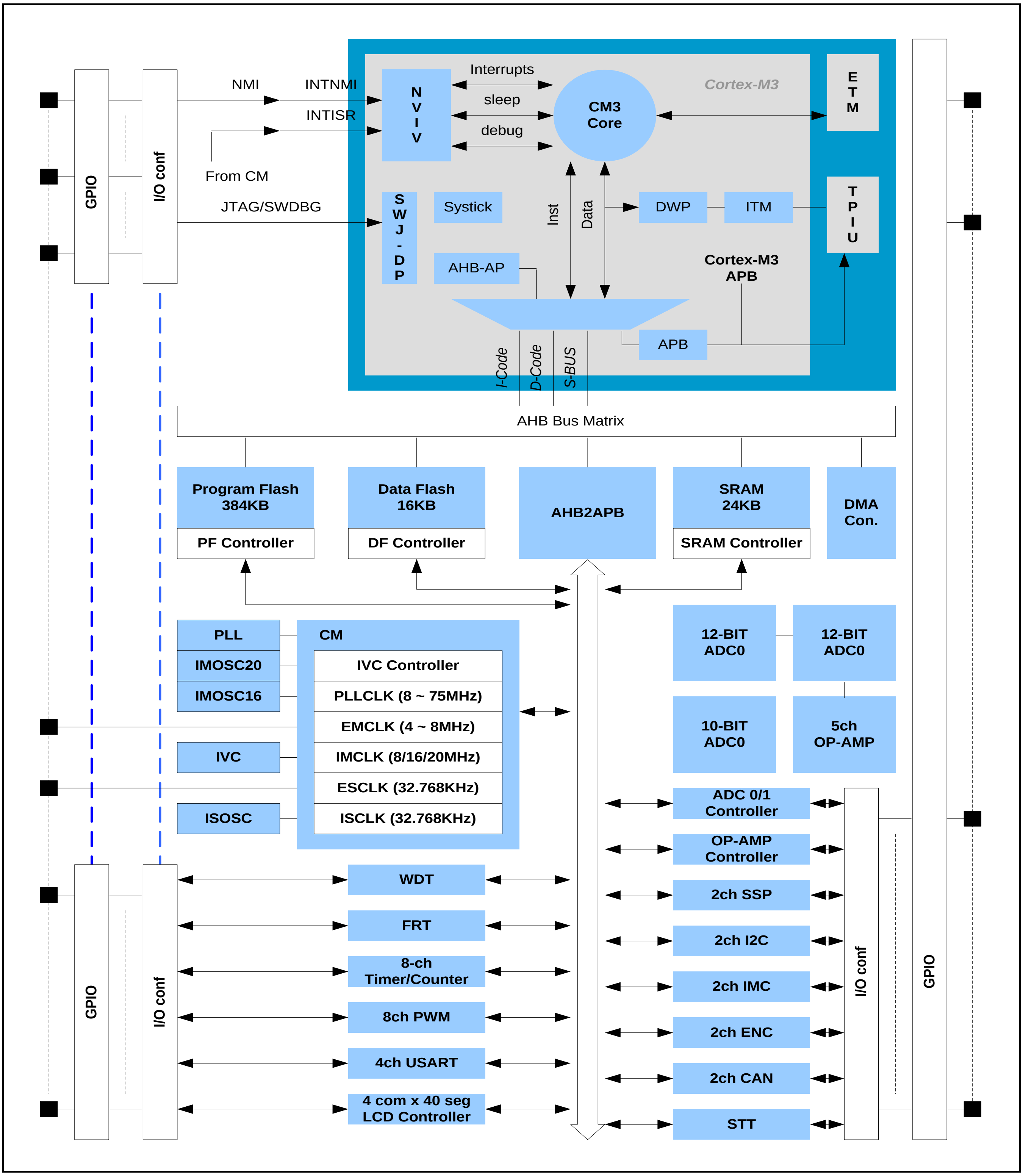


Figure 1-1 S3FM02G Block Diagram

2 Pin Configuration

2.1 Pin Configuration

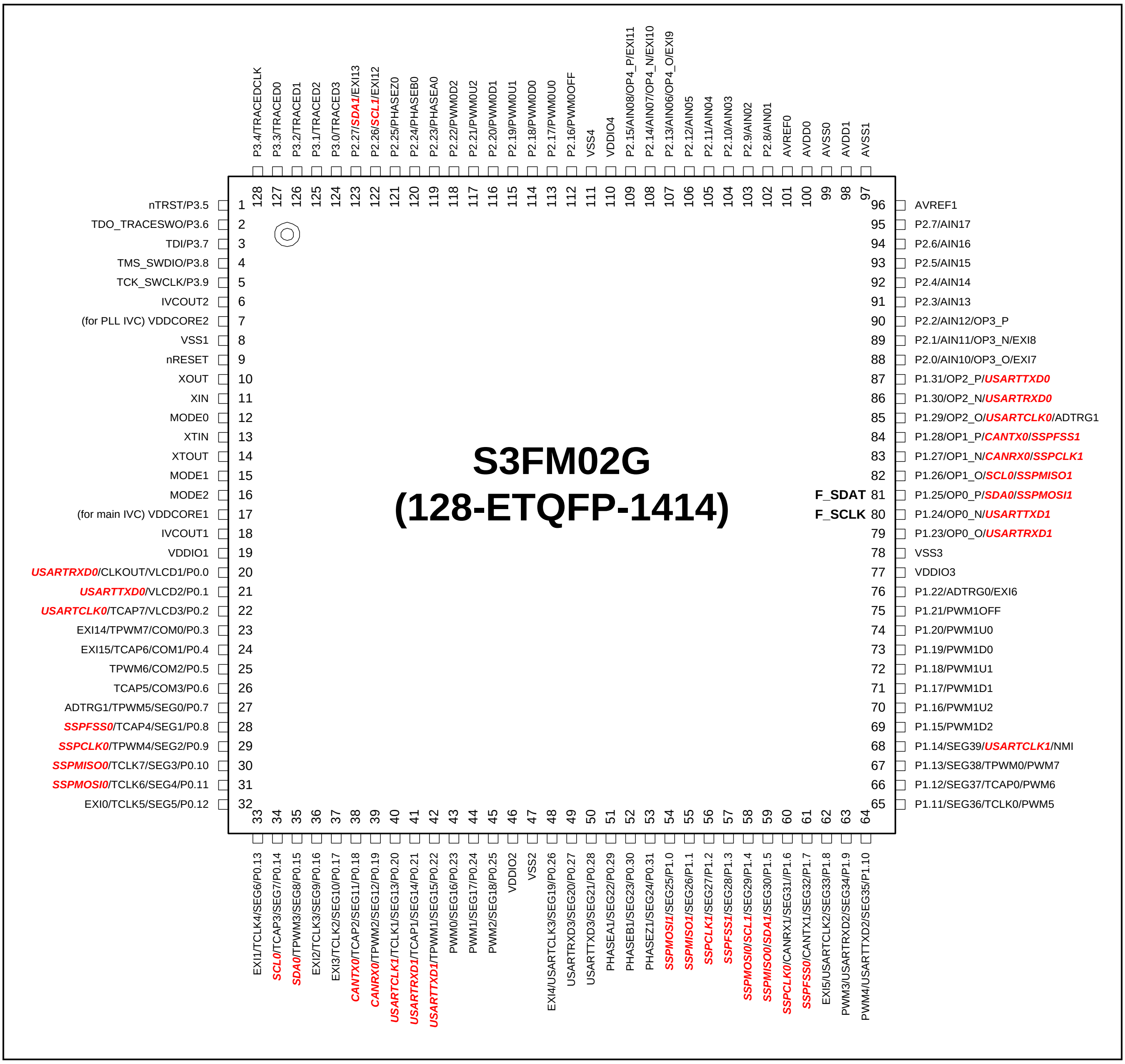


Figure 2-1 Pin Configuration

2.2 Pin Assignments

- D: Digital, A: Analog
- IO: Input and Output (Bi-direction), O: Output, I: Input, P: Power, G: Ground

Table 2-1 Pin Assignments – Pin Number Order

Num	PIN Name				Default @RESET	PULL up/dn @RESET	I/O state @RESET
	1st	2nd	3rd	4th			
1	P3.5	nTRST	–	–	nTRST	PULL-UP	I
2	P3.6	TDO/TRACESWO	–	–	TDO/ TRACESWO	–	O
3	P3.7	TDI	–	–	TDI	PULL-UP	I
4	P3.8	TMS/SWDIO	–	–	TMS/SWDIO	PULL-UP	I
5	P3.9	TCK/SWCLK	–	–	TCK/SWCLK	PULL-UP	I
6	IVCOUT2	IVCOUT2	IVCOUT2	IVCOUT2	IVCOUT2	–	O
7	VDDCORE2	VDDCORE2	VDDCORE2	VDDCORE2	VDDCORE2	–	P
8	VSS1	VSS1	VSS1	VSS1	VSS1	–	G
9	nRESET	nRESET	nRESET	nRESET	nRESET	PULL-UP	I
10	X_{OUT}	X_{OUT}	X_{OUT}	X_{OUT}	X_{OUT}	–	O
11	X_{IN}	X_{IN}	X_{IN}	X_{IN}	X_{IN}	–	I
12	MODE0	MODE0	MODE0	MODE0	MODE0	PULL-DN	I
13	X_{TIN}	X_{TIN}	X_{TIN}	X_{TIN}	X_{TIN}	–	I
14	X_{TOUT}	X_{TOUT}	X_{TOUT}	X_{TOUT}	X_{TOUT}	–	O
15	MODE1	MODE1	MODE1	MODE1	MODE1	PULL-DN	I
16	MODE2	MODE2	MODE2	MODE2	MODE2	PULL-DN	I
17	VDDCORE1	VDDCORE1	VDDCORE1	VDDCORE1	VDDCORE1	–	P
18	IVCOUT1	IVCOUT1	IVCOUT1	IVCOUT1	IVCOUT1	–	O
19	VDDIO1	VDDIO1	VDDIO1	VDDIO1	VDDIO1	–	P
20	P0.0	VLCD1	CLKOUT	***USARTRXD0***	P0.0	–	I
21	P0.1	VLCD2	–	***USARTTXD0***	P0.1	–	I
22	P0.2	VLCD3	TCAP7	***USARTCLK0***	P0.2	–	I
23	P0.3	COM0	TPWM7	EXI14	P0.3	–	I
24	P0.4	COM1	TCAP6	EXI15	P0.4	–	I
25	P0.5	COM2	TPWM6	–	P0.5	–	I
26	P0.6	COM3	TCAP5	–	P0.6	–	I
27	P0.7	SEG0	TPWM5	ADTRG1	P0.7	–	I
28	P0.8	SEG1	TCAP4	***SSPFSS0***	P0.8	–	I
29	P0.9	SEG2	TPWM4	***SSPCLK0***	P0.9	–	I

Num	PIN Name				Default @RESET	PULL up/dn @RESET	I/O state @RESET
	1st	2nd	3rd	4th			
30	P0.10	SEG3	TCLK7	***SSPMISO0***	P0.10	–	I
31	P0.11	SEG4	TCLK6	***SSPMOSI0***	P0.11	–	I
32	P0.12	SEG5	TCLK5	EXI0	P0.12	–	I
33	P0.13	SEG6	TCLK4	EXI1	P0.13	–	I
34	P0.14	SEG7	TCAP3	***SCL0***	P0.14	–	I
35	P0.15	SEG8	TPWM3	***SDA0***	P0.15	–	I
36	P0.16	SEG9	TCLK3	EXI2	P0.16	–	I
37	P0.17	SEG10	TCLK2	EXI3	P0.17	–	I
38	P0.18	SEG11	TCAP2	***CANTX0***	P0.18	–	I
39	P0.19	SEG12	TPWM2	***CANRX0***	P0.19	–	I
40	P0.20	SEG13	TCLK1	***USARTCLK1***	P0.20	–	I
41	P0.21	SEG14	TCAP1	***USARTRXD1***	P0.21	–	I
42	P0.22	SEG15	TPWM1	***USARTTXD1***	P0.22	–	I
43	P0.23	SEG16	PWM0	–	P0.23	–	I
44	P0.24	SEG17	PWM1	–	P0.24	–	I
45	P0.25	SEG18	PWM2	–	P0.25	–	I
46	VDDIO2	VDDIO2	VDDIO2	VDDIO2	VDDIO2	–	P
47	VSS2	VSS2	VSS2	VSS2	VSS2	–	G
48	P0.26	SEG19	USARTCLK3	EXI4	P0.26	–	I
49	P0.27	SEG20	USARTRXD3	–	P0.27	–	I
50	P0.28	SEG21	USARTTXD3	–	P0.28	–	I
51	P0.29	SEG22	PHASEA1	–	P0.29	–	I
52	P0.30	SEG23	PHASEB1	–	P0.30	–	I
53	P0.31	SEG24	PHASEZ1	–	P0.31	–	I
54	P1.0	SEG25	***SSPMOSI1***	–	P1.0	–	I
55	P1.1	SEG26	***SSPMISO1***	–	P1.1	–	I
56	P1.2	SEG27	***SSPCLK1***	–	P1.2	–	I
57	P1.3	SEG28	***SSPFSS1***	–	P1.3	–	I
58	P1.4	SEG29	***SCL1***	***SSPMOSI0***	P1.4	–	I
59	P1.5	SEG30	***SDA1***	***SSPMISO0***	P1.5	–	I
60	P1.6	SEG31	CANRX1	***SSPCLK0***	P1.6	–	I
61	P1.7	SEG32	CANTX1	***SSPFSS0***	P1.7	–	I
62	P1.8	SEG33	USARTCLK2	EXI5	P1.8	–	I
63	P1.9	SEG34	USARTRXD2	PWM3	P1.9	–	I
64	P1.10	SEG35	USARTTXD2	PWM4	P1.10	–	I

Num	PIN Name				Default @RESET	PULL up/dn @RESET	I/O state @RESET
	1st	2nd	3rd	4th			
65	P1.11	SEG36	TCLK0	PWM5	P1.11	–	I
66	P1.12	SEG37	TCAP0	PWM6	P1.12	–	I
67	P1.13	SEG38	TPWM0	PWM7	P1.13	–	I
68	P1.14	SEG39	***USARTCLK1***	NMI	P1.14	–	I
69	P1.15	PWM1D2	–	–	P1.15	–	I
70	P1.16	PWM1U2	–	–	P1.16	–	I
71	P1.17	PWM1D1	–	–	P1.17	–	I
72	P1.18	PWM1U1	–	–	P1.18	–	I
73	P1.19	PWM1D0	–	–	P1.19	–	I
74	P1.20	PWM1U0	–	–	P1.20	–	I
75	P1.21	PWM1OFF	–	–	P1.21	–	I
76	P1.22	ADTRG0	EXI6	–	P1.22	–	I
77	VDDIO3	VDDIO3	VDDIO3	VDDIO3	VDDIO3	–	P
78	VSS3	VSS3	VSS3	VSS3	VSS3	–	G
79	P1.23	OP0_O	***USARTRXD1***	–	P1.23	–	I
80	P1.24	OP0_N	***USARTTXD1***	–	P1.24	–	I
81	P1.25	OP0_P	***SDA0***	***SSPMOSI1***	P1.25	–	I
82	P1.26	OP1_O	***SCL0***	***SSPMISO1***	P1.26	–	I
83	P1.27	OP1_N	***CANRX0***	***SSPCLK1***	P1.27	–	I
84	P1.28	OP1_P	***CANTX0***	***SSPFSS1***	P1.28	–	I
85	P1.29	OP2_O	***USARTCLK0***	ADTRG1	P1.29	–	I
86	P1.30	OP2_N	***USARTRXD0***	–	P1.30	–	I
87	P1.31	OP2_P	***USARTXD0***	–	P1.31	–	I
88	P2.0	AIN10	OP3_O	EXI7	P2.0	–	I
89	P2.1	AIN11	OP3_N	EXI8	P2.1	–	I
90	P2.2	AIN12	OP3_P	–	P2.2	–	I
91	P2.3	AIN13	–	–	P2.3	–	I
92	P2.4	AIN14	–	–	P2.4	–	I
93	P2.5	AIN15	–	–	P2.5	–	I
94	P2.6	AIN16	–	–	P2.6	–	I
95	P2.7	AIN17	–	–	P2.7	–	I
96	AVREF1	AVREF1	AVREF1	AVREF1	AVREF1	–	I
97	AVSS1	AVSS1	AVSS1	AVSS1	AVSS1	–	G
98	AVDD1	AVDD1	AVDD1	AVDD1	AVDD1	–	P
99	AVSS0	AVSS0	AVSS0	AVSS0	AVSS0	–	G

Num	PIN Name				Default @RESET	PULL up/dn @RESET	I/O state @RESET
	1st	2nd	3rd	4th			
100	AVDD0	AVDD0	AVDD0	AVDD0	AVDD0	–	P
101	AVREF0	AVREF0	AVREF0	AVREF0	AVREF0	–	I
102	P2.8	AIN01	–	–	P2.8	–	I
103	P2.9	AIN02	–	–	P2.9	–	I
104	P2.10	AIN03	–	–	P2.10	–	I
105	P2.11	AIN04	–	–	P2.11	–	I
106	P2.12	AIN05	–	–	P2.12	–	I
107	P2.13	AIN06	OP4_O	EXI9	P2.13	–	I
108	P2.14	AIN07	OP4_N	EXI10	P2.14	–	I
109	P2.15	AIN08	OP4_P	EXI11	P2.15	–	I
110	VDDIO4	VDDIO4	VDDIO4	VDDIO4	VDDIO4	–	P
111	VSS4	VSS4	VSS4	VSS4	VSS4	–	G
112	P2.16	PWM0OFF	–	–	P2.16	–	I
113	P2.17	PWM0U0	–	–	P2.17	–	I
114	P2.18	PWM0D0	–	–	P2.18	–	I
115	P2.19	PWM0U1	–	–	P2.19	–	I
116	P2.20	PWM0D1	–	–	P2.20	–	I
117	P2.21	PWM0U2	–	–	P2.21	–	I
118	P2.22	PWM0D2	–	–	P2.22	–	I
119	P2.23	PHASEA0	–	–	P2.23	–	I
120	P2.24	PHASEB0	–	–	P2.24	–	I
121	P2.25	PHASEZ0	–	–	P2.25	–	I
122	P2.26	***SCL1***	EXI12	–	P2.26	–	I
123	P2.27	***SDA1***	EXI13	–	P2.27	–	I
124	P3.0	TRACED3	–	–	P3.0	–	I
125	P3.1	TRACED2	–	–	P3.1	–	I
126	P3.2	TRACED1	–	–	P3.2	–	I
127	P3.3	TRACED0	–	–	P3.3	–	I
128	P3.4	TRACECLK	–	–	P3.4	–	I

2.3 Pin Description

- D: Digital, A: Analog
- IO: Input and Output (Bi-direction), O: Output, I: Input, P: Power, G: Ground

2.3.1 Miscellaneous

Name	I/O	Description	D/A
nRESET	I	Chip Reset Signal (active “Low”) This nRESET pin contains an internal pull up resistor 250. Setting this pin to low level initialize the internal state of the device. Thereafter, setting the input to high release the reset status. The S3FM02G waits for the system clock to be stable, and the PC to the reset interrupt vector. Internal Reset is generated after clock stabilization.	D
MODE[2:0]	I	Factory test input pins. This pins should be connected to Ground	D

2.3.2 Clock Manager

Name	I/O	Description	D/A
X_{IN}	I	External Main Oscillator Input (4MHz ~ 8MHz)	A
X_{OUT}	O	External Main Oscillator Output	A
XT_{IN}	I	External Sub-Oscillator Input (32.768KHz)	A
XT_{OUT}	O	External Sub-Oscillator Output.	A
CLKOUT	O	Internal Clock Output signals.	D

2.3.3 External Interrupt

Name	I/O	Description	D/A
EXI0 ~ EXI15	I	External interrupt/wakeup input pins	D

2.3.4 DEBUG Interface

Name	I/O	Description	D/A
TCK/SWCLK	I	JTAG Test Clock/ Serial Wire Clock, Pull-up	D
TMS/SWDIO	I	JTAG Test Mode Select/ Serial Wire Data Input Output, Pull-up	D
TDI	I	JTAG Test Data Input, Pull-up	D
TDO/TRACESWO	O	JTAG Test Data Out / Trace Serial Wire Viewer Data	D
nTRST	I	JTAG Test nRESET, Pull-up	D
TRACECLK	O	Trace Clock	D
TRACED[3:0]	O	Trace Data	D

2.3.5 USART Interface

Name	I/O	Description	D/A
USARTCLK[3:0]	IO	External Clock Signal Input / Internal Clock Signal Output	D
USARTRXD[3:0]	I	Receive Data Input	D
USARTTXD[3:0]	O	Transmit Data Output	D

2.3.6 Encoder Interface

Name	I/O	Description	D/A
PHASEA[1:0]	I	Phase A input pin	D
PHASEB[1:0]	I	Phase B input pin	D
PHASEZ[1:0]	I	Phase Z input pin	D

2.3.7 IMC Interface

Name	I/O	Description	D/A
PWM[1:0]U[2:0]	O	PWM output for inverter motor	D
PWM[1:0]D[2:0]	O	PWM output for inverter motor	D
PWM[1:0]OFF	I	Input pin for PWM output off	D

2.3.8 TIMER Interface

Name	I/O	Description	D/A
TCLK[7:0]	I	External Clock Input	D
TCAP[7:0]	I	External Capture Input	D
TPWM[7:0]	O	PWM Output	D

2.3.9 PWM Interface

Name	I/O	Description	D/A
PWM[7:0]	O	Pulse width modulation output	D

2.3.10 I2C Interface

Name	I/O	Description	D/A
SCL[1:0]	IO	Serial Clock	D
SDA[1:0]	IO	Serial Data	D

2.3.11 CAN Interface

Name	I/O	Description	D/A
CANTX[1:0]	O	Transmit Data Output	D
CANRX[1:0]	I	Receive Data Input	D

2.3.12 SSP Interface

Name	I/O	Description	D/A
SSPCLK[1:0]	IO	SSP Master Clock Output / Slave Clock Input	D
SSPMISO[1:0]	IO	SSP Master Input / Slave Output	D
SSPMOSI[1:0]	IO	SSP Master Output / Slave Input	D
SSPFSS[1:0]	IO	SSP Master Chip Select Output / Slave Chip Select Input	D

2.3.13 LCD Controller Interface

Name	I/O	Description	D/A
VLCD[3:1]	P	LCD Power Supply	D
COM[3:0]	O	COM drive signal	D
SEG[39:0]	O	SEG drive signal	D

2.3.14 ADC Interface

Name	I/O	Description	D/A
AIN0[8:1]	I	Analog Input pins for 8-channels	A
AIN1[7:0]	I	Analog Input pins for 8-channels	A
ADTRG[1:0]	I	ADC External Trigger Input pin	D
AVREF[1:0]	I	ADC Reference Top Voltage.	A

2.3.15 OP-AMP Interface

Name	I/O	Description	D/A
OP[4:0]_O	O	OP-AMP Output	A
OP[4:0]_N	I	OP-AMP N Input	A
OP[4:0]_P	I	OP-AMP P Input	A

2.3.16 GPIOs

Name	I/O	Description	D/A
P0[31:0]	IO	General Purpose I/O port 0	D
P1[31:0]	IO	General Purpose I/O port 1	D
P2[27:0]	IO	General Purpose I/O port 2	D
P3[9:0]	IO	General Purpose I/O port 3	D

2.3.17 FLASH

Name	I/O	Description	D/A
F_SDAT	IO	Serial Data pin (Output when reading, Input when writing)	D
F_SCLK	I	Serial Clock	D

2.3.18 Power

Name	I/O	Description
VDDCORE[2:1]	P	Digital Power for interval IVC (2.7V ~ 5.5V)
VDDIO[4:1]	P	Digital IO Power 2.7 V ~ 5.5V
VSS[4:1]	G	Digital Ground
IVCOUT[2:1]	P	Cap Output port from internal Regulators (connect to GND through a 1uF capacitor)
AVDD[1:0]	P	Analog Power
AVSS[1:0]	G	Analog Ground

MODE2	MODE1	MODE0	Mode Description
0	0	0	User Normal/Debug Mode
0	0	1	User Flash Writing Tool Mode
0	1	0	User UART SPGM Tool Mode
1	0	1	SCAN Mode (Only for Test)

3 System Memory Management

3.1 Default Memory Map

The S3FM02G has memory space allocation as below.

Table 3-1 Memory Map

Address	Memory
Reserved	Reserved
0xE00F_FFFF ~ 0xE000_0000	Cortex$^{\text{TM}}$-M3 Internal Peripheral Registers
Reserved	Reserved
0x8000_3FFF ~ 0x8000_0000	16 Kbytes Internal Data Flash Memory
Reserved	Reserved
0x400F_FFFF ~ 0x4000_0000	Special Function Registers
Reserved	Reserved
0x2000_5FFF ~ 0x2000_0000	24 Kbytes Internal SRAM Memory
Reserved	Reserved
0x0005_FFFF ~ 0x0000_0000	384 Kbytes Internal Program Flash Memory

3.2 Special Function Register Map

3.2.1 Core Special Function Register Map

Table 3-2 Core Special Function Register Map

Base Address	Peripheral	Description
0xE00F_F000	ROM Table	ROM memory table
0xE004_2000	External PPB	Private Peripheral Bus
0xE004_1000	ETM	Embedded Trace Macro Register
0xE004_0000	TPIU	Trace Port Interface
0xE000_F000	Reserved	–
0xE000_E000	SCS	System Control Space
0xE000_3000	Reserved	–
0xE000_2000	FPB	Flash Patch & Break Pint
0xE000_1000	DWT	Data Watch Point & Trace
0xE000_0000	ITM	Instrumentation Trace Macro-cell

3.2.2 Peripheral Special Function Register Map

Table 3-3 Peripheral Memory Map

Peripheral Group	Base Address	Peripheral	Description
DMAC	0x400F_0000	DMAC	Direct Memory Access Controller
CAN	0x400E_1000	CAN1	Controller Area Network 1
	0x400E_0000	CAN0	Controller Area Network 0
LCDC	0x400D_0000	LCDC	LCD Controller
ENC	0x400C_1000	ENC1	Encoder Counter 1
	0x400C_0000	ENC0	Encoder Counter 0
IMC	0x400B_1000	IMC1	Inverter Motor Controller1
	0x400B_0000	IMC0	Inverter Motor Controller0
I2C	0x400A_1000	I2C1	Inter-Integrated Circuit 1
	0x400A_0000	I2C0	Inter-Integrated Circuit 0
SSP	0x4009_1000	SSP1	Synchronous Serial Port 1
	0x4009_0000	SSP0	Synchronous Serial Port 0
USART	0x4008_3000	USART3	Universal Sync/Async Receiver/Transmitter 3
	0x4008_2000	USART2	Universal Sync/Async Receiver/Transmitter 2
	0x4008_1000	USART1	Universal Sync/Async Receiver/Transmitter 1
	0x4008_0000	USART0	Universal Sync/Async Receiver/Transmitter 0
STT	0x4007_8000	STT	Stamp Timer
PWM	0x4007_7000	PWM7	Pulse Width Modulation 7 (16bit)

Peripheral Group	Base Address	Peripheral	Description
	0x4007_6000	PWM6	Pulse Width Modulation 6 (16bit)
	0x4007_5000	PWM5	Pulse Width Modulation 5 (16bit)
	0x4007_4000	PWM4	Pulse Width Modulation 4 (16bit)
	0x4007_3000	PWM3	Pulse Width Modulation 3 (16bit)
	0x4007_2000	PWM2	Pulse Width Modulation 2 (16bit)
	0x4007_1000	PWM1	Pulse Width Modulation 1 (16bit)
	0x4007_0000	PWM0	Pulse Width Modulation 0 (16bit)
TIMER/ Counter	0x4006_7000	TC7	Timer/Counter 7 (16bit)
	0x4006_6000	TC6	Timer/Counter 6 (16bit)
	0x4006_5000	TC5	Timer/Counter 5 (16bit)
	0x4006_4000	TC4	Timer/Counter 4 (16bit)
	0x4006_3000	TC3	Timer/Counter 3 (16bit)
	0x4006_2000	TC2	Timer/Counter 2 (16bit)
	0x4006_1000	TC1	Timer/Counter 1 (16bit)
	0x4006_0000	TC0	Timer/Counter 0 (16bit)
GPIO	0x4005_8000	IOCONF	IO Configuration
	0x4005_3000	GPIO3	General Purpose IO Group 3
	0x4005_2000	GPIO2	General Purpose IO Group 2
	0x4005_1000	GPIO1	General Purpose IO Group 1
	0x4005_0000	GPIO0	General Purpose IO Group 0
OPAMP	0x4004_2000	OPAMP	OP-AMP
ADC	0x4004_1000	ADC1	Analog to Digital Converter 1 (10bit)
	0x4004_0000	ADC0	Analog to Digital Converter 0 (12bit)
FRT	0x4003_1000	FRT	Free Running Timer (32bit)
WDT	0x4003_0000	WDT	Watchdog Timer
SYSTEM	0x4002_0000	CM	System(Clock, Reset and Power) Manager
MEMORY	0x4001_1000	DFC	Data Flash Controller
	0x4001_0000	PFC	Program Flash Controller
SFM	0x4000_0000	–	Device information including Chip ID

4 Analog to Digital Converter (ADC0)

4.1 Overview

This section provides a complete functional description of the ADC controller, detailing the operation of the design from the end user perspective in a number of sub-section.

4.1.1 Features

The following is the distinctive features that are described in detail in this user's manual:

- Resolution: 12-bit
- 8 Input Channels, AIN0[8:1]
- Four input channels with OP-AMP
- Conversion Start Sources:
  - Software Start, External Trigger Input (ADTRG0), Internal Peripheral Trigger Signal (IMC0/1), Timer Match (TC)
- Differential Linearity Error: ± 1.5 LSB (Max.)
- Integral Linearity Error: ± 3.5 LSB (Max.)
- Offset Error: ± 43 mV (Max. @5.5V)
- Maximum Conversion Rate: 1us (@5MHz clock)
- Low Power Consumption
- Power Supply Voltage: 2.7 ~ 5.5V
- Analog Input Range: 0 – AV_{REF0}

4.1.2 Pin Description

Table 4-1 ADC0 Pin Description

Pin Name	Function	I/O Type	Comments
AVREF0	Analog Reference Voltage	Analog Input	–
AIN0[8:1]	Analog Input	Analog Input	–
ADTRG0	External Start Trigger Signal asserted via ADTRG Pin	Digital Input	–

4.1.3 Block Diagram

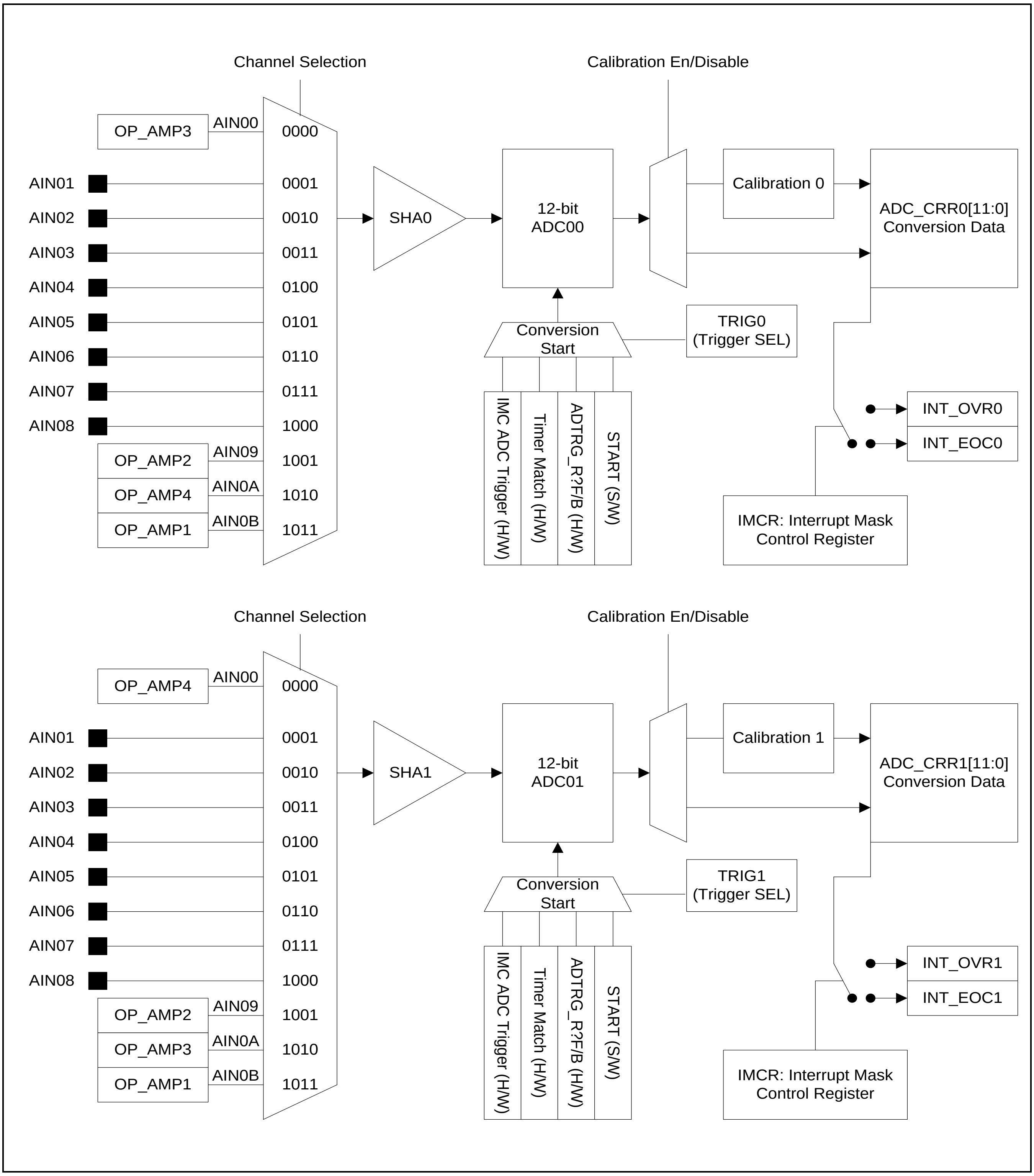


Figure 4-1 ADC0 Block Diagram

4.2 General Operation

4.2.1 ADC Input and Output

ADC operation is to convert the signal asserted via AIN0x input pin to digital data. The valid input signal can be the followings:

Voltage range is from reference bottom to top.

AIN0x function pins are used for an analog input source to convert by ADC. Input signal range is followed by the boundary of reference, Reference TOP and Reference BOTTOM.

Input Voltage Range: 0.0V ~ 5.0V

Reference Bottom = 0.0V, Reference Top = 5.0V

$$Dout = \frac{V_{IN}}{V_{FS}} = \frac{b_{N-1}}{2} + \frac{b_{N-2}}{2^2} + \ldots\ldots + \frac{b_0}{2^N}$$

$$1\ LSB = \frac{\text{Reference Top - Reference Bottom}}{2^{\text{Resolution}}} = \frac{5.0V - 0.0V}{2^{12}} = \frac{5.0V}{4096} \approx 1.22mV$$

Table 4-2 ADC Input & Output Range

Index	AINx input voltage (V)	Digital Output (Binary)	Digital Output (HEX)
0	0.00000 ~ 0.00122	0000 0000 0000	0x000
1	–	0000 0000 0001	0x001
2	–	0000 0000 0010	0x002
~	~	~	~
511	–	0001 1111 1111	0x1FF
512	–	0010 0000 0000	0x200
513	–	0010 0000 0001	0x201
~	~	~	
4092	–	1111 1111 1100	0xFFC
4093	–	1111 1111 1101	0xFFD
4094	–	1111 1111 1110	0xFFE
4095	–	1111 1111 1111	0xFFF

4.2.2 Analog Cell Clock Frequency

The clock for ADC operation is obtained from PCLK. ADC clock must not be over 5MHz. That means the maximum ADC clock is 5MHz. When ADC operates with 5MHz, the conversion is done in the minimum time (1us).

4.2.3 Conversion Sequence Definition

A conversion sequence is a sequence of analog inputs to be converted. User can configure ADC block to make conversions of some of the 8 inputs in its own order. The length of the sequence (in other terms the number of conversions) is defined by setting the CCNT field in the ADC_MR (ADC Mode Register). The following table gives the relation between the CCNT field and the number of conversion performed in a sequence:

Table 4-3 CCNT[2:0] Values and the Number of Conversions

CCNT[2:0]	Count Value	Description
000	1	One conversion operation of ICNUM0[2:0] channel
001	2	2 times conversion operation from ICNUM0[2:0] to ICNUM1[2:0]
010	3	3 times conversion operation from ICNUM0[2:0] to ICNUM2[2:0]
011	4	4 times conversion operation from ICNUM0[2:0] to ICNUM3[2:0]
100	5	5 times conversion operation from ICNUM0[2:0] to ICNUM4[2:0]
101	6	6 times conversion operation from ICNUM0[2:0] to ICNUM5[2:0]
110	7	7 times conversion operation from ICNUM0[2:0] to ICNUM6[2:0]
111	8	8 times conversion operation from ICNUM0[2:0] to ICNUM7[2:0]

This means that, even configured in ‘one shot’ mode, the ADC will run the specified number of conversion after a start request. The data register will be updated with the conversion results by the end of each conversion which composes the sequence.

The composition of a sequence is programmed in the ADC_CCSR register. The ICNUM0 field defines the first input to be converted in the sequence. The ICNUM1 field defines the second input to be converted and so on. Find in the table below the relation between the ICNUMx values and input selected:

Table 4-4 ICNUMx Values and Selected Input

ICNUMx Values	Selected Pin
0000	AIN 00 with OP amp
0001	AIN 01
0010	AIN 02
0011	AIN 03
0100	AIN 04
0101	AIN 05
0110	AIN 06
0111	AIN 07
1000	AIN 08
1001	AIN 09 with OP amp
1010	AIN 0A with OP amp
1011	AIN 0B with OP amp

For example, assuming that:

- CCNT[2:0] = 0x2
- ICNUM0[3:0] = 0x5 (AIN05), ICNUM1[3:0] = 0x2 (AIN02) and ICNUM2[3:0] = 0x1 (AIN01)

After a start conversion request, ADC converts input 5 (AIN05), followed by input 2 (AIN02) and it finishes by converting input 1 (AIN01).

4.2.4 One-shot and Continuous Mode

The ADC can be programmed in two modes: one shot and continuous conversion mode.

One shot conversion mode is enabled by setting CMODE bit of control register to '0'. In this mode, upon conversion start request, the ADC performs the only complete conversions sequence and then stops and waits for another start request.

The ADC cannot be stopped until it finishes the conversions sequence.

Continuous conversion mode is enabled by setting CMODE bit of control register to '1'. In this mode, upon conversion start request, the ADC repetitively performs conversions sequences until it is forced to stopped. To stop continuous conversion, the CPU must write STOP bit in the control register.

When a stop is requested, the ADC finishes its current conversion and updates the data register with this last conversion result. No other conversions are performed even if the conversions sequence is not finished.

NOTE: However, user should be vigilant, because after a stop command in continuous mode, the ADC finishes the on-going conversion and this may look like to an extra conversion.

4.2.5 Conversion Start Sources

User can select the start signal for conversion by TRIG[2:0] field of ADC_MR(mode register).

- START: Conversions are started by the CPU (writing the START bit in the ADC_CSR).
- ADTRG: Conversions can also be started by an external device using a dedicated input pin (ADTRG).
- Peripheral (IMC, Timer): Conversions can also be started by a peripherals like IMC, timer.

4.2.6 Power Management

The ADC peripheral includes power management features that can be used to minimize power consumption. Power can be saved on two sides: analog and digital.

- Analog power saving: To reduce analog power consumption, CPU shall disable the ADC module (write ADCEN bit to '0') which has for effect to set the analog cell to 'standby' mode.
- Digital power saving: To reduce digital power consumption, CPU shall disable the ADC clock (write CLKEN bit to'0') which has for effect to disable all incoming clocks. Then, digital consumption is reduced close to 0.

4.2.7 Flags and Interrupt

ADC generates an interrupt if at least one of EOC or OVR is active (set to '1') while it is enabled (corresponding bit in ADC_MISR is read as '1').

Each interrupt bit can be enabled or disabled using interrupt mask control register.

The ADC generates an EOC interrupt when conversion is completed while ADC interrupt is enabled. You can know if whether that interrupt occurs or not by reading the interrupt pending register. This interrupt bit can be enabled or disabled using respectively the interrupt enable register and interrupt disable register.

4.2.7.1 BUSY

The BUSY bit indicates ADC is during converting analog data or not.

4.2.7.2 EOC (End Of Conversion)

The EOC bit of the status register flags the presence of a new valid data in data register.

- If EOC is '0', it means that no conversion has been done since last reset of this bit or that the last conversion result has been read by the CPU in the data register.
- If EOC is '1', it means that a conversion has been completed and the new conversion result has not been read yet in data register.

EOC flag is reset to '0' each time a read access is performed in data register (ADC_CRR).

4.2.7.3 OVR (Overrun)

This flag indicates that a new data overwrites a previous converted data which has not been read. The previous data has been lost.

OVR flag can be clear by CPU (writing OVR bit in clear status register).

4.2.8 Software Sequence for Conversion

The following lines list the basic sequence of operations after a reset for using ADC peripheral:

- Enable ADC clock, CLKEN in ADC_CEDR
- Enable ADC IP for ADC Operation
- Configure ADC clock in the ADC_CEDR. CDIV fields shall be programmed in order to have an analog frequency clock less than 5MHz.
- Configure conversion channel (ICNUMx in ADC_CCSR).
- Select the conversion start trigger source (TRIG in ADC_MR).
- Check BUSY bit in ADC_SR. When the flag is 0, the ADC is ready to start conversion.
- Start conversion by writing START bit in the ADC_CSR.
- The analog input voltage is sampled and conversion completes after 5 ADC clock cycles from the start command. The digital 12-bit conversion result is stored into ADC_CRR and the EOC bit in ADC_RISR sets to ‘1’. If EOC flag was already set, then the OVR bit is also set.
- The CPU can then read the digital value in ADC_CRR, which automatically clears the EOC.

4.2.9 Calibration

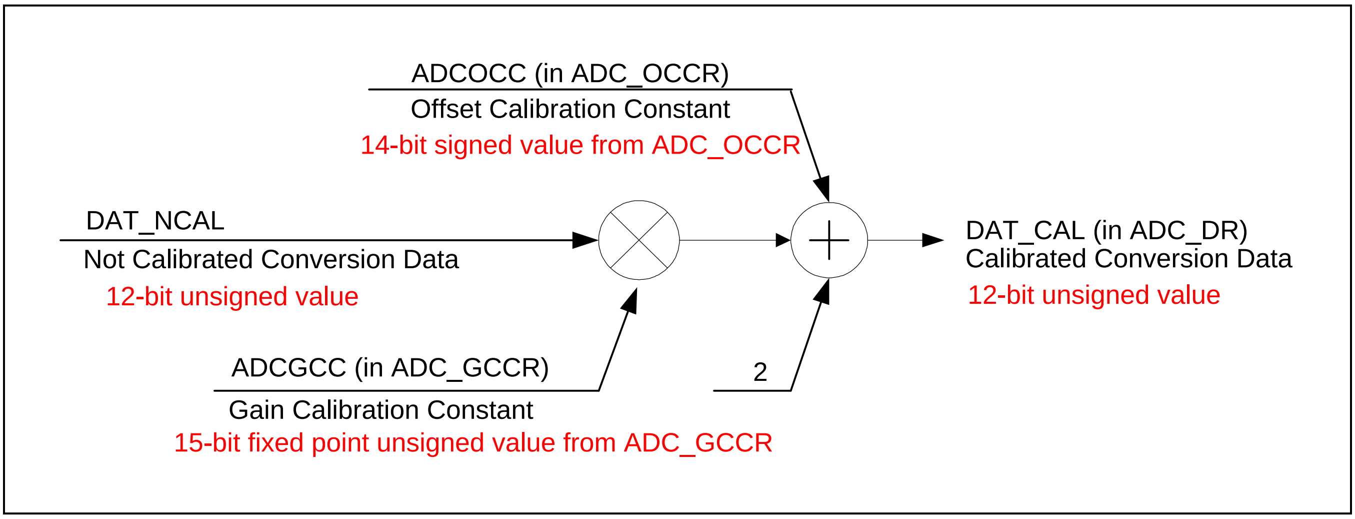
- DAT_NCAL: Converted Result Data without Calibration
- DAT_CAL: Converted Result Data with Calibration
- ADC_GCC: Gain Calibration Constant
- ADC_OCC: Offset Calibration Constant

If user wants to use calibration function, user should get constants for a calibration unit. Constants may have a different value depending on the noise level of a real target system.

ADC_GCC and ADC_OCC are determined by taking two samples of known reference voltages and using these samples to calculate their values. Once calculated, ADD_GCC is stored in ADC_GCR and ADC_OCC in ADC_OCR. A conversion result is calibrated according to the status of CALEN bit. User can get the calibrated conversion result data by setting CALEN bit to “1”.

If the CALEN is asserted, the ADC will automatically calculate the calibrated result before sending the result to the ADC_CRR (ADC Convert Data Register). If the CALEN bit is negated, it bypasses the calibration unit, and is directly sent to ADC_CRR (ADC Convert Data Register).

4.2.9.1 Calibration Unit


ADCOCC (in ADC_OCCR)
Offset Calibration Constant
14-bit signed value from ADC_OCCR
DAT_NCAL
Not Calibrated Conversion Data
12-bit unsigned value
DAT_CAL (in ADC_DR)
Calibrated Conversion Data
12-bit unsigned value
ADCGCC (in ADC_GCCR)
Gain Calibration Constant
15-bit fixed point unsigned value from ADC_GCCR
2

The process to get calibrated ADC Conversion Data consists of two steps:

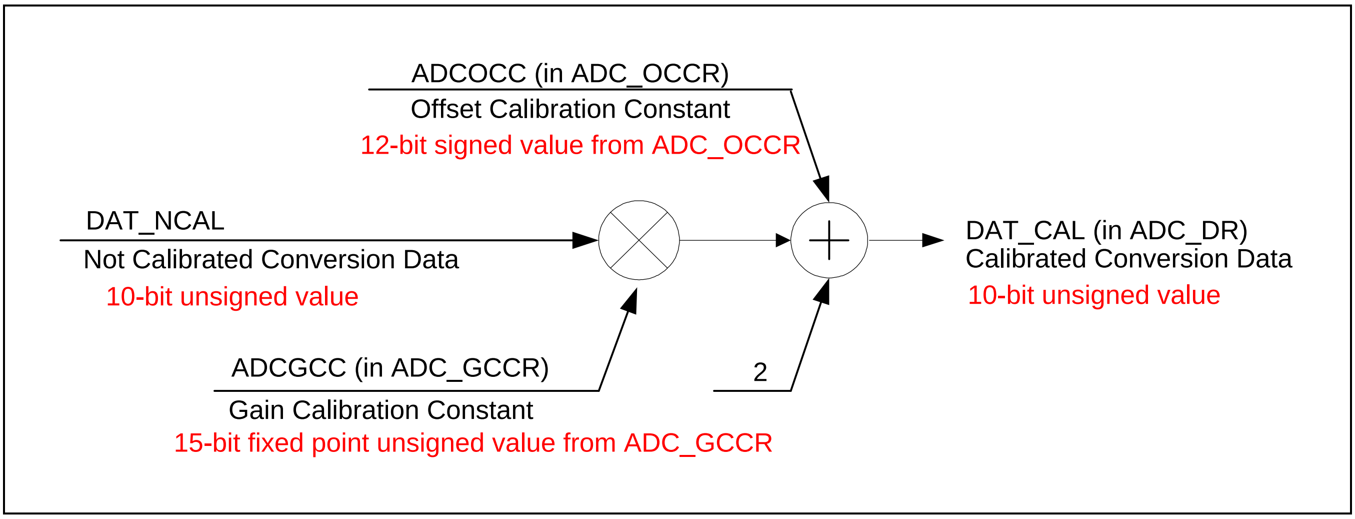
- Determine the gain and offset calibration constants
- Run calibration unit with the non-calibrated data generated by the direct converting on ADC macro-cell

Calibration Unit Operation;

```
DAT_CAL = ADC_GCC * DAT_NCAL + ADC_OCC + 2
```

Before using calibration function, user determines these constants and writes corresponding registers (ADC_CCR and ADC_OCR). User can determine with two pairs of expected (DAT_CAL) and measured (DAT_NCAL) values being available for two reference voltages. User can choose between internal and external voltage. Reference voltages are 25% and 75% of VREF.

- VREF: Reference voltage value to convert any ADC input voltage selected by user in Voltage Range
- 1/4VREF: 25% Point voltage of VREF voltage value
- 3/4VREF: 75% Point voltage of VREF voltage value


ADCOCC (in ADC_OCCR)
Offset Calibration Constant
12-bit signed value from ADC_OCCR
DAT_NCAL
Not Calibrated Conversion Data
10-bit unsigned value
DAT_CAL (in ADC_DR)
Calibrated Conversion Data
10-bit unsigned value
ADCGCC (in ADC_GCCR)
2
Gain Calibration Constant
15-bit fixed point unsigned value from ADC_GCCR

The non-calibrated results for these input voltages are obtained by converting these channels with conversion commands that have the CAL bit negated. The transfer equations for when sampling these reference voltages are:

```
DAT_CAL@3/4VREF = ADC_GCC * DAT_NCAL@3/4VREF + ADC_OCC +2
DAT_CAL@1/4VREF = ADC_GCC * DAT_NCAL@1/4VREF + ADC_OCC +2
```

Thus;

```
ADC_GCC = (DAT_CAL@3/4VREF - DAT_CAL@1/4VREF) / (DAT_NCAL@3/4VREF - DAT_NCAL@1/4VREF)
ADC_OCC = DAT_CAL@3/4VREF - ADC_GCC*DAT_NCAL@3/4VREF - 2
```

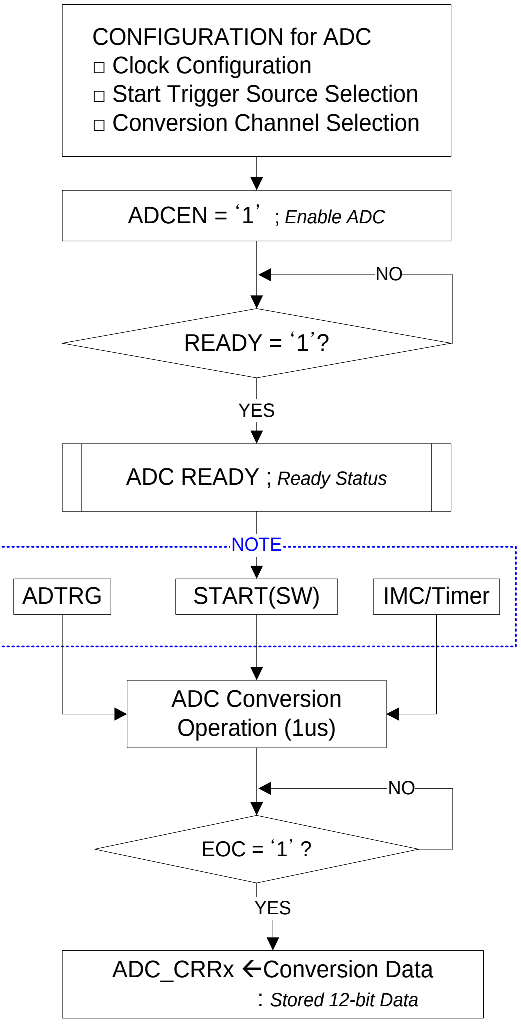
or

```
ADC_OCC = DAT_CAL@1/4VREF - ADC_GCC*DAT_NCAL@1/4VREF - 2
```

After being calculated, the ADC_GCC and ADC_OCC values must be written to ADC_GCCR and ADC_OCCR.

When user sets “CALEN” bit, the ADC will automatically calibrate the results using the ADC_GCC and ADC_OCC values stored in the ADC calibration registers.

4.2.10 Operation Sequence



NOTE: User should configure ADC start conversion by only one trigger signal among ADTRG input signal, START bit, IMC ADC trigger signal or timer match signal.

Figure 4-2 ADC Flow chart

4.3 Register Description

4.3.1 Register Map Summary

- Base Address: 0x4004_0000

Table 4-5 ADC0 Control Special Function Registers

Register	Offset	Description	Reset Value
ADC_IDR	0x000	ID Register	0x0001_001F
ADC_CEDR	0x004	Clock Enable/Disable Register	0x0000_0000
ADC_SRR	0x008	Software Reset Register	0x0000_0000
ADC_CSR	0x00C	Control Set Register	0x0000_0000
ADC_CCR	0x010	Control Clear Register	0x0000_0000
ADC_CDR	0x014	Clock Divider Register	0x0000_0000
ADC_MR	0x018	Mode Register	0x0000_0000
ADC_CCSR0	0x01C	Conversion Channel Sequence Register	0x0000_0000
ADC_CCSR1	0x020	Conversion Channel Sequence Register	0x0000_0000
ADC_SR	0x024	Status Register	0x0000_0000
ADC_IMSCR	0x028	Interrupt Mask Set/Clear Register	0x0000_0000
ADC_RISR	0x02C	Raw Interrupt Status Register	0x0000_0000
ADC_MISR	0x030	Masked Interrupt Status Register	0x0000_0000
ADC_ICR	0x034	Interrupt Clear Register	0x0000_0000
ADC_CRR0	0x038	Conversion Result Register	0x0000_0000
ADC_CRR1	0x03C	Conversion Result Register	0x0000_0000
ADC_GCR0	0x040	Gain Calibration Register	0x0000_0000
ADC_OCR0	0x044	Offset Calibration Register	0x0000_0000
ADC_GCR1	0x048	Gain Calibration Register	0x0000_0000
ADC_OCR1	0x04C	Offset Calibration Register	0x0000_0000
ADC_DMACR	0x050	DMA Control Register	0x0000_0000

4.3.1.1 ADC_IDR (ADC ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_001F

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register This field stores the ID code for the corresponding IP.	0x0001_001F

4.3.1.2 ADC_CEDR (ADC Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	ADC Clock Enable/Disable Bit. 0 = Disable ADC Clock 1 = Enables ADC Clock ADC software reset does not affect CLKEN bit status.	0
DBGEN	[31]	RW	Debug mode enable. 0 = Disable debug mode ADC not halted during processor debug mode. 1 = Enable debug mode ADC is halted during processor debug mode.	0

4.3.1.3 ADC_SRR (ADC Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset 0 = No effect 1 = Perform ADC Software Reset operation	0

4.3.1.4 ADC_CSR (ADC Control Set Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																					STOP1	START1	ADCEN1	RSVD					STOP0	START0	ADCEN0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	R	R	R	R	R	W	W	W

Name	Bit	Type	Description	Reset Value
ADCENx	[0, 8]	W	ADC0x Enable Bit 0 = No effect 1 = Enable ADC Core **NOTE:** After ADC block is enabled (ADCEN==1) TBD us stabilization time must be needed. ADCSTABLE bit in SR (status register) will be set after stabilization.	0
STARTx	[1, 9]	W	ADC0x Conversion Start Bit 0 = No effect 1 = Start ADC Conversion. This is the one of ADC trigger sources, software trigger.	0
STOPx	[2, 10]	W	ADC Conversion Stop in continuous conversion. This bit is auto-clear bit. 0 = No effect 1 = Stop the continuous conversion. **NOTE:** Before starting conversion, users should ensure that ADC is ready for conversion (BUSY bit is set to logical one in ADC_SR).	0

4.3.1.5 ADC_CCR (ADC Control Clear Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																							ADCEN1	RSVD							ADCEN0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
ADCENx	[0, 8]	W	ADC0x Disable Control Bit 0 = No effect 1 = Disable ADC0x Core	0

4.3.1.6 ADC_CDR (ADC Clock Divider Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																										CDIV					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
CDIV	[5:0]	RW	ADC Clock Divider Selection Field This field can determine the ADC0 clock frequency. That frequency is from 1MHz to 5MHz. For fast conversion, user should generate the maximum frequency with clock divider. ADC clock source is PCLK. `FADC = PCLK / (N+1)` **NOTE:** The clock rate to the ADC must not be exceeded 5 MHz.	00000'b

4.3.1.7 ADC_MR (ADC Mode Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
CCNT1[2:0]			CMODE1	RSVD	EICR1	ICRV1	CALEN1	CCNT0[2:0]			CMODE0	RSVD	EICR0	ICRV0	CALEN0	TRIG1[2:0]			RSVD					TRIG0[2:0]			RSVD				
0	0	0	0	0	0	0	0	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	R	RW	RW	RW	RW	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	R	R	R	R	R	RW	RW	RW	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
TRIGx[2:0]	[5:7] [13:15]	RW	ADC Start Trigger Signal Selection Bits This field can determine the trigger signal for ADC. 000 = Software (START bit in ADC_CR) 001 = ADTRG – Rising 010 = ADTRG – Falling 011 = ADTRG – Rising or Falling (Both) 100 = TCx – Timer/Counter Match 101 = IMC0 – ADC Trigger Value 110 = IMC1 – ADC Trigger Value Others = Reserved	000'b
CALENx	[16] [24]	RW	Calibration Enable/Disable Control Bit 0 = Disable calibration. The conversion data of ADC0 is generated without calibration. 1 = Enable calibration. The conversion data of ADC0 is generated with calibration. To use the calibration, user should initialize before enable calibration. For initialization, user should obtain the conversion data of 1/4 and 3/4 reference voltage. That need to configuration 'ICRV' or 'CRVS' control bits.	0
ICRVx	[17] [25]	RW	Internal Calibration Reference (Voltage) Value 0 = Select the internal voltage source 1/4AVDD to determinate the calibration constants 1 = Select the internal voltage source 3/4AVDD to determinate the calibration constants	0
EICRx	[18] [26]	RW	External/Internal Calibration Reference 0 = Select the external input voltage sources (1/4AVDD and 3/4AVDD) to determinate the calibration constants via ADC input channel. In this case, it's the same the normal operation mode. It's just different that the input voltage for conversion is 1/4AVDD and 3/4AVDD. 1 = Select he internal reference voltage sources (1/4AVDD and 3/4AVDD) to determinate the calibration constants	0

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td><b>NOTE:</b>
1. When user can’t use ADC without calibration, CALEN bit is ‘0’. In normal operation, EICR should be ‘0’.
2. When internal calibration is set, the 1/4 and 3/4 values of V_{REF} are stored in ADC_OCR registers but these values can't be used for getting accurate compensation (calibration) because of the offset difference in the customer board environment. Therefore external calibration is better used for more accurate calibration.
<table>
<tr><th>–</th><th>EICR</th><th>ICRV</th><th>Remark</th></tr>
<tr><td>Use 1/4 or/and 3/4 reference voltage supplied externally</td><td>0</td><td>X</td><td>User should use any ADC channel (AINx) to obtain external reference voltage. User can get the conversion data by 1/4 reference voltage with AINx. For the conversion data of 3/4 reference voltage, the same or other channel can be used.</td></tr>
<tr><td>Use 1/4 reference voltage supplied internally</td><td>1</td><td>0</td><td>Not use ADC channel(AINx)</td></tr>
<tr><td>Use 3/4 reference voltage supplied internally</td><td>1</td><td>1</td><td>Not use ADC channel(AINx)</td></tr>
</table>
X means ‘don’t care’.</td><td></td></tr>
<tr><td>CMODEx</td><td>[20]
[28]</td><td>RW</td><td>Conversion Mode Control Bit
0 = One shot mode. ADC converts as much inputs as specified by the CCNT[2:0] in the order specified in the ADC_MR, and stops.
1 = Continuous mode. ADC converts as much inputs as specified by the CCNT[2:0] in the order specified in the ADC_MR, and repeats. This bit is initialized to 0.
<b>NOTE:</b> In continuous mode, after a stop command, the ADC finishes the on-going conversion and this may look like to an extra conversion.</td><td>0</td></tr>
<tr><td>CCNTx[2:0]</td><td>[23:21]
[31:29]</td><td>RW</td><td>Conversion Count Field
<b>NOTE:</b> Even in one shot mode, ADC will run multiple conversions if the CCNT[2:0] is greater than 0000b.
<b>Table 4-6 CCNT[2:0] Values and the Count Value of Conversions</b>
<table>
<tr><th>CCNT[2:0]</th><th>Count for Conversion</th></tr>
<tr><td>000</td><td>One conversion operation of ICNUM0[2:0] channel</td></tr>
</table></td><td>000'b</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><td>001</td><td>2 times conversion operation from ICNUM0[2:0] to ICNUM1[2:0]</td></tr>
<tr><td>010</td><td>3 times conversion operation from ICNUM0[2:0] to ICNUM2[2:0]</td></tr>
<tr><td>011</td><td>4 times conversion operation from ICNUM0[2:0] to ICNUM3[2:0]</td></tr>
<tr><td>100</td><td>5 times conversion operation from ICNUM0[2:0] to ICNUM4[2:0]</td></tr>
<tr><td>101</td><td>6 times conversion operation from ICNUM0[2:0] to ICNUM5[2:0]</td></tr>
<tr><td>110</td><td>7 times conversion operation from ICNUM0[2:0] to ICNUM6[2:0]</td></tr>
<tr><td>111</td><td>8 times conversion operation from ICNUM0[2:0] to ICNUM7[2:0]</td></tr>
</table>
</td><td></td></tr>
</table>

4.3.1.8 ADC_CCSR (ADC Conversion Channel Sequence Register)

- Address = Base Address + 0x001C, 0x0020, Reset Value = 0x0000_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
ICNUM7[2:0]				ICNUM6[2:0]				ICNUM5[2:0]				ICNUM4[2:0]				ICNUM3[2:0]				ICNUM2[2:0]				ICNUM1[2:0]				ICNUM0[2:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>ICNUMx[3:0]</td><td>[31:0]</td><td>RW</td><td>Analog Input Channel Number Selection Field
Conversion sequence can consist of from first conversion channel defined by ICNUM0[3:0] to last conversion channel defined by ICNUMx[3:0]. ‘x’ can be from 1 to 7.

<b>Table 4-7 ICNUMx[2:0] Values</b>
<table>
<tr><th>ICNUMx Values</th><th>Selected Pin</th></tr>
<tr><td>0000</td><td>AIN 00 with OP amp</td></tr>
<tr><td>0001</td><td>AIN 01</td></tr>
<tr><td>0010</td><td>AIN 02</td></tr>
<tr><td>0011</td><td>AIN 03</td></tr>
<tr><td>0100</td><td>AIN 04</td></tr>
<tr><td>0101</td><td>AIN 05</td></tr>
<tr><td>0110</td><td>AIN 06</td></tr>
<tr><td>0111</td><td>AIN 07</td></tr>
<tr><td>1000</td><td>AIN 08</td></tr>
<tr><td>1001</td><td>AIN 09 with OP amp</td></tr>
<tr><td>1010</td><td>AIN 0A with OP amp</td></tr>
<tr><td>1011</td><td>AIN 0B with OP amp</td></tr>
</table>
</td><td>0x0</td></tr>
</table>

4.3.1.9 ADC_SR (ADC Status Register)

- Address = Base Address + 0x024, Reset Value = 0x0000_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
RSVD																				CTCVS1	ADCEN1	BUSY1	ADCSTABLE1	RSVD				CTCVS0	ADCEN0	BUSY0	ADCSTABLE0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ADCSTABLEx	[0] [8]	R	ADC Stabilization Status 0 = ADC is not stabilized. 1 = ADC is stabilized. This bit will be set after initialization time.	0
BUSYx	[1] [9]	R	ADC Status monitoring bit This bit can notify the status of ADC. 0 = ADC is not on a conversion operating. 1 = ADC is on a conversion operating **NOTE:** To change the configuration of ADC, You must check ADC_SR (ADC Status Register).	0
ADCENx	[2] [10]	R	ADC Enable / Disable Status 0 = ADC is disabled. 1 = ADC is enabled.	0
CTCVSx	[3] [11]	R	Continuous mode status 0 = One shot mode with help of microprocessor. 1 = Continuous mode, the peripheral is stand-alone. This bit is initialized to 0 and changes when there is a change of mode. This bit never generates any interruption.	0

4.3.1.10 ADC_IMSCR (ADC Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x028 Reset Value = 0x0000_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
RSVD																						OVR1	EOC1	RSVD						OVR0	EOC0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R	R	R	R	R	R	R W	R W

Name	Bit	Type	Description	Reset Value
EOCx	[0] [8]	RW	End of Conversion Interrupt Mask 0 = EOC interrupt is masked. (Disable the interrupt) 1 = EOC interrupt is not masked. (Enable the interrupt)	0
OVRx	[1] [9]	RW	Overrun Interrupt Mask 0 = OVR interrupt is masked. (Disable the interrupt) 1 = OVR interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

4.3.1.11 ADC_RISR (ADC Raw Interrupt Status Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																						OVR1	EOC1	RSVD						OVR0	EOC0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EOCx	[0] [8]	R	End of Conversion Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the EOC interrupt	0
OVRx	[1] [9]	R	Overrun Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the OVR interrupt	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

4.3.1.12 ADC_MISR (ADC Masked Interrupt Status Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																						OVR1	EOC1	RSVD						OVR0	EOC0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EOCx	[0] [8]	R	End of Conversion Masked Interrupt State Gives the masked interrupt status(after masking) of the EOC interrupt	0
OVRx	[1] [9]	R	Overrun Masked Interrupt State Gives the masked interrupt status(after masking) of the OVR interrupt	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

4.3.1.13 ADC_ICR (ADC Interrupt Clear Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																						OVR1	EOC1	RSVD						OVR0	EOC0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	R	R	R	R	R	R	W	W

Name	Bit	Type	Description	Reset Value
EOCx	[0] [8]	W	End of conversion interrupt 0 = No effect 1 = Clears the EOC interrupt	0
OVRx	[1] [9]	W	Overrun interrupt 0 = No effect 1 = Clears the OVR interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

4.3.1.14 ADC_CRR (ADC Conversion Result Register)

- Address = Base Address + 0x0038, + 0x003C, Reset Value = 0x0000_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
RSVD																				DATA											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DATA	[11:0]	R	Conversion Result Data of ADC A/D converter Output Data Result : 0x000 – 0xFFF When A/D conversion is finished, the conversion result can be read from the ADC_CRR register. The conversion data register should be read after the conversion is finished, EOC bit.	0x000

4.3.1.15 ADC_GCR (ADC Gain Calibration Register)

- Address = Base Address + 0x0040, + 0x0048, Reset Value = 0x0000_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
RSVD																	GCC_INT	GCC_FRAC													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
GCC_FRAC	[13:0]	RW	ADC Gain Calibration Constant Value. ADCGCC field is fixed point unsigned value. ADCGCC[14:0] = Fixed point unsigned value GCC × 2^{15} The Fixed point unsigned value GCC gets from the following (For more detail, refer to *4.2.9.1*) GCC = (IDLE75% – IDLE25%)/(RAW75% – RAW25%) The calculated result (GCC) should be limited from 0.5 to 1.99999.	0x0000
GCC_INT	[14]	RW	–	0

4.3.1.16 ADC_OCR (ADC Offset Calibration Register)

- Address = Base Address + 0x0044, + 0x004C, Reset Value = 0x0000_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
RSVD																		ADCOCC													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ADCOCC	[13:0]	RW	ADC Offset Calibration Constant Value. ADCOCC field is signed value. Negative values should be expressed using the 2's complement.	0x0000

4.3.1.17 ADC_DMACR (ADC DMA Control Register)

- Address = Base Address + 0x0050, Reset Value = 0x0000_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
RSVD																														DMAE1	DMAE0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W

Name	Bit	Type	Description	Reset Value
DMAEx	[1:0]	RW	DMA for ADC Enable/Disable Control Bit. 0 = Disable 1 = Enable	0

5 Analog to Digital Converter (ADC1)

5.1 Overview

This chapter describes Analog to Digital Converter (ADC). The 10-bit Analog to Digital Converter (ADC) module supports 8's input channels.

5.1.1 Features

- A cyclic type monolithic ADC with a power down function
- 10 Input Channels, AIN1[7:0]
- Resolution: 10-bit
- Maximum Conversion Rate: 20 us (@700KHz clock)
- A programmable conversions sequence: Sequence of analog inputs to be converted.
  This allows the user to make conversions of some of the 8 inputs in its own order. The length of a sequence (number of conversion) is defined by setting the CCNT field in the ADC_MR register. The composition of sequence is defined in the ADC_CCSR register.
- 2 conversion modes: one shot (or single) mode or continuous mode
  In one shot mode, inputs specified in the conversion sequence are successively converted after the start command, conversion results are successively stored in the data register and then ADC stops.
  In continuous mode, inputs specified in the conversions sequence are converted one after the other continuously until a stop is requested. Each time a conversions sequence is completed, the ADC re-starts conversion of the selected inputs in the order of the sequence. In this mode, the microprocessor starts the ADC which is then completely independent.
- ADC starts conversion using the followings:
  Software Start, START bit in ADC_CSR register
  External Trigger Input Signal via ADTRG1 Pin
- Adjustable gain and offset calibration
- Power management features for reducing power consumption
- DMA transfer

5.1.2 Pin Description

Table 5-1 ADC1 Pin Description

Pin Name	Function	I/O Type	Active Level	Comments
AIN1[7:0]	Analog Input	Analog Input	–	–
AVREF1	Analog Reference Voltage	Analog Input	–	–
ADTRG1	External Start Input 0	Digital Input	–	–

5.1.3 Block Diagram

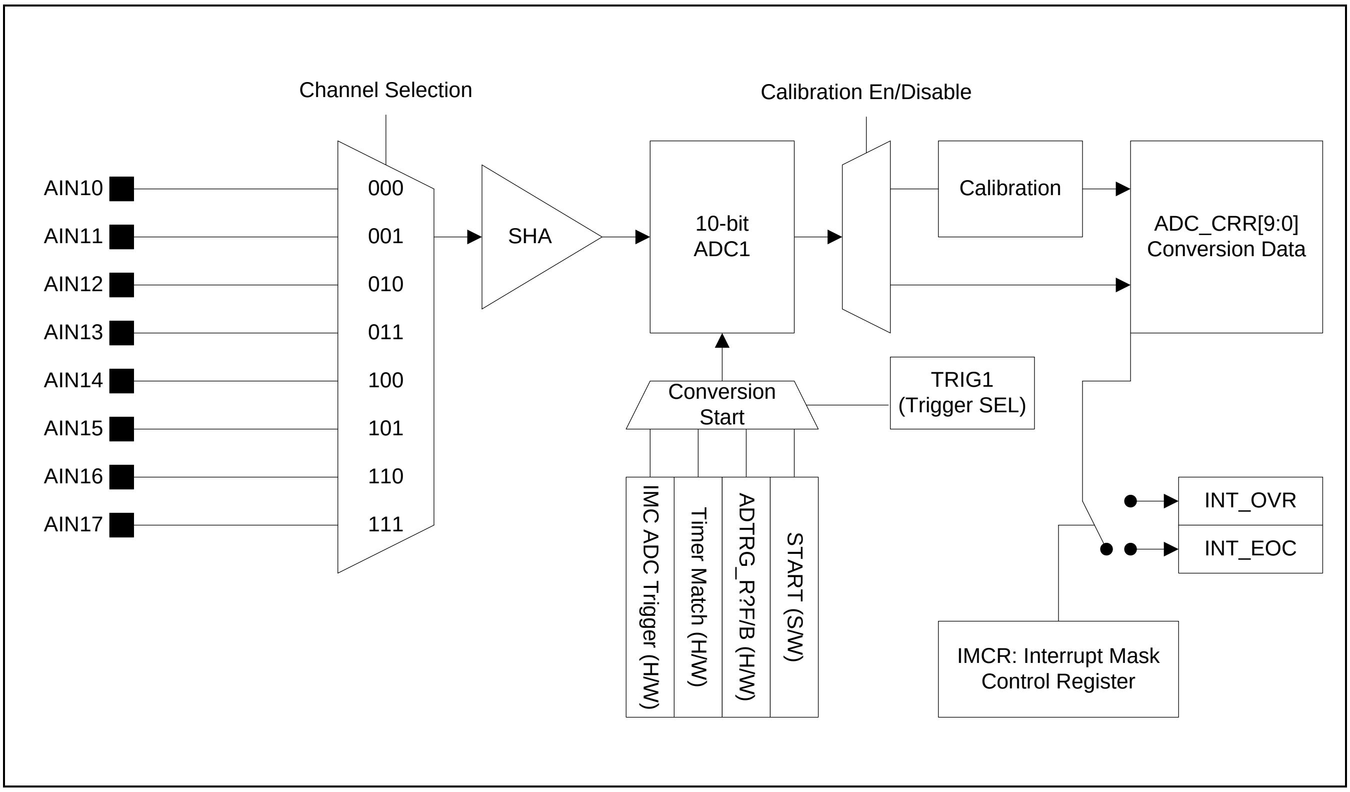


Figure 5-1 ADC1 Block Diagram

5.2 General Operation

5.2.1 ADC Input and Output

AIN1[7:0] function pins are used for an analog input source to convert by ADC. Input signal range is followed by the boundary of reference, Reference TOP and Reference BOTTOM.

Input Voltage Range: 0.0V ~ 5.0V

Reference Bottom = 0.0V, Reference Top = 5.0V

$$1LSB = \frac{Reference\ Top - Reference\ Bottom}{2^{Resolution}} = \frac{5.0V - 0.0V}{2^{10}} = \frac{5.0V}{1024} \approx 4.88mV$$

Table 5-2 ADC1 Input & Output Range

Index	AINx input voltage (V)	Digital Output (Binary)	Digital Output (HEX)
0	~ 0.00488	00 0000 0000	0x000
1	0.00488 ~ 0.00976	00 0000 0001	0x001
2	0.00976 ~ 0.01464	00 0000 0010	0x002
~	~	~	~
511	2.49511 ~ 2.50000	01 1111 1111	0x1FF
512	2.50000 ~ 2.50488	10 0000 0000	0x200
513	2.50488 ~ 2.50976	10 0000 0001	0x201
~	~	~	~
1021	4.98535 ~ 4.99023	11 1111 1101	0x3FD
1022	4.99023 ~ 4.99511	11 1111 1110	0x3FE
1023	4.99511 ~	11 1111 1111	0x3FF

5.2.2 Analog Cell Clock Frequency

Analog clock frequency can be tuned (**less than maximum 700KHz**) whichever system clock frequency. This feature allows to be configurable the sampling frequency of analog inputs. As specified in the electrical characteristics (see Table "ADC characteristics"), the analog cell clock frequency is limited to 700KHz, whereas system clock is usually far higher. So, the ADC module provides a clock frequency divider based on a 6-bit counter.

Total 10-bit conversion time is as follows:

- A/D converter clock =700KHz
- Conversion speed = 700KHz / 14 clock cycles = 50KHz → Conversion time = 20 us

NOTE: This A/D converter was designed to operate at maximum 700KHz clock.

5.2.3 Conversion Sequence Definition

A conversion sequence is a sequence of analog inputs to be converted. User can configure ADC block to make conversions of some of the 8 inputs in its own order. The length of the sequence (in other terms the number of conversions) is defined by setting the CCNT field in the ADC_MR (ADC Mode Register). The following table gives the relation between the CCNT field and the number of conversion performed in a sequence:

Table 5-3 CCNT[2:0] Values and the Number of Conversions

CCNT[2:0]	Count Value	Description
000	1	One conversion operation of ICNUM0[2:0] channel
001	2	2 times conversion operation from ICNUM0[2:0] to ICNUM1[2:0]
010	3	3 times conversion operation from ICNUM0[2:0] to ICNUM2[2:0]
011	4	4 times conversion operation from ICNUM0[2:0] to ICNUM3[2:0]
100	5	5 times conversion operation from ICNUM0[2:0] to ICNUM4[2:0]
101	6	6 times conversion operation from ICNUM0[2:0] to ICNUM5[2:0]
110	7	7 times conversion operation from ICNUM0[2:0] to ICNUM6[2:0]
111	8	8 times conversion operation from ICNUM0[2:0] to ICNUM7[2:0]

This means that, even configured in ‘one shot’ mode, the ADC will run the specified number of conversion after a start request. The data register will be updated with the conversion results by the end of each conversion which composes the sequence.

The composition of a sequence is programmed in the ADC_CCSR register. The ICNUM0 field defines the first input to be converted in the sequence. The ICNUM1 field defines the second input to be converted and so on. Find in the table below the relation between the ICNUMx values and input selected:

Table 5-4 ICNUMx Values and Selected Input

ICNUMx Values	Selected Pin
000	AIN 10
001	AIN 11
010	AIN 12
011	AIN 13
100	AIN 14
101	AIN 15
110	AIN 16
111	AIN 17

For example, assuming that:

- CCNT[2:0] = 0x2,
- ICNUM0[2:0] = 0x5 (AIN15), ICNUM1[2:0] = 0x2 (AIN12) and ICNUM2[2:0] = 0x0 (AIN10)

After a start conversion request, ADC converts input 5 (AIN15), followed by input 2 (AIN12) and it finishes by converting input 0 (AIN10).

5.2.4 One-shot and Continuous Mode

The ADC can be programmed in two modes: one shot and continuous conversion mode.

One shot conversion mode is enabled by setting CMODE bit of control register to ’0’. In this mode, upon conversion start request, the ADC performs the only complete conversions sequence and then stops and waits for another start request.

The ADC can not be stopped until it finishes the conversions sequence.

Continuous conversion mode is enabled by setting CMODE bit of control register to ’1’. In this mode, upon conversion start request, the ADC repetitively performs conversions sequences until it is forced to stopped. To stop continuous conversion, the CPU must write STOP bit in the control register.

When a stop is requested, the ADC finishes its current conversion and updates the data register with this last conversion result. No other conversions are performed even if the conversions sequence is not finished.

NOTE: However, user should be vigilant, because after a stop command in continuous mode, the ADC finishes the on-going conversion and this may look like to an extra conversion.

5.2.5 Conversion Start Sources

Conversion start source can select TRIG[2:0] field in ADC_MR.

1. Conversions are started by the CPU (writing the START bit in the ADC_CSR).
2. Peripheral Timer Match Signal: Match for programmed count value occurs
3. Conversions can also be started by an external device using a dedicated input pin (ADTRG1).

5.2.6 Power Management

The ADC peripheral includes power management features that can be used to minimize power consumption. Power can be saved on two sides: analog and digital.

Analog power saving: To reduce analog power consumption, CPU shall disable the ADC module (write ADCEN bit to ‘0’) which has for effect to set the analog cell to ’standby’ mode.

Digital power saving: To reduce digital power consumption, CPU shall disable the ADC clock (write CLKEN bit to’0’) which has for effect to disable all incoming clocks. Then, digital consumption is reduced close to 0.

5.2.7 Flags and Interrupt

The ADC generates an interrupt if at least one of EOC, or OVR bit is active (set to ‘1’) in the status register while it is enabled (corresponding bit in ADC_MISR is read as ‘1’).

Each interrupt bit can be enabled or disabled using interrupt mask control register.

5.2.7.1 BUSY

The BUSY bit indicates ADC is during converting analog data or not.

5.2.7.2 EOC (End Of Conversion)

The EOC bit of the status register flags the presence of a new valid data in data register.

If EOC is ‘0’, it means that no conversion has been done since last reset of this bit or that the last conversion result has been read by the CPU in the data register.

If EOC is ‘1’, it means that a conversion has been completed and the new conversion result has not been read yet in data register.

EOC flag is reset to ’0’ each time a read access is performed in data register (ADC_CRR).

5.2.7.3 OVR (Overrun)

This flag indicates that a new data overwrites a previous converted data which has not been read. The previous data has been lost.

OVR flag can be clear by CPU (writing OVR bit in clear status register).

5.2.8 Software Sequence for Conversion

The following lines list the basic sequence of operations after a reset for using ADC peripheral:

- Enable the clock in the ADC_CEDR.
- Configure ADC mode in the ADC_MR. CDIV field shall be programmed in order to have an analog frequency clock less than 700KHz. Indicate if the conversion is or not continuous. Define the conversions sequence: number of conversion (CCNT field in ADC_MR) and which inputs are converted (ICNUMx fields in ADC_CCSR).
- Enable ADC (ADCEN in ADC_CSR).
- Wait for BUSY bit in ADC_SR. When the flag is 0, the ADC is ready to start conversion.
- Initiate conversion by writing START bit in the ADC_CSR.
- ADC selects the analog input associated with conversion number 1 of conversion sequence.
- The analog input voltage is sampled and conversion completes after 14 ADC clock cycles from the start command. The digital 10-bit conversion result is stored into ADC_CRR and the EOC bit in ADC_RISR rises up. If EOC flag was already set, then the OVR bit is also set.
- The CPU can then read the digital value in ADC_CRR, which automatically clears the EOC. If the CPU decides that no more data should be converted in continuous mode, then it respectively writes the STOP bit. In this case, the ADC stops operating and waits for next start request. Note that in 'one shot' mode, the ADC can not be stopped until the end of all conversions specified in the conversion sequence.
- If CCNT is non-null, the ADC selects the analog input associated to the next conversion number and the operation flow restarts from step 6.
- If CMODE is read as '1', the ADC restarts another conversion sequence from step 5.

5.2.9 Calibration

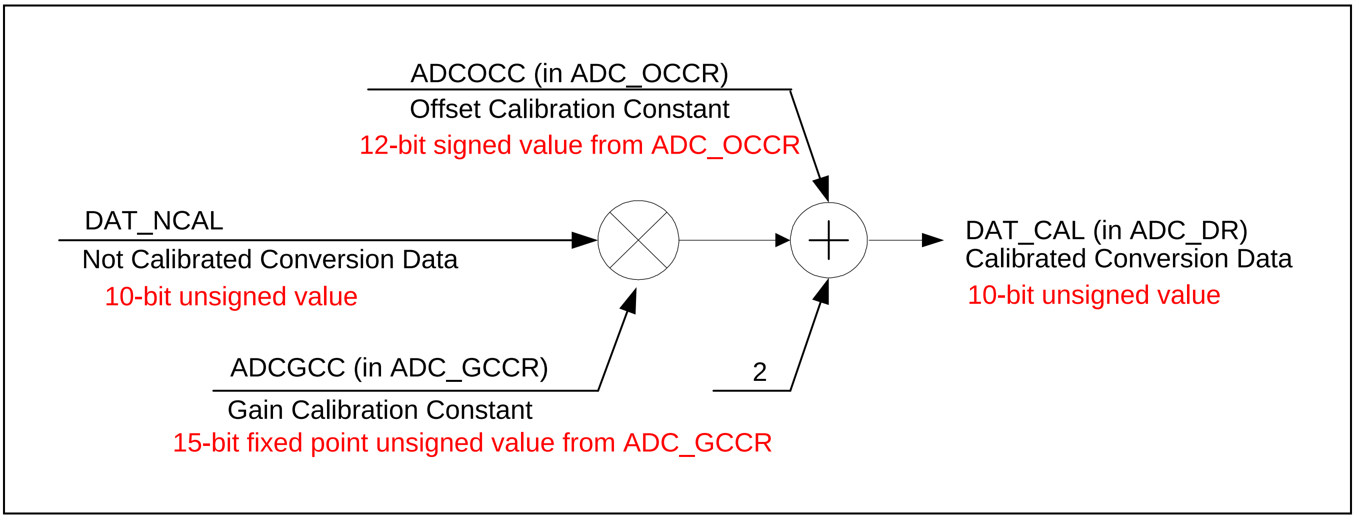
- DAT_NCAL: Converted Result Data without Calibration
- DAT_CAL: Converted Result Data with Calibration
- ADC_GCC: Gain Calibration Constant
- ADC_OCC: Offset Calibration Constant

If user wants to use calibration function, user should get constants for a calibration unit. Constants may have a different value depending on the noise level of a real target system.

ADC_GCC and ADC_OCC are determined by taking two samples of known reference voltages and using these samples to calculate their values. Once calculated, ADD_GCC is stored in ADC_GCCR and ADC_OCC in ADC_OCCR. A conversion result is calibrated according to the status of CALEN bit. User can get the calibrated conversion result data by setting CALEN bit to "1".

If the CALEN is asserted, the ADC will automatically calculate the calibrated result before sending the result to the ADC_CRR (ADC Convert Data Register). If the CALEN bit is negated, it bypasses the calibration unit, and is directly sent to ADC_CRR (ADC Convert Data Register).

5.2.9.1 Calibration Unit


ADCOCC (in ADC_OCCR)
Offset Calibration Constant
12-bit signed value from ADC_OCCR
DAT_NCAL
Not Calibrated Conversion Data
10-bit unsigned value
DAT_CAL (in ADC_DR)
Calibrated Conversion Data
10-bit unsigned value
ADCGCC (in ADC_GCCR)
2
Gain Calibration Constant
15-bit fixed point unsigned value from ADC_GCCR

The process to get calibrated ADC Conversion Data consists of two steps:

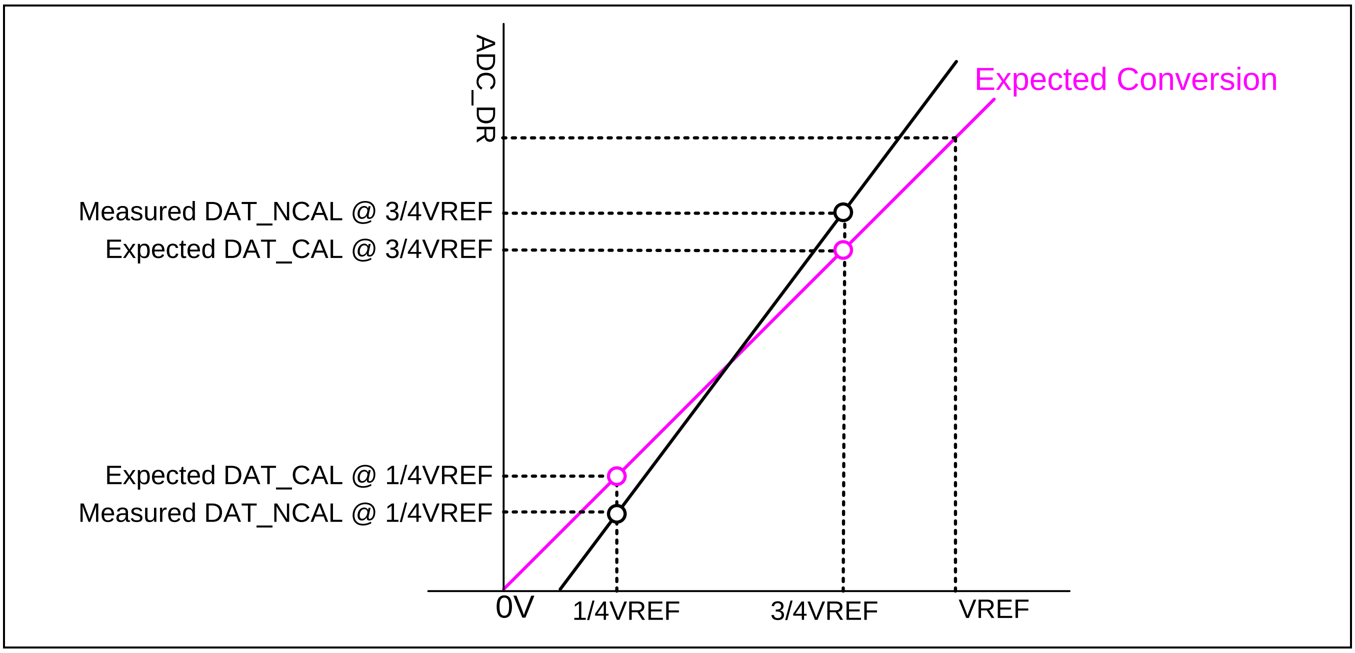
- Determine the gain and offset calibration constants
- Run calibration unit with the non-calibrated data generated by the direct converting on ADC macro-cell

Calibration Unit Operation;

```
DAT_CAL = ADC_GCC * DAT_NCAL + ADC_OCC + 2
```

Before using calibration function, user determines these constants and writes corresponding registers (ADC_GCCR and ADC_OCCR). User can determine with two pairs of expected (DAT_CAL) and measured (DAT_NCAL) values being available for two reference voltages. User can choose only external voltage because 10bit ADC has no internal reference voltage generation logic. Reference voltages are 25% and 75% of VREF.

- VREF: Reference voltage value to convert any ADC input voltage selected by user in Voltage Range
- 1/4VREF: 25% Point voltage of VREF voltage value
- 3/4VREF: 75% Point voltage of VREF voltage value



The non-calibrated results for these input voltages are obtained by converting these channels with conversion commands that have the CAL bit negated. The transfer equations for when sampling these reference voltages are:

```
DAT_CAL@3/4VREF = ADC_GCC * DAT_NCAL@3/4VREF + ADC_OCC +2
DAT_CAL@1/4VREF = ADC_GCC * DAT_NCAL@1/4VREF + ADC_OCC +2
```

Thus;

```
ADC_GCC = (DAT_CAL@3/4VREF - DAT_CAL@1/4VREF) / (DAT_NCAL@3/4VREF - DAT_NCAL@1/4VREF)
ADC_OCC = DAT_CAL@3/4VREF - ADC_GCC*DAT_NCAL@3/4VREF - 2
```

or

```
ADC_OCC = DAT_CAL@1/4VREF - ADC_GCC*DAT_NCAL@1/4VREF - 2
```

After being calculated, the ADC_GCC and ADC_OCC values must be written to ADC_GCCR and ADC_OCCR.

When user sets “CALEN” bit, the ADC will automatically calibrate the results using the ADC_GCC and ADC_OCC values stored in the ADC calibration registers.

5.3 Register Description

5.3.1 Register Map Summary

- Base Address: 0x4004_1000

Table 5-5 ADC1 Special Function Registers

Register	Offset	Description	Reset Value
ADC_IDR	0x000	ID Register	0x0011_001F
ADC_CEDR	0x004	Clock Enable/Disable Register	0x0000_0000
ADC_SRR	0x008	Software Reset Register	0x0000_0000
ADC_CSR	0x00C	Control Set Register	0x0000_0000
ADC_CCR	0x010	Control Clear Register	0x0000_0000
ADC_CDR	0x014	Clock Divider Register	0x0000_0000
ADC_MR	0x018	Mode Register	0x0000_0000
ADC_CCSR	0x01C	Conversion Channel Sequence Register	0x0000_0000
ADC_SR	0x020	Status Register	0x0000_0000
ADC_IMSCR	0x024	Interrupt Mask Set/Clear Register	0x0000_0000
ADC_RISR	0x028	Raw Interrupt Status Register	0x0000_0000
ADC_MISR	0x02C	Masked Interrupt Status Register	0x0000_0000
ADC_ICR	0x030	Interrupt Clear Register	0x0000_0000
ADC_CRR	0x034	Conversion Result Register	0x0000_0000
ADC_GCR	0x038	Gain Calibration Register	0x0000_0000
ADC_OCR	0x03C	Offset Calibration Register	0x0000_0000
ADC_DMACR	0x040	DMA Control Register	0x0000_0000

5.3.1.1 ADC_IDR (ADC ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0011_001F

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Register This field stores the ID code for the corresponding IP.	0x0011_001F

5.3.1.2 ADC_CEDR (ADC Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	ADC Clock Enable/Disable Bit 0 = Disable ADC Clock 1 = Enables ADC Clock ADC software reset dose not affect ADCCLK bit status.	0
DBGEN	[31]	RW	Debug enable bit 0 = Disable debug mode ADC is not halted during processor debug mode. 1 = Enable debug mode ADC is halted during processor debug mode.	0

5.3.1.3 ADC_SRR (ADC Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	ADC Software Reset 0 = No effect 1 = Perform ADC Software Reset operation and auto-cleared	0

5.3.1.4 ADC_CSR (ADC Control Set Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																													STOP	START	ADCEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W

Name	Bit	Type	Description	Reset Value
ADCEN	[0]	W	ADC Enable Bit 0 = No effect 1 = Enable ADC Core **NOTE:** After ADC block is enabled (ADCEN==1), TBD us stabilization time must be needed. ADCSTABLE bit in SR (status register) will be set after stabilization.	0
START	[1]	W	ADC Conversion Start Bit: This bit generates software ADC trigger signal. This bit is auto-clear bit. 0 = No effect 1 = Start	0
STOP	[2]	W	ADC Conversion Stop in continuous conversion. This bit is auto-clear bit. 0 = No effect 1 = Stop the continuous conversion. **NOTE:** Before starting conversion, users should ensure that ADC is ready for conversion (BUSY bit is set to logical one in ADC_SR).	0

5.3.1.5 ADC_CCR (ADC Control Clear Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																															ADCEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
ADCEN	[0]	W	ADC Disable Control Bit 0 = No effect 1 = Disable ADC Core	0

5.3.1.6 ADC_CDR (ADC Clock Divider Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																										CDIV					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
CDIV	[5:0]	RW	ADC Clock Divider Selection Field This field can determine the ADC clock frequency. That frequency is from 1MHz to 5MHz. For fast conversion, user should generate the maximum frequency with clock divider. ADC clock source is PCLK. `FADC = PCLK / (2*(N+1))` **NOTE:** The clock rate to the ADC must not be exceeded 700KHz.	00000'b

5.3.1.7 ADC_MR (ADC Mode Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
CCNT			CMODE	RSVD	EICR1	RSVD	CALEN	RSVD																TRIG[2:0]			RSVD				
0	0	0	0	0	0	0	0	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	R	RW	R	RW	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
TRIG[2:0]	[5:7]	RW	ADC Start Signal Selection Bits: This field can determine the trigger signal for ADC. 000 = Software 001 = ADTRG – Rising 010 = ADTRG – Falling 011 = ADTRG – Rising or Falling (both) 100 = TMCx	0000'b
CALEN	[24]	RW	Calibration Enable 0 = Disable Calibration 1 = Enable Calibration, ADC_DR register obtains the conversion data passed calibration process.	0
EICR	[26]	RW	External/Internal Calibration Reference 0 = Select the external reference voltage sources (1/4AVDD and 3/4AVDD) to determinate the calibration constants 1 = Select he internal reference voltage sources (1/4AVDD and 3/4AVDD) to determinate the calibration constants **NOTE:** When user can't use ADC without calibration, CALEN bit is '0', EICR should be '0'. User must choose only external voltage because 10bit ADC has no internal reference voltage generation logic	0
CMODE	[28]	RW	Conversion Mode Control Bit 0 = One shot mode. ADC converts as much inputs as specified by the CCNT[2:0] in the order specified in the ADC_MR, and stops. 1 = Continuous mode. ADC converts as much inputs as specified by the CCNT[2:0] in the order specified in the ADC_MR, and repeats. This bit is initialized to 0. **NOTE:** In continuous mode, after a stop command, the ADC finishes the on-going conversion and this may look like to an extra conversion.	0
CCNT[2:0]	[29:31]	RW	Conversion Count Field	0

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td>

NOTE: Even in one shot mode, ADC will run multiple conversions if the CCNT[2:0] is greater than 0000b.

Table 5-6 CCNT[2:0] Values and the Count Value of Conversions

<table>
<tr><th>CCNT[2:0]</th><th>Count for Conversion</th></tr>
<tr><td>000</td><td>One conversion operation of ICNUM0[2:0] channel</td></tr>
<tr><td>001</td><td>2 times conversion operation from ICNUM0[2:0] to ICNUM1[2:0]</td></tr>
<tr><td>010</td><td>3 times conversion operation from ICNUM0[2:0] to ICNUM2[2:0]</td></tr>
<tr><td>011</td><td>4 times conversion operation from ICNUM0[2:0] to ICNUM3[2:0]</td></tr>
<tr><td>100</td><td>5 times conversion operation from ICNUM0[2:0] to ICNUM4[2:0]</td></tr>
<tr><td>101</td><td>6 times conversion operation from ICNUM0[2:0] to ICNUM5[2:0]</td></tr>
<tr><td>110</td><td>7 times conversion operation from ICNUM0[2:0] to ICNUM6[2:0]</td></tr>
<tr><td>111</td><td>8 times conversion operation from ICNUM0[2:0] to ICNUM7[2:0]</td></tr>
</table>

</td><td></td></tr>
</table>

5.3.1.8 ADC_CCSR (ADC Conversion Channel Sequence Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD	ICNUM7[2:0]			RSVD	ICNUM6[2:0]			RSVD	ICNUM5[2:0]			RSVD	ICNUM4[2:0]			RSVD	ICNUM3[2:0]			RSVD	ICNUM2[2:0]			RSVD	ICNUM1[2:0]			RSVD	ICNUM0[2:0]		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ICNUMx[2:0]	[2:0] [6:4] [10:8] [14:12] [18:16] [22:20] [26:24] [30:28]	RW	Analog Input Channel Number Selection Field Conversion sequence can consist of from first conversion channel defined by ICNUM0[2:0] to last conversion channel defined by ICNUMx[2:0]. ‘x’ can be from 1 to 7.	000'b

Table 5-7 ICNUMx[2:0] Values

ICNUMx Values	Selected Pin
000	AIN 10
001	AIN 11
010	AIN 12
011	AIN 13
100	AIN 14
101	AIN 15
110	AIN 16
111	AIN 17

5.3.1.9 ADC_SR (ADC Status Register)

- Address = Base Address + 0x020, Reset Value = 0x0000_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
RSVD																												CTCVS	ADCEN	BUSY	ADCSTABLE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ADCSTABLE	[0]	R	ADC Stabilization Status 0 = ADC is not stabilized 1 = ADC is stabilized. This bit will be set after initialization time(TBD).	0
BUSY	[1]	R	ADC status monitoring bit 0 = ADC is not on a conversion operating 1 = ADC is on a conversion operating **NOTE:** To change the configuration of ADC, you must check ADC_SR (ADC status register).	0
ADCEN	[2]	R	ADC Enable / Disable Status 0 = ADC is disabled. 1 = ADC is enabled.	0
CTCVS	[3]	R	Continuous mode status 0 = One shot mode with help of microprocessor. 1 = Continuous mode, the peripheral is stand-alone. This bit is initialized to 0 and changes when there is a change of mode. This bit never generates any interruption.	0

5.3.1.10 ADC_IMSCR (ADC Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x024 Reset Value = 0x0000_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
RSVD																														OVR	EOC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W

Name	Bit	Type	Description	Reset Value
EOC	[0]	RW	End of Conversion Interrupt Mask 0 = EOC interrupt is masked. (Disable the interrupt) 1 = EOC interrupt is not masked. (Enable the interrupt)	0
OVR	[1]	RW	Overrun Interrupt Mask 0 = OVR interrupt is masked. (Disable the interrupt) 1 = OVR interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

5.3.1.11 ADC_RISR (ADC Raw Interrupt Status Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																														OVR	EOC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EOC	[0]	R	End of Conversion Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the EOC interrupt	0
OVR	[1]	R	Overrun Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the OVR interrupt	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

5.3.1.12 ADC_MISR (ADC Masked Interrupt Status Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																						RSVD	RSVD	RSVD						OVR0	EOC0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EOC	[0]	R	End of Conversion Masked Interrupt State Gives the masked interrupt status(after masking) of the EOC interrupt	0
OVR	[1]	R	Overrun Masked Interrupt State Gives the masked interrupt status(after masking) of the OVR interrupt	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

5.3.1.13 ADC_ICR (ADC Interrupt Clear Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																														OVR	EOC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W

Name	Bit	Type	Description	Reset Value
EOC	[0]	W	End of conversion interrupt 0 = No effect 1 = Clears the EOC interrupt	0
OVR	[1]	W	Overrun interrupt 0 = No effect 1 = Clears the OVR interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

5.3.1.14 ADC_CRR (ADC Conversion Result Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																						DATA									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DATA	[9:0]	R	Conversion Result Data of ADC The result data from an analog to digital conversion is latched into this register at the end of a conversion and remains valid until a new conversion is completed. When this register is read, the EOC bit in the ADC_SR register is cleared. **NOTE:** When debugging, to avoid clearing EOC bit, users should use ghost registers.	0x000

5.3.1.15 ADC_GCR (ADC Gain Calibration Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																	GCC_INT	GCC_FRAC													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
GCC_FRAC	[13:0]	RW	ADC Gain Calibration Constant Value. ADCGCC field is fixed point unsigned value. ADCGCC[14:0] = Fixed point unsigned value GCC × 2^{15} The Fixed point unsigned value GCC gets from the following (For more detail, refer to *4.2.9.1*) `GCC = (IDLE75% - IDLE25%) / (RAW75% - RAW25%)` The calculated result (GCC) should be limited from 0.5 to 1.99999.	0x0000
GCC_INT	[14]	RW	–	0

5.3.1.16 ADC_OCR (ADC Offset Calibration Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																				ADCOCC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADCOCC	[11:0]	RW	ADC Offset Calibration Constant Value. ADCOCC field is signed value. Negative values should be expressed using the 2’s complement.	0x0000

5.3.1.17 ADC_DMACR (ADC DMA Control Register)

- Address = Base Address + 0x0040, Reset Value = 0x0000_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
RSVD																															DMAE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
DMAE	[0]	RW	DMA for ADC Enable/Disable Control Bit. 0 = Disable 1 = Enable	0

6 Controller Area Network (CAN)

6.1 Overview

This chapter describes the Controller Area Network (CAN) Block.

6.1.1 Features

The CAN module consists of the components CAN Core, Message RAM, Message Handler, Control Registers and Module Interface.

The CAN Core performs communication according to the CAN protocol version 2.0 part A and B. The bit rate can be programmed to values up to 1MBit/s depending on the used technology. For the connection to the physical layer additional transceiver hardware is required.

For communication on a CAN network, individual Message Objects are configured. The Message Objects and Identifier Masks for acceptance filtering of received messages are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the Message Handler. Those functions are the acceptance filtering, the transfer of messages between the CAN Core and the Message RAM, and the handling of transmission requests as well as the generation of the module interrupt.

The CAN implements the following features:

- Supports CAN protocol version 2.0 part A and B
- Bit rates up to 1 MBit/s
- 32 Message Objects
- Each Message Object has its own identifier mask
- Programmable FIFO mode (concatenation of Message Objects)
- Maskable interrupt
- Disabled Automatic Retransmission mode for Time Triggered CAN applications
- Programmable loop-back mode for self-test operation
- Power management block and wake up mode allowing optimization of power consumption
- CAN_TX output pin configurable in open drain allowing connection without external transceiver

6.1.2 Pin Description

Table 6-1 Controller Area Network (CAN) Pin Description

Pin Name	Function	I/O Type	Active Level	Comments
CANTX[1:0]	Transmit Data Output	O	Low	–
CANRX[1:0]	Receive Data Input	I	Low	–

6.2 Functional Description

6.2.1 Block Diagram

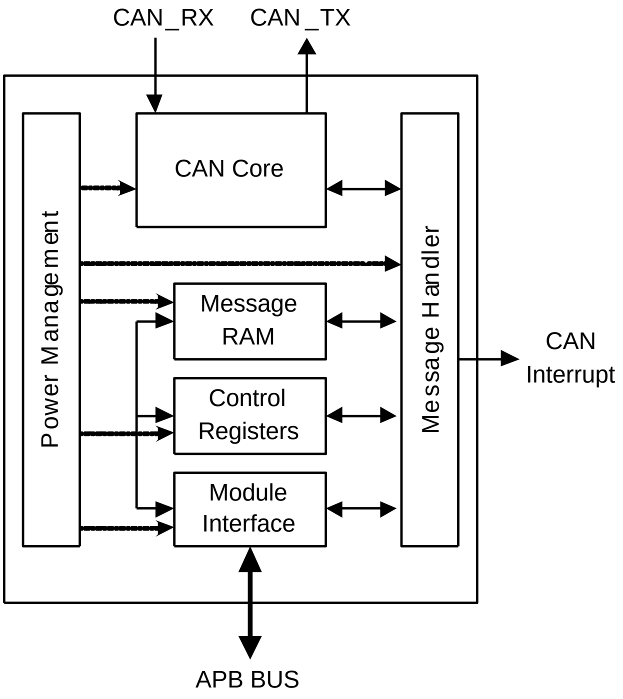


Figure 6-1 Controller Area Network (CAN) Block Diagram

6.2.2 General Operation

6.2.2.1 Introduction of CAN Protocol

The Controller Area Network (CAN) is a serial communication protocol which efficiently supports distributed real-time control with a very high level of security.

Its domain of application ranges from high speed networks to low-cost multiplex wiring.
For example, in automotive, electronics, engine control units, sensors, anti-skid systems, etc. may be connected using CAN with bit rates up to 1 Mbit/s. At the same time, it is cost effective to build into vehicle body electronics such as lamp clusters, electric windows, etc... to replace the wiring harness otherwise required.

The intention of this specification is to achieve compatibility between any two CAN implementations. Compatibility, however, has different aspects regarding e.g. electrical features and the interpretation of data to be transferred.

To achieve design transparency and implementation flexibility, CAN has been subdivided into different layers according to the ISO/OSI reference model:

- The Data Link Layer
  - The Logical Link Control (LLC) sub-layer
  - The Medium Access Control (MAC) sub-layer

- The Physical Layer

NOTE: In previous versions of CAN specification, services and functions of the LLC and MAC sub-layers of the Data Link Layer were described in layers denoted as ’object layer’ and ’transfer layer’.

The scope of the LLC sub-layer includes: determining which messages received by the LLC sub-layer are actually to be accepted; providing services for data transfer and for remote data request; and providing the means for recovery management and overload notifications. There is much freedom in defining object handling.

The scope of the MAC sub-layer mainly is the transfer protocol, i.e. controlling the framing, performing arbitration, error checking, error and fault confinement. Within the MAC sub-layer it is decided whether the bus is free for starting signaling a new transmission or whether a reception is just starting. Also some general features of the bit timing are regarded as part of the MAC sub-layer. It is in the nature of the MAC sub-layer that there is no freedom for modifications.

The scope of the physical layer is the actual transfer of the bits between the different nodes with respect to all electrical properties. Within a network the physical layer, of course, has to be the same for all nodes. There are, however, many possible implementations of physical layer.

The scope of this specification is to define the MAC sub-layer and a small part of the LLC sub-layer of the Data Link Layer and to describe the consequences of the CAN protocol on the surrounding layers.

6.2.3 Basic Concepts

6.2.3.1 CAN Properties

CAN has the following properties:

- Prioritization of messages
- Guarantee of latency times
- Configuration flexibility
- Multicast reception with time synchronization
- System wide data consistency
- Multi-master
- Error detection and error signaling
- Automatic retransmission of corrupted messages as soon as the bus is idle again
- Distinction between temporary errors and permanent failures of nodes and autonomous switching off of defect nodes

6.2.3.2 Layered Structure of a CAN Node

Layered architecture of CAN according to the ISO/OSI reference model:

- The LLC sub-layer is concerned with message filtering, overload notification and recovery management.
- The MAC sub-layer represents the kernel of the CAN protocol. It presents messages received from the LLC sub-layer and accepts messages to be transmitted to the LLC sub-layer. The MAC sub-layer is responsible for Message Framing, Arbitration, Acknowledgement, Error Detection and Signaling. The MAC sub-layer are supervised by a management entity called Fault Confinement which is self-checking mechanism for distinguishing short disturbances from permanent failures.
- The physical layer defines how signals are actually transmitted and therefore deals with the description of bit timing, bit encoding, and synchronization. Within this specification the physical layer is not defined, as it will vary according to the requirements of individual applications (for example, transmission medium and signal level implementations).

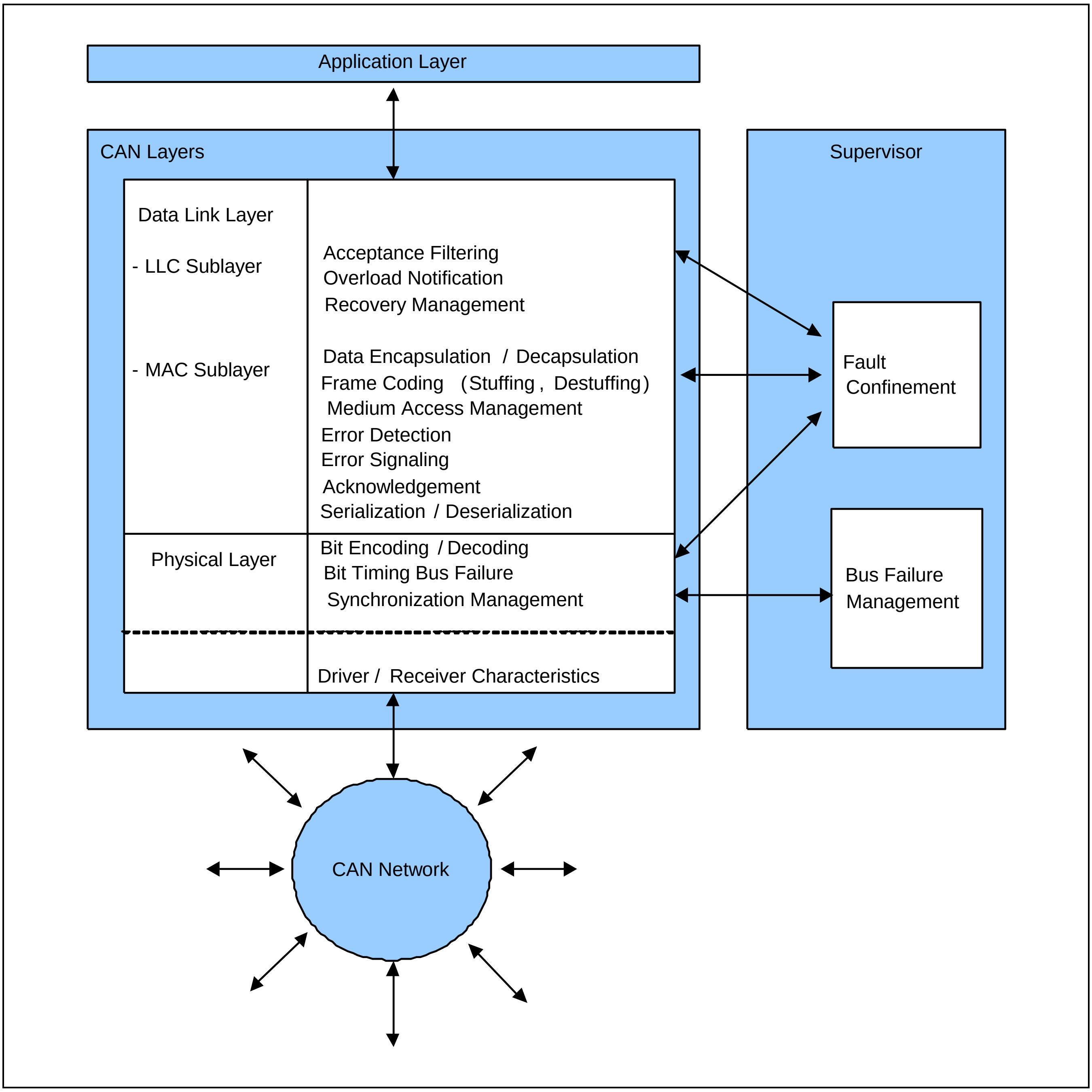


Figure 6-2 CAN Layers Description

- LLC = Logical Link Control
- MAC = Medium Access Control

The scope of this specification is to define the Data Link Layer and the consequences of the CAN protocol on the surrounding layers.

6.2.3.3 Messages

Information on the bus is sent in fixed format messages of different but limited length (see section *Message Transfer* below). When the bus is free, any connected unit may start to transmit a new message.

6.2.3.4 Information Routing

- In CAN systems, a CAN node does not make use of any information about the system configuration (e.g. node addresses). This has several important consequences:
- System flexibility: nodes can be added to the CAN network without requiring any change in the software or hardware of any node and application layer.
- Message routing: the content of a message is named by an identifier. The identifier does not indicate the destination of the message, but describes the meaning of the data, so that all nodes in the network are able to decide by Message Filtering whether the data is to be acted upon by them or not.
- Multicast: as a consequence of the concept of Message Filtering any number of nodes can receive and simultaneously act upon the same message.
- Data consistency: within a CAN network it is guaranteed that a message is simultaneously accepted either by all nodes or by no node. Thus data consistency of a system is achieved by the concepts of multicast and by error handling.

6.2.3.5 Bit Rate

The speed of CAN may be different in different systems. However, in a given system the bit-rate is uniform and fixed.

6.2.3.6 Properties

The identifier defines a static message priority during bus access.

6.2.3.7 Remote Data Request

By sending a Remote Frame a node requiring data may request another node to send the corresponding Data Frame. The Data Frame and the corresponding Remote Frame are named by the same Identifier.

6.2.3.8 Multi-Master

When the bus is free any unit may start to transmit a message. The message unit to be transmitted with the highest priority gains bus access.

6.2.3.9 Arbitration

Whenever the bus is free, any unit may start to transmit a message. If two or more units start transmitting messages at the same time, the bus access conflict is resolved by bit-wise arbitration using the Identifier. The mechanism of arbitration guarantees that neither information nor time is lost. If a Data Frame and a Remote Frame with the same Identifier are initiated at the same time, the Data Frame prevails over the Remote Frame. During arbitration every transmitter compares the level of the bit transmitted with the level that is monitored on the bus. If these levels are equal, the unit may continue to send. When a ’recessive’ level is sent and a ’dominant’ level is monitored (see *Bus Values*), the unit has lost arbitration and must withdraw without sending one more bit.

6.2.3.10 Safety

In order to achieve the utmost safety of data transfer, powerful measures for error detection, signalling and self-checking are implemented in every CAN node.

6.2.3.11 Error Detection

For detecting errors the following measures have been taken:

- Monitoring (transmitters compare the bit levels to be transmitted with the bit levels detected on bus).
- Cyclic Redundancy Check (CRC).
- Bit stuffing.
- Message frame check

6.2.3.12 Performance of Error Detection

The error detection mechanisms have the following properties:

- All global errors are detected (by means of monitoring).
- All local errors at transmitters are detected (by means of monitoring).
- Up to 5 randomly distributed errors in a message are detected (by means of CRC).
- Burst errors of length less than 15 in a message are detected (by means of CRC).
- Errors of any odd number in a message are detected (by means of CRC).

Total residual error probability for undetected corrupted messages is less than: message error rate $\times$ 4.7 $\times$ 10^{-11}.

6.2.3.13 Error Signaling and Recovery Time

Corrupted messages are flagged by any node detecting an error. Such messages are aborted and will be retransmitted automatically. The recovery time from detecting an error until the start of the next message is at most 31 bit times, if there is no further error.

6.2.3.14 Fault Confinement

CAN nodes are able to distinguish short disturbances from permanent failures. Defective nodes are switched off.

6.2.3.15 Connections

The CAN serial communication link is a bus to which a number of units may be connected. This number has no theoretical limit. Practically the total number of units will be limited by delay times and/or electrical loads on the bus line.

6.2.3.16 Single Channel

The bus consists of a single bi-directional channel that carries bits. From this data resynchronization information can be derived. The way in which this channel is implemented is not fixed in this specification. E.g. single wire (plus ground), two differential wires, optical fibers, etc.

6.2.3.17 Bus Values

The bus can have one of two complementary logical values: ’dominant’ or ’recessive’. During simultaneous transmission of ’dominant’ and ’recessive’ bits, the resulting bus value will be ’dominant’. For example, in case of a wired-AND implementation of the bus, the ‘dominant’ level would be represented by a logical ’0’ and the ’recessive’ level by a logical ’1’. Physical states (e.g. electrical voltage, light) that represent the logical levels are not given in this specification.

6.2.3.18 Acknowledgement

All receivers check the consistency of the message being received and will acknowledge a consistent message and flag an inconsistent message.

6.2.3.19 Sleep Mode / Wake-Up

To reduce the system’s power consumption, a CAN-device may be set into sleep mode without any internal activity and with disconnected bus drivers. The sleep mode is finished with a wake-up by any bus activity or by internal conditions of the system. On wake-up, the internal activity is restarted, although the MAC sublayer will be waiting for the system’s oscillator to stabilize and it will then wait until it has synchronized itself to the bus activity (by checking for eleven consecutive ’recessive’ bits), before the bus drivers are set to "on-bus" again.

6.2.4 Message Transfer

6.2.4.1 Definition of Transmitter/Receiver

A node originating a message is called transmitter of that message. The node stays transmitter until the bus is idle or the node loses arbitration. A node is called receiver of a message, if it is not transmitter of that message and the bus is not idle.

6.2.4.2 Frame Formats

There are two different formats which differ in the length of the identifier field:

Table 6-2 Identifier Length within Standard and Extended Frame

Frame Format	Identifier Length
Standard frame	11 bit
Extended frame	29 bit

6.2.4.3 Frame Types

Message transfer is manifested and controlled by four different frame types:

- A data frame carries data from a transmitter to the receivers.
- A remote frame is transmitted by a bus unit to request the transmission of the data frame with the same identifier.
- An error frame is transmitted by any unit on detecting a bus error.
- An overload frame is used to provide for an extra delay between the preceding and the succeeding data or remote frames.

Data frames and remote frames can be used both in standard frame format and extended frame format. They are separated from preceding frames by an Inter-frame space.

6.2.4.3.1 Data Frame

A data frame is composed of seven different bit fields:

- Start of Frame (SOF)
- Arbitration field
- Control field
- Data field
- CRC field
- ACK field
- End of Frame (EOF)

The data field can be of length zero.

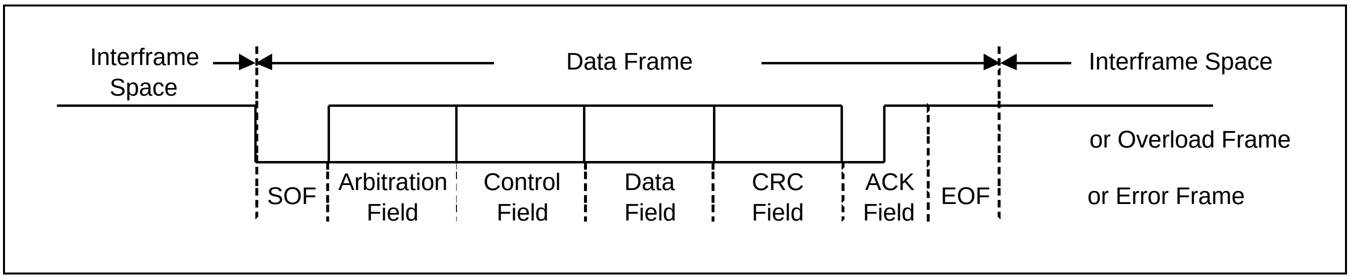


Figure 6-3 Data Frame

6.2.4.3.2 Start of Frame (SOF, Standard and Extended Format)

The Start of Frame (SOF) marks the beginning of Data Frames and Remote Frames. It consists of a single ’dominant’ bit.

A node is only allowed to start transmission when the bus is idle (see '*Inter-frame Spacing*'). All nodes have to synchronize to the leading edge caused by Start of Frame (see '*Hard Synchronization*') of the node starting transmission first.

6.2.4.3.3 Arbitration Field

The format of the Arbitration Field is different for standard format and extended format frames.

In standard format the Arbitration Field consists of the 11 bit identifier and the RTR bit. The identifier bits are denoted ID[28:18].

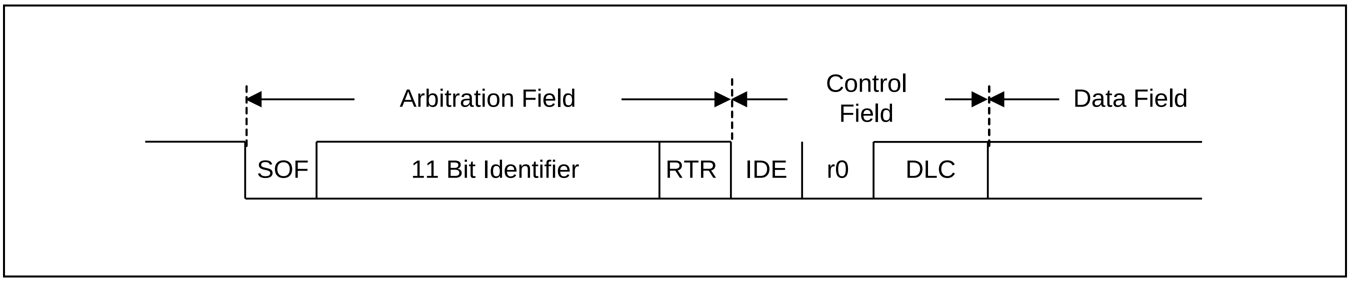


Figure 6-4 Standard Format

In Extended Format the Arbitration Field consists of the 29 bit Identifier, the SRR bit, the IDE bit, and the RTR bit. The Identifier bits are denoted ID[28:0].

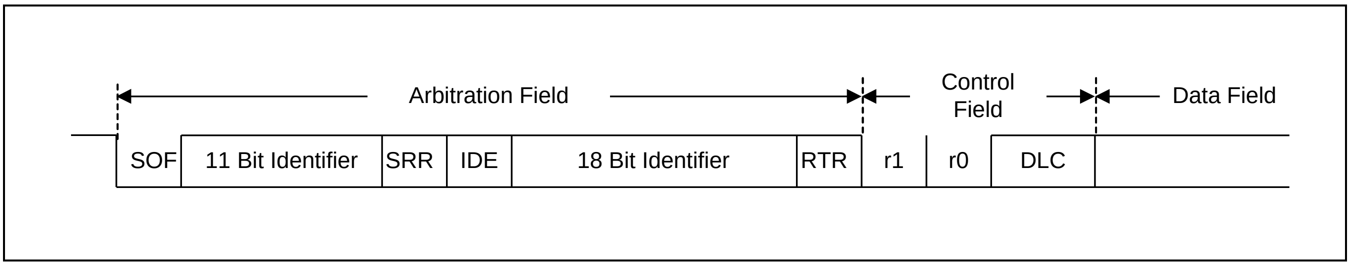


Figure 6-5 Extended Format

In order to distinguish between Standard Format and Extended Format the reserved bit r1 in previous CAN specifications version 1.0 - 1.2 now is denoted as IDE bit.

- Identifier
  - IDENTIFIER - Standard Format
    The IDENTIFIER's length is 11 bits and corresponds to the base ID in Extended Format. These bits are transmitted in the order from ID28 to ID18. The least significant bit is ID18. The 7 most significant bits ID[28:22] must not be all 'recessive'.
  - IDENTIFIER - Extended Format
    In contrast to the Standard Format the Extended Format consists of 29 bits. The format comprises two sections: Base ID with 11 bits and the Extended ID with 18 bits.
    - Base ID: the Base ID consists of 11 bits. It is transmitted in the order from ID28 to ID18. It is equivalent to format of the Standard Identifier. The base ID defines the Extended Frame's base priority.
    - Extended ID: the Extended ID consists of 18 bits. It is transmitted in the order of ID17 to ID0.

In a Standard Frame the IDENTIFIER is followed by the RTR bit.

- RTR bit (Standard and Extended Format)
  Find in the table below the RTR bit logical value according to frame type:

Table 6-3 RTR Bit

Frame Type	RTR Bit Logical Value
Data frame	'dominant'
Remote frame	'recessive'

In an Extended Frame the base ID is transmitted first, followed by the IDE bit and the SRR bit. The extended ID is transmitted after the SRR bit.

- SRR bit
  The SRR bit (Substitute Remote Request) is a recessive bit. It is transmitted in Extended Frames at the position of the RTR bit in Standard Frames and so substitutes the RTR bit in the Standard Frame. Therefore, collisions of a Standard Frame and an Extended Frame, the base ID (see 'Extended IDENTIFIER' below) of which is the same as the Standard Frame's Identifier, are resolved in such a way that the Standard Frame prevails the Extended Frame.
- IDE bit (Extended Format)
  Find in the table below the IDE bit logical value and membership according to frame format type:

Table 6-4 IDE Bit

Frame Format Type	IDE Bit Belongs to	IDE Bit Logical Value
Standard frame format	the control field	'dominant'
Extended frame format	the arbitration field	'recessive'

6.2.4.3.4 Control Field (Standard and Extended Format)

The CONTROL FIELD consists of six bits. The format of the CONTROL FIELD is different for Standard Format and Extended Format. Frames in Standard Format include the DATA LENGTH CODE, the IDE bit, which is transmitted 'dominant' (see above), and the reserved bit r0. Frames in Extended Format include the DATA LENGTH CODE and two reserved bits r1 and r0. The reserved bits have to be sent 'dominant', but Receivers accept 'dominant' and 'recessive' bits in all combinations.

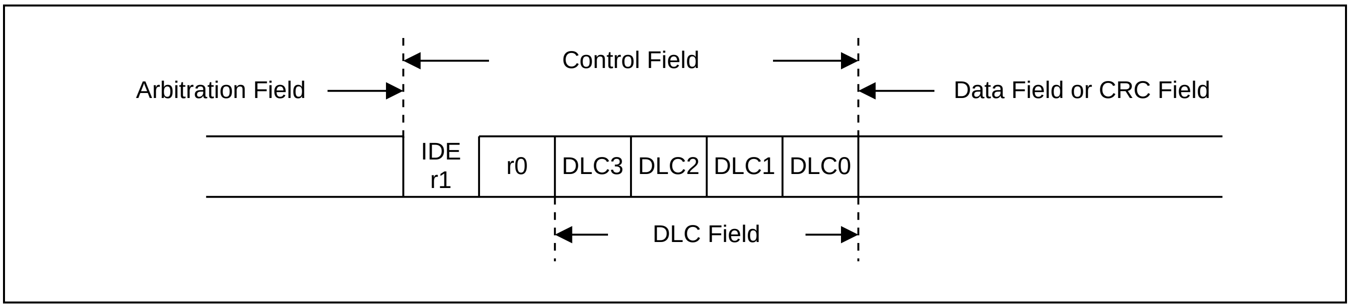


Figure 6-6 CAN Control Field

- Data Length Code (Standard and Extended format)
  The number of bytes in the DATA FIELD is indicated by the DATA LENGTH CODE. The DLC field is four bits wide and is transmitted within the CONTROL FIELD. Coding of the number of data bytes by the DLC field is described in the table below:

Table 6-5 Data Length Code (NOTE)

Number of Data (Bytes)	Data Length Code			
	DLC 3	DLC 2	DLC 1	DLC 0
0	D	D	D	D
1	D	D	D	R
2	D	D	R	D
3	D	D	R	R
4	D	R	D	D
5	D	R	D	R
6	D	R	R	D
7	D	R	R	R
8	R	D	D	D

NOTE: In table, D means dominant bit, R means recessive bit.

6.2.4.3.5 Data Frame (Standard and Extended Format)

The admissible numbers of data bytes is 0 to 8. Other values may not be used.

6.2.4.3.6 Data Field (Standard and Extended Format)

The DATA FIELD consists of the data to be transferred within a DATA FRAME. It can contain from 0 to 8 bytes, which each contain 8 bits which are transferred MSB first.

6.2.4.3.7 CRC Field (Standard and Extended Format)

The CRC (Cyclic Redundancy Check) FIELD contains the CRC SEQUENCE followed by a CRC DELIMITER.

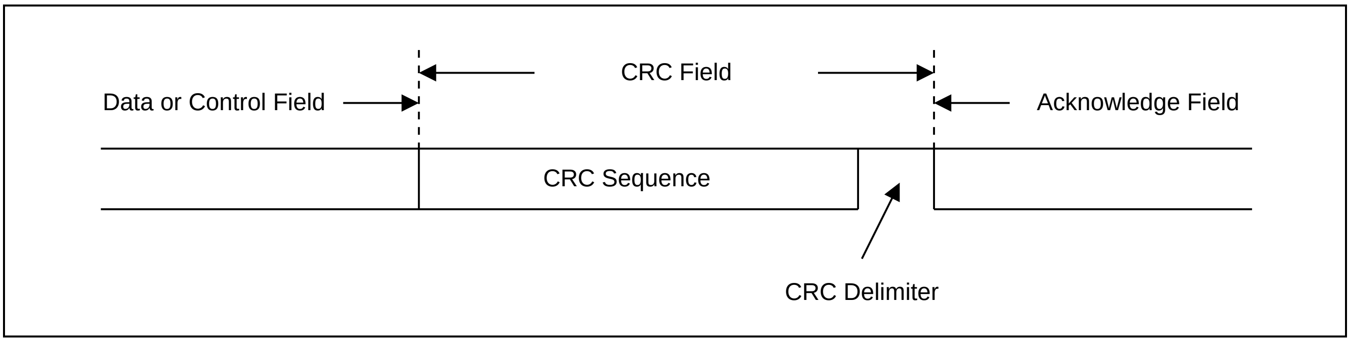


Figure 6-7 CRC Field

- CRC Sequence (Standard and Extended Format)
  The frame check sequence is derived from a cyclic redundancy code best suited for frames with bit counts less than 127 bits (BCH code).

  In order to carry out the CRC calculation, the polynomial to be divided is defined as the polynomial, the coefficients of which are given by the de-stuffed bit stream consisting of START OF FRAME, ARBITRATION FIELD, CONTROL FIELD, DATA FIELD (if present) and, for the 15 lowest coefficients, by 0. This polynomial is divided (the coefficients are calculated modulo-2) by the generator-polynomial:

  $X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1.$

  The remainder of this polynomial division is the CRC SEQUENCE transmitted over the bus.

  In order to implement this function, a 15 bit shift register CRC_RG (14:0) can be used. If NXTBIT denotes the next bit of the bit stream, given by the de-stuffed bit sequence from START OF FRAME until the end of the DATA FIELD, the CRC SEQUENCE is calculated as follows:

  ```
  CRC_RG = 0// initialize shift register
  REPEAT CRCNXT = NXTBIT EXOR CRC_RG[14]
  CRC_RG[14:1] = CRC_RG[13:0]// shift left by
  CRC_RG[0] = 0// 1 position
  IF CRCNXT THEN
  CRC_RG[14:0] = CRC_RG[14:0] EXOR 0x4599
  ENDIF
  UNTIL (CRC sequence starts or there is an ERROR condition)
  ```

  After the transmission/reception of the last bit of the DATA FIELD, CRC_RG contains the CRC sequence.

- CRC Delimiter (standard and extended format)
  The CRC SEQUENCE is followed by the CRC DELIMITER which consists of a single ’recessive’ bit.

6.2.4.3.8 ACK Field (Standard and Extended Format)

The ACK (acknowledgement) FIELD is two bits long and contains the ACK SLOT and the ACK DELIMITER. In the ACK FIELD the transmitting node sends two ’recessive’ bits.

A RECEIVER which has received a valid message correctly, reports this to the TRANSMITTER by sending a ’dominant’ bit during the ACK SLOT (it sends ’ACK’).

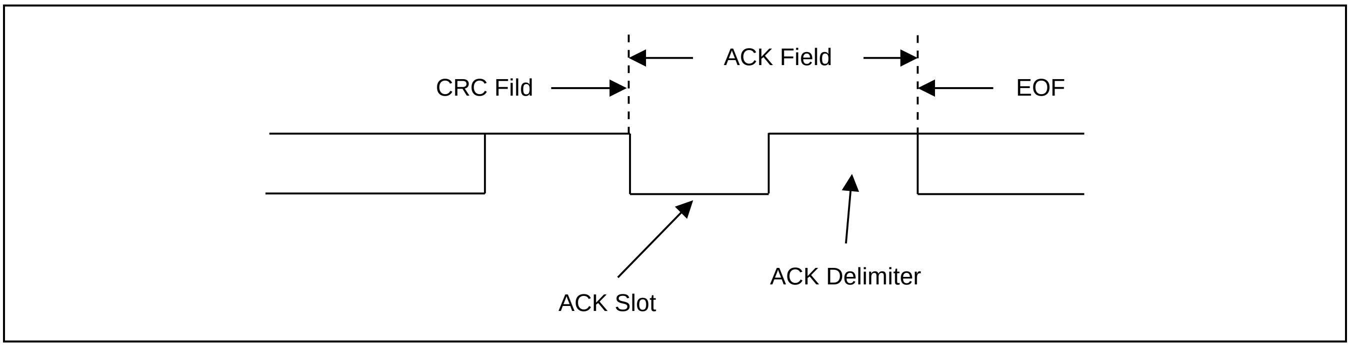


Figure 6-8 ACK Field

- ACK SLOT:
  All stations having received the matching CRC SEQUENCE report this within the ACK SLOT by superscribing the ‘recessive’ bit of the TRANSMITTER by a ‘dominant’ bit.

- ACK DELIMITER:
  The ACK DELIMITER is the second bit of the ACK FIELD and has to be a ‘recessive’ bit. As a consequence, the ACK SLOT is surrounded by two ‘recessive’ bits (CRC DELIMITER, ACK DELIMITER).

6.2.4.3.9 End of Frame (Standard and Extended Format)

Each DATA FRAME and REMOTE FRAME is delimited by seven ’recessive’ bits. These bits form the END OF FRAME SEQUENCE (EOF).

6.2.4.3.10 Remote Frame

A node acting as a RECEIVER for certain data can initiate the transmission of the respective data by its source node by sending a REMOTE FRAME. A REMOTE FRAME exists both in Standard Format and in Extended Format. In both case it is composed of six different bit fields:

Start of Frame (SOF)

- Arbitration field
- Control field
- CRC field
- ACK field
- End of Frame

Contrary to DATA FRAMEs, the RTR bit of REMOTE FRAMEs is 'recessive'. There is no DATA FIELD, independent of the values of the DATA LENGTH CODE which may be signed any value within the admissible range from 0 to 8. The value is the DATA LENGTH CODE of the corresponding DATA FRAME.

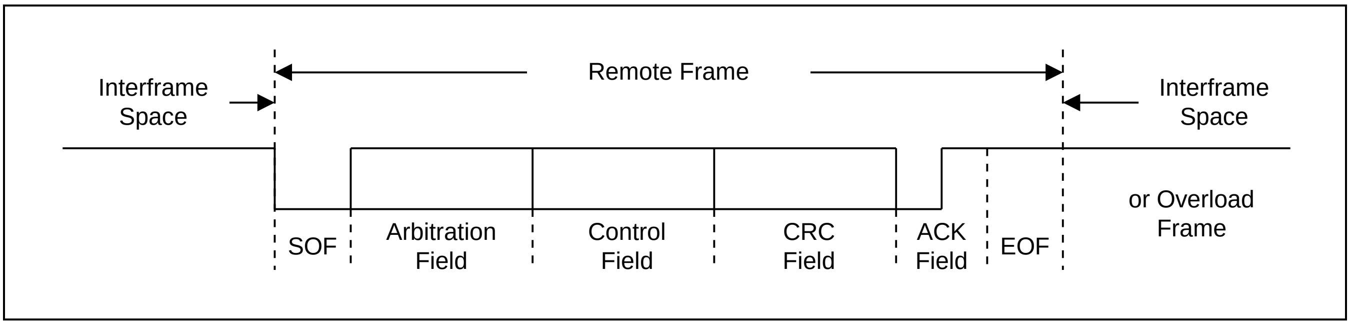


Figure 6-9 Remote Frame

The polarity of the RTR bit indicates whether a transmitted frame is a DATA FRAME (RTR bit 'dominant') or a REMOTE FRAME (RTR bit 'recessive').

6.2.4.3.11 Error Frame

The ERROR FRAME consists of two different fields. The first field is given by the superposition of ERROR FLAGs contributed from different nodes. The following second field is the ERROR DELIMITER. In order to terminate an ERROR FRAME correctly, an 'error passive' node may need the bus to be 'bus idle' for at least three bit times (if there is a local error at an 'error passive' receiver). Therefore the bus should not be loaded to 100%.

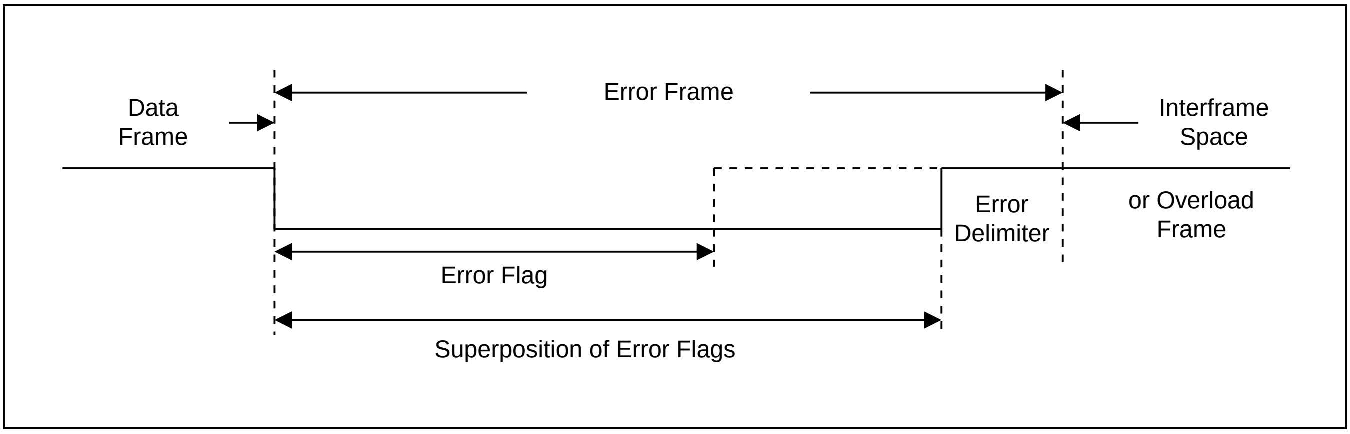


Figure 6-10 Error Frame

6.2.4.3.12 Error Flag

There are two forms of an ERROR FLAG: an ACTIVE ERROR FLAG and a PASSIVE ERROR FLAG.

- The ACTIVE ERROR FLAG consists of six consecutive 'dominant' bits.
- The PASSIVE ERROR FLAG consists of six consecutive 'recessive' bits unless it is overwritten by 'dominant' bits from other nodes.

An 'error active' node detecting an error condition signals this by transmission of an ACTIVE ERROR FLAG. The ERROR FLAG's form violates the law of bit stuffing (see *Coding*) applied to all fields from START OF FRAME to CRC DELIMITER or destroys the fixed form ACK FIELD or END OF FRAME field. As a consequence, all other nodes detect an error condition and on their part start transmission of an ERROR FLAG. So the sequence of 'dominant' bits which actually can be monitored on the bus results from a superposition of different ERROR FLAGs transmitted by individual nodes. The total length of this sequence varies between a minimum of six and a maximum of twelve bits.

An 'error passive' node detecting an error condition tries to signal this by transmission of a PASSIVE ERROR FLAG. The 'error passive' node waits for six consecutive bits of equal polarity, beginning at the start of the PASSIVE ERROR FLAG. The PASSIVE ERROR FLAG is complete when these six equal bits have been detected.

6.2.4.3.13 Error Delimiter

The ERROR DELIMITER consists of eight 'recessive' bits.

After transmission of an ERROR FLAG each node sends 'recessive' bits and monitors the bus until it detects a 'recessive' bit. Afterwards it starts transmitting seven more 'recessive' bits.

6.2.4.3.14 Overload Frame

The OVERLOAD FRAME contains the two bit fields OVERLOAD FLAG and OVERLOAD DELIMITER. There are two kinds of OVERLOAD conditions, which both lead to the transmission of an OVERLOAD FLAG:

- The internal conditions of a receiver, which requires a delay of the next DATA FRAME or REMOTE FRAME.
- Detection of a 'dominant' bit at the first and second bit of INTERMISSION.

If a CAN node samples a dominant bit at the eighth bit (the last one) of an ERROR DELIMITER or OVERLOAD DELIMITER, it will start transmitting an OVERLOAD FRAME (not an ERROR FRAME). The Error Counters will not be incremented.

The start of an OVERLOAD FRAME due to OVERLOAD condition 1 is only allowed to be started at the first bit time of an expected INTERMISSION, whereas OVERLOAD FRAMEs due to OVERLOAD condition 2 and condition 3 start one bit after detecting the 'dominant' bit.

At most two OVERLOAD FRAMEs may be generated to delay the next DATA or REMOTE FRAME.

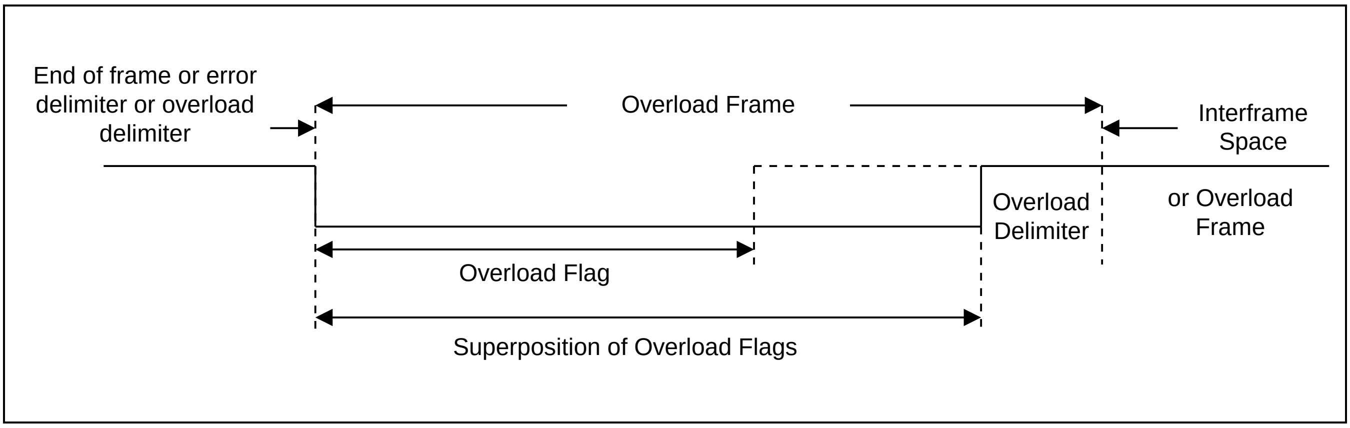

End of frame or error delimiter or overload delimiter
Overload Frame
Interframe Space
or Overload Frame
Overload Delimiter
Overload Flag
Superposition of Overload Flags

Figure 6-11 Overload Frame

6.2.4.3.15 Overload Flag

Consists of six 'dominant' bits. The overall form corresponds to that of the ACTIVE ERROR FLAG.

The OVERLOAD FLAG's form destroys the fixed form of the INTERMISSION field. As a consequence, all other nodes also detect an OVERLOAD condition and on their part start transmission of an OVERLOAD FLAG. In case that there is a 'dominant' bit detected during the 3rd bit of INTERMISSION then it will interpret this bit as START OF FRAME.

NOTE: Controllers based on the CAN Specification version 1.0 and 1.1 have another interpretation of the 3rd bit of INTERMISSION: if a dominant bit was detected locally at some node, the other nodes will not interpret the OVERLOAD FLAG correctly, but interpret the first of these six 'dominant' bits as START OF FRAME, the sixth 'dominant' bit violates the rule of bit stuffing causing an error condition.

6.2.4.3.16 Overload Delimiter

Consists of eight ’recessive’ bits.

The OVERLOAD DELIMITER is of the same form as the ERROR DELIMITER. After transmission of an OVERLOAD FLAG the node monitors the bus until it detects a transition from a ’dominant’ to a ’recessive’ bit. At this point of time every bus node has finished sending its OVERLOAD FLAG and all nodes start transmission of seven more ’recessive’ bits in coincidence.

6.2.4.3.17 Inter-frame Spacing

DATA FRAMEs and REMOTE FRAMEs are separated from preceding frames whatever type they are (DATA FRAME, REMOTE FRAME, ERROR FRAME, OVERLOAD FRAME) by a bit field called INTERFRAME SPACE. In contrast, OVERLOAD FRAMEs and ERROR FRAMEs are not preceded by an INTERFRAME SPACE and multiple OVERLOAD FRAMEs are not separated by an INTERFRAME SPACE.

6.2.4.3.18 Inter-frame Space

Contains the bit fields INTERMISSION and BUS IDLE and, for ’error passive’ nodes, which have been TRANSMITTER of the previous message, SUSPEND TRANSMISSION.

For nodes which are not ’error passive’ or have been RECEIVER of the previous message:

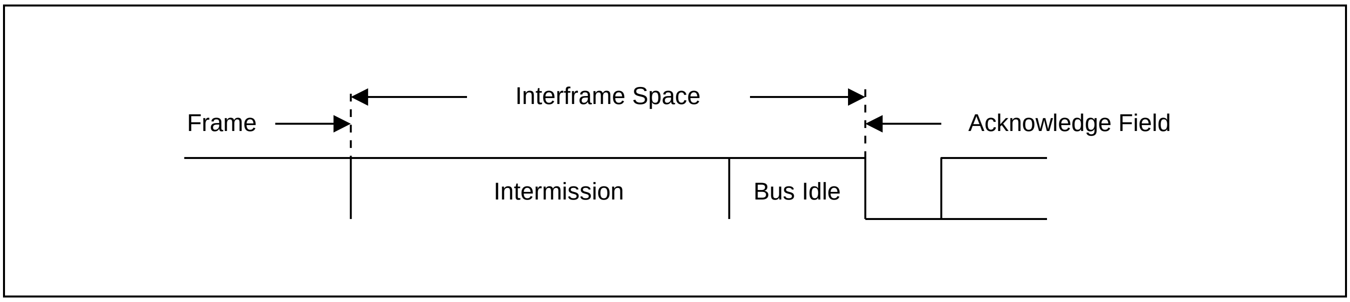


Figure 6-12 Interframe Space for Receiver

For ’error passive’ nodes which have been TRANSMITTER of the previous message:

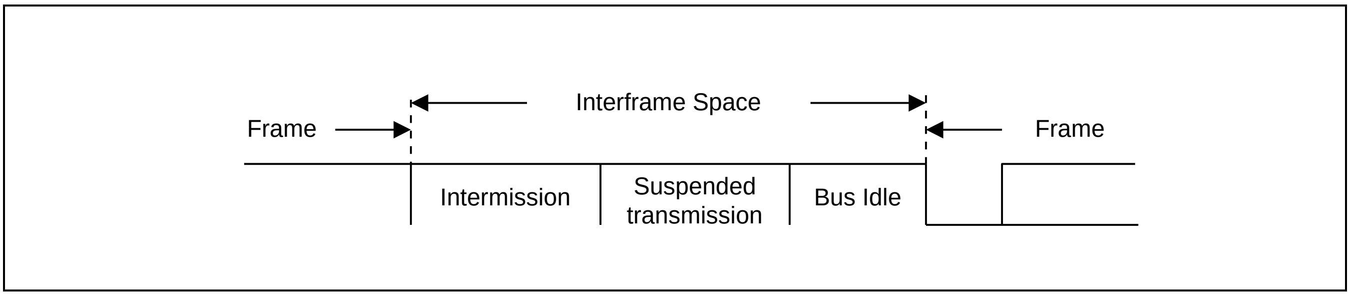


Figure 6-13 Interframe Space for Transmitter

6.2.4.3.19 Intermission

Consists of three ’recessive’ bits.

During INTERMISSION the only action to be taken is signaling an OVERLOAD condition and no node is allowed to actively start a transmission of a DATA FRAME or REMOTE FRAME.

NOTE: If a CAN node has a message waiting for transmission and it samples a dominant bit at the third bit of NTERMISSION, it will interpret this as a START OF FRAME bit, and, with the next bit, start transmitting its message with the first bit of its IDENTIFIER without first transmitting a START OF FRAME bit and without becoming receiver.

6.2.4.3.20 Bus Idle

The period of BUS IDLE may be of arbitrary length. The bus is recognized to be free and any node having something to transmit can access the bus. A message, which is pending for transmission during the transmission of another message, is started in the first bit following INTERMISSION.

The detection of a ’dominant’ bit on the bus is interpreted as START OF FRAME.

6.2.4.3.21 Suspend transmission

After an ’error passive’ node has transmitted a message, it sends eight ’recessive’ bits following INTERMISSION, before starting to transmit a further message or recognizing the bus to be idle. If meanwhile a transmission (caused by another node) starts, the node will become receiver of this message.

6.2.4.3.22 Conformance with Regard to Frame Formats

The Standard Format is equivalent to the Data/Remote Frame Format as it is described in the CAN Specification 1.2. In contrast the Extended Format is a new feature of the CAN protocol. In order to allow the design of relatively simple controllers, the implementation of the Extended Format to its full extend is not required (e.g. send messages or accept data from messages in Extended Format), whereas the Standard Format must be supported without restriction.

New controllers are considered to be in conformance with this CAN Specification, if they have at least the following properties with respect to the Frame Formats defined before.

- Every new controller supports the Standard Format
- Every new controller can receive messages of the Extended Format. This requires that Extended Frames are not destroyed just because of their format. It is, however, not required that the Extended Format must be supported by new controllers.

6.2.5 Message Filtering

Message filtering is based upon the whole Identifier. Mask registers that allow any Identifier bit to be set ’don’t care’ for message filtering, may be used to select groups of Identifiers to be mapped into the attached receive buffers. If mask registers are implemented every bit of the mask registers must be programmable, i.e. they can be enabled or disabled for message filtering. The length of the mask register can comprise the whole IDENTIFIER or only part of it.

6.2.6 Message validation

The point of time at which a message is taken to be valid, is different for the transmitter and the receivers of the message.

6.2.6.1 Transmitter

The message is valid for the transmitter, if there is no error until the end of END OF FRAME. If a message is corrupted, retransmission will follow automatically and according to prioritization. In order to be able to compete for bus access with other messages, retransmission has to start as soon as the bus is idle.

6.2.6.2 Receivers

The message is valid for the receivers, if there is no error until the last but one bit of END OF FRAME. The value of the last bit of END OF FRAME is treated as ’don’t care’, a dominant value does not lead to a FORM ERROR.

6.2.7 Coding

BIT STREAM CODING: the frame segments START OF FRAME, ARBITRATION FIELD, CONTROL FIELD, DATA FIELD and CRC SEQUENCE are coded by the method of bit stuffing. Whenever a transmitter detects five consecutive bits of identical value in the bit stream to be transmitted it automatically inserts a complementary bit in the actual transmitted bit stream.

The remaining bit fields of the DATA FRAME or REMOTE FRAME (CRC DELIMITER, ACK FIELD and END OF FRAME) are of fixed form and not stuffed. The ERROR FRAME and the OVERLOAD FRAME are of fixed form as well and not coded by the method of bit stuffing.

The bit stream in a message is coded according to the Non-Return-to-Zero (NRZ) method. This means, that during the total bit time the generated bit level is either ’dominant’ or ’recessive’.

6.2.8 Error Handling

6.2.8.1 Error Detection

There are 5 different error types (which are not mutually exclusive):

- BIT ERROR: A unit that is sending a bit on the bus also monitors the bus. A BIT ERROR has to be detected at that bit time, when the bit value that is monitored is different from the bit value that is sent. An exception is the sending of a ’recessive’ bit during the stuffed bit stream of the ARBITRATION FIELD or during the ACK SLOT. Then no BIT ERROR occurs when a ’dominant’ bit is monitored. A TRANSMITTER sending a PASSIVE ERROR FLAG and detecting a ’dominant’ bit does not interpret this as a BIT ERROR.
- STUFF ERROR: A STUFF ERROR has to be detected at the bit time of the 6th consecutive equal bit level in a message field that should be coded by the method of bit stuffing.
- CRC ERROR: The CRC sequence consists of the result of the CRC calculation by the transmitter. The receivers calculate the CRC in the same way as the transmitter. A CRC ERROR has to be detected, if the calculated result is not the same as that received in the CRC sequence.
- FORM ERROR: A FORM ERROR has to be detected when a fixed-form bit field contains one or more illegal bits. (Note, that for a Receiver a dominant bit during the last bit of END OF FRAME is not treated as FORM ERROR).
- ACKNOWLEDGEMENT ERROR: An ACKNOWLEDGEMENT ERROR has to be detected by a transmitter whenever it does not monitor a ’dominant’ bit during ACK SLOT.

6.2.8.2 Error Signaling

A node detecting ‘an error condition’ signals this by transmitting an ERROR FLAG. For an ’error active’ node it is an ACTIVE ERROR FLAG, for an ’error passive’ node it is a PASSIVE ERROR FLAG. Whenever a BIT ERROR, a STUFF ERROR, a FORM ERROR or an ACKNOWLEDGEMENT ERROR is detected by any node, transmission of an ERROR FLAG is started at the respective node at the next bit.

Whenever a CRC ERROR is detected, transmission of an ERROR FLAG starts at the bit following the ACK DELIMITER, unless an ERROR FLAG for another error condition has already been started.

6.2.9 Fault Confinement

With respect to fault confinement a unit may be in one of three states:

- ’Error active’
- ’Error passive’
- ’Bus off’

An ’error active’ unit can normally take part in bus communication and sends an ACTIVE ERROR FLAG when an error has been detected.

An ’error passive’ unit must not send an ACTIVE ERROR FLAG. It takes part in bus communication, but when an error has been detected only a PASSIVE ERROR FLAG is sent. Also after a transmission, an ’error passive’ unit will wait before initiating a further transmission (See SUSPEND TRANSMISSION).

A ’bus off’ unit is not allowed to have any influence on the bus (E.g. output drivers switched off).

For fault confinement two counts are implemented in every bus unit:

- TRANSMIT ERROR COUNT
- RECEIVE ERROR COUNT

These counts are modified according to the following rules (note that more than one rule may apply during a given message transfer):

- When a RECEIVER detects an error, the RECEIVE ERROR COUNT will be increased by 1, except when the detected error was a BIT ERROR during the sending of an ACTIVE ERROR FLAG or an OVERLOAD FLAG.
- When a RECEIVER detects a ’dominant’ bit as the first bit after sending an ERROR FLAG the RECEIVE ERROR COUNT will be increased by 8.
- When a TRANSMITTER sends an ERROR FLAG the TRANSMIT ERROR COUNT is increased by 8 except:
  - Exception 1: If the TRANSMITTER is ’error passive’ and detects an ACKNOWLEDGEMENT ERROR because of not detecting a ’dominant’ ACK and does not detect a ’dominant’ bit while sending its PASSIVE ERROR FLAG.
  - Exception 2: If the TRANSMITTER sends an ERROR FLAG because a STUFF ERROR occurred during ARBITRATION, and should have been ’recessive’, and has been sent as ’recessive’ but monitored as ’dominant’.

In exceptions 1 and 2 the TRANSMIT ERROR COUNT is not changed.

- If an TRANSMITTER detects a BIT ERROR while sending an ACTIVE ERROR FLAG or an OVERLOAD FLAG the TRANSMIT ERROR COUNT is increased by 8.
- If an RECEIVER detects a BIT ERROR while sending an ACTIVE ERROR FLAG or an OVERLOAD FLAG the RECEIVE ERROR COUNT is increased by 8.
- Any node tolerates up to 7 consecutive 'dominant' bits after sending an ACTIVE ERROR FLAG, PASSIVE ERROR FLAG or OVERLOAD FLAG. After detecting the 14th consecutive 'dominant' bit (in case of an ACTIVE ERROR FLAG or an OVERLOAD FLAG) or after detecting the 8th consecutive 'dominant' bit following a PASSIVE ERROR FLAG, and after each sequence of additional eight consecutive 'dominant' bits every TRANSMITTER increases its TRANSMIT ERROR COUNT by 8 and every RECEIVER increases its RECEIVE ERROR COUNT by 8.
- After the successful transmission of a message (getting ACK and no error until END OF FRAME is finished) the TRANSMIT ERROR COUNT is decreased by 1 unless it was already 0.
- After the successful reception of a message (reception without error up to the ACK SLOT and the successful sending of the ACK bit), the RECEIVE ERROR COUNT is decreased by 1, if it was between 1 and 127. If the RECEIVE ERROR COUNT was 0, it stays 0, and if it was greater than 127, then it will be set to a value between 119 and 127.
- A node is 'error passive' when the TRANSMIT ERROR COUNT equals or exceeds 128, or when the RECEIVE ERROR COUNT equals or exceeds 128. An error condition letting a node become 'error passive' causes the node to send an ACTIVE ERROR FLAG.
- A node is 'bus off' when the TRANSMIT ERROR COUNT is greater than or equal to 256.
- An 'error passive' node becomes 'error active' again when both the TRANSMIT ERROR COUNT and the RECEIVE ERROR COUNT are less than or equal to 127.
- An node which is 'bus off' is permitted to become 'error active' (no longer 'bus off') with its error counters both set to 0 after 128 occurrences of 11 consecutive 'recessive' bits have been monitored on the bus.

NOTE:

1. An error count value greater than about 96 indicates a heavily disturbed bus. It may be of advantage to provide means to test for this condition.
2. Start-up / Wake-up: If during system start-up only 1 node is online, and if this node transmits some message, it will get no acknowledgement, detect an error and repeat the message. It can become 'error passive' but not 'bus off' due to this reason.

6.2.10 Oscillator Tolerance

A maximum oscillator tolerance of 1.58% is given and therefore a ceramic resonator can be used at a bus speed of up to 125 Kbits/s as a rule of thumb. For a more precise evaluation refer to: Dais, S; Chapman, M:

"Impact of Bit Representation on Transport Capacity and Clock Accuracy in Serial Data Streams",

SAE Technical Paper Series 890532, Multiplexing in Automobile SP-773, March 1989.
For the full bus speed range of the CAN protocol, a quartz oscillator is required.
The chip of the CAN network with the highest requirement for its oscillator accuracy determines the oscillator accuracy which is required from all the other nodes.

NOTE: CAN controllers following this CAN Specification and controllers following the previous versions 1.0 and 1.1, used in one and the same network, must all be equipped with a quartz oscillator. That means ceramic resonators can only be used in a network with all the nodes of the network following CAN Protocol Specification versions 1.2 or later.

6.2.11 Bit Timing Requirements

6.2.11.1 Nominal Bit Rate

The Nominal Bit Rate is the number of bits per second transmitted in the absence of resynchronization by an ideal transmitter.

6.2.11.2 Nominal Bit Time

The nominal bit time of the network is uniform throughout the network and is given by:

- NOMINAL BIT TIME = 1 / NOMINAL BIT RATE

The Nominal Bit Time can be thought of as being divided into separate non-overlapping time segments. These segments are:

- Synchronization Segment (SYNC_SEG)
- Propagation Time Segment (PROP_SEG)
- Phase Buffer Segment1 (PHASE_SEG1)
- Phase Buffer Segment2 (PHASE_SEG2)

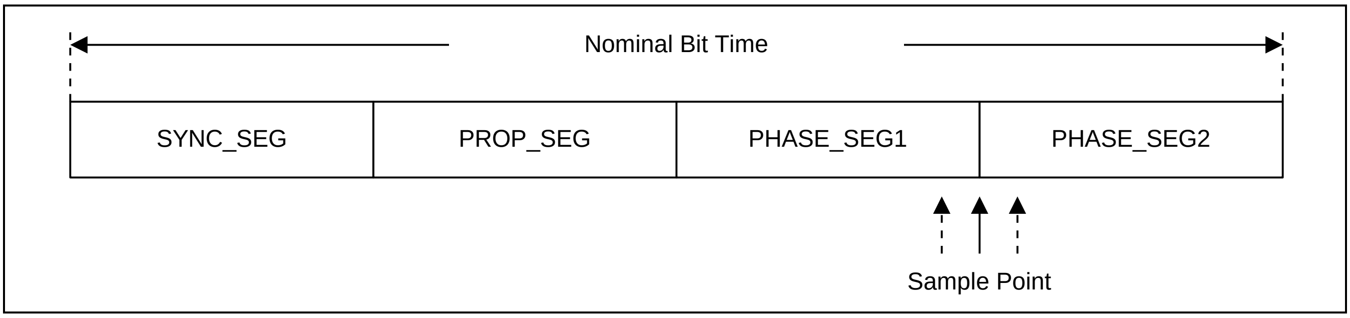


Figure 6-14 Partition of Bit Time

The duration of each segment is set in the mode register and is expressed as a multiple of time quantum (TCAN).

6.2.11.3 Time Quantum

The TIME QUANTUM (T_Q) is a fixed unit of time derived from the oscillator period. It is programmable by the way of a pre-scalar, called the BAUD RATE PRESCALAR (BD[9:0] in CAN_MR register). The formula between TQ, CORECLK and BD[9:0] is the following:

- `TQ = (BD[9:0] +1) / CORECLK`

Each segment (SYNC_SEG, PROP_SEG, PHASE_SEG1 and PHASE_SEG2) is an integer multiple of TIME QUANTUM (T_Q).

6.2.11.4 Synchronization Segment

This part of the bit time is used to synchronize the various nodes on the bus. An edge is expected to lie within this segment. The duration of the synchronization segment, SYNC_SEG, is not programmable and is fixed at one time quantum.

6.2.11.5 Propagation Segment

This part of the bit time is used to compensate for the physical delay times within the network. It is twice the sum of the signal’s propagation time on the bus line, the input comparator delay, and the output driver delay.

The duration of the propagation segment PROP_SEG may be between 1 and 8 time quanta. It is programmable by the way of the PROP bits in the MR, and it is equal to:

6.2.11.6 Phase Segment 1 and Phase Segment 2

These PHASE BUFFER SEGMENTs are used to compensate for edge phase errors. These segments can be lengthened or shortened by resynchronization.

6.2.11.7 Sample Point

The SAMPLE POINT is the point of time at which the bus level is read and interpreted as the value of that respective bit. It’s location is at the end of PHASE_SEG1.

6.2.11.8 Information Processing Time (IPT)

The INFORMATION PROCESSING TIME is the time segment starting with the SAMPLE POINT reserved for calculation the subsequent bit level.

6.2.11.9 Length of Time Segments

- SYNC_SEG is 1 TIME QUANTUM long.
- PROP_SEG is programmable to be 1, 2, ..., 8 TIME QUANTA long.
- PHASE_SEG1 is programmable to be 1, 2, ..., 8 TIME QUANTA long.
- PHASE_SEG2 is the maximum of PHASE_SEG1 and the INFORMATION PROCESSING TIME.
- The INFORMATION PROCESSING TIME is less than or equal to 2 TIME QUANTA long.

The total number of TIME QUANTA in a bit time has to be programmable at least from 8 to 25.

NOTE: It is often intended that control units do not make use of different oscillators for the local CPU and its communication device. Therefore the oscillator frequency of a CAN device tends to be that of the local CPU and is determined by the requirements of the control unit. In order to derive the desired bit rate, programmability of the bit timing is necessary. In case of CAN implementations that are designed for use without a local CPU the bit timing cannot be programmable. On the other hand these devices allow to choose an external oscillator in such a way that the device is adjusted to the appropriate bit rate so that the programmability is dispensable for such components. The position of the sample point, however, should be selected in common for all nodes. Therefore the bit timing of CAN devices without local CPU should be compatible to the following definition of the bit time:

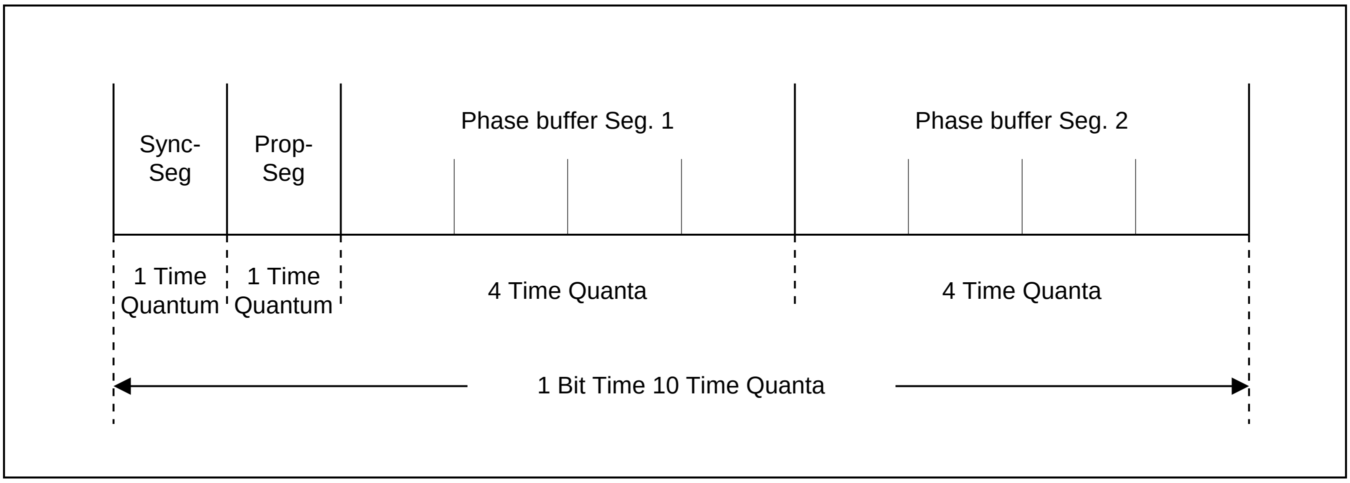


Figure 6-15 Example of Nominal Bit Time

6.2.12 Hard Synchronization

After a HARD SYNCHRONIZATION the internal bit time is restarted with SYNC_SEG. Thus HARD SYNCHRONIZATION forces the edge which has caused the HARD SYNCHRONIZATION to lie within the SYNCHRONIZATION SEGMENT of the restarted bit time.

6.2.12.1 Resynchronization Jump Width

As a result of RESYNCHRONIZATION PHASE_SEG1 may be lengthened or PHASE_SEG2 may be shortened. The amount of lengthening or shortening of the PHASE BUFFER SEGMENTs has an upper bound given by the RESYNCHRONIZATION JUMP WIDTH. The RESYNCHRONIZATION JUMP WIDTH shall be programmable between 1 and min (4, PHASE_SEG1). Clocking information may be derived from transitions from one bit value to the other. The property that only a fixed maximum number of successive bits have the same value provides the possibility of resynchronizing a bus unit to the bit stream during a frame. The maximum length between two transitions which can be used for resynchronization is 29 bit times.

6.2.12.2 Phase Error of an Edge

The PHASE ERROR of an edge is given by the position of the edge relative to SYNC_SEG,measured in TIME QUANTA The sign of PHASE ERROR is defined as follows:

- E = 0 if the edge lies within SYNC_SEG.
- E > 0 if the edge lies before the SAMPLE POINT.
- E < 0 if the edge lies after the SAMPLE POINT of the previous bit.

6.2.12.3 Resynchronization

The effect of a RESYNCHRONIZATION is the same as that of a HARD SYNCHRONIZATION, when the magnitude of the PHASE ERROR of the edge which causes the RESYNCHRONIZATION is less than or equal to the programmed value of the RESYNCHRONIZATION JUMP WIDTH. When the magnitude of the PHASE ERROR is larger than the RESYNCHRONIZATION JUMP WIDTH,

- And if the PHASE ERROR is positive, then PHASE_SEG1 is lengthened by an amount equal to the RESYNCHRONIZATION JUMP WIDTH.
- And if the PHASE ERROR is negative, then PHASE_SEG2 is shortened by an amount equal to the RESYNCHRONIZATION JUMP WIDTH.

6.2.12.4 Synchronization Rules

HARD SYNCHRONIZATION and RESYNCHRONIZATION are the two forms of SYNCHRONIZATION. They obey the following rules:

- Only one SYNCHRONIZATION within one bit time is allowed.
- An edge will be used for SYNCHRONIZATION only if the value detected at the previous SAMPLE POINT (previous read bus value) differs from the bus value immediately after the edge.
- HARD SYNCHRONIZATION is performed whenever there is a ’recessive’ to ’dominant’ edge during BUS IDLE.
- All other ’recessive’ to ’dominant’ edges fulfilling the rules 1 and 2 will be used for RESYNCHRONIZATION with the exception that a node transmitting a dominant bit will not perform a RESYNCHRONIZATION as a result of a ’recessive’ to ’dominant’ edge with a positive PHASE ERROR, if only ’recessive’ to ’dominant’ edges are used for resynchronization.

6.3 Operating Modes

6.3.1 Software Initialization

The software initialization is started by disabling CAN controller, either by software (setting the bit CANDIS in the CAN Control Register) or by a hardware reset, or by going bus off.

While CAN is disabled, all message transfer from and to the CAN bus is stopped, the status of the CAN bus output CAN_TX is recessive (HIGH). The counter of the ERCR (see CAN_ERCR (CAN Error Counter Register) is unchanged. Disabling CAN does not change any configuration register.

To initialize the CAN Controller, the CPU has to set up the Mode Register and each Message Object. If a Message Object is not needed, it is sufficient to set MSGVAL of CAN_IRx to not valid. Otherwise, the whole Message Object has to be initialized because the RAM is not initialized at power up.

Access to the Mode Register for the configuration of the bit timing is allowed when CAN is disabled (CANENS of CAN_SR is reset to zero) and bit CCENS in the CAN_SR is set (See CAN_SR (CAN Status Register)).

Enabling the CAN (by CPU only) finishes the software initialization (CANEN of CAN_CR is set to ‘1’.). Afterwards the Bit Stream Processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus Idle) before it can take part in bus activities and starts the message transfer.

The initialization of the Message Objects is independent of bit CANENS of CAN_SR and can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid(MSGVAL of CAN_IRx is set to ‘0’) before the BSP starts the message transfer. To change the configuration of a Message Object during normal operation, the CPU has to start by setting MSGVAL of CAN_IRx to 0. When the configuration is completed, MSGVAL is set to valid again.

6.3.2 CAN Message Transfer

Once the CAN is initialized and CAN is enabled, the CAN Core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored into their appropriate Message Objects if they pass the Message Handler's acceptance filtering. The whole message including all arbitration bits, DLC and eight data bytes is stored into the Message Object.

If the Identifier Mask is used, the arbitration bits which are masked to "don't care" may be overwritten in the Message Object.

The CPU may read or write each message any time via the Interface Registers, the Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the CPU. If a permanent Message Object (arbitration and control bits set up during configuration) exists for the message, only the data bytes are updated and then TXRQST bit with NEWDAT bit are set to start the transmission (See CAN_MCR (CAN Interface X Message Control Register)). If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time, they are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started.

Depending on the configuration of the Message Object, the transmission of a message may be requested automatically by the reception of a remote frame with a matching identifier.

Caution: The message object number 32 shall not be used in transmission.

6.3.3 Message Interface Register Sets

There are two sets of Interface Registers which are used to control the CPU access to the Message RAM. The Interface Registers avoid conflicts between CPU access to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object (See *6.3.4 Message Object*) or part of the Message Object may be transferred between the Message RAM and the IFx Message Buffer registers in one single transfer.

The function of the two interface register sets is identical (except for BASIC mode). They can be used the way that one set of registers is used for the data to transfer to the Message RAM while the other set of registers is used for the data transfer from the Message RAM, allowing both processes to be interrupted by each other. *Table 6-6* gives an overview of the two Interface Register sets.

Each set of Interface Registers consists of Message Buffer Registers controlled by their own Transfer Management Registers. The Transfer Management Register specifies the direction of the data transfer, which part of a Message Object will be transferred and is used to select a Message Object to the Message RAM as a target or source for the transfer and to start the action specified.

Table 6-6 Interface Register Sets

Offset	Interface Registers	Name	Access
Interface 0 Registers			
0x0100	Interface 0 Transfer Management Register	CAN_TMR0	Read/Write
0x0104	Interface 0 Data A Register	CAN_DAR0	Read/Write
0x0108	Interface 0 Data B Register	CAN_DBR0	Read/Write
0x010C	Interface 0 Mask Register	CAN_MSKR0	Read/Write
0x0110	Interface 0 Identifier Register	CAN_IR0	Read/Write
0x0114	Interface 0 Message Control Register	CAN_MCR0	Read/Write
0x0118	Interface 0 Stamp Register	CAN_STPR0	Read only
Interface 1 Registers			
0x0120	Interface 1 Transfer Management Register	CAN_TMR1	Read/Write
0x0124	Interface 1 Data A Register	CAN_DAR1	Read/Write
0x0128	Interface 1 Data B Register	CAN_DBR1	Read/Write
0x012C	Interface 1 Mask Register	CAN_MSKR1	Read/Write
0x0130	Interface 1 Identifier Register	CAN_IR1	Read/Write
0x0134	Interface 1 Message Control Register	CAN_MCR1	Read/Write
0x0138	Interface 1 Stamp Register	CAN_STPR1	Read only

6.3.4 Message Object

There are 32 Message Objects in Message RAM. They cannot be accessed directly by the CPU, these accesses are handled via the IFx Interface Registers.

The table below gives an overview of the Message Object content.

Table 6-7 Message Object Content

Name	Description
DAR	First data bytes message
DBR	Last data bytes message
MSKR	Identifier mask
IR	Message identifier
MCR	Message control
STPR	Stamp date of message

6.3.5 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the CAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. By default, this means for automatic retransmission is enabled. It can be disabled to enable the CAN to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment.

The Automatic Retransmission mode is disabled by programming bit AR in the CAN Mode Register (CAN_MR) to zero. In this operation mode the programmer has to consider the different behavior of bits TXRQST and NEWDAT in the Control Registers of the Message Buffers:

- When a transmission starts bit TXRQST of the respective Message Buffer is reset, while bit NEWDAT remains set.
- When the transmission completed successfully bit NEWDAT is reset.
- When a transmission failed (lost arbitration or error) bit NEWDAT remains set. To restart the transmission the CPU has to set TXRQST back to one.

6.3.6 Test Mode

The CAN module has three test mode that can be combined:

- The Silent mode
- The Loop back mode
- The Basic mode

Moreover, the pin CAN_TX can be controlled by software. It can also be configured in open drain allowing to connect directly CAN controllers together without external transceiver.

The pin CAN_TX control and test modes are enabled through the Test Register. Write access in the Test Register is protected by a control access key. To enable or disable a test function, it is necessary to write the correct bit pattern (TESTKEY = 0x0C75) to the control access key bits at the same time as the control bits are written.

The bit RX monitors the state of pin CAN_RX and therefore is only readable.

6.3.7 Silent Mode

The CAN Core can be set in Silent Mode by programming the Test Register bit SILENT to one.

In Silent Mode, the CAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent Mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits (Acknowledge Bits, Error Frames). The figure below shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Silent Mode.

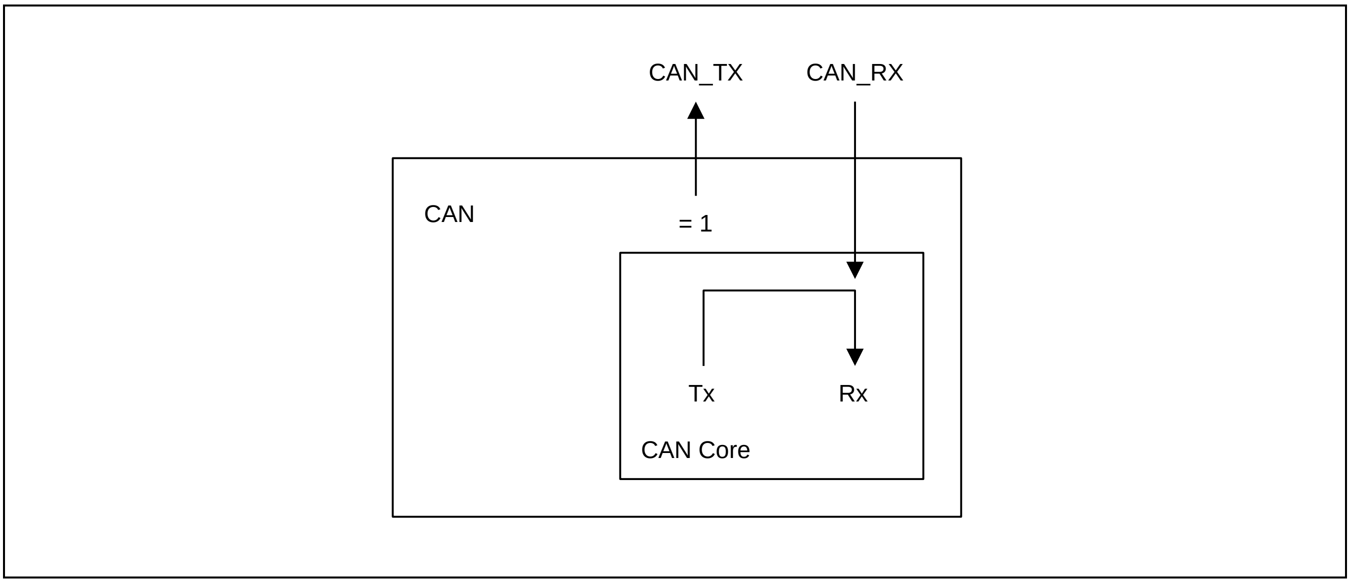


Figure 6-16 CAN Core in Silent Mode

In ISO 11898-1, the Silent Mode is called the Bus Monitoring Mode.

6.3.8 Loop Back Mode

The CAN Core can be set in Loop Back Mode by programming the Test Register bit LBACK to one. In Loop Back Mode, the CAN Core treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into a Receive Buffer. The figure below shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Loop Back Mode.

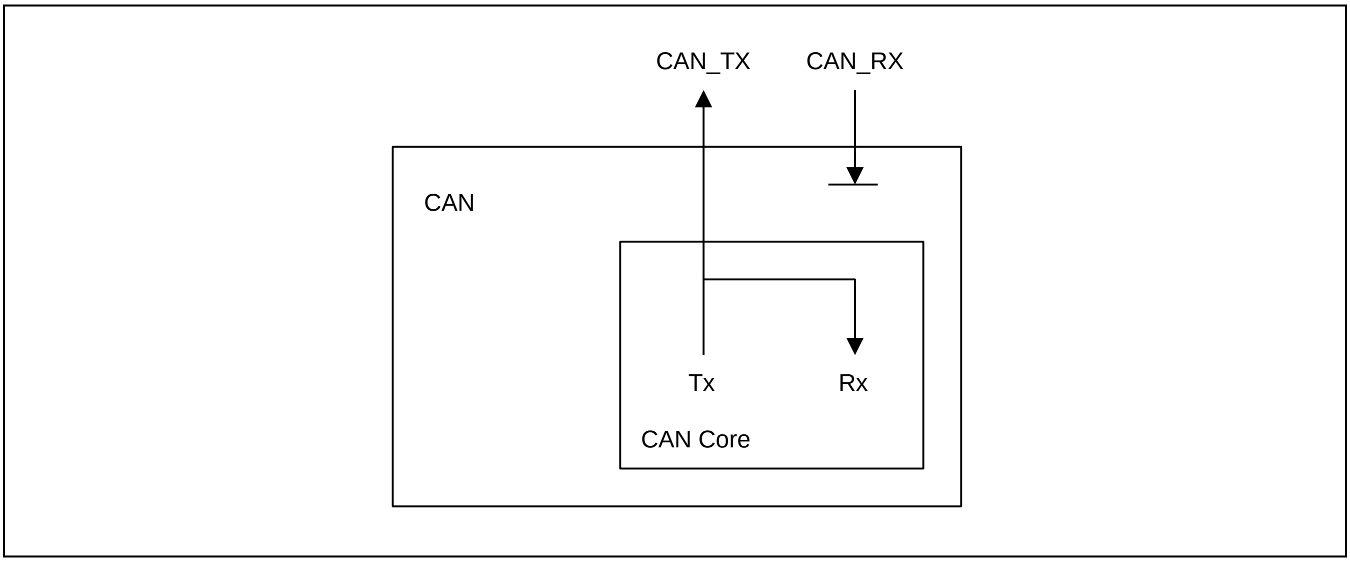


Figure 6-17 CAN Core in Loop Back Mode

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode. In this mode the CAN Core performs an internal feedback from its TX output to its RX input. The actual value of the CAN_RX input pin is disregarded by the CAN Core. The transmitted messages can be monitored at the CAN_TX pin.

6.3.9 Loop Back Mode Combined With Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by programming bits LBACK and SILENT to one at the same time. This mode can be used for a “Hot Self test”, meaning the CAN can be tested without affecting a running CAN system connected to the pins CAN_TX and CAN_RX. In this mode the CAN_RX pin is disconnected from the CAN Core and the CAN_TX pin is held recessive. Figure 8-18 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

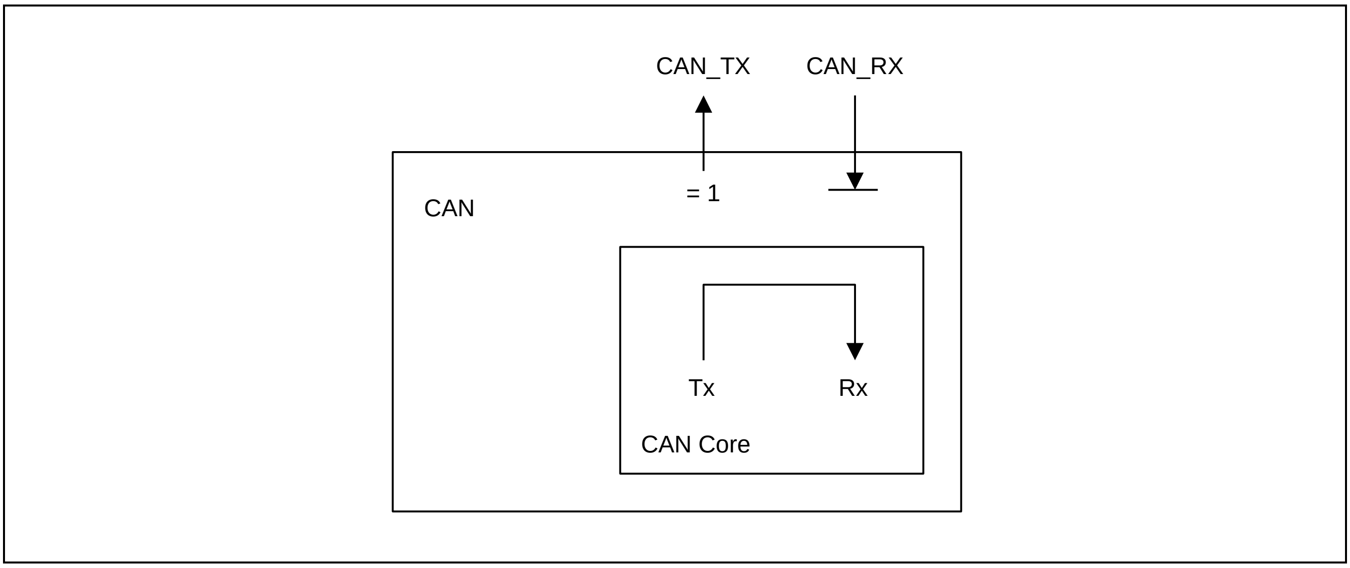


Figure 6-18 CAN Core in Loop Back Combined with Silent Mode

6.3.10 Basic Mode

The CAN Core can be set in Basic Mode by programming the Test Register bit BASIC to one. In this mode the CAN module runs without the Message RAM.

The IF0 Registers are used as Transmit Buffer. The transmission of the contents of the IF0 Registers is requested by writing the RQBTX bit of the CAN_CR register to '1'. The IF0 Registers are locked while the BTXPD bit is set in the CAN_SR register. The BTXPD bit indicates that the transmission is pending.

As soon the CAN bus is idle, the IF0 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has completed, the BTXPD bit is reset and the locked IF0 Registers are released.

A pending transmission can be aborted at any time by writing the ABBTX bit of the CAN_CR register. If the CPU aborted a pending transmission, a possible retransmission in case of lost arbitration or in case of an error is disabled.

The IF1 Registers are used as Receive Buffer. After the reception of a message the contents of the shift register is stored into the IF1 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing STSR bit of the CAN_CR register, the contents of the shift register is stored into the IF1 Registers.

In Basic Mode the evaluation of all Message Object related control and status bits and of the control bits of the Ifx Transfer Management Registers is turned off. The message number of the Transfer Management registers is not evaluated. The NEWDAT and MSGLST bits of the IF1 Message Control Register retain their function, DLC will show the received DLC, the other control bits will be read as '0'.

6.3.11 Control of pin CAN_TX

Four output functions are available for the CAN transmit pin CAN_TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor CAN Core's bit timing and it can drive constant dominant or recessive values. The last two functions, combined with the readable CAN receive pin CAN_RX, can be used to check the physical layer of CAN bus.

The output mode of pin CAN_TX is selected by programming the Test Register bits TX1 and TX0 as described in Test Register definition. (See *CAN_TSTR (CAN Test Register)*)

The three test functions for pin CAN_TX interfere with all CAN protocol functions. CAN_TX must be left in its default function when CAN message transfer or any of the test modes Loop Back Mode, Silent Mode, or Basic Mode are selected.

CAN_TX pin can be programmed as open drain (it can only drive a low level). This configuration is done by setting TXOPD to '1' in CAN_TSTR register. User should disable CAN (CANDIS in CAN_CR register) before switching between open drain and normal mode in order to avoid glitch on CAN_TX output. An external pull-up is necessary to guarantee a logic level (logical one) when the pin is not being driven. This feature can be useful in two cases:

- It allows to connect directly our CAN controller to another CAN controller providing the same open drain feature without external transceiver.

It allows to drive a 5V transceiver whereas the CAN controller is 3.3V, using an external 5V pull up resistor.

6.3.12 Bus off Recovery Sequence

When the CAN controller goes in bus off mode, it becomes disable (CANENS is reset) and does not take part any longer in bus activities. To resume in normal operation mode, CPU has to enable the CAN controller (set CANEN bit in *CAN_CR (CAN Control Register)*), at this point the CAN controller will wait for 129 successive occurrences of bus idle (a sequence of 11 consecutive recessive bits).

At the end of the bus off recovery sequence, the read and write error counter are reset and the BUSOFF bit is reset. The bus off recovery sequence cannot be shortened by resetting CAN (SWRST set to '1' in CAN_CR).

During the waiting time after CPU has enabled CAN controller, each time a sequence of 11 recessive bits has been monitored, BIT0 bit is set in the status register, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus off recovery sequence.

The bus off recovery sequence can be monitored by reading the REC counter: it shall be reset when CPU enables the CAN, and each time a sequence of 11 consecutive recessive bits occurred on the bus, the REC counter shall be increased by one. During the recovery sequence, BUSOFF, ERPASS and CANENS bits are set while ERWARN bit is reset. (See *CAN_SR (CAN Status Register)*)

6.4 CAN Application

6.4.1 Management of Message Objects

The configuration of the Message Objects in the Message RAM will (with the exception of the bits MSGVAL, NEWDAT, ITPND, and TXRQST) not be affected by resetting the chip. All the Message Objects must be initialized by the CPU or they must not be valid (MSGVAL of CAN_IRx = '0') and the bit timing must be configured before the CPU set the CANEN bit in the CAN Control Register (CAN_CR).

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data field of one of the two interface register sets to the desired values. By writing to the corresponding IFx Transfer Management Register (CAN_TMRx), the IFx Message Buffer Registers are loaded into the addressed Message Object in the Message RAM. A message transfer between the message RAM and the IFx message buffer register is started as soon as the CPU has written in the register CAN_TMRx.

When the CANEN bit in the CAN Control Register is set, the CAN Protocol Controller state machine of the CAN Core and the Message Handler State Machine control the CAN's internal data flow. Received messages that pass the acceptance filtering are stored into the Message RAM, messages with pending transmission request are loaded into the CAN Core's Shift Register and are transmitted via the CAN bus.

The CPU reads received messages and updates messages to be transmitted via the IFx Interface Registers. Depending on the configuration, the CPU is interrupted on certain CAN message and CAN error events.

6.4.2 Message Handler State Machine

The Message Handler controls the data transfer between the RX/TX Shift Register of the CAN Core, the Message RAM and the IFx Registers.

The Message Handler FSM controls the following functions:

- Data Transfer from IFx Registers to the Message RAM.
- Data Transfer from Message RAM to the IFx Registers.
- Data Transfer from Shift Register to the Message RAM.
- Data Transfer from Message RAM to Shift Register.
- Data Transfer from Shift Register to the Acceptance Filtering unit.
- Scanning of Message RAM for a matching Message Object.
- Handling of TXRQST flags. (See *CAN_MCR (CAN Interface X Message Control Register 0 -1)*)
- Handling of interrupts.

6.4.2.1 Data Transfer Between Interface Register and Message RAM

When the CPU initiates a data transfer between the IFx Registers and Message RAM, the Message Handler sets the BUSYx bit in the Status Register to ‘1’. After the transfer has completed, the BUSYx bit is set back to ‘0’ (*Figure 6-19*).

The respective Transfer Management Register (CAN_TMRx) specifies whether a complete Message Object or only parts of it will be transferred. But, due to the structure of the Message RAM it is not possible to write single bits/bytes of one Message Object, it is always necessary to write a complete Message Object into the Message RAM. Therefore the data transfer from the IFx Registers to the Message RAM requires of a read-modify-write cycle. Parts of the Message Object that are not to be changed are read from the Message RAM and then the complete contents of the Message Buffer Registers are into the Message Object.

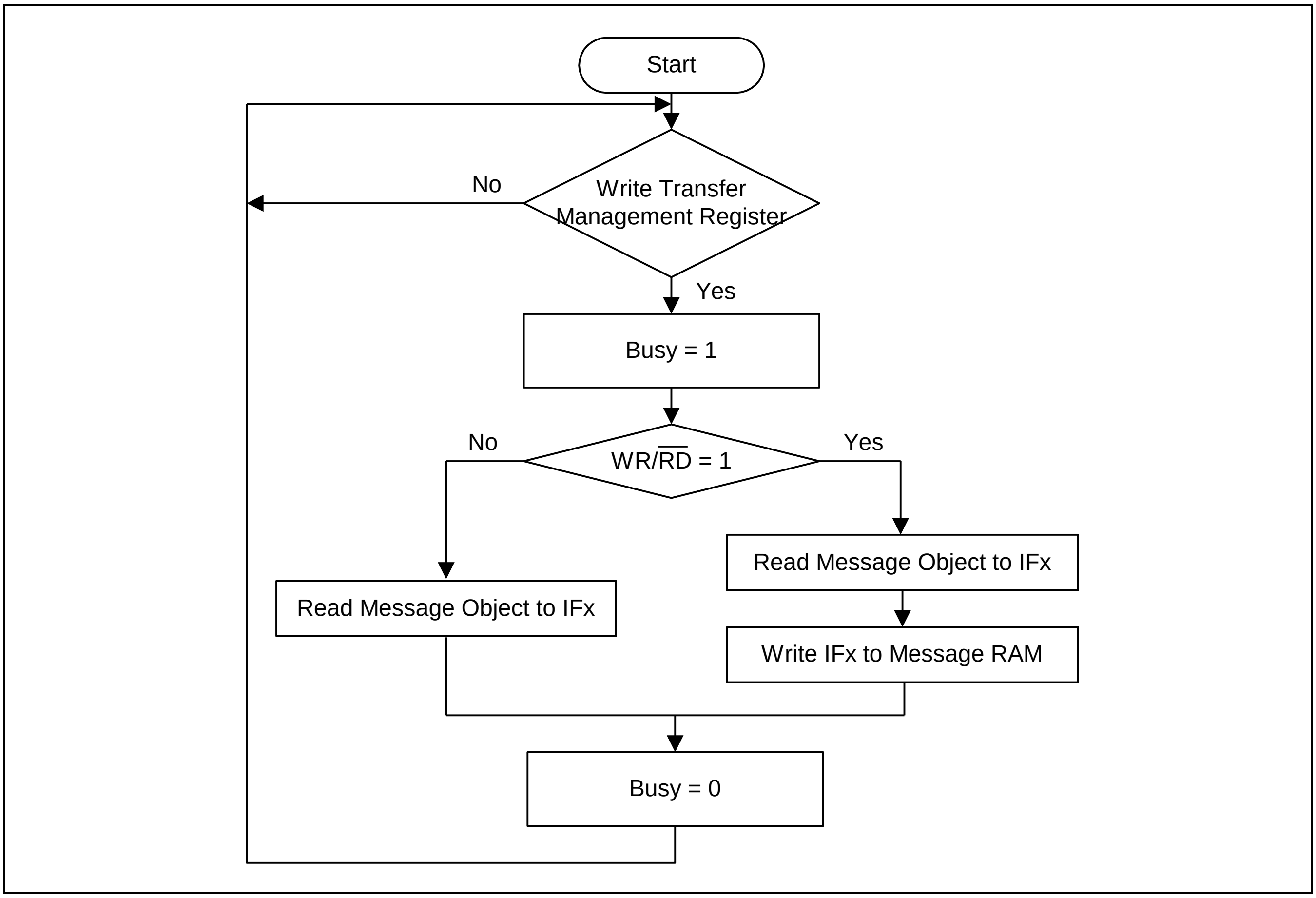

Start
Write Transfer Management Register
No
Yes
Busy = 1
WR/$\overline{RD}$ = 1
No
Yes
Read Message Object to IFx
Read Message Object to IFx
Write IFx to Message RAM
Busy = 0

Figure 6-19 Data Transfer Between IFx Registers and Message RAM

After the partial write of a Message Object, that Message Buffer Registers that are not selected in the Transfer Management Register will set to the actual contents of the selected Message Object.

After the partial read of a Message Object, that Message Buffer Registers that are not selected in the Transfer Management Register will be left unchanged.

6.4.2.2 Transmission of Messages from Message Object to Shift Register

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFx Registers and Message RAM, the MSGVAL bits in the CAN_IRx and TXRQST bits in the CAN_MCRx are took in consideration. The valid Message Object with the highest priority pending transmission request (see *6.4.2.4*) is loaded into the shift register by the Message Handler and the transmission is started. The Message Object's NEWDAT bit is reset (See CAN_MCR (CAN Interface X Message Control Register 0–1)).

After a successful transmission and if no new data was written to the Message Object (NEWDAT = '0') since the start of the transmission, the TXRQST bit will be reset (See CAN_MCR (CAN Interface X Message Control Register 0–1)).). If TXIE is set, ITPND will be set after a successful transmission. If the CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

6.4.2.3 Acceptance Filtering of Received Messages in the Shift Register

When the arbitration and control field (ID + IDE + RTR + DLC) of an incoming message is completely shifted into the RX/TX Shift Register of the CAN Core, the Message Handler FSM starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. Then the arbitration and mask fields (including MSGVAL, UMASK, NEWDAT, and OVERWRITE) of Message Object 1 are loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scanning is stopped and the Message Handler FSM proceeds depending on the type of frame (Data Frame or Remote Frame) received.

6.4.2.3.1 Reception of a Data Frame

The Message Handler FSM stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. Not only the data bytes, but also all arbitration bits and the Data Length Code are stored into the corresponding Message Object. This is implemented to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The NEWDAT bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU should reset NEWDAT when it reads the Message Object. If at the time of the reception the NEWDAT bit was already set, MSGLST is set to indicate that the previous data (supposedly not seen by the CPU) is lost. If the RXIE bit is set, the ITPND bit is set, causing the Interrupt Register to point to this Message Object.

The TXRQST bit of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

6.4.2.3.2 Reception of a Remote Frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

1. MDIR of CAN_IRx = ‘1’ (message direction = transmit), RMTEN of CAN_MCRx = ‘1’, UMASK of CAN_MCRx = ‘1’ or ’0’. At the reception of a matching Remote Frame, the TXRQST bit of this Message Object is set. The rest of the Message Object remains unchanged.
2. MDIR of CAN_IRx = ‘1’ (message direction = transmit), RMTEN of CAN_MCRx = ‘0’, UMASK of CAN_MCRx = ‘0’. At the reception of a matching Remote Frame, the TXRQST bit of this Message Object remains unchanged; the Remote Frame is ignored.
3. MDIR of CAN_IRx = ‘1’ (message direction = transmit), RMTEN of CAN_MCRx = ‘0’, UMASK of CAN_MCRx = ’1’. At the reception of a matching Remote Frame, the TXRQST bit of this Message Object is reset. The arbitration and control field (ID + IDE + RTR + DLC) from the shift register is stored into the Message Object in the Message RAM and the NEWDAT bit of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

6.4.2.4 Receive/Transmit Priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object.

6.4.3 Configuration of a Transmit Object

The table below shows how a Transmit object should be initialized.

Table 6-8 Initialization of a Transmit Object

MSGVAL	ID	DATA	MSK	OVER WRITE	MDIR	NEWDAT	MSGLST	RXIE	TXIE	ITPND	RMTEN	TXRQST
1	Appl	Appl	Appl	1	1	0	0	0	Appl	0	Appl	0

The Identifier Registers (ID and XTD of CAN_IRx) are given by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to ID[28:18]. And, ID[17:0] can then be disregarded.

If the TXIE of CAN_MCRx is set, the ITPND bit will be set after a successful transmission of the Message Object.

If the RMTEN of CAN_MCRx is set, a matching received Remote Frame will cause the TXRQST of CAN_MCRx to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Data Registers (DLC of CAN_MCRx, DATA[0:7]) are given by the application, TXRQST and RMTEN may not be set before the data is valid.

The Mask Registers (MSK, UMASK, MXTD, and MMDIR bits) may be used (UMASK=’1’) to allow groups of Remote Frames with similar identifiers to set the TXRQST bit. Handle with care. The MDIR bit should not be masked.

6.4.4 Updating a Transmit Object

The CPU may update the data bytes of a Transmit object any time via the IFx Interface registers, neither MSGVAL of CAN_IRx nor TXRQST of CAN_MCRx have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFx Data a Register or IFx Data B Register have to be valid before the content of that register is transferred to the Message Object. Either the CPU has to write all four bytes into the IFx Data Register or the Message Object is transferred to the IFx Data Register before the CPU writes the new data bytes.

To prevent the reset of TXRQST at the end of a transmission that may already be in progress while the data is updated, NEWDAT has to be set together with TXRQST. For details see section “Transmission of messages from message object to shift register”.

When NEWDAT is set together with TXRQST, NEWDAT will be reset as soon as the new transmission has started.

6.4.5 Configuration of a Receive Object

The table below shows how a Receive object should be initialized.

Table 6-9 Initialization of a Receive Object

MSGVAL	ID	DATA	MSK	OVER WRITE	MDIR	NEWDAT	MSGLST	RXIE	TXIE	ITPND	RMTEN	TXRQST
1	Appl	Appl	Appl	1	0	0	0	Appl	0	0	0	0

The Identifier Registers (ID and XTD bits of CAN_IRx) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier ("Standard Frame") is used, it is programmed to ID[28:18], ID[17:0] can then be disregarded. When a Data Frame with an 11-bit Identifier is received, ID[17:0] will be set to '0'.

If the RXIE bit is set, the ITPND bit will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (DLC of CAN_MCRx) is given by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by non specified values.

The Mask Registers (UMASK, MXTD, and MMDIR bits) may be used (UMASK='1') to allow groups of Data Frames with similar identifiers to be accepted. For details see "*Reception of a Data Frame*". The MDIR bit should not be masked in typical applications.

6.4.6 Handling of Received Message

The CPU may read a received message any time while the CAN is not transmitting a message — check TS flag in CAN_SR register — via the IFx Interface registers, the data consistency is guaranteed by the Message Handler state machine.

The CPU will write to the Transfer Management Register. This write will transfer the whole received message from the Message RAM into the Message Buffer Register. Additionally, the bits NEWDAT and ITPND are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits show which of the matching messages has been received. The actual value of NEWDAT shows whether a new message has been received since last time this Message Object was read.

The actual value of MSGLST shows whether more than one message has been received since last time this Message Object was read. MSGLST will not be automatically reset.

By means of a Remote Frame, the CPU may request another CAN node to provide new data for a receive object. Setting the TXRQST bit of a receive object will cause the transmission of a Remote Frame with the receive object's identifier.

This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the TXRQST bit is automatically reset.

6.4.7 Configuration of a FIFO Buffer

With the exception of the OVERWRITE of CAN_MCRx , the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see section “Configuration of a Receive object”.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the FIFO Buffer. The OVERWRITE bit of all Message Objects of a FIFO Buffer except the last have to be programmed to zero. The OVERWRITE bits of the last Message Object of a FIFO Buffer is set to one, configuring it as the End of the Block.

6.4.8 Reception of messages with FIFO Buffers

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer the NEWDAT bit of this Message Object is set. By setting NEWDAT while OVERWRITE is zero the Message Object is locked for further write accesses by the Message Handler until the CPU has written the NEWDAT bit back to zero.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing NEWDAT to zero, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite previous messages.

6.4.8.1 Reading from a FIFO Buffer

When the CPU transfers the contents of Message Object to the IFx Message Buffer registers by writing its number to the IFx Transfer Management Register(CAN_TMRx), the bits NEWDAT and ITPND should be reset to zero (TRND = ‘1’ and CLRIT of CAN_TMRx = ‘1’). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the CPU should read out the Message Objects starting at the FIFO Object with the lowest message number.

The figure below shows how a set of Message Objects which are concatenated to a FIFO Buffer can be handled by the CPU.

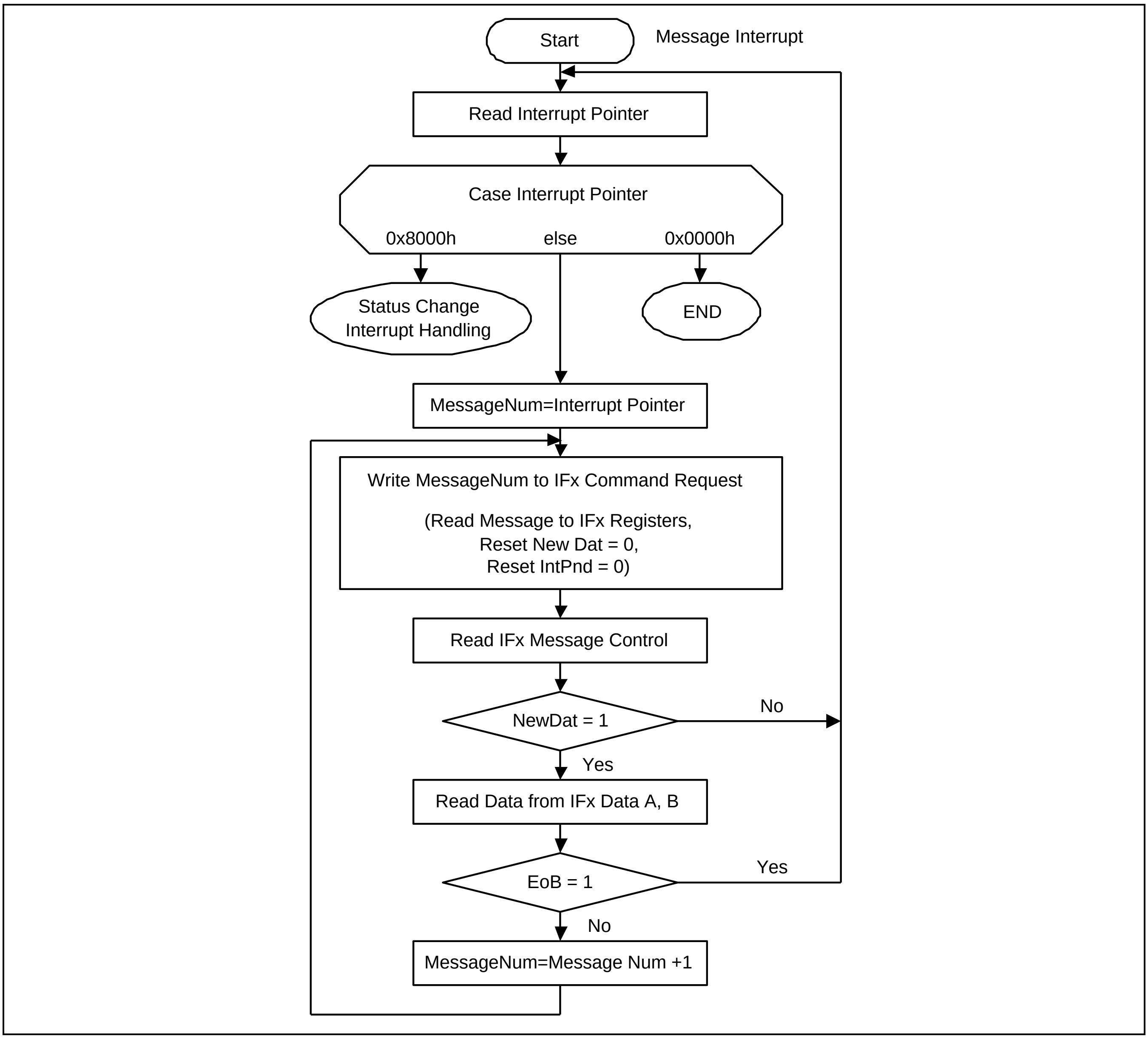

Start
Message Interrupt
Read Interrupt Pointer
Case Interrupt Pointer
0x8000h
else
0x0000h
Status Change Interrupt Handling
END
MessageNum=Interrupt Pointer
Write MessageNum to IFx Command Request
(Read Message to IFx Registers, Reset New Dat = 0, Reset IntPnd = 0)
Read IFx Message Control
NewDat = 1
No
Yes
Read Data from IFx Data A, B
EoB = 1
Yes
No
MessageNum=Message Num +1

Figure 6-20 CPU Handling of a FIFO Buffer

6.4.9 Handling of Interrupts

A message interrupt is cleared by clearing the Message Object's ITPND bit. The Status Interrupt is cleared by writing the Clear Status Register.

The CPU controls whether a change of the Status Register may cause an interrupt with the Interrupt Mask Register and bits TXIE and RXIE of IFx Message Control Register (CAN_TMRx).

6.4.10 Configuration of the bit Timing

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration however, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and of the CAN nodes' interaction on the CAN bus.

6.4.10.1 Bit Time and Bit Rate

CAN supports bit rates in the range of lower than 1 kBit/s up to 1000 kBit/s. Each member of the CAN network has its own clock generator, usually a quartz oscillator. The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes' oscillator periods (fosc) maybe different.

The frequencies of these oscillators are not absolutely stable. Small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by resynchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see *Figure 6-21*).
The Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1, and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (see *Table 6-10*).
The length of the time quantum (tq) , which is the basic time unit of the bit time, is defined by the CAN controller's system clock fsys and the Baud Rate Pre-scalar: tq = (BD+1) / fsys. The CAN's system clock f sys is the frequency of its CAN_CLK input.

The Synchronization Segment Sync_Seg is that part of the bit time where edges of the CAN bus level are expected to occur; the distance between an edge that occurs outside of Sync_Seg and the Sync_Seg is called the phase error of that edge. The Propagation Time Segment Prop_Seg is intended to compensate for the physical delay times within the CAN network. The Phase Buffer Segments Phase_Seg1 and Phase_Seg2 surround the Sample Point. The (Re-) Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

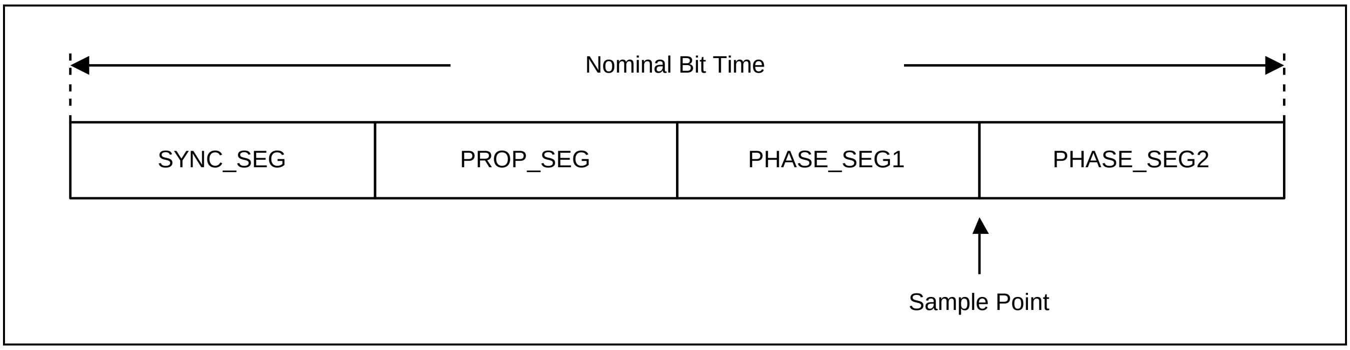


Figure 6-21 Bit Timing

Table 6-10 Parameters of the CAN Bit Time

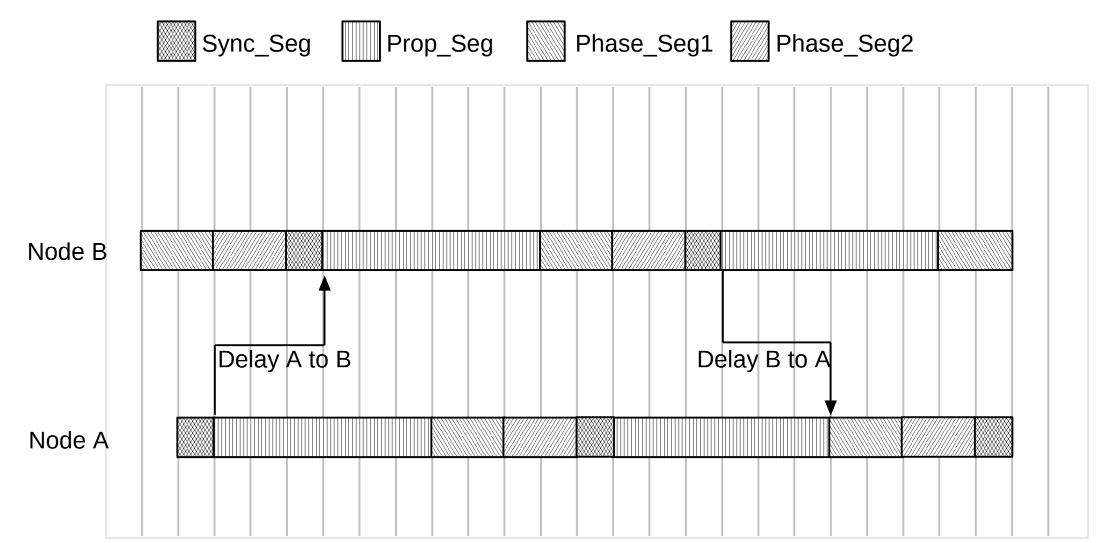
Parameter	Range	Remark
BRP	[1..32]	Defines the length of the time quantum tq
Sync_Seg	1 tq	Fixed length, synchronization of bus input to system clock
Prop_Seg	[1..8] tq	Compensates for the physical delay times
Phase_Seg1	[1..8] tq	May be lengthened temporarily by synchronization
Phase_Seg2	[1..8] tq	May be shortened temporarily by synchronization
SJW	[1..8] tq	May not be longer than either phase buffer segment
This table describes the minimum programmable ranges required by the CAN protocol		

A given bit rate may be met by different bit time configurations, but for the proper function of the CAN network the physical delay times and the oscillator’s tolerance range have to be considered.

6.4.10.2 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus will be out of phase with the transmitter of that bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's non-destructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages require that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in below shows the phase shift and propagation times between two CAN nodes.



Delay A to B ≥ node output delay (A) + bus line delay (A → B) + node input delay (B)
Prop_Seg ≥ Delay A to B + Delay B to A
Prop_Seg ≥ 2 • [max (node output delay + bus line delay + node input delay)]

Figure 6-22 Propagation Time Segment

In this example, both nodes A and B are transmitters performing an arbitration for the CAN bus. The node A has sent its Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay(A to B) after it has been transmitted, B's bit timing segments are shifted with regard to A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B to A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B would arrive at node A after the start of Phase_Seg1, it could happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

The error occurs only when two nodes arbitrate for the CAN bus that have oscillators of opposite ends of the tolerance range and that are separated by a long bus line; this is an example of a minor error in the bit timing configuration (Prop_Seg to short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3 Sample Mode the CAN does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 t q, requiring a longer Prop_Seg.

6.4.10.3 Phase Buffer Segments and Synchronization

The Phase Buffer Segments (Phase_Seg1 and Phase_Seg2) and the Synchronization Jump Width (SJW) are used to compensate for the oscillator tolerance. The Phase Buffer Segments may be lengthened or shortened by synchronization. Synchronizations occur on edges from recessive to dominant, their purpose is to control the distance between edges and Sample Points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous Sample Point. A synchronization may be done only if a recessive bit was sampled at the previous Sample Point and if the actual time quantum’s bus level is dominant.

An edge is synchronous if it occurs inside of Sync_Seg, otherwise the distance between edge and the end of Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist: Hard Synchronization and Resynchronization. A Hard Synchronization is done once at the start of a frame; inside a frame only Resynchronizations occur.

- Hard Synchronization
  After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error.
  Thus hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.
- Resynchronization
  Resynchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

When the phase error of the edge which causes Resynchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

When the phase error of the edge which causes Resynchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

When the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of Hard Synchronization and Resynchronization are the same. If the magnitude of the phase error is larger than SJW, the Resynchronization cannot compensate the phase error completely, an error of (phase error - SJW) remains.

Only one synchronization may be done between two Sample Points. The Synchronizations maintain a minimum distance between edges and Sample Points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize “hard” on the edge transmitted by the “leading” transceiver that started transmitting first, but due to propagation delay times, they cannot become ideally synchronized. The “leading” transmitter does not necessarily win the arbitration, therefore the receivers have to synchronize themselves to different transmitters that subsequently “take the lead” and that are differently synchronized to the previously “leading” transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that “takes the lead” in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator’s clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator’s tolerance range. The examples in the figure below show how the Phase Buffer Segments are used to compensate for phase errors.

There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a “late” edge, the lower drawing shows the synchronizations on an “early” edge, and the middle drawing is the reference without synchronization.

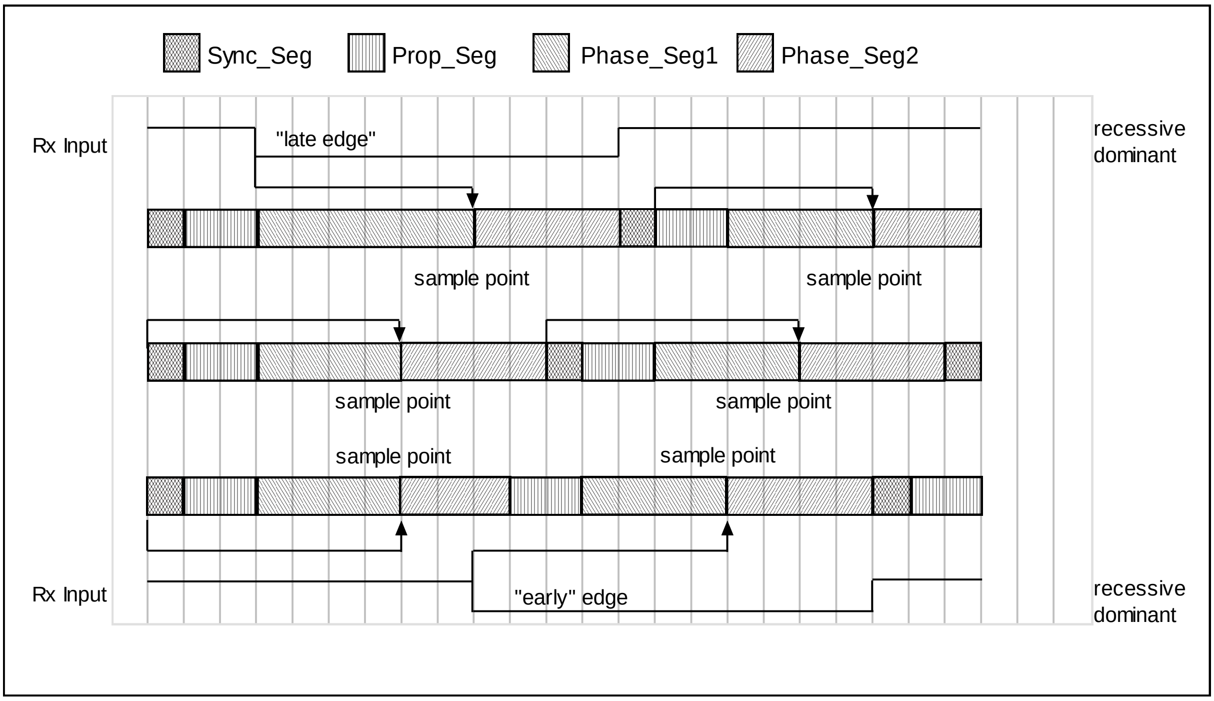


Figure 6-23 Synchronization on Late and Early Edges

In the first example an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is "late" since it occurs after the Sync_Seg. Reacting to the "late" edge, Phase_Seg1 is lengthened so that the distance from the edge to the Sample Point is the same as it would have been from the Sync_Seg to the Sample Point if no edge had occurred. The phase error of this "late" edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync_Seg.

In the second example an edge from recessive to dominant occurs during Phase_Seg2. The edge is "early" since it occurs before a Sync_Seg. Reacting to the "early" edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the Sample Point is the same as it would have been from an Sync_Seg to the Sample Point if no edge had occurred. As in the previous example, the magnitude of this "early" edge's phase error is less than SJW, so it is fully compensated. The Phase Buffer Segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation's state machine, where the bit time starts and ends at the Sample Points. The state machine omits Sync_Seg when synchronizing on an "early" edge because it cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

The examples in the next figure show how short dominant noise spikes are filtered by synchronizations. In both examples the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the Synchronization Jump Width is greater than or equal to the phase error of the spike's edge from recessive to dominant. Therefore the Sample Point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the Sample Point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

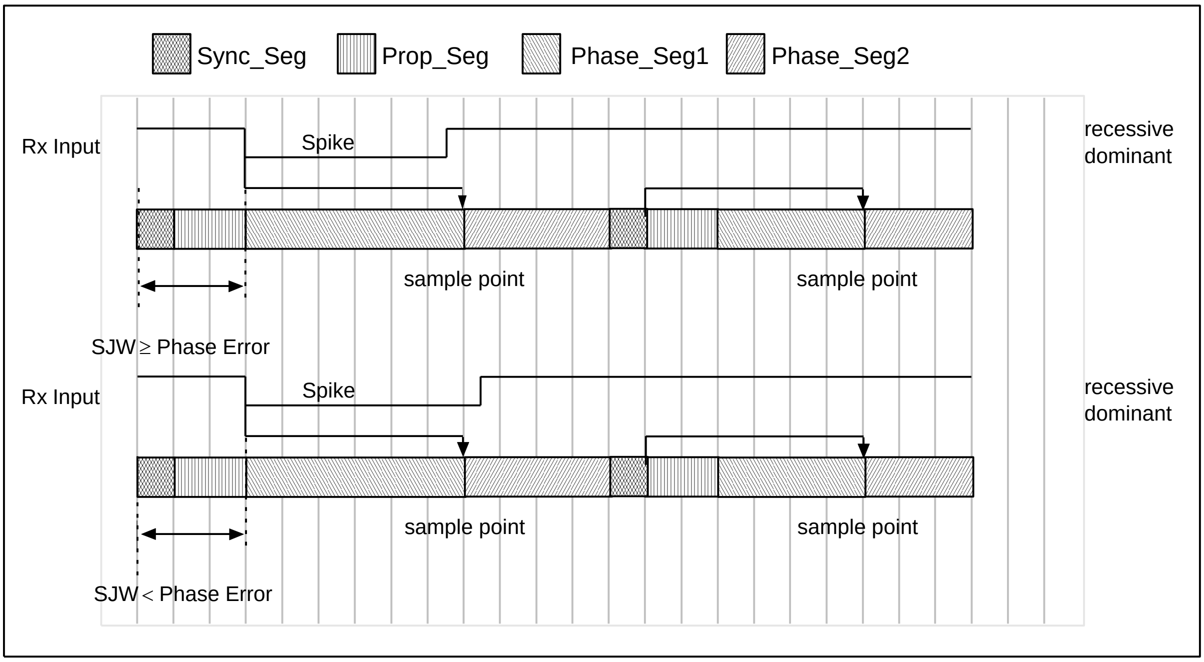


Figure 6-24 Filtering of Short Dominant Spikes

6.4.10.4 Oscillator Tolerance Range

The oscillator tolerance range was increased when the CAN protocol was developed from version 1.1 to version 1.2 (version 1.0 was never implemented in silicon). The option to synchronize on edges from dominant to recessive became obsolete, only edges from recessive to dominant are considered for synchronization. The only CAN controllers to implement protocol version 1.1 have been Intel 82526 and Philips 82C200, both are superseded by successor products.

The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range df for an oscillator's frequency f osc around the nominal frequency f nom with (1 – df) • f nom ≤ f osc ≤ (1 + df) • f nom depends on the proportions of Phase_Seg1, Phase_Seg2, SJW, and the bit time. The maximum tolerance df is the defined by two conditions (both shall be met):

```
I: df ≤ [min (Phase_Seg1, Phase_Seg2)/2 • (13•bit_time-Phase_Seg2)]

I: df ≤ [SJW/20 • bit_time]
```

It has to be considered that SJW may not be larger than the smaller of the Phase Buffer Segments and that the Propagation Time Segment limits that part of the bit time that may be used for the Phase Buffer Segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125kBit/s (bit time = 8ms) with a bus length of 40m.

6.4.10.5 Configuration of the CAN Protocol Controller

In most CAN implementations and also in this CAN, the bit timing configuration is programmed in two register bytes.

The sum of Prop_Seg and Phase_Seg1 (as TSEG1) is combined with Phase_Seg2 (as TSEG2) in one byte, SJW and BRP are combined in the other byte (see the figure below).

In these bit timing registers, the four components TSEG1, TSEG2, SJW, and BRP have to be programmed to a numerical value that is one less than its functional value; so instead of values in the range of [1..n], values in the range of [0..n-1] are programmed. That way, e.g. SJW (functional range of [1..4]) is represented by only two bits.

Therefore the length of the bit time is (programmed values) [TSEG1 + TSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

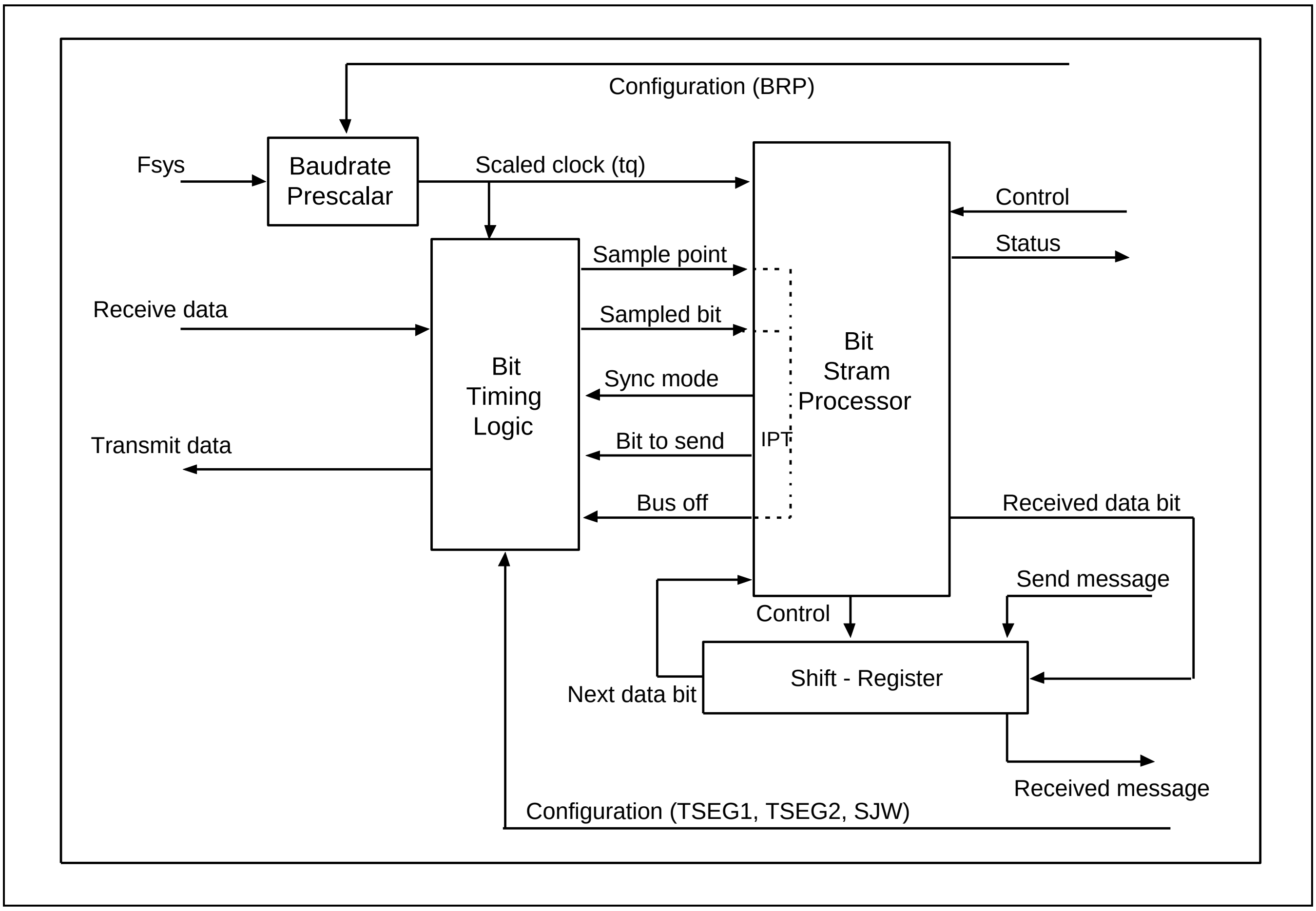


Figure 6-25 Structure of the CAN Controller Core

The data in the bit timing registers are the configuration input of the CAN protocol controller. The Baud Rate Prescalar (configured by BRP) defines the length of the time quantum, the basic time unit of the bit time; the Bit Timing Logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the BTL state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the Bit Stream Processor (BSP) state machine is evaluated once each bit time, at the Sample Point.

The Shift Register serializes the messages to be sent and parallelizes received messages. It's loading and shifting is controlled by the BSP.

The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the Sample Point and processes the sampled bus input bit. The time after the Sample point that is needed to calculate the next bit to be sent (e.g. data bit, CRC bit, stuff bit, error flag, or idle) is called the Information Processing Time (IPT).

The IPT is application specific but may not be longer than 2 tq; this CAN's IPT is 0 tq. Its length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.

6.4.10.6 Calculation of the Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting bit time (1/bit rate) must be an integer multiple of the system clock period.

The bit time may consist of 4 to 25 time quanta, the length of the time quantum tq is defined by the Baud Rate Pre-scalar with T_Q = (Baud Rate Pre-scalar + 1) / fsys. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop_Seg. Its length depends on the delay times measured in the system. A maximum bus length as well as a maximum node delay has to be defined for expandable CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of T_Q).

The Sync_Seg is 1 t_Q long (fixed), leaving (bit time - Prop_Seg - 1) T_Q for the two Phase Buffer Segments. If the number of remaining T_Q is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than the CAN controller’s Information Processing Time, which is, depending on the actual implementation, in the range of [0..2] T_Q.

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in section “*Oscillator Tolerance Range*”.

If more than one configuration is possible, that configuration allowing the highest oscillator tolerance range should be chosen.

CAN nodes with different system clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The CAN system’s oscillator tolerance range is limited by that node with the lowest tolerance range.

The calculation may show that bus length or bit rate have to be decreased or that the oscillator frequencies’ stability has to be increased in order to find a protocol compliant configuration of the CAN bit timing.

6.4.10.7 Example for bit Timing at High Baud Rate

In this example, the frequency of CAN module clock is 10MHz, BD (Baud rate prescalar) is 0, the bit rate is fixed at 1M Bit/s.

$T_Q = t_{CANCLK}$ = 100ns
Bit time = 1us = 10 × T_Q
Delay of bus driver: 50ns
Delay of receiver circuit: 30ns
Delay of bus line (40m): 220ns
Prog_Seg = 2 × (Delay of bus driver) + 2 × (Delay of receiver circuit) + 2 × (Delay of bus line) = 600ns = 6 × T_Q
The Sync_Seg is 1 T_Q long (fixed), leaving (bit time – Prop_Seg – 1) tq for the two Phase Buffer Segments. If the number of remaining tq is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1

T_{Sync_Seg} = 1 × T_Q
T_Q remaining = 3, then
Phase buffer Seg 1 = 1 × T_Q
Phase buffer Seg 2 = 2 × T_Q
So:
T_{PHS1} = Prop_Seg + Phase buffer Seg 1 = 7 × T_Q
T_{PHS2} = Phase buffer Seg 2 = 2 × T_Q
Bit time 1us = T_{Sync_Se}g + T_{PHS1} + T_{PHS2}
TSJW = min (4, Phase buffer Seg 1) = 1 × T_Q
PHSEG2[2:0] = (T_{PHS2} / T_Q) – 1 = 1
PHSEG1[3:0] = (T_{PHS1} / T_Q) – 1 = 6
AR = 1 (to be compliant with CAN standard)
SJW[1:0] = (T_{SJW} / T_Q) – 1 = 0
BD [9:0] = 0
CAN_MR = 0x00164000

6.4.10.8 Example for bit Timing at Low Baud Rate

In this example, the frequency of CAN module clock is 2MHz, BD(baud rate prescalar) is 1, the bit rate is fixed at 100K Bit/s.

$T_Q = 2 \times t_{CANCLK}$ = 1us
Bit time = 10us = 10 × T_Q
Delay of bus driver: 200ns
Delay of receiver circuit: 80ns
Delay of bus line (40m): 220ns
Prog_Seg = 2 × (Delay of bus driver) + 2 × (Delay of receiver circuit) + 2 × (Delay of bus line) = 1us = T_Q

The Sync_Seg is 1 tq long (fixed), leaving (bit time – Prop_Seg – 1) tq for the two Phase Buffer Segments. If the number of remaining tq is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1

T_{Sync_Seg} = 1 us = 1 × T_Q
TQ remaining = 8, then
Phase buffer Seg 1 = 4 × T_Q
Phase buffer Seg 2 = 4 × T_Q
So:
T_{PHS1} = Prop_Seg + Phase buffer Seg 1 = 5 × T_Q
T_{PHS2} = Phase buffer Seg 2 = 4 × T_Q
Bit time 10us = $T_{Sync\_Seg} + T_{PHS1} + T_{PHS2}$
T_{SJW} = min (4, Phase buffer Seg 1) = 4 × T_Q
PHSEG2[2:0] = (T_{PHS2} / T_Q) – 1 = 3
PHSEG1[3:0] = (T_{PHS1} / T_Q) – 1 = 4
AR = 1 (to be compliant with CAN standard)
SJW[1:0] = (T_{SJW} / T_Q) – 1 = 3
BD [9:0] = 1
CAN_MR = 0x00347001

6.4.10.9 Time Stamp

A 32.bit stamp allows to date all messages received.

The 32.bit register forming the second counter in the Stamp Timer (STT) is provided to the CAN module. After each reception of a CAN frame, the value of the current second counter will be automatically written in the corresponding CAN channel CAN_STPRX register.

6.5 Register Description

6.5.1 Register Map Summary

- Base Address: 0x400E_0000
- Base Address: 0x400E_1000

Register	Offset	Description	Reset Value
RSVD	0x0000–0x004C	Reserved	–
CAN_ECR	0x0050	Enable Clock Register	0x0000_0000
CAN_DCR	0x0054	Disable Clock Register	0x0000_0000
CAN_PMSR	0x0058	Power Management Status Register	0x0000_0000
RSVD	0x005C	Reserved	–
CAN_CR	0x0060	Control Register	0x0000_0000
CAN_MR	0x0064	Mode Register	0x0023_4001
RSVD	0x0068	Reserved	–
CAN_CSR	0x006C	Clear Status Register	0x0000_0000
CAN_SR	0x0070	Status Register	0x0000_0000
CAN_IER	0x0074	Interrupt Enable Register	0x0000_0000
CAN_IDR	0x0078	Interrupt Disable Register	0x0000_0000
CAN_IMR	0x007C	Interrupt Mask Register	0x0000_0000
RSVD	0x0080	Reserved	–
CAN_ISSR	0x0084	Interrupt Source Status Register	0x0000_0000
CAN_SIER	0x0088	Source Interrupt Enable Register	0x0000_0000
CAN_SIDR	0x008C	Source Interrupt Disable Register	0x0000_0000
CAN_SIMR	0x0090	Source Interrupt Mask Register	0x0000_0000
CAN_HPIR	0x0094	Highest Priority Interrupt Register	0x0000_0000
CAN_ERCR	0x0098	Error Counter Register	0x0000_0000
RSVD	0x009C–0x00FC	Reserved	–
CAN_TMR0	0x0100	Interface 0 transfer management register	0x0000_0001
CAN_DAR0	0x0104	Interface 0 data A register	0x0000_0000
CAN_DBR0	0x0108	Interface 0 data B register	0x0000_0000
CAN_MSKR0	0x010C	Interface 0 mask register	0xDFFF_FFFF
CAN_IR0	0x0110	Interface 0 identifier register	0x0000_0000
CAN_MCR0	0x0114	Interface 0 message control register	0x0000_0000
CAN_STPR0	0x0118	Interface 0 stamp register	0x0000_0000
RSVD	0x011C	Reserved	–
CAN_TMR1	0x0120	Interface 1 transfer management register	0x0000_0001
CAN_DAR1	0x0124	Interface 1 data A register	0x0000_0000
CAN_DBR1	0x0128	Interface 1 data B register	0x0000_0000

Register	Offset	Description	Reset Value
CAN_MSKR1	0x012C	Interface 1 mask register	0xDFFF_FFFF
CAN_IR1	0x0130	Interface 1 Identifier register	0x0000_0000
CAN_MCR1	0x0134	Interface 1 message control register	0x0000_0000
CAN_STPR1	0x0138	Interface 1 stamp register	0x0000_0000
RSVD	0x013C	Reserved	–
CAN_TRR	0x0140	Transmission request register	0x0000_0000
CAN_NDR	0x0144	New data register	0x0000_0000
CAN_MVR	0x0148	Message valid register	0x0000_0000
CAN_TSTR	0x014C	Test register	(Note)

NOTE: The value of this depends on CANRX pin value

6.5.1.1 CAN_ECR (CAN Enable Clock Register)

- Address = Base Address + 0x0050, Reset Value = 0x0000_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
DBGEN	RSVD																													CAN	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R

Name	Bit	Type	Description	Reset Value
CAN	[1]	W	CAN clock enable 0 = No effect 1 = Enable CAN clock	0
DBGEN	[31]	W	Debug mode enable 0 = No effect 1 = Enable debug mode	0

6.5.1.2 CAN_DCR (CAN Disable Clock Register)

- Address = Base Address + 0x0054, Reset Value = 0x0000_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
DBGEN	RSVD																													CAN	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R

Name	Bit	Type	Description	Reset Value
CAN	[1]	W	CAN clock disable 0 = No effect 1 = Disable CAN clock	0
DBGEN	[31]	W	Debug mode disable 0 = No effect 1 = Disable debug mode	0

6.5.1.3 CAN_PMSR (CAN Power Manager Status Register)

- Address = Base Address + 0x0058, Reset Value = 0x0000_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
DBGEN	RSVD																													CAN	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CAN	[1]	R	CAN clock status 0 = CAN clock disabled. 1 = CAN clock enabled. **NOTE:** 1. The CAN_PMSR register is not reset by software reset. 2. Disabling CAN clock while a transfer is occurring will stop immediately the current transfer. This transfer will finish as soon as the CPU enables CAN clock. To avoid incomplete transfer, software should ensure no transfer is occurring when disabling CAN clock.	0
DBGEN	[31]	R	Debug mode status 0 = Debug mode is disabled: dbgack_sclk input has no influence on CAN function. 1 = Debug mode is enabled: when the dbgack_sclk input signal is low, the CAN function is left unchanged. When this signal is active high, the CAN function is frozen, and in case a transfer is in progress, the transfer will be finished safely before the CAN is frozen. However, full read/write access to internal register is kept for debug purpose.	0

6.5.1.4 CAN_CR (CAN Control Register)

- Address = Base Address + 0x0060, Reset Value = 0x0000_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
RSVD																					STSR	ABBTX	RQBTX	RSVD			CCDIS	CCEN	CANDIS	CANEN	SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	R	R	R	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	CAN software reset 0 = No effect. 1 = Reset the CAN. A software reset triggered hardware reset of the CAN is performed. It reset all the registers except the CAN_PMSR. The data stored in the Message RAM is not affected by a hardware reset, except MSGVAL, TXRQST, NEWDAT and ITPND bits for each Message Objects. After power-on, the content of the Message RAM is undefined.	0
CANEN	[1]	W	CAN enable 0 = No effect. 1 = Enables the CAN. Enabling the CAN finishes the software initialization, transmitter and receiver state machines are reset. Afterwards the Bit Stream Processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus Idle) before it can take part in bus activities and starts the message transfer. This delay is part of the CAN standard. If CAN is disabled because it goes in bus off state, CAN could be enabled again, but it will take part in bus activities after 129 successive occurrences of bus idle (a sequence of 11 consecutive recessive bits) occurred on the bus. The CANENS bit is set only at the end of the 129 occurrence of 11 consecutive recessive bits. The bus off recovery sequence can be monitored by reading the REC counter: it shall be reset when CPU enables the CAN, and each time a sequence of 11 consecutive recessive bits occurred on the bus, the REC counter shall be increased by one.	0
CANDIS	[2]	W	CAN disable 0 = No effect. 1 = Disables the CAN. No data will be received or transmitted. Disabling CAN module while module is	0

Name	Bit	Type	Description	Reset Value
			transferring will stop immediately the current transfer. The status of the CAN bus output CAN_TX is 'recessive' (HIGH). The counter of the ERCR (Error Counter Register) is unchanged. Disabling CAN does not change any configuration register. In case both CANEN and CANDIS are equals to one when the control register is written, the CAN will be disabled.	
CCEN	[3]	W	Configuration change enable 0 = No effect. 1 = Enables the write access on Bit timing bits in CAN_MR register (while CAN is disabled: CANENS = 0 in CAN_SR register).	0
CCDIS	[4]	W	Configuration change disable 0 = No effect. 1 = Disables the write access on bit timing bits in CAN_MR register. In case both CCEN and CCDIS are equals to one when the control register is written, the configuration change will be disabled.	0
RQBTX	[8]	W	Request basic transmission 0 = No effect. 1 = Request the transmission of the contents of IF0 registers (Basic mode). This bit is used only in basic mode (BASIC = '1' in CAN_TSTR register).	0
ABBTX	[9]	W	Abort basic transmission 0 = No effect. 1 = Abort the transmission of the contents of IF0 registers (Basic mode). This bit is used only in basic mode (BASIC = '1' in CAN_TSTR register). In case both RQBTX and ABBTX are equals to one when the control register is written, the transmission is aborted.	0
STSR	[10]	W	Store shift register 0 = No effect. 1 = The contents of the shift register is stored into the IF1 Registers (Basic mode). It allows to monitor the actual contents of the shift register. This bit is used only in basic mode (BASIC = '1' in CAN_TSTR register).	0

6.5.1.5 CAN_MR (CAN Mode Register)

- Address = Base Address + 0x0064, Reset Value = 0x0023_4001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									RHSEG2			PHSEG1				RSVD	AR	SJW		RSVD	CSSEL	BD									
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
BD	[9:0]	RW	Baud rate pre-scalar Used for generating the time quantum (T_Q) using the following formula: T_Q = (BD + 1) / CANCLK The bit time is built up from a multiple of this quantum.	0x001
CSSEL	[10]	RW	CAN Clock(CANCLK) Source Selection Bit 0 = PCLK 1 = EMCLK	0'b
SJW	[13:12]	RW	Synchronization jump width The CAN controller re-synchronizes on each edge of the transmission. The (SWJ + 1) value defines the maximum number of CAN clock cycles a bit period may be shortened or lengthened. $T_{SWJ} = T_Q \times (SJW + 1)$	00'b
AR	[14]	RW	Automatic Retransmission 0 = No Automatic Retransmission 1 = Activate Automatic Retransmission (This bit must be set to 1 to be compliant with the CAN standard.)	1'b
PHSEG1	[19:16]	RW	Phase segment 1 value The time segment before the sample point. It is the sum of the Propagation Segment and Phase Buffer Segment 1. Zero is a prohibited value and will be written as a one. $T_{PHS1} = T_Q \times (PHSEG1 + 1)$	0011'b
PHSEG2	[22:20]	RW	Phase segment 2 value The time segment after the sample point. $T_{PHS2} = T_Q \times (PHSEG2 + 1)$	010'b

All the bits of this register, except AR bit, can be written only if CCENS (Configuration Change Enable) bit is high and CANENS (CAN Enable Status) bit is low. (See CAN_SR (CAN Status Register), P8-63).

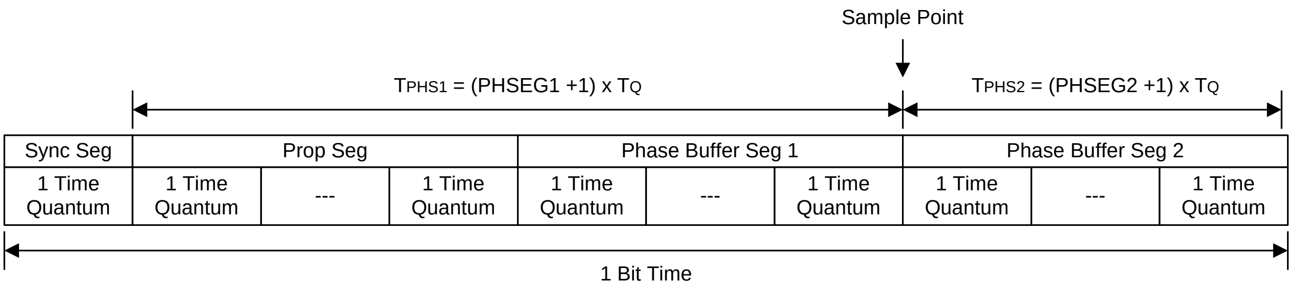


Figure 6-26 CAN Bit Time Built up from Time Quantum

Table 6-11 Bit rate and Minimal Core Frequency

Bit Rate	Minimal CANCLK
125Kbits	1MHz
250Kbits	2MHz
500Kbits	4MHz
1Mbits	8MHz

6.5.1.6 CAN_CSR (CAN Clear Status Register)

- Address = Base Address + 0x006C, Reset Value = 0x0000_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
RSVD																CRC	BIT0	BIT1	ACK	FORM	STUFF	TXOK	RXOK	RSVD			ACTVT	BUSOFFTR	ERPASSTR	ERWARNTR	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	R	R	R	W	W	W	W	R

Name	Bit	Type	Description	Reset Value
ERWARNTR	[1]	W	Clear error passive warning transition 0 = No effect. 1 = Clear error passive warning transition interrupt.	0
ERPASSTR	[2]	W	Clear error passive transition 0 = No effect. 1 = Clear error passive transition interrupt.	0
BUSOFFTR	[3]	W	Clear bus off transition 0 = No effect. 1 = Clear bus off transition interrupt.	0
ACTVT	[4]	W	Clear activity 0 = No effect. 1 = Clear activity interrupt.	0
RXOK	[8]	W	Clear successfully received message 0 = No effect. 1 = Clear successfully received message interrupt.	0
TXOK	[9]	W	Clear successfully transmit message 0 = No effect. 1 = Clear successfully transmit message interrupt.	0
STUFF	[10]	W	Clear stuff error 0 = No effect. 1 = Clear stuff error interrupt.	0
FORM	[11]	W	Clear form error 0 = No effect. 1 = Clear form error interrupt.	0
ACK	[12]	W	Clear acknowledge error 0 = No effect. 1 = Clear acknowledge error interrupt.	0
BIT1	[13]	W	Clear bit to one error 0 = No effect. 1 = Clear bit to one error interrupt.	0
BIT0	[14]	W	Clear bit to zero error 0 = No effect. 1 = Clear bit to zero error interrupt.	0
CRC	[15]	W	Clear CRC error 0 = No effect. 1 = Clear CRC error interrupt.	0

6.5.1.7 CAN_SR (CAN Status Register)

- Address = Base Address + 0x0070, Reset Value = 0x0000_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
RSVD						BTXPD	CCENS	TS	RS	BUSY1	BUSY0	BUSOFF	ERPASS	ERWARN	CANENS	CRC	BIT0	BIT1	ACK	FORM	STUFF	TXOK	RXOK	RSVD			ACTVT	BUSOFFTR	ERPASSTR	ERWARNTR	ISS
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ISS	[0]	R	Interrupt Source Status 0 = No interrupt in any channel. 1 = At least one interrupt occurred in a channel and the corresponding bit in Source Interrupt Mask Register (SIMR, P8-76) is enabled (read CAN_ISSR(P8-73) for more information).	0
ERWARNTR	[1]	R	Error passive warning transition 0 = No transition from or to error passive warning occurs since the last reset of this bit. 1 = At least one transition from or to error passive warning occurs (one of the error counters reached the error passive warning limit of 96) and this bit has not been reset.	0
ERPASSTR	[2]	R	Error passive transition 0 = No transition between error passive mode since last reset of this bit. 1 = At least one transition from or to error passive mode and this bit has not been reset.	0
BUSOFFTR	[3]	R	Bus off transition 0 = No passage in bus off mode since last reset of this bit. 1 = At least one transition from or to bus off mode and this bit has not been reset.	0
ACTVT	[4]	R	Activity 0 = No activity (no dominant level) has occurred on the CAN_RX pin since the last reset of this bit or CAN clock is enabled. 1 = Activity (dominant level) has occurred on the CAN_RX pin since the last reset of this bit while CAN clock is disabled (halt mode and / or CAN = ‘0’ in CAN_PMSR register).	0
RXOK	[8]	R	Successfully received a message 0 = No message received since last reset of this bit. 1 = At least one message has been successfully	0

Name	Bit	Type	Description	Reset Value
			received.	
TXOK	[9]	R	Successfully transmitted a message 0 = No message transmitted since last reset of this bit. 1 = At least one message has been successfully transmitted.	0
STUFF	[10]	R	Stuff error 0 = No stuff error occurred during last message transfer since last reset of this bit. 1 = During last message transfer, a stuff error occurred – more than five equal bits in a sequence have occurred in a part of a received message where this is not allowed.	0
FORM	[11]	R	Form error 0 = No form error occurred during last message transfer since last reset of this bit. 1 = During last message transfer, a form error occurred: a fixed format part of a received frame has the wrong format.	0
ACK	[12]	R	Acknowledge error 0 = No acknowledge error occurred during last message transfer since last reset of this bit. 1 = During last message transfer, an acknowledge error occurred: the message this CAN transmitted was not acknowledge by another node.	0
BIT1	[13]	R	Bit to one error 0 = No bit to one error occurred during last message transfer since last reset of this bit. 1 = During last message transfer, a bit to one error occurred: during the transmission of a message, with the exception of the identifier, the device wanted to send a recessive level, but monitored a dominant level.	0
BIT0	[14]	R	Bit to zero error 0 = No bit to zero error occurred during last message transfer since last reset of this bit. 1 = During last message transfer, a bit to zero error occurred: during the transmission of a message (or acknowledged bit, or active error flag, or overload flag), the device wanted to send a dominant level, but monitored a recessive level. During bus off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the bus off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).	0
CRC	[15]	R	CRC error 0 = No CRC error occurred during last message transfer (reception or transmission) since last reset of this bit. 1 = A CRC error occurred – the CRC check sum was incorrect in the message received – during last message	0

Name	Bit	Type	Description	Reset Value
			transfer (reception or transmission).	
CANENS	[16]	R	CAN enable status 0 = CAN is disabled. All message transfer from and to CAN bus is stopped, the status of the output pin CAN_TX is recessive and the error counters are unchanged. Disable the CAN does not change any configuration register. 1 = CAN is enabled. It can take part in bus activities and can start message transfer.	0
ERWARN	[17]	R	Error passive warning 0 = No error passive warning. 1 = At least one of the error counters reached the error passive warning limit of 96.	0
ERPASS	[18]	R	Error passive 0 = CAN is not in error passive mode. 1 = CAN is in error passive mode.	0
BUSOFF	[19]	R	Bus off 0 = CAN is not in bus off mode. 1 = CAN is in bus off mode. When the CAN controller goes in bus off mode, it becomes disable (CANENS is reset) and does not take part any longer in bus activities. To resume in normal operation mode, CPU has to enable the CAN controller (set CANEN bit in CAN_CR register), at this point the CAN controller will wait for 129 successive occurrences of bus idle (a sequence of 11 consecutive recessive bits). At the end of the bus off recovery sequence, the read and write error counter are reset and the BUSOFF bit is reset. The bus off recovery sequence cannot be shortened by resetting CAN (SWRST set to '1' in CAN_CR). During the waiting time after CPU has enabled CAN controller, each time a sequence of 11 recessive bits has been monitored, BIT0 bit is set in the status register, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus off recovery sequence.	0
BUSY0	[20]	R	Busy flag of interface 0 0 = Read/write action between interface 0 and Message RAM has finished. 1 = A transfer – read or write – between interface 0 and Message RAM is in progress.	0
BUSY1	[21]	R	Busy flag of interface 1 0 = Read/write action between interface 1 and Message RAM has finished. 1 = A transfer – read or write – between interface 1 and Message RAM is in progress.	0
RS	[22]	R	Receive status	0

Name	Bit	Type	Description	Reset Value
			0 = If the transmit status is also clear then the CAN controller is idle; otherwise it is in transmit mode. 1 = CAN controller entered receive mode from idle, or by losing arbitration during transmission. Note: If CAN controller lost arbitration during transmission, during a CAN clock period, RS and TS bit are set to '1'.	
TS	[23]	R	Transmit status 0 = If the receive status is also clear then the CAN controller is idle; otherwise it is in receive mode. 1 = CAN controller has started to transmit a message.	0
CCENS	[24]	R	Configuration change enable 0 = The CPU has no write access to the bit timing bits in CAN_MR register. 1 = The CPU has write access to the bit timing bits in CAN_MR register while CAN is disabled (CANENS = 0).	0
BTXPD	[25]	R	Basic transmission pending 0 = No transmission has been requested in basic mode or the transmission has completed or basic mode has been left. 1 = The transmission of the IF0 registers is pending or occurring at that moment. This bit is used only in basic mode (BASIC = '1' in CAN_TSTR register).	0

CRC, BIT0, BIT1, ACK, FORM and STUFF bits are mutually exclusives and are updated by the CAN controller at the end of each transfer (reception or transmission).

6.5.1.8 CAN_IER (CAN Interrupt Enable Register)

- Address = Base Address + 0x0074, Reset Value = 0x0000_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
RSVD																CRC	BIT0	BIT1	ACK	FORM	STUFF	TXOK	RXOK	RSVD			ACTVT	BUSOFFTR	ERPASSTR	ERWARNTR	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	R	R	R	W	W	W	W	R

Name	Bit	Type	Description	Reset Value
ERWARNTR	[1]	W	Error passive warning transition enable 0 = No effect. 1 = Enable error passive warning transition interrupt.	0
ERPASSTR	[2]	W	Error passive transition 0 = No effect. 1 = Enable error passive transition interrupt.	0
BUSOFFTR	[3]	W	Bus off transition 0 = No effect. 1 = Enable bus off transition interrupt.	0
ACTVT	[4]	W	Activity 0 = No effect. 1 = Enable activity interrupt.	0
RXOK	[8]	W	Successfully received a message 0 = No effect. 1 = Enable successfully received a message interrupt.	0
TXOK	[9]	W	Successfully transmitted a message 0 = No effect. 1 = Enable successfully transmit a message interrupt.	0
STUFF	[10]	W	Stuff error 0 = No effect. 1 = Enable stuff error interrupt.	0
FORM	[11]	W	Form error 0 = No effect. 1 = Enable form error interrupt.	0
ACK	[12]	W	Acknowledge error 0 = No effect. 1 = Enable acknowledge error interrupt.	0
BIT1	[13]	W	Bit to one error 0 = No effect. 1 = Enable bit to one error interrupt.	0

Name	Bit	Type	Description	Reset Value
BIT0	[14]	W	Bit to zero error 0 = No effect. 1 = Enable bit to zero error interrupt.	0
CRC	[15]	W	CRC error 0 = No effect. 1 = Enable CRC error interrupt.	0

6.5.1.9 CAN_IDR (CAN Interrupt Disable Register)

- Address = Base Address + 0x0078, Reset Value = 0x0000_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
RSVD																CRC	BIT0	BIT1	ACK	FORM	STUFF	TXOK	RXOK	RSVD			ACTVT	BUSOFFTR	ERPASSTR	ERWARNTR	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	R	R	R	W	W	W	W	R

Name	Bit	Type	Description	Reset Value
ERWARNTR	[1]	W	Error passive warning transition disable 0 = No effect. 1 = Disable error passive warning transition interrupt.	0
ERPASSTR	[2]	W	Error passive transition disable 0 = No effect. 1 = Disable error passive transition interrupt.	0
BUSOFFTR	[3]	W	Bus off disable 0 = No effect. 1 = Disable bus off transition interrupt.	0
ACTVT	[4]	W	Activity disable 0 = No effect. 1 = Disable activity interrupt.	0
RXOK	[8]	W	Successfully received a message disable 0 = No effect. 1 = Disable successfully received a message interrupt.	0
TXOK	[9]	W	Successfully transmitted a message disable 0 = No effect. 1 = Disable successfully transmit a message interrupt.	0
STUFF	[10]	W	Stuff error disable 0 = No effect. 1 = Disable stuff error interrupt.	0
FORM	[11]	W	Form error disable 0 = No effect. 1 = Disable form error interrupt.	0
ACK	[12]	W	Acknowledge error disable 0 = No effect. 1 = Disable acknowledge error interrupt.	0
BIT1	[13]	W	Bit to one error disable 0 = No effect. 1 = Disable bit to one error interrupt.	0

Name	Bit	Type	Description	Reset Value
BIT0	[14]	W	Bit to zero error disable 0 = No effect. 1 = Disable bit to zero error interrupt.	0
CRC	[15]	W	CRC error disable 0 = No effect. 1 = Disable CRC error interrupt.	0

6.5.1.10 CAN_IMR (CAN Interrupt Mask Register)

- Address = Base Address + 0x007C, Reset Value = 0x0000_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
RSVD																CRC	BIT0	BIT1	ACK	FORM	STUFF	TXOK	RXOK	RSVD			ACTVT	BUSOFFTR	ERPASSTR	ERWARNTR	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ERWARNTR	[1]	R	Error passive warning mask 0 = Error passive warning interrupt is disabled. 1 = Error passive warning interrupt is enabled.	0
ERPASSTR	[2]	R	Error passive mask 0 = Error passive interrupt is disabled. 1 = Error passive interrupt is enabled.	0
BUSOFFTR	[3]	R	Bus off mask 0 = Bus off interrupt is disabled. 1 = Bus off interrupt is enabled.	0
ACTVT	[4]	R	Activity mask 0 = Activity interrupt is disabled. 1 = Activity interrupt is enabled.	0
RXOK	[8]	R	Successfully received a message mask 0 = Successfully received a message interrupt is disabled. 1 = Successfully received a message interrupt is enabled.	0
TXOK	[9]	R	Successfully transmitted a message mask 0 = Successfully transmit a message interrupt is disabled. 1 = Successfully transmit a message interrupt is enabled.	0
STUFF	[10]	R	Stuff error mask 0 = Stuff error interrupt is disabled. 1 = Stuff error interrupt is enabled.	0
FORM	[11]	R	Form error mask 0 = Form error interrupt is disabled. 1 = Form error interrupt is enabled.	0
ACK	[12]	R	Acknowledge error mask 0 = Acknowledge error interrupt is disabled. 1 = Acknowledge error interrupt is enabled.	0
BIT1	[13]	R	Bit to one error mask 0 = Bit to one error interrupt is disabled.	0

Name	Bit	Type	Description	Reset Value
			1 = Bit to one error interrupt is enabled.	
BIT0	[14]	R	Bit to zero error mask 0 = Bit to zero error interrupt is disabled. 1 = Bit to zero error interrupt is enabled.	0
CRC	[15]	R	CRC error mask 0 = CRC error interrupt is disabled. 1 = CRC error interrupt is enabled.	0

6.5.1.11 CAN_ISSR (CAN Interrupt Source Status Register)

- Address = Base Address + 0x0084, Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	R	Channel X interrupt 0 = No interrupt occurred on channel X. 1 = An interrupt occurred on channel X: ITPND = ‘1’ on channel X.	0

6.5.1.12 CAN_SIER (CAN Source Interrupt Enable Register)

- Address = Base Address + 0x0088, Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	W	Channel X interrupt enable 0 = No effect. 1 = Enable Channel X interrupt.	0

6.5.1.13 CAN_SIDR (CAN Source Interrupt Disable Register)

- Address = Base Address + 0x008C, Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	W	Channel X interrupt disable 0 = No effect. 1 = Disable Channel X interrupt.	0

6.5.1.14 CAN_SIMR (CAN Source Interrupt Mask Register)

- Address = Base Address + 0x0090, Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	R	Channel X interrupt mask 0 = Channel X interrupt is disabled. 1 = Channel X interrupt is enabled.	0

6.5.1.15 CAN_HPIR (CAN Highest Priority Interrupt Register)

- Address = Base Address + 0x0094, Reset Value = 0x0000_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
RSVD																INTID															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
INTID	[15:0]	R	Channel X interrupt mask 0 = No interrupt is pending. • 0x0001-0x0020: The number of Message Object which caused the interrupt. • 0x0021-0x7FFF: Reserved. • 0x8000: Status interrupt: at least one interrupt raised up in status register (except ACTVT and ISS) and while enabled (corresponding bit in IMR is set). Read CAN_SR for more information. • 0x8001-0xFFFF: Reserved. If several interrupt are pending, the CAN highest priority interrupt register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it. The status interrupt has the highest priority. Among the channel interrupt, the Message object's interrupt priority decreases with increasing message number.	0x0000

6.5.1.16 CAN_ERCR (CAN Error Counter Register)

- Address = Base Address + 0x0098, Reset Value = 0x0000_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
RSVD																TEC								REP	REC						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
REC	[6:0]	R	Reception error counter Here is the value of the reception error counter.	0x00
REP	[7]	R	Receive Error Passive 0 = The receive Error Counter is below the error passive level as defined in the CAN specification. 1 = The receive Error Counter has reached the error passive level.	0
TEC	[15:8]	R	Transmit error counter Here is the value of the transmit error counter.	0x00

6.5.1.17 CAN_TMR (CAN Interface X Transfer Management Register 0–1)

- Address = Base Address + 0x0100, Reset Value = 0x0000_0001
- Address = Base Address + 0x0120, Reset Value = 0x0000_0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																CLRIT	TRND	RSVD	AMCA	AIR	AMSKR	ADBR	ADAR	WR	RSVD	NUMBER					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
NUMBER	[5:0]	RW	Message number Message number to read or write. Only values between 0x01 and 0x20 (0x01 and 0x20 included) are valid. When a message number that is not valid (0x00 and 0x21 to 0x3F) is written to this register, the message number is transformed into a valid message value and that Message is transferred.	0x00
WR	[7]	RW	Write or read direction 0 = Read: transfer from the message object NUMBER into the selected interface X registers. 1 = Write: transfer direction from the selected interface X registers to the message object NUMBER.	0
ADAR	[8]	RW	Access Data A register 0 = Data bytes 0-3 unchanged. 1 = Transfer data bytes 0-3 between message object and CAN_DARx register.	0
ADBR	[9]	RW	Access Data B register 0 = Data bytes 4-7 unchanged. 1 = Transfer data bytes 4-7 between message object and CAN_DBRx register.	0
AMSKR	[10]	RW	Access mask register 0 = Mask bits unchanged. 1 = Transfer mask bits between message object and CAN_MSKRx register.	0
AIR	[11]	RW	Access identifier register 0 = Identifier bits unchanged. 1 = Transfer identifier, direction, extended and message valid bits between message object and CAN_IRx register.	0
AMCR	[12]	RW	Access message control register 0 = Control bits unchanged.	0

Name	Bit	Type	Description	Reset Value
			1 = Transfer control bits between message object and CAN_MCRx register.	
TRND	[14]	RW	Set TXRQST bit or clear NEWDAT 0 = TXRQST or NEWDAT bit unchanged. 1 = When writing (WR = 1), it set TXRQST bit in the message object. When reading (WR = 0), it clears NEWDAT bit in the message object. In write mode (WR = 1), if a transmission is requested by programming bit TXRQST in this register, bit TXRQST in the CAN_MCRx register will be ignored. In read mode (WR = 0), a read access to a message object can be combined with the reset of the control bits ITPND and NEWDAT. The value of these bits transferred to the CAN_MCRx register always reflect the status before resetting these bits.	0
CLRIT	[15]	RW	Clear interrupt pending bit 0 = ITPND bit remains unchanged. 1 = Clear ITPND bit in the message object. This bit is ignored in write mode.	0

A message transfer between the Message RAM and the IFx Message buffer registers is started as soon as the CPU has written in this register. With this write operation the BUSYx bit is automatically set to ‘1’ in the CAN_SR, and it is reset to ’0’ once the transfer between the Interface Registers and the Message RAM has completed. Registers of interface X (DAR, DBR, MSKR, IR and MCR) must not be accessed while the BUSYx bit is read as a ’1’ in the CAN_SR. Most of the bit of this register have different function depending on WR bit.

Caution: The message object number 32 shall not be used in transmission.

6.5.1.18 CAN_DAR (CAN Interface X Data A Register 0–1)

- Address = Base Address + 0x0104, Reset Value = 0x0000_0000
- Address = Base Address + 0x0124 Reset Value = 0x0000_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
DATA3								DATA2								DATA1								DATA0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA0	[7:0]	RW	Data0 of interface Data number 0 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA1	[15:8]	RW	Data1 of interface Data number 1 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA2	[23:16]	RW	Data2 of interface Data number 2 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA3	[31:24]	RW	Data3 of interface Data number 3 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00

Access to this register is prohibited while BUSYx bit is set.

6.5.1.19 CAN_DBR (CAN Interface X Data B Register 0–1)

- Address = Base Address + 0x0108, Reset Value = 0x0000_0000
- Address = Base Address + 0x0128 Reset Value = 0x0000_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
DATA3								DATA2								DATA1								DATA0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA0	[7:0]	RW	Data0 of interface Data number 0 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA1	[15:8]	RW	Data1 of interface Data number 1 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA2	[23:16]	RW	Data2 of interface Data number 2 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00
DATA3	[31:24]	RW	Data3 of interface Data number 3 of interface. In a CAN data frame, DATA0 is the first, DATA7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.	0x00

Access to this register is prohibited while BUSYx bit is set.

6.5.1.20 CAN_MSKR (CAN Interface X Mask Register 0–1)

- Address = Base Address + 0x010C, Reset Value = 0xDFFF_FFFF
- Address = Base Address + 0x012C Reset Value = 0xDFFF_FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MXTD	MMDIR	RSVD	BASEMASK											EXTMASK																	
1	1	0	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
RW	RW	R	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

Name	Bit	Type	Description	Reset Value
EXTMASK	[17:0]	RW	Extended identifier mask 18-bit mask for extended identifier (only extended identifier). If CAN controller received a 29-bit identifier (extended frame), all mask bits (BASEMASK [28:18] and EXTMASK[17:0]) are used. When a bit within EXTMASK bits is set to one, the corresponding identifier bit is used for acceptance filtering. When a bit within EXTMASK bits is set to zero, the corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering.	0x7FFF
BASEMASK	[28:18]	RW	Base identifier mask 11-bit mask for base identifier (standard and extended identifier). If CAN controller received a 11-bit identifier (standard frame), the lower bits (EXTMASK[17:0]) are not used. When a bit within BASEMASK bits is set to one, the corresponding identifier bit is used for acceptance filtering. When a bit within BASEMASK bits is set to zero, the corresponding identifier bit cannot inhibit the match in the acceptance filtering.	0x1FFF
MMDIR	[30]	RW	Message direction mask 0 = The MDIR bit of CAN_IRx register has no effect on the acceptance filtering. 1 = The MDIR bit of CAN_IRx register is used for acceptance filtering.	1
MXTD	[31]	RW	XTD bit mask 0 = The XTD bit of CAN_IRx register has no effect on the acceptance filtering. 1 = The XTD bit of CAN_IRx register is used for acceptance filtering.	1

Access to this register is prohibited while BUSYx bit is set.

6.5.1.21 CAN_IR (CAN Interface X Identifier Register 0–1)

- Address = Base Address + 0x0110, Reset Value = 0x0000_0000
- Address = Base Address + 0x0130 Reset Value = 0x0000_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
MSGVAL	XTD	MDIR	BASEID											EXTID																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
EXTID	[17:0]	RW	Extended identifier of interface X 18-bit value for extended identifier. If CAN sends a 29-bit identifier, all bits (BASEID and EXTID) are used. If CAN sends a 29-bit identifier, base identifier (BASEID) are sent first, extended identifier (EXTID) later.	0x00000
BASEID	[28:18]	RW	Base identifier of interface X 11-bit value for base identifier (standard and extended identifier). If CAN sends only a 11-bit identifier, the lower bits (ID[17:0]) are not used. If CAN sends a 29-bit identifier, base identifier (BASEID) are sent first, extended identifier (EXTID) later.	0x000
MDIR	[29]	RW	Message direction 0 = Receive. If TXRQST is set to ‘1’, a remote frame with the identifier of this message object is transmitted. If a data frame with matching identifier is received, that message is stored in this message object. 1 = Transmit. If TXRQST is set to ‘1’, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, if RMTEN = ‘1’ the TXRQST bit of this message object is set, if RMTEN = ‘0’ and UMASK = ’0’ the remote frame is ignored, if RMTEN = ‘0’ and UMASK = ‘1’ the arbitration and control field (ID + IDE + RTR + DLC) from the received message are stored into the Message Object in the Message RAM and the NEWDAT bit of this Message Object is set.	0
XTD	[30]	RW	Extended identifier 0 = The 11-bit (“standard”) identifier will be used for this message object. 1 = The 29-bit (“extended”) identifier will be used for this message object.	0
MSGVAL	[31]	RW	Message Valid	0

Name	Bit	Type	Description	Reset Value
			0 = The message object is ignored by the message handler. 1 = The message object is configured and should be considered by the message handler. The CPU must reset the MSGVAL bit of all unused message objects during initialization before it enables CAN (CANEN in the CAN_CR register). This bit must also be reset before the identifier (BASEID and EXTID), the control bits XTD and MDIR or the data length code DLC are modified, or if the messages object is no longer required.	

Access to this register is prohibited while BUSYx bit is set.

6.5.1.22 CAN_MCR (CAN Interface X Message Control Register 0–1)

- Address = Base Address + 0x0114, Reset Value = 0x0000_0000
- Address = Base Address + 0x0134, Reset Value = 0x0000_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
RSVD																NEWDAT	MSGLST	ITPND	UMASK	TXIE	RXIE	RMTEN	TXRQST	OVERWRITE	RSVD			DLC			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	R	R	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
DLC	[3:0]	RW	Data length code This is the number of bytes in the data field of a message (from 0 to 8). This value is updated whenever a frame is received (data or remote).	0x0
OVERWRITE	[7]	RW	Overwrite mode 0 = Message object is configured in normal mode. In normal mode, new frames do not overwrite the previous frame. 1 = Message object is configured in overwrite mode. In overwrite mode, new frames overwrite the previous frame. This bit is used to concatenate two or more messages objects (up to 32) to build a FIFO buffer. For single message objects (not belonging to a FIFO buffer) this bit must always be set to one.	0
TXRQST	[8]	RW	Transmit request 0 = This message object is not waiting for transmission. 1 = The transmission of this message object is requested and is not yet done.	0
RMTEN	[9]	RW	Remote enable 0 = At the reception of a remote frame, TXRQST is left unchanged. 1 = At the reception of a remote frame, TXRQST is set.	0
RXIE	[10]	RW	Receive interrupt enable 0 = ITPND will be left unchanged after a successful reception of a frame. 1 = ITPND will be set after a successful reception of a frame.	0
TXIE	[11]	RW	Transmit interrupt enable 0 = ITPND will be left unchanged after a successful transmission of a frame. 1 = ITPND will be set after a successful transmission of a	0

Name	Bit	Type	Description	Reset Value
			frame.	
UMASK	[12]	RW	Use acceptance mask 0 = Mask ignored. 1 = Use mask (BASEMASK, EXTMASK, MXTD and MMDIR) for acceptance filtering.	0
ITPND	[13]	RW	Interrupt pending 0 = This message is not the source of an interrupt. 1 = This message object is the source of an interrupt.	0
MSGLST	[14]	RW	Message lost 0 = No message lost since last time this bit was reset by the CPU. 1 = The message handler stored a new message into this object when NEWDAT was still set, the CPU lost a message. This is only valid for message objects with direction = receive.	0
NEWDAT	[15]	RW	New data 0 = No new data has been written into the data portion of this message object by the message handler since last time this flag was reset by the CPU. 1 = The message handler or the CPU has written new data into the data portion of this message object.	0

Access to this register is prohibited while BUSYx bit is set. In basic mode, for interface 0 (transmit interface), only DLC bits retain their functions. For interface 1 (receive interface), only NEWDAT, MSGLST and DLC bits retains theirs functions, and when a message is received other bits are read as ‘0’.

6.5.1.23 CAN_STPR (CAN Interface X Stamp Register 0–1)

- Address = Base Address + 0x0118, Reset Value = 0x0000_0000
- Address = Base Address + 0x0138 Reset Value = 0x0000_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
STAMP																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
STAMP	[31:0]	R	Stamp value These 32-bits stamp the date at which the message linked with the channel X has been received. The value is copied from the STT second counter register.	0x0000_0000

6.5.1.24 CAN_TRR (CAN Transmission Request Register)

- Address = Base Address + 0x0140 Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	R	Transmission request on channel x 0 = This message object is not waiting for a transmission. 1 = The transmission object of this message object is requested and is not yet done.	0

6.5.1.25 CAN_NDR (CAN New Data Register)

- Address = Base Address + 0x0144 Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	R	New data on channel x 0 = No new data has been written into the data portion of this message object by the message handler since last time this flag was cleared by the CPU. 1 = The message handler or the CPU has written new data into the data portion of this message object.	0

6.5.1.26 CAN_MVR (CAN Message Valid Register)

- Address = Base Address + 0x0148 Reset Value = 0x0000_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
CH32	CH31	CH30	CH29	CH28	CH27	CH26	CH25	CH24	CH23	CH22	CH21	CH20	CH19	CH18	CH17	CH16	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx	[x-1]	R	Message valid on channel x 0 = This message object is ignored by the message handler. 1 = This message object is configured and should be considered by the message handler.	0

6.5.1.27 CAN_TSTR (CAN Test Register)

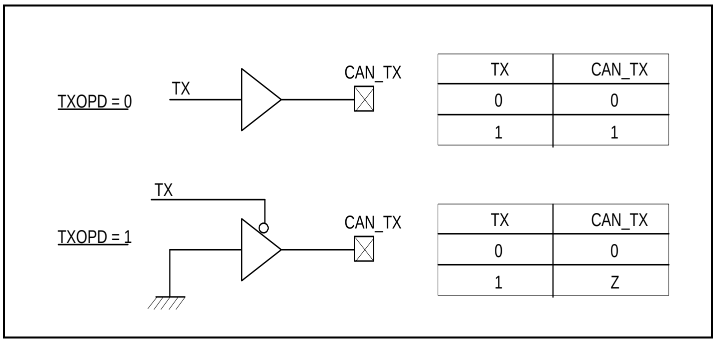
- Address = Base Address + 0x014C Reset Value = 0x0000_0000

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
TSTKEY																RSVD									RX	TXOPD	TX		LBACK	SILENT	BASIC
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
BASIC	[0]	RW	Basic mode 0 = Basic mode disabled. 1 = Interface 0 registers used as TX buffer, Interface 1 registers used as RX buffer. For further details, see "*Basic Mode*".	0
SILENT	[1]	RW	Silent mode 0 = Silent mode is disabled. 1 = The module is in silent mode.	0
LBACK	[2]	RW	Loop Back Mode 0 = Loop back mode is disabled. 1 = Loop back mode is enabled.	0
TX	[4:3]	RW	Control of CAN_TX pin. TX must be left in its default function (controlled by CAN core) when CAN message transfer or any of the test modes Loop Back Mode, Silent Mode, or Basic Mode are selected. The three test functions (monitoring, drive '0' and drive '1') interfere with all CAN protocol functions.	0

Table 6-12 Control of CAN_TX Pin

TX[1:0]	Control
00	CAN_TX is controlled by CAN core
01	Sample point is monitored at CAN_TX pin (CAN controller shall be enabled)
10	CAN_TX pin drives a dominant ('0') value
11	CAN_TX pin drives a recessive ('1') value

Name	Bit	Type	Description	Reset Value
TXOPD	[5]	RW	TX Open drain. 0 = Output CAN_TX pin is not configured as an open drain. 1 = Output CAN_TX pin is configured as an open drain. It allows to connect directly CAN controller without external transceiver or to drive properly a 5V transceiver with a 3.3V CAN controller with an additional external pull up. **NOTE:** When switching this bit, user should disable CAN in order to avoid to generate glitch on CAN_TX pin.  TXOPD = 0 TX CAN_TX TX CAN_TX 0 0 1 1 TX TXOPD = 1 CAN_TX TX CAN_TX 0 0 1 Z **Figure 6-27 Open Drain Mode**	0
RX	[6]	RW	Monitor the value of CAN_RX pin. 0 = The CAN bus is dominant (CAN_RX = '0') 1 = The CAN bus is recessive (CAN_RX = '1')	–
TSTKEY	[31:16]	RW	Test access key. Used only when writing CAN_TSTR. TSTKEY is read as 0. 0x0C75 = Write access in CAN_TSTR is allowed. Other value = Write access in CAN_TSTR is prohibited.	0

7 Clock & Power Manager

7.1 Overview

This chapter describes the management for system clock and power according to the operation mode.

The system clock tree consists of the following clock sources.

- EMCLK: External Main Clock, frequency selected by user, from 4 to 8 MHz
- IMCLK: Internal Main Clock, frequency selected by smart option (8, 16, or 20 MHz)
- ESCLK: External Sub-Clock, 32.768 KHz
- ISCLK: Internal Sub-Clock, 32.768 KHz

The clock control logic in S3FM02G can generate the required clock signals including HCLK/FCLK for CPU, HCLK for the AHB bus peripherals, and PCLK for the APB bus peripherals and so on. The S3FM02G has a Phase Locked Loop PLL for system clock.

The clock control logic can connect/disconnect the clock to each peripheral block by software, which will reduce the power consumption. For the power control logic, it has several power management schemes to keep optimal power consumption for a given task. The power management block can activate several modes.

7.1.1 Features

- Clock Management
  - External Oscillator 4 ~ 8 MHz (EMCLK: External Main Clock)
  - Programmable PLL with operational frequency from 8 to 75 MHz
  - Programmable clock divider (SDIV and PDIV) for SYSCLK and PCLK
  - External main and sub-clock failure detection with internal main and sub-clock (clock monitor function)
  - Clock Out Port (CLKOUT)

- Clock sources
  - EMCLK: External Main Clock, External 4 MHz ~ 8 MHz Oscillator
  - ESCLK: External Sub-Clock, External 32.768KHz Oscillator
  - IMCLK: Internal Main Clock, Internal 8/16/20 MHz RC oscillator.
  - ISCLK: Internal Sub-Clock, Internal 32.768KHz Ring oscillator
  - Four clock sources are enabled by default after reset

- SYSTEM PLL
  - Obtain an input clock from EMCLK (4 ~ 8 MHz)
  - Configurable output frequency from 8 MHz to 75 MHz
  - Configurable counter for PLL stabilization time

- Reset sources
  - NRST: External Input Pin nRESET Reset
  - EMCMRST: External Main Clock Monitor Fail Reset
  - ESCMRST: External Sub-Clock Monitor Fail Reset
  - LVDRST: LVD Reset (Called LVR)
  - WDTRST: Watchdog Timer Reset
  - SWRST: Software Reset
  - PORST: Power-On Reset
  - SYSRST: Reset by CPU request

- Power Management
  - NORMAL MODE: CPU runs by one of four clock sources
    - Main operating Mode: CPU runs by EMCLK or IMCLK.
    - Sub operating Mode: CPU runs by ESCLK or ISCLK.
  - HIGH-SPEED MODE: CPU runs by PLLCLK.
  - IDLE MODE: CPU halts operation.
    - HCLK stop.
    - STCLK or PCLK are configurable en/disable by S/W
    - Enter by Sleep command belongs to Cortex^{TM}-M3
    - Sub IDLE mode means all clock source are ISCLK or ESCLK.
  - STOP MODE: All stop except ISCLK. It can stop or not by S/W.
    - The other clocks (EMCLK, ESCLK, PLLCLK and IMCLK) stop.

7.2 Clock Management

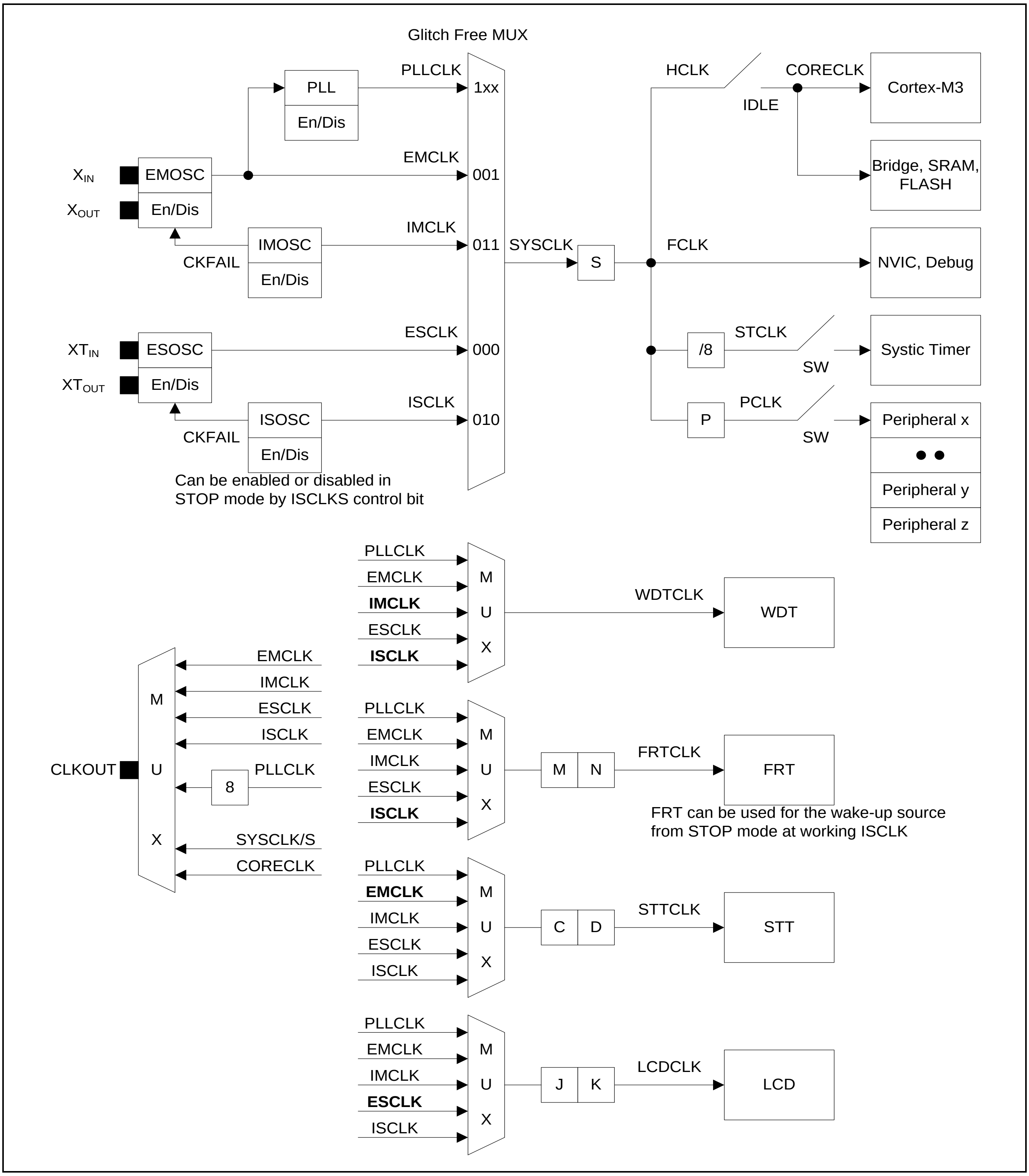


Figure 7-1 System Clock Tree Block Diagram

7.2.1 SYSCLK and Flash Smart Option

Reset status can be different by smart option. That includes SYSCLK, IMCLK and BTDIV.

Table 7-1 Summary of Smart Option for Clock Manager

SO_CSR	Smart Option	After Reset

[1:0]	POCCS[1:0] = 0
	POCCS[1:0] = 1
	POCCS[1:0] = 2
	POCCS[1:0] = 3
		SYSCLK = ESCLK
		SYSCLK = EMCLK
		SYSCLK = ISCLK
		SYSCLK = IMCLK

[7:6]	IMSEL[1:0] = 1
	IMSEL[1:0] = 2
	IMSEL[1:0] = 3
		IMCLK = 8MHz
		IMCLK = 16MHz
		IMCLK = 20MHz

[15:12]	BTDIV[3:0] = 3 BTDIV[3:0] = 4 BTDIV[3:0] = 5 ~ BTDIV[3:0] = 15	BT DIVIDER = 1 BT DIVIDER = 2 BT DIVIDER = 4 ~ BT DIVIDER = 4096
SO_CSR: Smart Option Configuration Status Register (In detail, refer to the PFLASH chapter)		

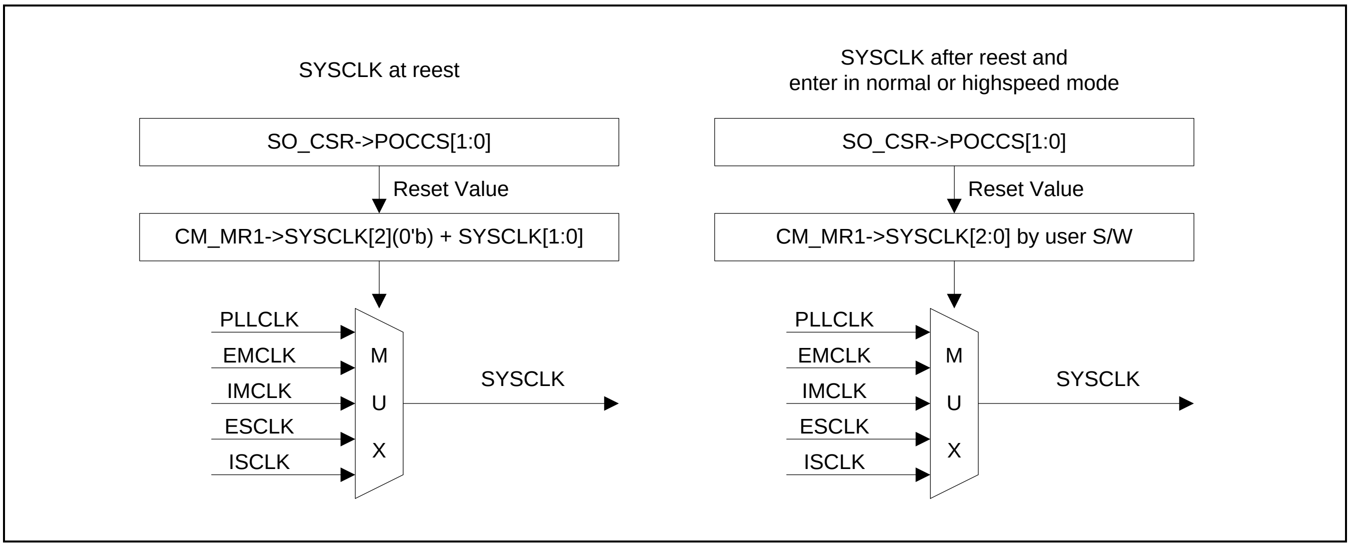


Figure 7-2 System Clock Selection

The following table shows clock sources for system operation.

Table 7-2 Clock Definition

Name	Definition	Description
EMCLK	External Main Clock or External Main Oscillator Clock	From 4MHz to 20MHz
IMCLK	Internal Main Clock or Internal Main Oscillator Clock	8, 16 or 20MHz
ESCLK	External Sub Clock or External Sub Oscillator Clock	32.768KHz
ISCLK	Internal Sub Clock or Internal Sub Oscillator Clock	32.768KHz
PLLCLK	PLL Output Clock	From 8MHz to 75MHz
SYSCLK	System Clock	From 32.768KHz to 75MHz
FCLK	Free Running Clock for Cortex-M3	SYSCLK/SDIV
CORECLK/HCLK	Cortex-M3 Clock and AHB Bus Peripherals Clock	SYSCLK/SDIV
STCLK	Sys-Tick Timer Clock in Cortex-M3	SYSCLK/SDIV/8
PCLK	Peripherals Clock	SYSCLK/SIDV/PDIV
WDTCLK	Watchdog Timer Clock	EMCLK, IMCLK, ESCLK, ISCLK or PLLCLK
LCDCLK	LCD Clock	EMCLK, IMCLK, ESCLK, ISCLK or PLLCLK
FRTCLK	Free Running Timer Clock	EMCLK, IMCLK, ESCLK, ISCLK or PLLCLK
STTCLK	Stamp Timer Clock	EMCLK, IMCLK, ESCLK, ISCLK or PLLCLK

Table 7-3 Clock Status at Reset and Wakeup

Clock	Reset	Wake-Up from STOP
EMCLK	RUN	Status before STOP
IMCLK	RUN	If FWAKE is 0, status before STOP If FWAKE is 1, RUN
ESCLK	RUN	Status before STOP
ISCLK	RUN	Status before STOP
PLLCLK	STOP	Status before STOP
SYSCLK	RUN	Status before STOP
FCLK	RUN	RUN
HCLK	RUN	RUN
STCLK	STOP	Status before STOP
PCLK	RUN	Status before STOP
WDTCLK/LCDCLK/FRTCLK	RUN	Status before STOP
STTCLK	STOP	Status before STOP

7.2.2 Clock Control State Machine

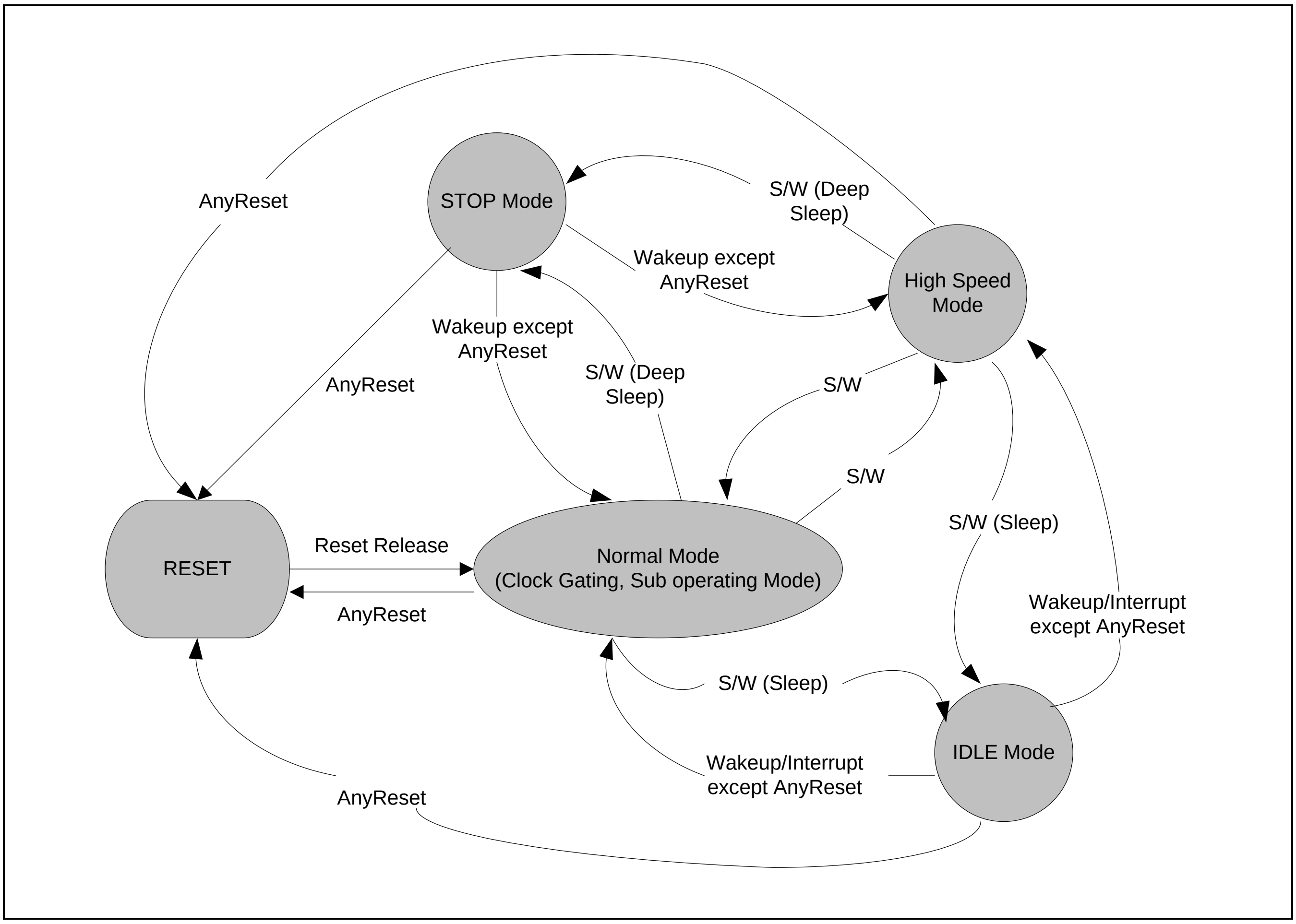


Figure 7-3 Clock Control State Machine

S3FM02G defines power modes (called operation mode) by SYSCLK. Other clocks (WDTCLK, LCDCLK, FRTCLK, STTCLK) are optional. Each mode has different states as following table.

Table 7-4 Operation Mode Definition

MODE	CORECLK	Always	Optional Alive/Dead Clock
NORMAL	One of four clock sources (NOTE)	Alive HCLK, FCLK	EMCLK, IMCLK, ESCLK, ISCLK, STCLK, PCLK
HIGH-SPEED	PLLCLK	Alive EMCLK before PLL enable, FCLK, HCLK	IMCLK, ESCLK, ISCLK, STCLK, PCLK
IDLE	(NOTE)	Alive FCLK Dead HCLK	EMCLK, IMCLK, ESCLK, ISCLK, STCLK, PCLK, PLLCLK
STOP	(NOTE)	Dead HCLK, FCLK, PCLK, STCLK, PLLCLK, EMCLK, ESCLK, IMCLK	ISCLK

NOTE: Four clock sources include EMCLK, ESCLK, IMCLK, and ISCLK
Core (Cortex-M3) can't take CORECLK because IDLE and STOP make CORECLK disconnect.

7.2.3 EMCLK

EMCLK means an external main clock. The frequency range of external clock oscillator is allowed from 4 MHz to 8 MHz.

When the resonators and load capacitors are used as the external oscillator, they have to be placed as close to the chip as possible in order to have the stable clock and minimize the stabilization time. The value of load capacitor should be chosen according to the external oscillator.

EMCLK can be used for the SYSCLK or PLL input clock source.

After the external oscillator is enabled and stabilized, it is required to configure CM_MR1 register in order to supply SYSCLK after checking EMCLK bit in CM_SR register. When the operation mode transition is completed, the STABLE bit in CM_SR is set to ‘1’.

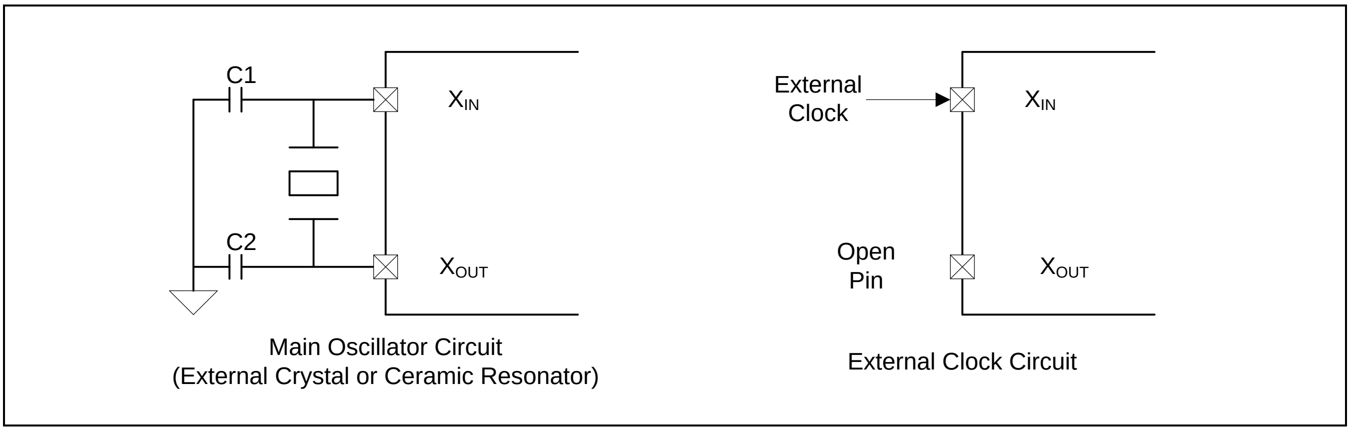


Figure 7-4 Crystal/Ceramic Resonator/External Clock Operation

7.2.3.1 Reset

After any reset, EMCLK is enabled by default and can be used for SYSCLK by POCCS bit of SO_CSR (Refer to the Program Flash). After reset, it can be disabled or enabled by controlling the EMCLK bit in CM_CCR/CM_CSR register.

7.2.3.2 Clock Monitor

A clock monitor function is to monitor the availability of external main oscillator by means of IMCLK. When any reset is generated, IMCLK is enabled and also a clock monitor function is enabled. If SYSCLK is EMCLK, clock monitor function is enabled and clock fail is detected, EMCMRST will be occurred and SYSCLK will be IMCLK, not EMCLK.

7.2.4 ESCLK

ESCLK means an external sub clock 32.768 KHz. ESCLK can be used for SYSCLK for specific case defined by user (Called Sub Operating Mode). When the operation mode transition is completed, the STABLE bit in CM_SR is set to ‘1’. It is not impossible to use as PLL input clock source.

7.2.4.1 Reset

After any reset, ESCLK is enabled by default and can be used for SYSCLK by POCCS bit of SO_CSR (Refer to the Program Flash). After reset, it can be disabled or enabled by controlling the ESCLK bit in CM_CCR/CM_CSR register.

7.2.4.2 Clock Monitor

A clock monitor function is to monitor the availability of external sub oscillator by means of ISCLK. If SYSCLK is ESCLK, clock monitor function is enabled and clock fail is detected, ESCMRST will be occurred and SYSCLK will be ISCLK, not ESCLK.

7.2.5 IMCLK

IMCLK means an internal main clock 8/16/20MHz.

It supplies the clock to the SYSCLK when the SYSCLK field in CM_MR1 register is ‘011’b’. When the operation mode transition is completed, the STABLE bit in CM_SR is set to ‘1’. It should also run as the SYSCLK when the external oscillator clock is determined as clock fail condition by the clock monitor function.

IMCLK cannot be used for PLL input clock source.

7.2.5.1 Reset

After any reset, IMCLK is enabled by default and can be used for SYSCLK by POCCS bit of SO_CSR (Refer to the Program Flash). After reset, it can be disabled or enabled by controlling the IMCLK bit in CM_CCR/CM_CSR register.

7.2.5.2 Clock Monitor

This clock is used the reference clock for external main clock monitor function. If user wants to use the clock monitor function of EMCLK, IMCLK is enabled and also a clock monitor function is enabled.

7.2.6 ISCLK

ISCLK means an internal sub- clock 32.768 KHz. ISCLK can be used for SYSCLK for specific case defined by user (Called Sub Operating Mode). When the operation mode transition is completed, the STABLE bit in CM_SR is set to ‘1’. It is not impossible to use as PLL input clock source.

The internal oscillator can be used for a watch dog timer, free running timer as a clock source.

ISCLK can be optionally enabled or not by controlling ISCLKS bin in CM_CSR/CM_CCR register in STOP mode.

7.2.6.1 Reset

After any reset, ISCLK is enabled by default and can be used for SYSCLK (main clock) by POCCS bit of SO_CSR (Refer to the Program Flash). After reset, it can be disabled or enabled by controlling the ISCLK bit in CM_CCR/CM_CSR register.

7.2.6.2 Clock Monitor

It is used the reference clock for external sub-clock monitor function. If user wants to use the clock monitor function of ESCLK, ISCLK is enabled and also a clock monitor function is enabled.

7.2.7 PLL (Phase Locked Loop)

The PLL multiplies the EMCLK and provides the clock to SYSCLK. The input range of source is from 4MHz to 8MHz and the PLL amplifies the source clock to PLLCLK. The PLL configuration must be finished before the PLL is enabled by writing a '1' to PLL bit in CM_CSR register. In addition, the input clock (EMCLK) must be stabilized before PLL is on. After the PLL is enabled and stabilized, it is allowed to configure CM_MR register in order to supply the PLL clock to SYSCLK. When the clock transition is completed, the STABLE bit in CM_SR is set to '1'.

The output of PLL can be obtained as below.

F_{IN} = Input frequency of PLL = EMCLK	PLL multipliers = M[7:0] (1 ~ 255)	F_{OUT} = Output frequency of PLL = PLLCLK = ((M+8) x Fin) / ((P+2)x(2^S)
	PLL pre-divider = P[5:0] (1 ~ 63)	
	PLL post-scaler = S[1:0] (0 ~ 3)	

After reset, PLL is disabled by default.

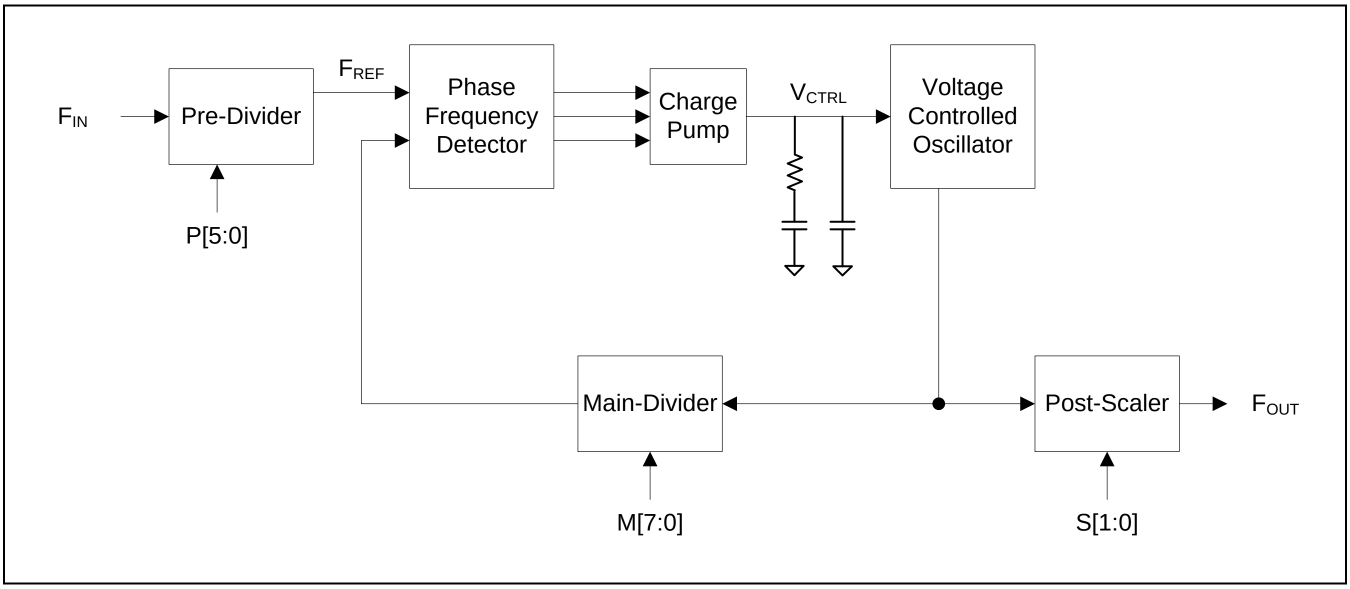


Figure 7-5 PLL (Phase-Locked Loop) Block Diagram

The PLL within the clock generator is the circuit that synchronizes the output signal with a reference or input signal in frequency as well as in phase. It is composed of the voltage controlled oscillator to generate the output frequency, the divider P to divide the reference frequency by p, the divider M to divide the VCO output frequency by m, the divider S to divide the VCO output frequency, the phase detector, charge pump, and loop filter. The output clock frequency F_{OUT} is related to the reference input clock frequency F_{IN} by the following equation.

```
FOUT = (m * FIN) / (p * (2^S))
m = M (the value for divider M) + 8, p = P (the value for divider P) + 2
```

Phase Frequency Detector

The phase detector monitors the phase difference between the F_{REF} (the reference frequency) and F_{VCO} (the feedback frequency), and generates a control signal when it detects difference between the two.

Charge Pump

The charge pump converts the phase frequency detector control signal to a charge in voltage across the internal filter that drives the VCO.

Voltage Controlled Oscillator (VCO)

The output voltage from the loop filter drives the VCO, causing its oscillation frequency to increase or decrease as a function of variations in voltage. When the VCO output matches the system clock in frequency and phase, the phase detector stops sending a control signal to the charge pump, which in turn stabilizes the input voltage to the loop filter. The VCO frequency then remains constant, and the PLL remains locked onto the system clock.

PLL VALUE CHANGE STEPS

If the PLL setting needs to be changed when F_{OUT} (PLLCLK) is used as SYSCLK, the PLL transition noise may be asserted to system. So the PLL configuration has to be changed in NORMAL mode. Do the following steps to change the PLL configuration.

After reset

1. Enable an external main oscillator (EMCLK) by controlling EMCLK bit in CM_CSR register.
2. Check an external main stable by monitoring EMCLK bit in CM_SR register.
3. Set PLL stabilization time to prevent abnormal operation.
4. Change PMS value in CM_PDPR register
5. Enable PLL by controlling PLL bit in CM_CSR register
6. Check a PLL stable by monitoring PLL bit in CM_SR register
7. If PLL is stable, set SYSCLK fields in CM_MR1 register, and then SYSCLK is fed from PLLCLK.

PLL configuration change in the high speed mode

1. Change system clock to external main oscillator(EMCLK)
2. PLL disable by setting PLL bit in CM_CCR
3. Change PMS value in CM_PDPR register
4. Enable PLL by controlling PLL bit in CM_CSR register
5. Check a PLL stable by monitoring PLL bit in CM_SR register
6. If PLL is stable, set SYSCLK fields in CM_MR1 register, and then SYSCLK is fed from PLLCLK.

7.2.8 SYSCLK Source Clock Selection

System clock (SYSCLK) input source can use PLLCLK, EMCLK, IMCLK, ESCLK or ISCLK as its clock source by setting SYSCLK fields in CM_MR register only after stable. Otherwise, it might bring the chip into unexpected status of operation. Also it might generate command error.

Following figure shows SYSCLK change timing diagram. Other case is same operation

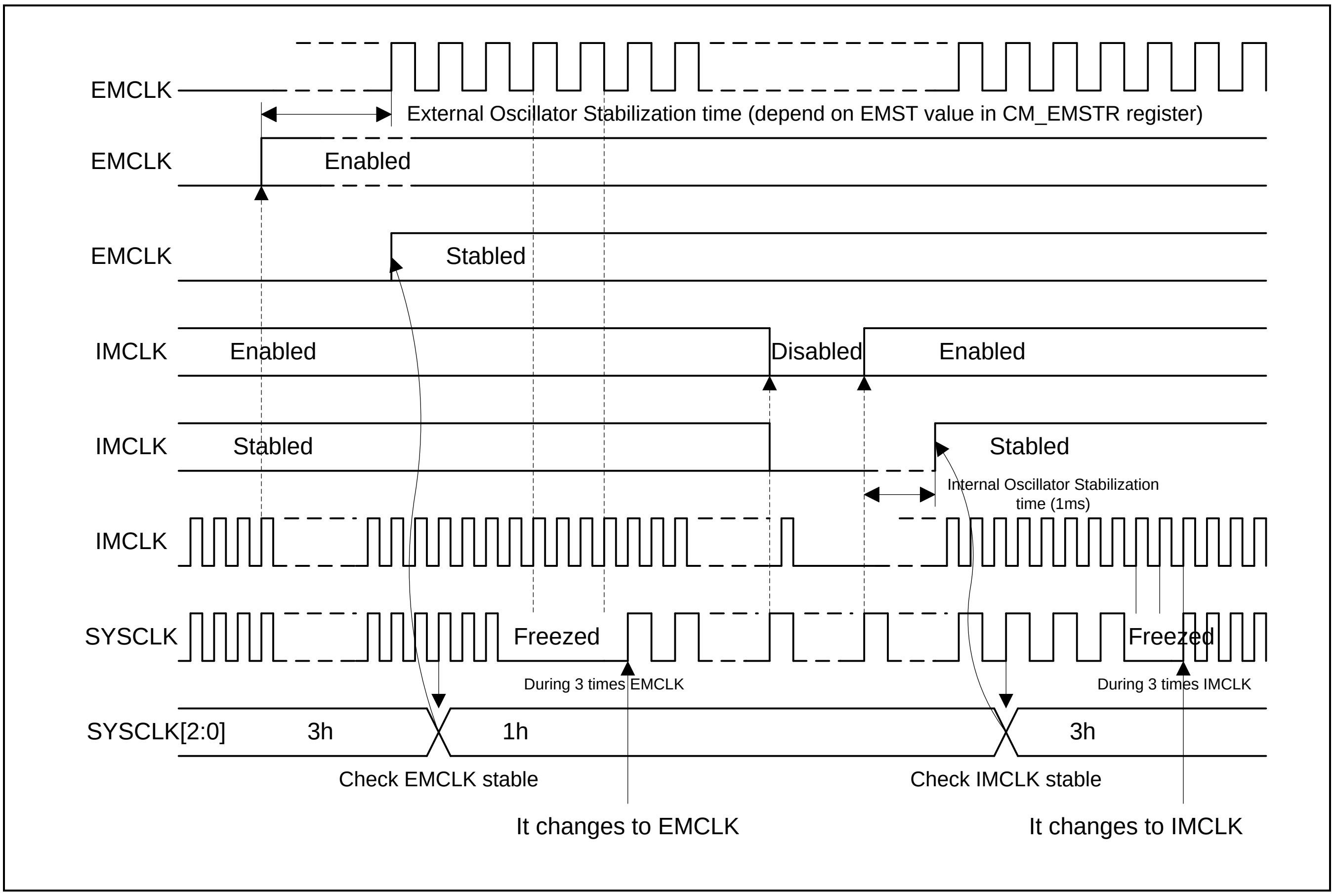


Figure 7-6 The case that changes clock source for SYSCLK

7.2.9 Clock Monitor

The clock monitor is a function to monitor the availability of external main clock or external sub clock oscillator. When the clock monitoring is enabled, the internal main or sub clock oscillator must be also enabled. Clock monitor has clock monitor function enable/disable control bit, clock fail detection flag, clock recovery flag, and clock monitor reset control.

Table 7-5 Clock Monitor Function

SYSCLK before Clock Fail	Condition	After Clock Fail
EMCLK or PLLCLK	EMCLK or EMCLK/PLL enable IMCLK enable EMCM/EMCMRST enable	EMCKFAIL bit in CM_SR register is set to '1'. Chip reset by EMCMRST will occur. After reset, SYSCLK is IMCLK, not EMCLK (or PLLCLK)
	EMCLK or EMCLK/PLL enable IMCLK enable EMCM enable EMCMRST disable	EMCKFAIL bit in CM_SR register is set to '1'. SYSCLK is the same before a clock fail detection. The issue of SYSCLK fail might bring the chip into unexpected status of operation.
IMCLK	EMCLK/IMCLK enable EMCM/EMCMRST enable	EMCKFAIL bit in CM_SR register is set to '1'. Chip reset by EMCMRST will occur. After reset, SYSCLK is IMCLK.
	EMCLK/IMCLK enable EMCM enable EMCMRST disable	EMCKFAIL bit in CM_SR register is set to '1'. SYSCLK is the same before a clock fail detection. Although EMCLK failed, SYSCLK doesn't have any problem. So chip runs expect parts to use EMCLK.
ESCLK	ESCLK/ISCLK enable ESCM/ESCMRST enable	ESCKFAIL bit in CM_SR register is set to '1'. Chip reset by ESCMRST will occur After reset, SYSCLK is ISCLK, not ESCLK
	ESCLK/ISCLK enable ESCM enable ESCMRST disable	ESCKFAIL bit in CM_SR register is set to '1'. SYSCLK is the same before a clock fail detection. The issue of SYSCLK fail might bring the chip into unexpected status of operation.
ISCLK	ESCLK enable ISCLK enable ESCM/ESCMRST enable	ESCKFAIL bit in CM_SR register is set to '1'. Chip reset by ESCMRST will occur. After reset, SYSCLK is ISCLK
	ESCLK enable ISCLK enable ESCM enable ESCMRST disable	ESCKFAIL bit in CM_SR register is set to '1'. SYSCLK is the same before a clock fail detection. Although ESCLK failed, SYSCLK doesn't have any problem. So chip runs expect parts to use ESCLK.

Table 7-6 Clock Monitor Control

Clock Monitor	External Main Clock	External Sub-Clock
Clock Monitor Fail Function Enable/Disable	EMCM	ESCM
Clock Monitor Fail Reset Function Enable/Disable	EMCMRST	ESCMRST
Clock Fail Detection Flag	EMCKFAIL	ESCKFAIL
Clock Recovery Flag	EMCKFAIL_END	ESCKFAIL_END

7.2.9.1 External Main Clock Monitor

The external main clock oscillator is sampled by the internal main clock (IMCLK) with 25-divider. If the sampled value is identical during three consecutive times, the external oscillator failure is detected. After detect, the operation is EMCMRST or EMCKFAIL status update.

If SYSCLK is EMCLK or PLLCLK, user can make reset when clock failure is detected. After reset by EMCMRST, system runs by IMCLK (internal main clock) and the reset status register is updated by hardware.

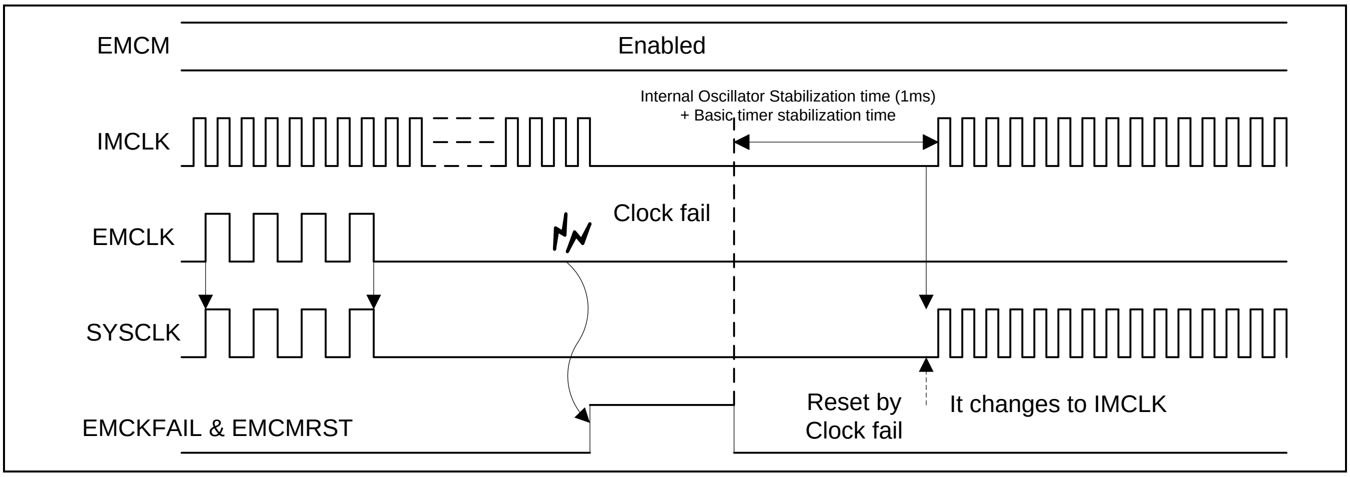


Figure 7-7 External Main Oscillator Fail with Reset

7.2.9.2 External Sub Clock Monitor

In case of external sub clock oscillator when sampled by the internal sub clock (ISCLK) with 25-divider. If the sampled value is identical during three consecutive times, the external oscillator failure is detected. After detect, the operation is ESCMRST or ESCKFAIL status update.

If SYSCLK is ESCLK, user can make reset when clock failure is detected. After reset by ESCMRST, system runs by ISCLK (internal sub clock) and the reset status register is updated by hardware.

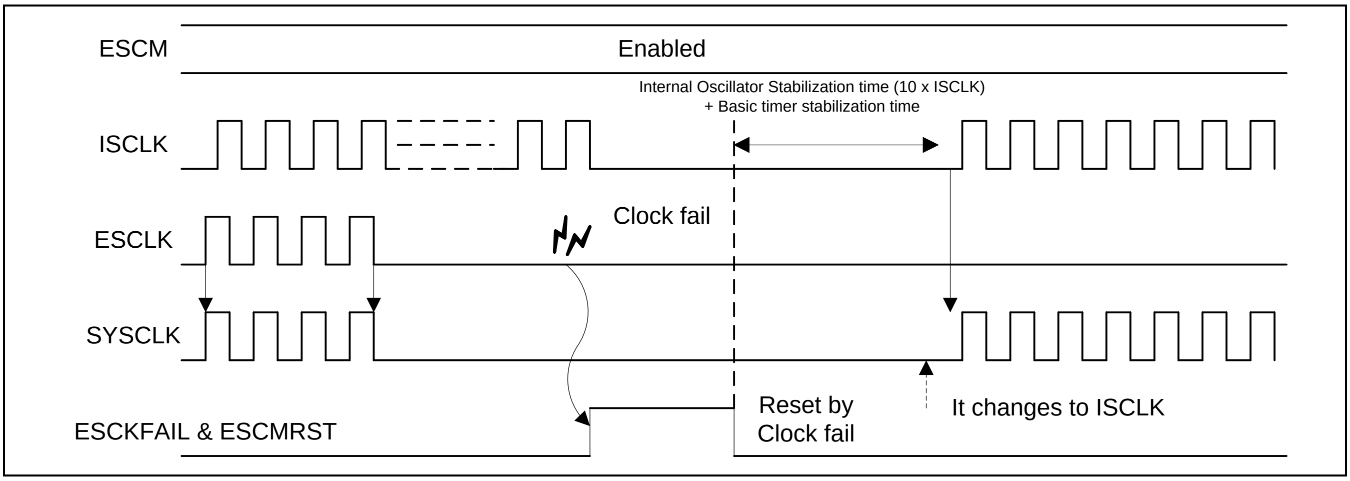


Figure 7-8 External Sub Oscillator Fail with Reset

7.2.9.3 End of Clock Fail

The clock monitor function should be used after the external main or sub oscillator is enabled and stabilized.

The EMCKFAIL_END/ESCMFAIL_END interrupt indicates the end of external main or sub oscillator clock failure. (To check the end of external main or sub oscillator fail, the external oscillator (main, sub) and IMCLK, ISCLK shall be first enabled, and the clock monitor function shall be enabled.)

When the failure disappears (i.e. EMCKFAIL_END/ESCMFAIL_END bit in CM_SR register is set to ‘1’ by edge recognition of the external oscillator by HW), the EMCKFAIL/ESCKFAIL bit will be cleared automatically.

The software is able to use the external oscillator again as the source clock. Before switching to EMCLK or ESCLK, the software must refer to following recommendations:

- Check EMCLK/ESCLK bit in CM_SR register
- Select the EMCLK or ESCLK by configuring SYSCLK fields in CM_MR1 register (Make sure if the external main or sub oscillator is stabilized)

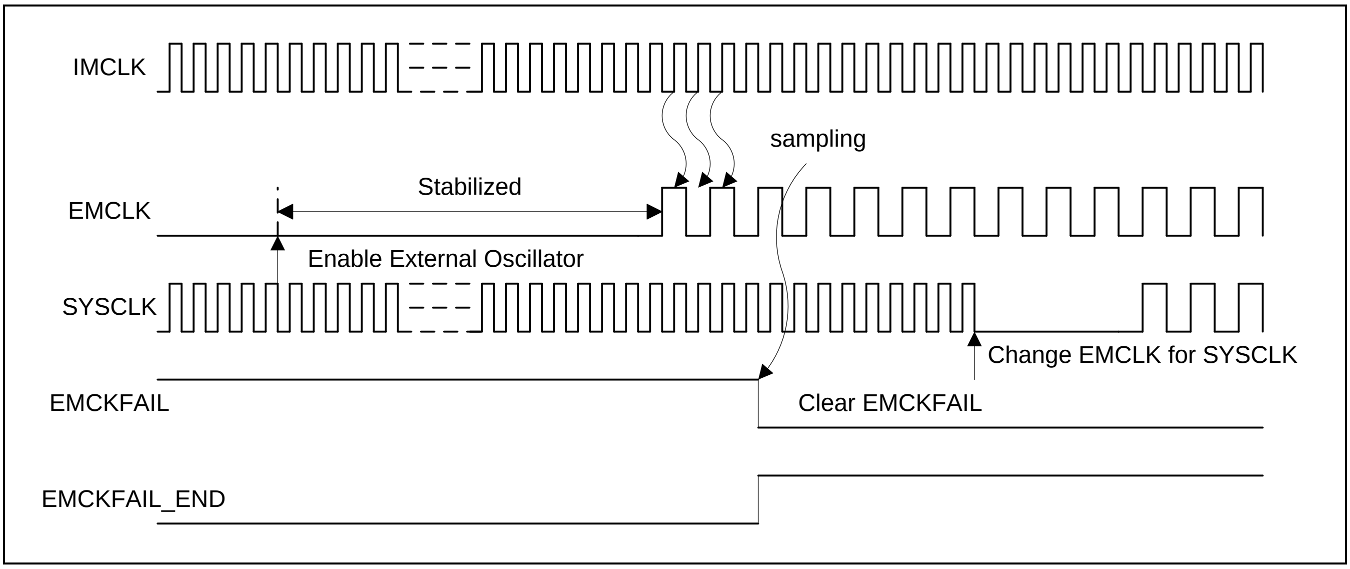


Figure 7-9 End of Clock Fail

Upon entry of STOP mode, the clock monitor function doesn’t be affected by HW. After wake-up from stop, the clock monitor function stays the state, enable or disable before the entry of STOP.

7.3 Power Management

After reset, the S3FM02G is in NORMAL mode. Several a low power modes are available to save power when does not need to be kept running, for example when waiting for an external event.

7.3.1 Power Modes

NORMAL mode is used to supply one of four clock sources (EMCLK, IMCLK, ESCLK, or ISCLK) to CPU as well as all peripherals. The power consumption will be increased when all peripherals are enabled. Both ON and OFF of clock gating of the individual clock source of each peripheral are performed by controlling each corresponding clock source enable bit of CM_PCSR0 and CM_PCSR1 registers. If SYSCLK is operated by ESCLK or ISCLK, it is called as sub operating mode.

HIGH-SPEED mode is used to supply PLLCLK to CPU as well as all peripherals.

IDLE mode is one of the low power modes. In IDLE mode disconnecting the clock to CPU halts the operation and some peripherals remain active by software.

STOP mode makes all logic stop basically. But ISCLK can be run by software. As the low power mode, the power consumption can be the lowest. The wake-up from STOP mode can be done by activating external interrupt, USARTRXn, SSPRXn, CANRXn or a chip reset. STOP mode can enter from normal or high-speed mode.

7.3.2 Low Power Mode and Wake-Up

There are two low-power modes that are IDLE and STOP mode. The watchdog and LVD can be disabled when it is not necessary in low power modes. It also reduces the current consumption.

- WFE: The wait for event instruction, WFE, causes entry to IDLE/STOP mode conditional on the value of an one-bit event register. When the processor executes a WFE instruction, it checks this register:
  - If the register is 0 the processor stops executing instructions and enters IDLE/STOP mode
  - If the register is 1 the processor clears the register to 0 and continues executing instructions without entering IDLE/STOP mode

- WFI: The wait for interrupt instruction, WFI, causes immediate entry to IDLE/STOP mode. When the processor executes a WFI instruction stops executing instructions and enters IDLE/STOP mode.

7.3.2.1 Enter IDLE Mode

The Idle mode is applicable when wake-up with the minimum latency is required. The clock supplied to Cortex-M3 is disconnected. Two options are available to select the IDLE mode entry mechanism, depending on the SLEEPONEXIT bit in Cortex-M3 System Control Register.

- Entry condition:
  - IDLE (sleep-now): If the SLEEPONEXIT bit of System Control Register in Cortex-M3 is cleared, the S3FM02G enters IDLE mode as soon as WFI or WFE instruction is executed.
  - IDLE (sleep-on-exit): If the SLEEPONEXIT bit of System Control Register in Cortex-M3 is set, the S3FM02G enters IDLE mode as soon as it exits the lowest priority ISR.

- Entry sequence:
  - Configure wakeup source
  - Clock source configuration
  - Oscillator and PLL configuration
  - If user wants to use IDLEW bit , copy NVIC Interrupt Set-Enable Registers into CM_NISRx
  - Execute entry condition

The sleep-on-exit bit (SLEEPONEXIT) is located in System Control Register of Cortex-M3 NVIC. If it is set, when the processor completes the execution of an exception handler it returns to Thread mode and immediately enters into the IDLE mode. Use this mechanism in applications that only require the processor to run when an exception occurs.

When IMCLK is only used without the external oscillator, the accuracy-critical code and PLL should not run just after exit of IDLE mode. The processor should wait till EMCLK is enabled and stabilized in this case.

By configuring PCLK in CM_CSR register, it is possible to disconnect the clock supply to peripherals. Thus, the current can be further optimized.

7.3.2.2 Exit IDLE Mode

If the WFI instruction is used to enter IDLE mode, any peripheral interrupt acknowledge by the nested vectored interrupt controller (NVIC) can wake up the device from IDLE mode.

If the WFE instruction is used to enter IDLE mode, the S3FM02G exits IDLE mode as soon as an event occurs.

Table 7-7 IDLE on Sleep-now

Sleep-now mode	Description
Mode Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: • SLEEPDEEP = 0 and • SLEEPONEXIT = 0 Refer to Cortex-M3 the System Control Register
Mode exit	If WFI was used for entry: Interrupt If WFE was used for entry: Wakeup event
Wakeup latency	None

Table 7-8 IDLE on Sleep-on-Exit

Sleep-on-exit mode	Description
Mode Entry	WFI (Wait for Interrupt) while: • SLEEPDEEP = 0 and • SLEEPONEXIT = 1 Refer to Cortex-M3 the System Control Register
Mode exit	Interrupt
Wakeup latency	None

7.3.2.3 Enter STOP Mode

The STOP mode is based upon the SLEEPDEEP in Cortex-M3 core. The all clocks (EMCLK, IMCLK, ESCLK, and ISCLK) stop operating. However, the status of internal 32.768KHz oscillator (ISCLK) are configurable. The clock supply to SYSCLK is disconnected in entry of STOP mode and connected in exit of STOP mode.

- Entry condition
  - STOP (sleep-now) : If the SLEEPDEEP bit of System Control Register in Cortex-M3 is set, the S3FM02G enters STOP mode as soon as WFI or WFE instruction is executed
  - STOP (sleep-on-exit): If the SLEEPONEXIT and SLEEPDEEP bits of System Control Register in Cortex-M3 are set, the S3FM02G enters STOP mode as soon as it exits the lowest priority ISR.

- Entry sequence
  - Configure wakeup sources
  - Clock source configuration
  - Oscillator and PLL configuration.
  - If user wants to use IDLEW bit, copy NVIC Interrupt Set-Enable Registers into CM_NISRx.
  - Enable internal 32.768KHz oscillator(ISCLK) when it is necessary
  - Execute entry condition

The sleep-on-exit bit (SLEEPONEXIT) is located in System Control Register of Cortex-M3 NVIC. If it is set, when the processor completes the execution of an exception handler it returns to Thread mode and immediately enters into the STOP mode.

7.3.2.4 Exit STOP Mode

Exit STOP mode by issuing an interrupt or a wakeup event.

The clock source of SYSCLK upon exit of Stop mode can differ by condition before entering STOP mode.
Wakeup from IDLE / STOP Mode

Before the entry of IDLE or STOP mode, executing WFE or WFI allows the processor to wakeup by an event or interrupts. As described in previous sections, the interrupt which is used for wakeup source should be enabled before the entry of IDLE or STOP mode.

Table 7-9 Clock Status on STOP and Wakeup

–	Normal or High Speed Mode	STOP	Wake-Up
EMCLK	RUN	STOP	RUN
	STOP	STOP	STOP
IMCLK	RUN	STOP	RUN
	STOP	STOP	STOP @FWAKE=0 RUN @FWAKE=1
ESCLK	RUN	STOP	RUN
	STOP	STOP	STOP
ISCLK	RUN	STOP	RUN
	STOP	STOP	STOP

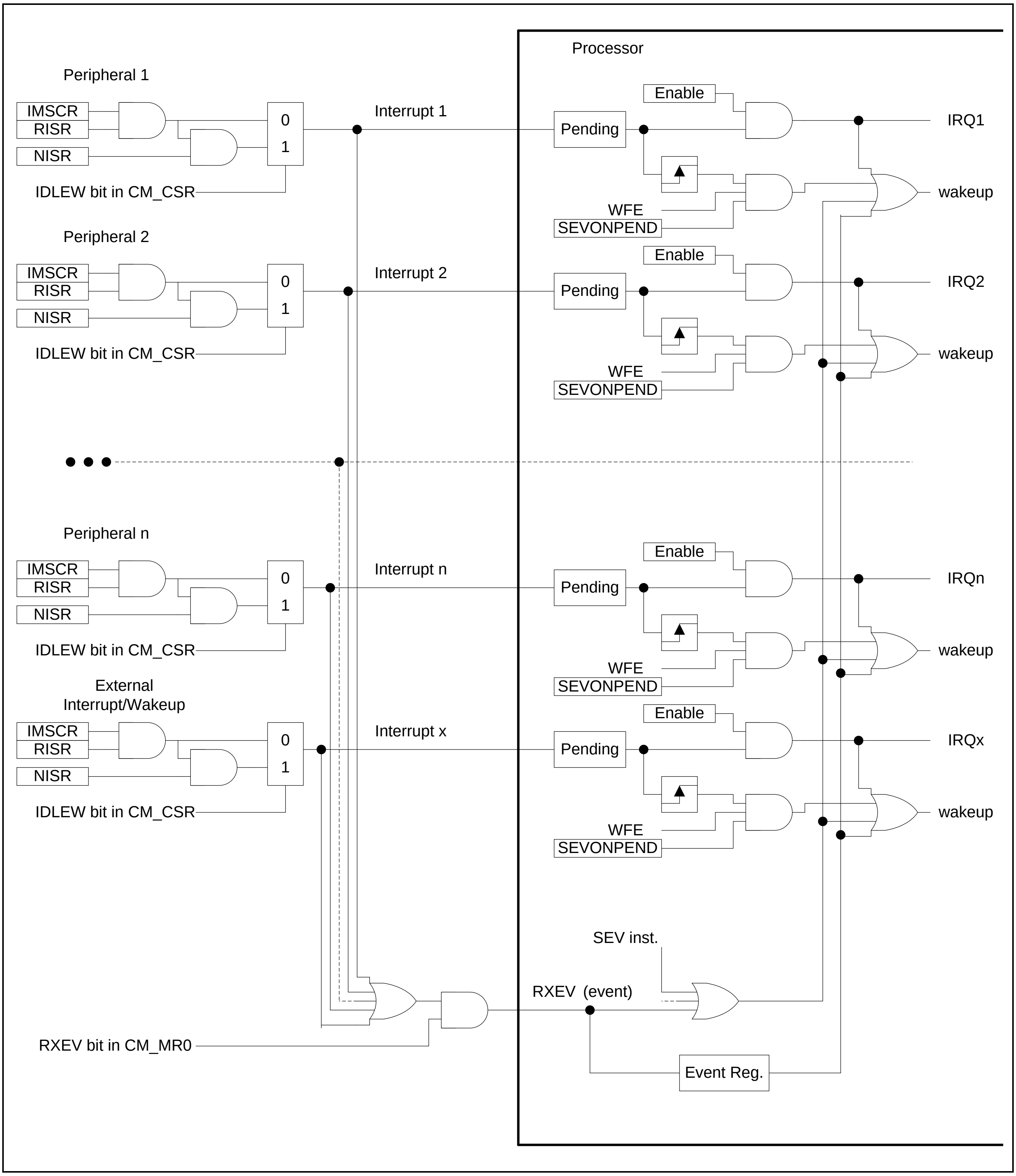


Figure 7-10 Interrupt and Event

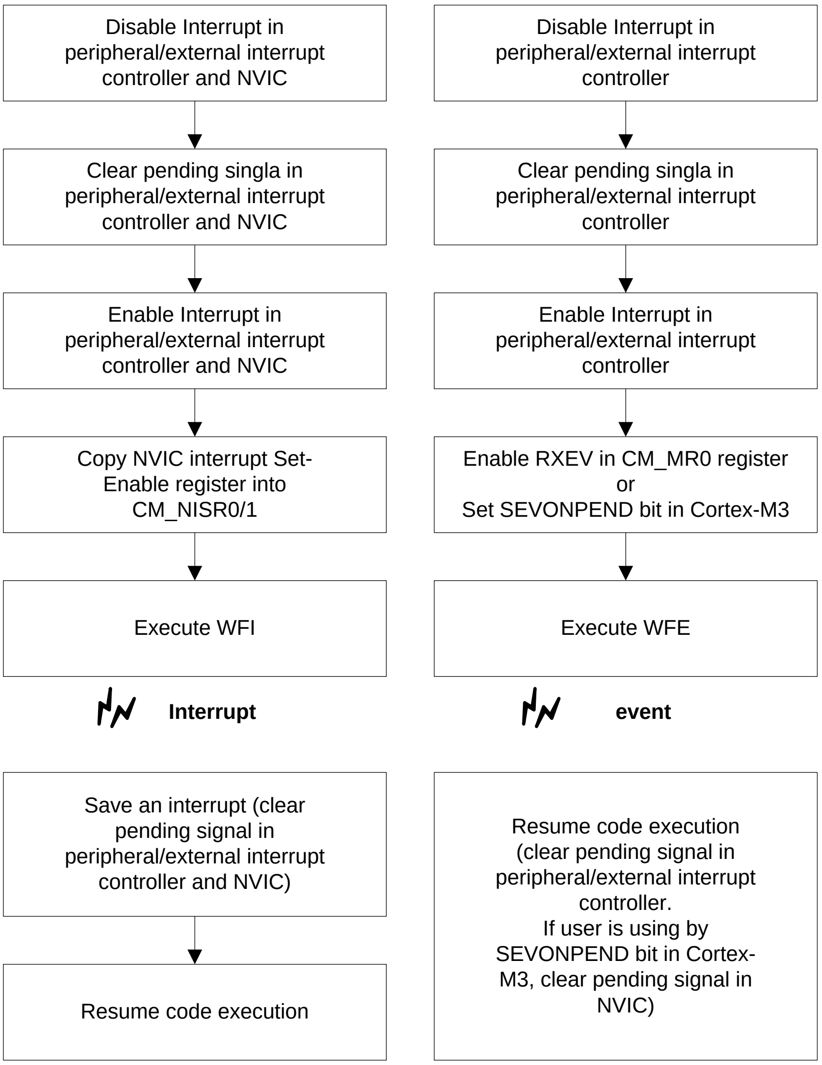

Disable Interrupt in peripheral/external interrupt controller and NVIC
Clear pending singla in peripheral/external interrupt controller and NVIC
Enable Interrupt in peripheral/external interrupt controller and NVIC
Copy NVIC interrupt Set-Enable register into CM_NISR0/1
Execute WFI
Interrupt
Save an interrupt (clear pending signal in peripheral/external interrupt controller and NVIC)
Resume code execution
Disable Interrupt in peripheral/external interrupt controller
Clear pending singla in peripheral/external interrupt controller
Enable Interrupt in peripheral/external interrupt controller
Enable RXEV in CM_MR0 register or Set SEVONPEND bit in Cortex-M3
Execute WFE
event
Resume code execution (clear pending signal in peripheral/external interrupt controller. If user is using by SEVONPEND bit in Cortex-M3, clear pending signal in NVIC)

Figure 7-11 Different Handling Process for Interrupt and Event in IDLE or STOP Mode

7.4 Reset Management

The S3FM02G has eight types of resets and reset controller can place the system the predefined states.

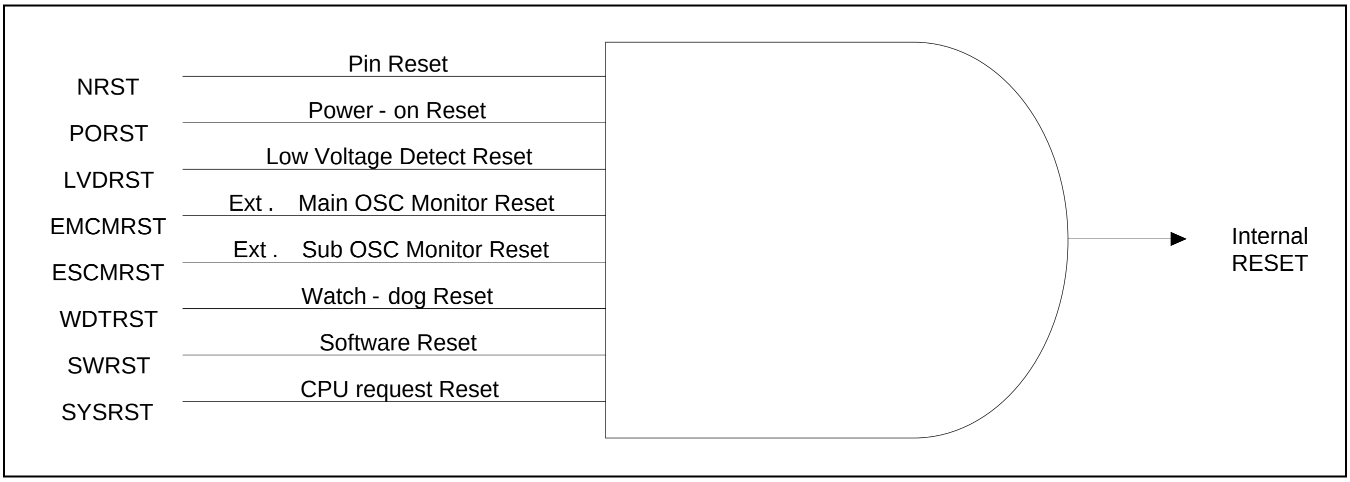


Figure 7-12 RESET Sources

Clock Manager has status bits related to the reset logging. CM_SR (Status Register) includes the information of reset logging ID from bit.24 to bit.31. When reset occurs, each source triggered a reset updates the status of each reset ID. All reset ID flags can be cleared by user (S/W). Reset status bits except PORRSTS and LVDRSTS can be cleared by other resets (H/W).

Table 7-10 Reset ID Flag

Reset Status Bit	CM_SR	Description
SWRSTS	[24]	The last reset ware generated by software (register control)
NRSTS	[25]	The last reset ware generated by a signal asserted via nRESET pin
LVDRSTS	[26]	The last reset ware generated by a low voltage detection
WDTRSTS	[27]	The last reset ware generated by a watchdog timer
PORRSTS	[28]	The last reset ware generated by power-on
ESCMRSTS	[29]	The last reset ware generated by a clock monitor function for EMCLK
EMCMRSTS	[30]	The last reset ware generated by a clock monitor function for ESCLK
SYSRSTS	[31]	The last reset ware generated by CPU request

7.4.1 Pin Reset (NRST)

Pin Reset is invoked when the nRESET pin is asserted and all units in the system are initialized to known states. When the unmaskable nRESET pin is asserted as "Low", the internal hardware reset signal is generated. Upon assertion of nRESET, the S3FM02G enters into reset state regardless of the previous state. After reset by nRESET, NRSTS bit in CM_SR register sets to '1'. This bit can be cleared by user (S/W) or other resets (H/W).

7.4.2 Power-on Reset (PORST)

The power-on reset circuit is built on S3FM02G. When power is initially applied to the S3FM02G, or when V_{DD} drops below the V_{POR}, the POR circuit holds the S3FM02G in reset until V_{DD} has risen above the V_{LVR} level (LVD reset level). After power-on reset, PORRSTS bit in CM_SR register sets to '1'. Although a reset by other reset sources occurs, the status (1 or 0) of this bit isn't cleared and remained. This bit can cleared by user only.

7.4.3 LVD Reset (LVDRST)

LVD reset is enabled by default at reset (see LVDRSTEN bit in CM_MR0 register). After reset, LVD reset can be configured by LVDRST bit in CM_SR register. The level of LVD reset is also configurable by LVDRL[2:0] fields in CM_MR0 register. After LVD reset, LVDRSTS bit in CM_SR register sets to '1'. Although a reset by other reset sources occurs, the status (1 or 0) of this bit isn't cleared and remained. This bit can be cleared by user (S/W) only.

7.4.3.1 LVD Interrupt

LVD interrupts can be enabled / disabled by controlling LVDINTEN bit in CM_MR0 register. And LVD interrupt can be configured by LVDINT bit in CM_IMSCR register. The LVD interrupt pending bit is asserted when the LVD interrupt flag bit (LVDINT) in CM_MISR is set to '1'. The level of LVD interrupt is configurable by LVDIL[2:0] fields in CM_MR0 register.

The pending bit (LVDINT) in CM_MISR should be cleared before LVD interrupt in NVIC is enabled in order to prevent unexpected interrupt. The pending interrupt must be cleared by first clearing the status bit (LVDINT) in CM_ICR and corresponding pending bit in NVIC.

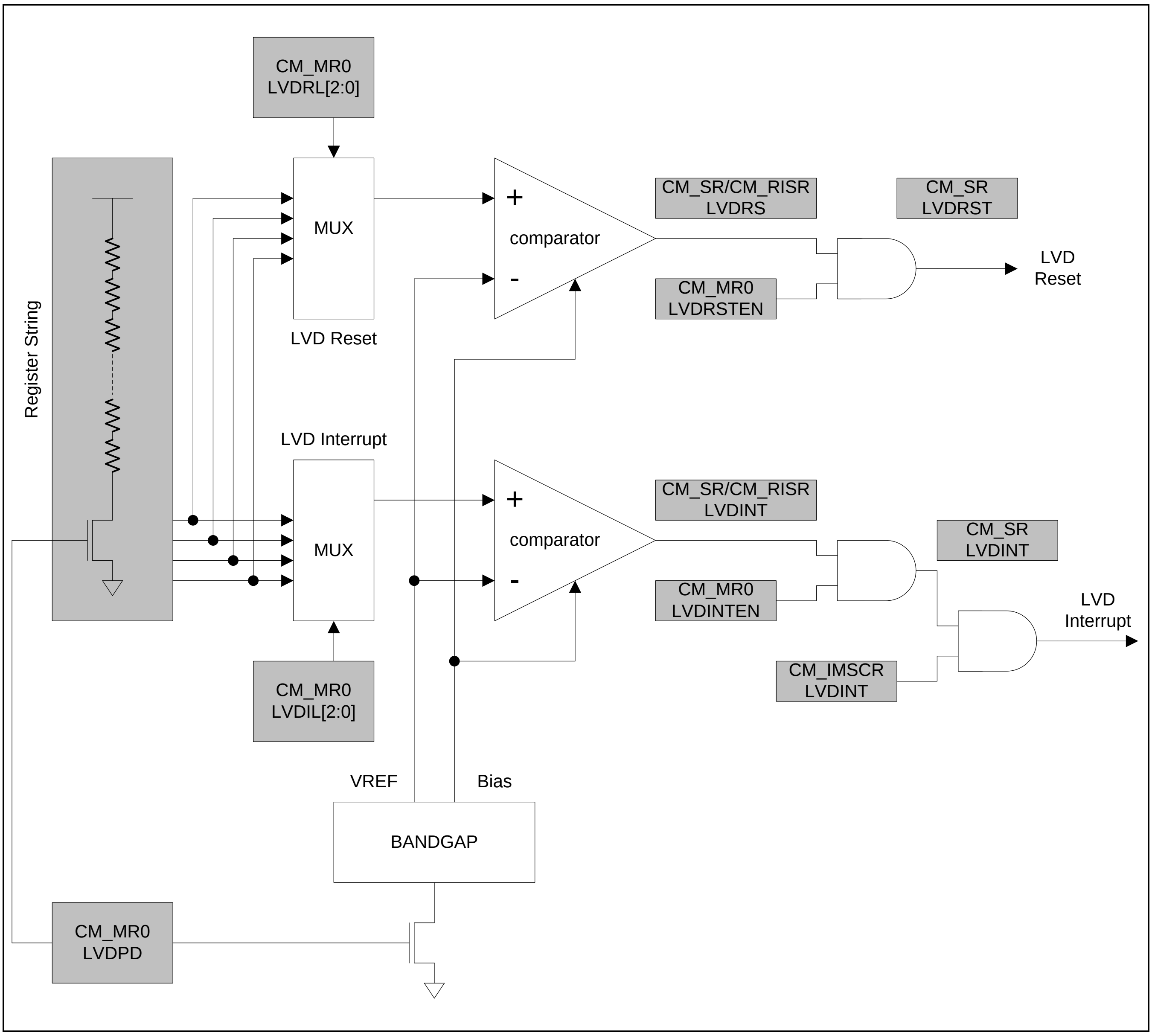


Figure 7-13 LVD Block Diagram

7.4.4 External Main Clock Monitor Reset (EMCMRST)

See the Clock Monitor.

7.4.5 External Sub Clock Monitor Reset (ESCMRST)

See the Clock Monitor

7.4.6 Watchdog Reset (WDTRST)

Watchdog reset is invoked timer overflows under the condition that both watchdog timer and reset are enabled. After watchdog timer reset, WDTRSTS bit in CM_SR register sets to '1'. This bit can be cleared by user (S/W) or other reset (H/W).

7.4.7 Software Reset (SWRST)

Software can initialize the device state itself when it writes '1' to CM_SRR register. After software reset, SWRSTS bit in CM_SR register sets to '1'. This bit can be cleared by user (S/W) or other resets (H/W).

7.4.8 CPU Request Reset (SYSRST)

CPU can initialize the device state itself when it writes '1' to SYSRESETREQ bit in Application Interrupt and Reset Control Register of Cortex-M3. After system reset, SYSRSTS bit in CM_SR register sets to '1'. This bit can be cleared by user (S/W) or other resets (H/W).

7.5 Basic Timer

The basic timer is a release timer at reset or wakeup from STOP. It blocks the clock supply to system and controls the reset/STOP release time for the predefined time while the power and clock supply is unstable as soon as the processor is powered on or the oscillator starts to run.

The reference clock for timer is SYSCLK defined by smart option. They generate the source clock for basic timer with clock divider. Basic timer can count with a different clock divider. The reset value of clock divider is defined by smart option.

After reset, user can control by changing BTCDIV[3:0] field in CM_BTCDR. For instance, to make the wakeup latency shorter in the exit of STOP mode, the software can configure the number of basic timer divider or count value before the entry of STOP mode.

The reset count value of basic timer is 0x100. In other words, when the 8th bit on the basic timer sets to ‘1’, 256 counts, system reset or wakeup signal are released.

This table shows the time by BT when BT count value is 256 and each divider value splits from 1 to 4096. For several example, BT input clock selects 1MHz, 4MHz, 8MHz, 16MHz, 20MHz, and 32.768KHz.

BT count - 256	/4096	/2048	/1024	/512	/256	/128
1MHz(us)	1048576.0	524288.0	262144.00	131072.000	65536.0000	2048.00000
4MHz(us)	262144.0	131072.0	65536.00	32768.000	16384.0000	512.00000
8MHz(us)	131072.0	65536.0	32768.00	16384.000	8192.0000	64.00000
16MHz(us)	65536.0	32768.0	16384.00	8192.000	4096.0000	128.00000
20MHz(us)	52428.8	26214.4	13107.20	6553.600	3276.8000	102.40000
32.768KHz(ms)	32000.0	16000.0	8000.00	4000.000	2000.0000	62.50000

BT count - 256	/64	/32	/16	/8	/4	/2	/1
1MHz(us)	16384.00000	8192.00000	4096.0	2048.00000	1024.00000	512.00000	256.00000
4MHz(us)	4096.00000	2048.00000	1024.0	512.00000	256.00000	128.00000	64.00000
8MHz(us)	2048.00000	1024.00000	512.0	256.00000	128.00000	64.00000	32.00000
16MHz(us)	1024.00000	512.00000	256.0	128.00000	64.00000	32.00000	16.00000
20MHz(us)	819.20000	409.60000	204.8	102.40000	51.20000	25.60000	12.80000
32.768KHz(ms)	500.00000	250.00000	16000.0	62.50000	31.25000	15.62500	7.81250

BT timer’s clock frequency is defined by a clock source and BT clock divider in smart option. That clock will be SYSCLK after reset. To finish BT’s counting means that the system reset occurs and CPU runs. S3FM02G recommends BT counting value the according to the selected clock source. To optimize on system, user can change the reset release time by BT. That is done by flash smart option re-program.

The following figures show the reset release timing diagram using BT timer.

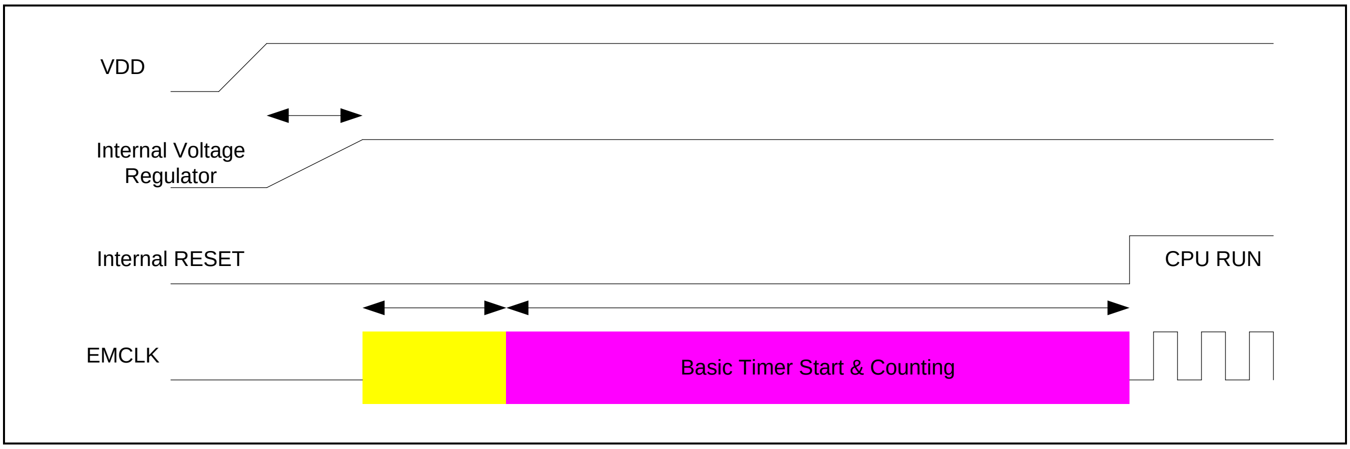


Figure 7-14 RESET @EMCLK

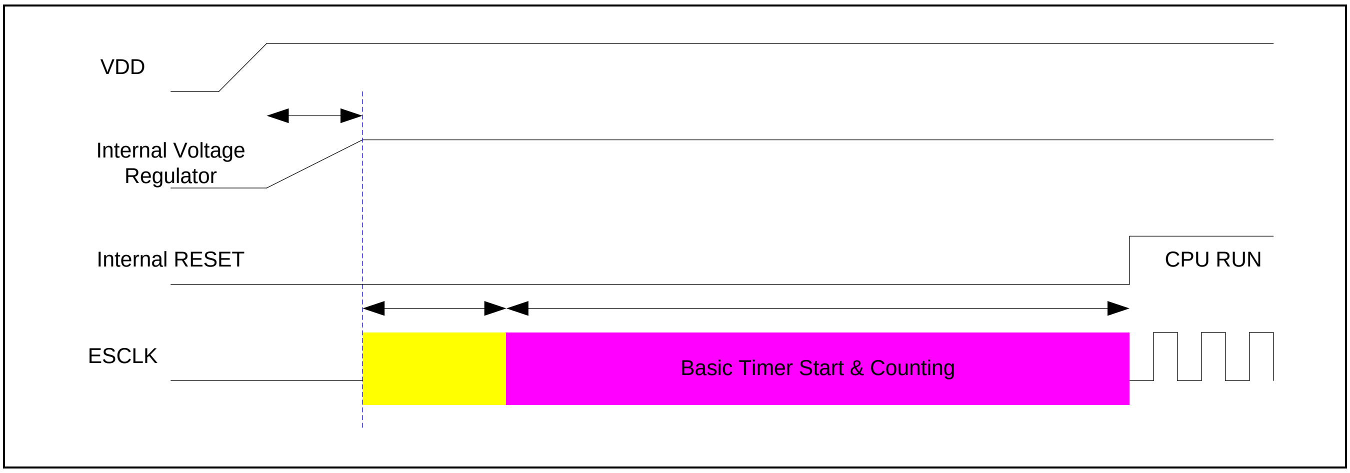


Figure 7-15 RESET @ESCLK

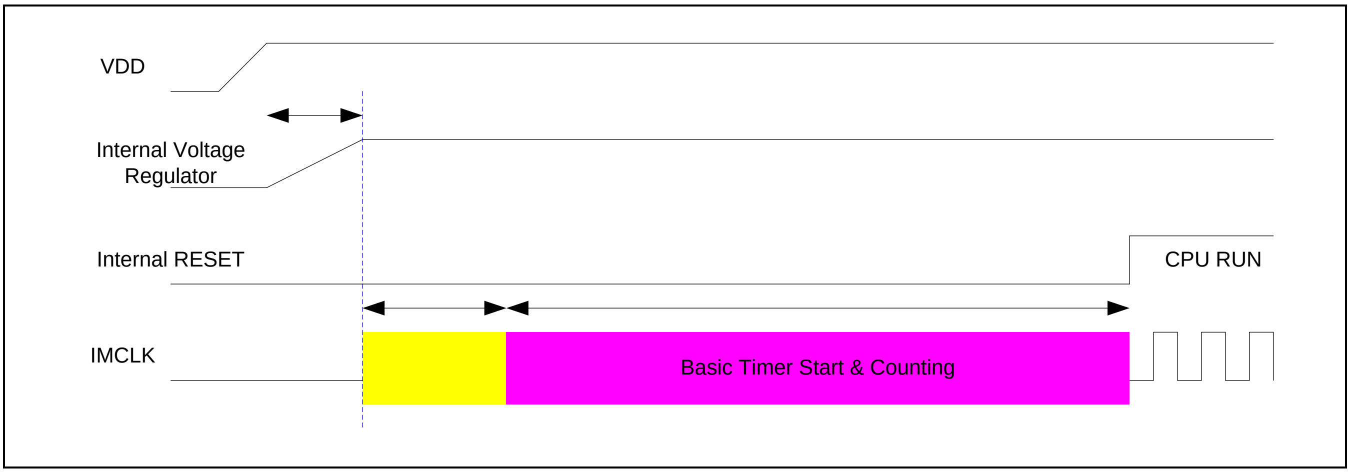


Figure 7-16 RESET @IMCLK

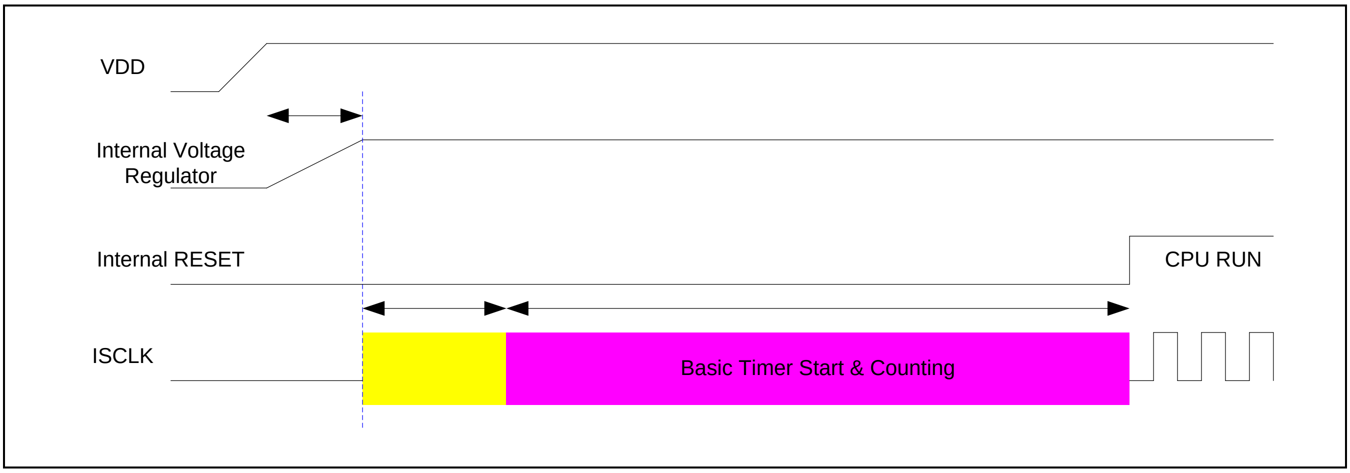


Figure 7-17 RESET @ISCLK

7.5.1 Fast Wake-up

For fast wake-up from STOP, user can use ‘FWAKE’ control bit. FWAKE bit should be enabled before STOP mode. When released from STOP mode by fast wake-up, SYSCLK will be IMCLK regardless of enable/disable status before STOP mode. It doesn’t need to use the fast wake-up when SYSCLK is IMCLK before STOP.

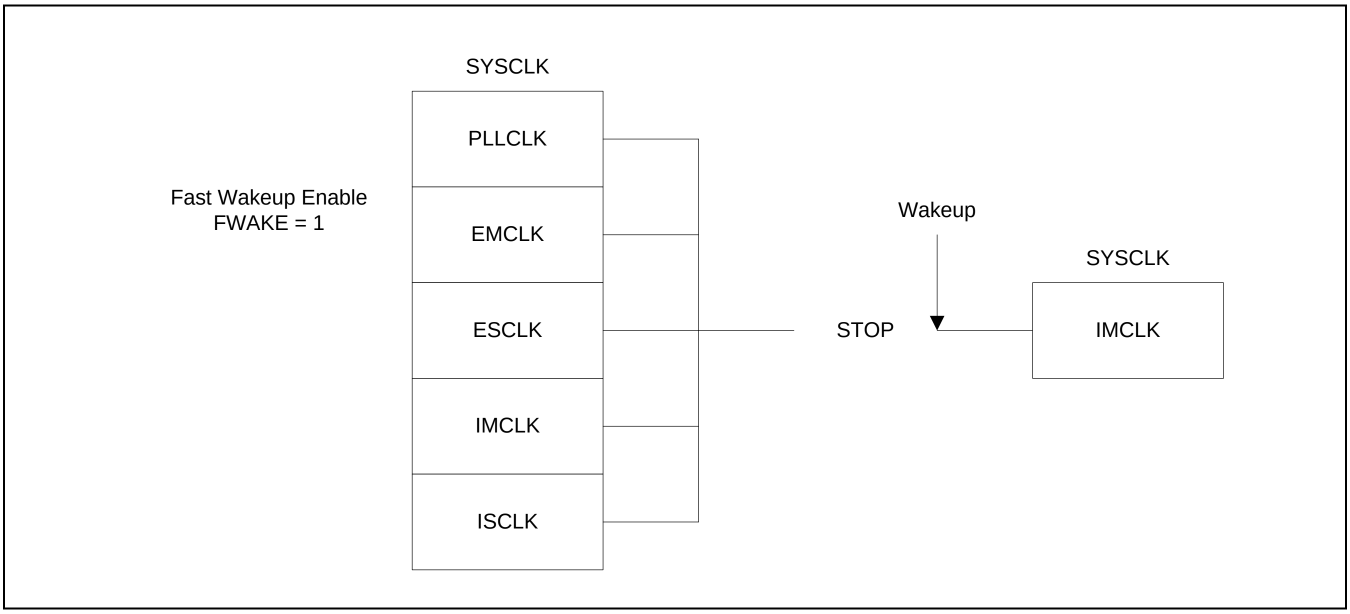


Figure 7-18 Fast Wake-up

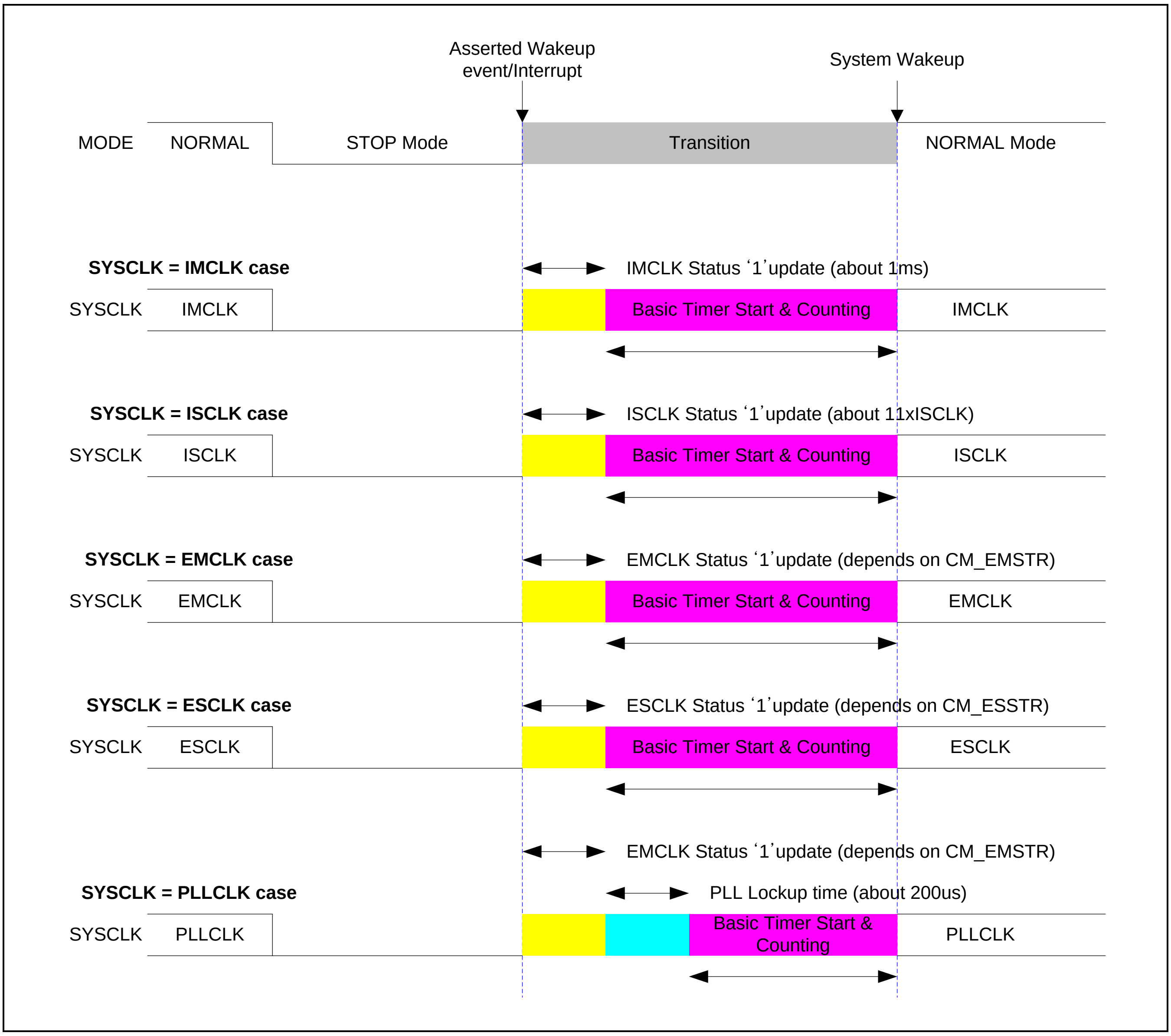

Asserted Wakeup event/Interrupt
System Wakeup
MODE
NORMAL
STOP Mode
Transition
NORMAL Mode
SYSCLK = IMCLK case
IMCLK Status '1' update (about 1ms)
SYSCLK
IMCLK
Basic Timer Start & Counting
IMCLK
SYSCLK = ISCLK case
ISCLK Status '1' update (about 11xISCLK)
SYSCLK
ISCLK
Basic Timer Start & Counting
ISCLK
SYSCLK = EMCLK case
EMCLK Status '1' update (depends on CM_EMSTR)
SYSCLK
EMCLK
Basic Timer Start & Counting
EMCLK
SYSCLK = ESCLK case
ESCLK Status '1' update (depends on CM_ESSTR)
SYSCLK
ESCLK
Basic Timer Start & Counting
ESCLK
EMCLK Status '1' update (depends on CM_EMSTR)
SYSCLK = PLLCLK case
PLL Lockup time (about 200us)
SYSCLK
PLLCLK
Basic Timer Start & Counting
PLLCLK

Figure 7-19 Basic Timer and Exit of STOP mode when FWAKE is "0"

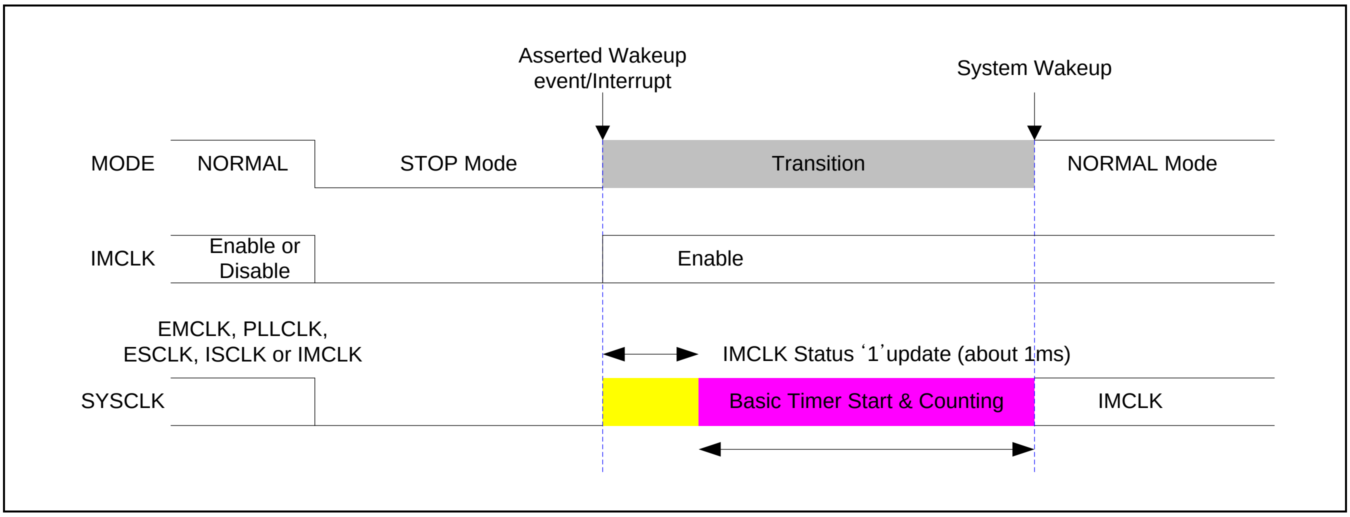


Figure 7-20 Basic Timer and Exit of STOP mode when FWAKE is "1" (SYSCLK = IMCLK)

7.6 Register Description

7.6.1 Register Map Summary

- Base Address: 0x4002_0000

Register	Offset	Description	Reset Value
CM_IDR	0x000	ID Register	0x0001_001C
CM_SRR	0x004	Software Reset Register	0x0000_0000
CM_CSR	0x008	Control Set Register	0x0000_0000
CM_CCR	0x00C	Control Clear Register	0x0000_0000
CM_PCSR0	0x010	Peripheral Clock Set Register 0	0x0000_0000
CM_PCSR1	0x014	Peripheral Clock Set Register 1	0x0000_0000
CM_PCCR0	0x018	Peripheral Clock Clear Register 0	0x0000_0000
CM_PCCR1	0x01C	Peripheral Clock Clear Register 1	0x0000_0000
CM_PCKSR0	0x020	Peripheral Clock Status Register 0	0x0000_000C
CM_PCKSR1	0x024	Peripheral Clock Status Register 1	0x0000_0000
CM_MR0	0x028	Mode Register0	0x0000_084B
CM_MR1	0x02C	Mode Register 1	0x0002_031X
CM_IMSCR	0x030	Interrupt Mask Set/Clear Register	0x0000_0000
CM_RISR	0x034	Raw Interrupt Status Register	0x0000_001F
CM_MISR	0x038	Masked Interrupt Status Register	0x0000_0000
CM_ICR	0x03C	Interrupt Clear Register	0x0000_0000
CM_SR	0x040	Status Register	0xYYA0_001F
CM_SCDR	0x044	System Clock Divider Register	0x0000_0007
CM_PCDR	0x048	Peripheral Clock Divider Register	0x0000_0000
CM_FCDR	0x04C	FRT Clock Divider Register	0x0000_0000
CM_STCDR	0x050	STT Clock Divider Register	0x0000_0000
CM_LCDR	0x054	LCD Clock Divider Register	0x0000_0000
CM_PSTR	0x058	PLL Stabilization Time Register	0x0000_0154
CM_PDPR	0x05C	PLL Divider Parameter Register	0x0008_0000
RSVD	0x060	Reserved. User must not access this register	–
RSVD	0x064	Reserved. User must not access this register	–
CM_EMSTR	0x068	External Main Clock Stabilization Time Register	0x0000_2710
CM_ESSTR	0x06C	External Sub-Clock Stabilization Time Register	0x0000_FFFF
CM_BTCDR	0x070	Basic Timer Clock Divider Register	0x0000_000X
CM_BTR	0x074	CM Basic Timer Register	0x0000_0100
CM_WCR0	0x078	Wakeup Control Register 0	0x0000_0000
CM_WCR1	0x07C	Wakeup Control Register 1	0x0000_0000
CM_WCR2	0x080	Wakeup Control Register 2	0x0000_0000

Register	Offset	Description	Reset Value
CM_WCR3	0x084	Wakeup Control Register 3	0x0000_0000
CM_WIMSCR	0x088	Wakeup Interrupt Mask Set/Clear Register	0x0000_0000
CM_WRISR	0x08C	Wakeup Raw Interrupt Status Register	0x0000_0000
CM_WMISR	0x090	Wakeup Masked Interrupt Status Register	0x0000_0000
CM_WICR	0x094	Wakeup Interrupt Clear Register	0x0000_0000
CM_NISR0	0x098	NVIC Interrupt Status Register 0	0x0000_0000
CM_NISR1	0x09C	NVIC Interrupt Status Register 1	0x0000_0000
RSVD	0x0A0	Reserved. User must not access this register	–
RSVD	0x0A4	Reserved. User must not access this register	–

NOTE: X can be changed by smart option. Y can be changed by real sources at reset time.

7.6.1.1 CM_IDR (ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_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
RSVD						IDCODE																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Code Register This field stores the ID code for the corresponding IP.	0x01_001C

7.6.1.2 CM_SRR (Software Reset Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	CM Software Reset 0 = No effect 1 = Perform Software Reset operation and auto-cleared. This reset makes chip reset. In other words, it is one of reset sources.	0

7.6.1.3 CM_CSR (Control Set Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD								EMCM	EMCMRST	ESCM	ESCMRST	IDLESP	RSVD							IDLEW	ISCLKS	PCLK	STCLK	PLL	RSVD	FWAKE	RSVD	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	W	W	W	W	W	R	R	R	R	R	R	R	W	W	W	W	W	R	W	R	W	W	W	W

Name	Bit	Type	Description	Reset Value
EMCLK	[0]	W	External Main Clock Enable Control Bit[1] 1 = Enable EMCLK clocking If user wants to enable EMCLK, IMCLK should be enabled. After checking EMCLK stable bit in CM_SR register, user can disable IMCLK.	0
IMCLK	[1]	W	Internal Main Clock Enable Control Bit[1] 1 = Enable IMCLK clocking	0
ESCLK	[2]	W	External Sub-Clock Enable Control Bit[1] 1 = Enable ESCLK clocking If user wants to enable ESCLK, ISCLK should be enabled. After checking ESCLK stable bit in CM_SR register, user can disable ISCLK.	0
ISCLK	[3]	W	Internal Sub-Clock Enable Control Bit[1] 1 = Enable ISCLK clocking	0
FWAKE	[5]	W	Fast Wake-up Enable Control Field 1 = Enable FWAKE If a MCU enters STOP mode after being enable a fast wake-up, SYSCLK becomes IMCLK after wake-up.	0
PLL	[7]	W	PLL ON Control Bit 1 = Enable PLL After enable PLL, user should check the PLL status bit in CM_SR register to use stabled PLL Clock.	0
STCLK	[8]	W	STCLK Enable Control in IDLE mode 1 = Enable STCLK in IDLE mode This is one of an option with IDLE mode. If STCLK bit sets to '1' before entering IDLE mode, STCLK is supplied to the sys-tick timer in IDLE mode.	0
PCLK	[9]	W	PCLK Enable Control bit in IDLE mode 1 = Connect PCLK supplied to peripherals in IDLE Mode PCLK is supplied to peripherals in IDLE mode.	0

Name	Bit	Type	Description	Reset Value
ISCLKS	[10]	W	ISCLK(Internal Sub Clock) Enable Control in STOP mode 1 = Enable ISCLK in STOP mode. When ISCLKS bit is enabled in STOP mode, internal Sub-Clock (ISCLK) will be enabled even if ISCLK bit is disabled. In this case, only FRT can be running with ISCLK.	0
IDLEW	[11]	W	Using NISR register for interrupt or wakeup source 1 = Using NISR0/NISR1 register	0
IDLESP	[19]	W	IVC Configuration Bit 1 = Use STOP IVC During the Sub-IDLE Mode, the Stop IVC will only be used**.** User wants to set to '1' in this bit, LVDPD bit in CM_MR0 register should be set to 0 before entering IDLE.	0
ESCMRST	[20]	W	External Sub Clock Monitor Reset Enable Control Bit 1 = Enable reset by Sub Clock monitor. When clock fail is detected, chip reset will be generated by clock monitor block	0
ESCM	[21]	W	External Sub Clock Monitor Function Enable Control Bit(2) 1 = Enable Sub Clock Monitor Function. This function is to detect the failure of an external sub clock.	0
EMCMRST	[22]	W	External Main Clock Monitor Reset Enable Control Bit 1 = Enable reset by Main Clock monitor. When a clock failure is detected, chip reset will be generated by clock monitor block.	0
EMCM	[23]	W	External Main Clock Monitor Function Enable Control Bit(2) 1 = Enable Main Clock Monitor Function This function is to detect the failure of an external main clock.	0

NOTE:

1. The complement of ‘Enable’ action means that each clock EMCLK, IMCLK, ESCLK, or ISCLK is enabled and stabled. That result of execution will be updated status register by hardware.
2. Before user sets to '1' to enable an external main or sub-clock monitor (EMCM or ESCM), the reference clock, IMCLK or ISCLK, should be enabled. To enable a clock monitor without a reference clock brings the command error.
3. This register is write-only. The opposite control should use CM_CCR. All defined control bits have a no-effect on writing '0'.

7.6.1.4 CM_CCR (Control Clear Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD								EMCM	EMCMRST	ESCM	ESCMRST	IDLESP	RSVD							IDLEW	ISCLKS	PCLK	STCLK	PLL	RSVD	FWAKE	RSVD	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	W	W	W	W	W	R	R	R	R	R	R	R	W	W	W	W	W	R	W	R	W	W	W	W

Name	Bit	Type	Description	Reset Value
EMCLK	[0]	W	External Main Clock Disable Control Bit 1 = Disable EMCLK clocking	0
IMCLK	[1]	W	Internal Main Clock Disable Control Bit 1 = Disable IMCLK clocking If EMCM bit in CM_SR register is '1', there is no effect (no disable) even if writing '1' to this bit.	0
ESCLK	[2]	W	External Sub-Clock Disable Control Bit 1 = Disable ESCLK clocking	0
ISCLK	[3]	W	Internal Sub-Clock Disable Control Bit 1 = Disable ISCLK clocking If ESCM bit in CM_SR register is '1', there is no effect (no disable) even if writing '1' to this bit.	0
FWAKE	[5]	W	Fast Wake-up Disable Control Field 1 = Disable FWAKE (After wake-up from STOP mode, SYSCLK source is the same before entry of STOP.) If a MCU enters STOP mode after being disable a fast wake-up, SYSCLK is the same before entry of STOP	0
PLL	[7]	W	PLL OFF Control Bit 1 = Disable PLL	0
STCLK	[8]	W	STCLK Disable Control in IDLE mode 1 = Disconnect STCLK supplied to a SYSTICK timer of Cortex-M3 in IDLE Mode	0
PCLK	[9]	W	PCLK Disable Control bit in IDLE mode 1 = Disconnect PCLK supplied to peripherals in IDLE Mode	0
ISCLKS	[10]	W	ISCLK(Internal Sub Clock) Disable Control in STOP mode 1 = Disable ISCLK in STOP mode	0
IDLEW	[11]	W	Ignore NISR register for interrupt or wakeup source 1 = Ignore NISR0/NISR1 register	0

Name	Bit	Type	Description	Reset Value
IDLESP	[19]	W	During the Sub-IDLE Mode, the Stop IVC will only be used 1 = Not use STOP IVC	0
ESCMRST	[20]	W	External Sub Clock Monitor Reset Disable Control Bit 1 = Disable reset by Sub Clock monitor. When clock fail is detected, clock fail flag will be reported.	0
ESCM	[21]	W	External Sub Clock Monitor Function Disable Control Bit 1 = Disable Sub Clock Monitor Function.	0
EMCMRST	[22]	W	External Main Clock Monitor Reset Disable Control Bit 1 = Disable reset by Main Clock monitor. When clock fail is detected, clock fail flag will be reported	0
EMCM	[23]	W	External Main Clock Monitor Function Disable Control Bit 1 = Disable Main Clock Monitor Function.	0

NOTE:

1. This register is write-only. The opposite control should use CM_CSR. All defined control bits have a no-effect on writing '0'.
2. It must be careful to write '1' to the each bit. Each clock can be a source of any block to operate. Before disable any clock, it need to check if whether that clock is used or not by any block. For example, if FRTCLK uses ISCLK as a timer clock source, user should stop the operation of FRTCLK before disable ISCLK. If not, CMDERR bit of CM_SR register will be set to '1' when CPU accesses CM_MR1 register.

7.6.1.5 CM_PCSR0 (Peripheral Clock Set Register 0)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
STTCLK	IOCLK	PFCCLK	RSVD	I2C1CLK	I2C0CLK	SPI1CLK	SPI0CLK	LCDCLK	ADCCLK	CAN1CLK	CAN0CLK	USART3CLK	USART2CLK	USART1CLK	USART0CLK	TC7CLK	TC6CLK	TC5CLK	TC4CLK	TC3CLK	TC2CLK	TC1CLK	TC0CLK	IMCCLK	ENCCLK	PWM1CLK	PWM0 CLK	FRTCLK	WDTCLK	OPACLK	SFMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	R	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
All Bit	[x]	W	PCLK (APB Clock) Gating Control 0 = No effect 1 = Enable each peripheral clock	0

NOTE: IOCLK includes GPIO clock and IOCONF clock

.

7.6.1.6 CM_PCSR1 (Peripheral Clock Set Register 1)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																						DFCCLK	ADC1CLK	IMC1CLK	ENC1CLK	PWM7CLK	PWM6CLK	PWM5CLK	PWM4CLK	PWM3CLK	PWM2CLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
All Bit	[x]	W	PCLK (APB Clock) Gating Control 0 = No effect 1 = Enable each peripheral clock	0

.

7.6.1.7 CM_PCCR0 (Peripheral Clock Clear Register 0)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
STTCLK	IOCLK	PFCCLK	RSVD	I2C1CLK	I2C0CLK	SPI1CLK	SPI0CLK	LCDCLK	ADCCLK	CAN1CLK	CAN0CLK	USART3CLK	USART2CLK	USART1CLK	USART0CLK	TC7CLK	TC6CLK	TC5CLK	TC4CLK	TC3CLK	TC2CLK	TC1CLK	TC0CLK	IMC0CLK	ENC0CLK	PWM1CLK	PWM0CLK	FRTCLK	WDTCLK	OPACLK	SFMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	R	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
All Bit	[x]	W	PCLK (APB Clock) Gating Control 0 = No effect 1 = Disable each peripheral clock	0

NOTE: IOCLK includes GPIO clock and IOCONF clock.

7.6.1.8 CM_PCCR1 (Peripheral Clock Clear Register 1)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																						DFCCLK	ADC1CLK	IMC1CLK	ENC1CLK	PWM7CLK	PWM6CLK	PWM5CLK	PWM4CLK	PWM3CLK	PWM2CLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
All Bit	[x]	W	PCLK (APB Clock) Gating Control 0 = No effect 1 = Disable each peripheral clock	0

7.6.1.9 CM_PCKSR0 (Peripheral Clock Status Register 0)

- Address = Base Address + 0x0020, Reset Value = 0x0000_000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STTCLK	IOCLK	PFCCLK	RSVD	I2C1CLK	I2C0CLK	SPI1CLK	SPI0CLK	LCDCLK	ADC0CLK	CAN1CLK	CAN0CLK	USART3CLK	USART2CLK	USART1CLK	USART0CLK	TC7CLK	TC6CLK	TC5CLK	TC4CLK	TC3CLK	TC2CLK	TC1CLK	TC0CLK	IMC0CLK	ENC0CLK	PWM1CLK	PWM0CLK	FRTCLK	WDTCLK	OPACLK	SFMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
All Bit	[x]	R	0 = Each peripheral clock is disabled (disconnected). It's impossible to control (write) SFR register of a target peripheral. 1 = Each peripheral clock is enabled (connected). It's possible to control (write) SFR register of a target peripheral	0 or 1

NOTE: IOCLK includes GPIO clock and IOCONF clock.

7.6.1.10 CM_PCKSR1 (Peripheral Clock Status Register 1)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																						DFCCLK	ADC1CLK	IMC1CLK	ENC1CLK	PWM7CLK	PWM6CLK	PWM5CLK	PWM4CLK	PWM3CLK	PWM2CLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
All Bit	[x]	R	0 = Each peripheral clock is disabled (disconnected). It's impossible to control (write) SFR register of a target peripheral. 1 = Each peripheral clock is enabled (connected). It's possible to control (write) SFR register of a target peripheral	0

7.6.1.11 CM_MR0 (Mode Register 0)

- Address = Base Address + 0x0028, Reset Value = 0x0000_084B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	CLKOUT[2:0]			LVDPD	STCLKEN	RXEV	RSVD	LVDINTEN	LVDIL[2:0]			LVDRSTEN	LVDRL[2:0]		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
LVDRL[2:0]	[2:0]	RW	LVD Reset Level (Refer to electrical specification) 011 = LVD_LEVEL0 (Typ. 2.6V) - Reset value 100 = LVD_LEVEL1 (Typ. 2.8V) 101 = LVD_LEVEL2 (Typ. 3.75V) 110 = LVD_LEVEL3 (Typ. 4.25V) Others = User must not set other values.	011'b
LVDRSTEN	[3]	RW	LVD Reset Enable/Disable Control Bit This bit selects if whether reset by LVD generates or not. The target voltage detected by LVD is decided by LVDRL fields. 0 = Disable 1 = Enable	1'b
LVDIL[2:0]	[6:4]	RW	LVD Interrupt threshold Level (Refer to electrical specification) 011 = LVD_LEVEL0 (Typ. 2.6V) - Reset value 100 = LVD_LEVEL1 (Typ. 2.8V) 101 = LVD_LEVEL2 (Typ. 3.75V) 110 = LVD_LEVEL3 (Typ. 4.25V) Others = User must not set other values. **NOTE:** If LVDIL[2:0] and LVDRL[2:0] are selected as the same voltage level, when detect voltage, reset operation will execute by LVD.	100'b
LVDINTEN	[7]	RW	LVD Interrupt Enable/Disable Control Bit This bit selects if whether interrupt by LCD generates or not. The target voltage detected by LCD is decided by LVDIL[2:0] fiels. 0 = Disable 1 = Enable	0'b

Name	Bit	Type	Description	Reset Value
RXEV	[9]	RW	RXEV Enable/Disable Control Bit 0 = Disable RXEV 1 = Enable RXEV It causes the Cortex-M3 defined Event Register to be set. This causes WFE instruction to complete. It also awakens the processor if it is sleeping as the result of encountering a WFE instruction when the Event Register is clear.	0'b
STCLKEN	[10]	RW	Systic Timer Clock Enable/Disable Control Bit 0 = Disable 1 = Enable	0'b
LVDPD	[11]	RW	LVD Power Down Control Bit 0 = LVD Power Down 1 = LVD Power Up When clear this bit, user can't use LVD interrupt and LVD reset function. To reduce current, user can configure this bit. For use LVD reset (LVDRST) or/and LVD interrupt, user should set the LVDPD bit. At reset, LVD is power-up state.	1'b
CLKOUT[2:0]	[14:12]	RW	Several Clock Output Control Bits 000 = EMCLK 001 = IMCLK 010 = ESCLK 011 = ISCLK 100 = PLLCLK/8 101 = RSVD 110 = SYSCLK/S 111 = CORECLK	000'b

7.6.1.12 CM_MR1 (Mode Register 1)

- Address = Base Address + 0x002C, Reset Value = 0x0002_031X

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													LCDCLK[2:0]			RSVD	STTCL[2:0]K			RSVD	FRTCLK[2:0]			RSVD	WDTCLK[2:0]K			RSVD	SYSCLK[2:0]		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	X	X
R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
SYSCLK[2:0]	[2:0]	RW	Select one among different clock for SYSCLK 000 = ESCLK, 32.768 KHz 001 = EMCLK, 4~8 MHz 010 = ISCLK, 32.768 KHz 011 = IMCLK, 8/16/20 MHz 100 = PLLCLK, 8~75 MHz Others = Prohibited Be careful to change SYSCLK. The SYSCLK is the clock for system and very important. So, if prohibited value is written (101'b, 110'b,, or 111'b), the value will be the same without the change. The end of system clock transition is indicated by the STABLE bit in CM_SR register set to '1'. **NOTE:** SYSCLK[1:0] fields reset value depends on the POCCS[1:0] field in smart option configuration status register (Refer to the Program Flash chapter).	0XX'b
WDTCLK[2:0]	[6:4]	RW	Watchdog Timer Clock Source Selection Bits If the clock source to change is disabled, the value won't be changed and command error will occur. That makes CMDERR status interrupt bit set to '1'. To reduce the current in no-operation, it is recommend to sue 'Disconnect WDTCLK'. 000 = EMCLK 001 = IMCLK 010 = ESCLK 011 = ISCLK 100 = PLLCLK Others = Disconnect WDTCLK	001'b
FRTCLK[2:0]	[10:8]	RW	Free Running Timer Clock Source Selection Bits If the clock source to change is disabled, the value won't be changed and command error will occur. That makes CMDERR status interrupt bit set to '1'.	011'b

Name	Bit	Type	Description	Reset Value
			000 = EMCLK 001 = IMCLK 010 = ESCLK 011 = ISCLK 100 = PLLCLK Others = Disconnect FRTCLK	
STTCLK[2:0]	[14:12]	RW	Stamp Timer Clock Source Selection Bits If the clock source to change is disabled, the value won't be changed and command error will occur. That makes CMDERR status interrupt bit set to '1'. 000 = EMCLK 001 = IMCLK 010 = ESCLK 011 = ISCLK 100 = PLLCLK Others = Disconnect STTCLK	000'b
LCDCLK[2:0]	[18:16]	RW	LCD Clock Source Selection Bits If the clock source to change is disabled, the value won't be changed and command error will occur. That makes CMDERR status interrupt bit set to '1'. 000 = EMCLK 001 = IMCLK 010 = ESCLK 011 = ISCLK 100 = PLLCLK Others = Disconnect LCDCLK **NOTE:** Before using LCD, clock source for LCD should be stabilized and LCD configuration should be done	010'b

7.6.1.13 CM_IMSCR (Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD													CMDERR	RSVD	LVDINT	EMCKFAIL	EMCKFAIL_END	ESCKFAIL	ESCKFAIL_END	RSVD				PLL	RSVD		STABLE	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R	R W	R W	R W	R W	R W	R	R	R	R	R W	R	R	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
Interrupt Source X	–	RW	0 = Disable Source X interrupt 1 = Enable Source X interrupt	–
EMCLK	[0]	RW	External Main Clock Stable Interrupt 0 = Stable Interrupt is masked. (Disable an interrupt) 1 = Stable interrupt is not masked. (Enable an interrupt)	0
IMCLK	[1]	RW	Internal Main Clock Stable Interrupt	0
ESCLK	[2]	RW	External Sub Clock Stable Interrupt	0
ISCLK	[3]	RW	Internal Sub Clock Stable Interrupt	0
STABLE	[4]	RW	Clock (SYSCLK) Switching Stable Interrupt	0
PLL	[7]	RW	PLL Stable Interrupt	0
ESCKFAIL_END	[12]	RW	External Sub Clock Failure End Interrupt	0
ESCKFAIL	[13]	RW	External Sub Clock Failure Interrupt	0
EMCKFAIL_END	[14]	RW	External Main Clock Failure End Interrupt	0
EMCKFAIL	[15]	RW	External Main Clock Failure Interrupt	0
LVDINT	[16]	RW	Interrupt Level Detect of LVD Interrupt	0
CMDERR	[18]	RW	Command Error Interrupt	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of '1' to the particular bit clears the mask, enabling the interrupt to be read. A write of '0' sets the corresponding mask.

7.6.1.14 CM_RISR (Raw Interrupt Status Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_001F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													CMDERR	LVDRS	LVDINT	EMCKFAIL	EMCKFAIL_END	ESCKFAIL	ESCKFAIL_END	RSVD				PLL	RSVD		STABLE	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EMCLK	[0]	R	External Main Clock Stable Interrupt Gives the raw interrupt state (prior to masking) of the EMCLK interrupt	1
IMCLK	[1]	R	Internal Main Clock Stable Interrupt Gives the raw interrupt state (prior to masking) of the IMCLK interrupt	1
ESCLK	[2]	R	External Sub Clock Stable Interrupt Gives the raw interrupt state (prior to masking) of the ESCLK interrupt	1
ISCLK	[3]	R	Internal Sub Clock Stable Interrupt Gives the raw interrupt state (prior to masking) of the ISCLK interrupt	1
STABLE	[4]	R	Clock (SYSCLK) Switching Stable Interrupt Gives the raw interrupt state (prior to masking) of the STABLE interrupt	1
PLL	[7]	R	PLL Stable Interrupt Gives the raw interrupt state (prior to masking) of the PLL interrupt	0
ESCKFAIL_END	[12]	R	External Sub Clock Failure End Interrupt Gives the raw interrupt state (prior to masking) of the ESCKFAIL_END interrupt	0
ESCKFAIL	[13]	R	External Sub Clock Failure Interrupt Gives the raw interrupt state (prior to masking) of the ESCKFAIL interrupt	0
EMCKFAIL_END	[14]	R	External Main Clock Failure End Interrupt Gives the raw interrupt state (prior to masking) of the EMCKFAIL_END interrupt	0
EMCKFAIL	[15]	R	External Main Clock Failure Interrupt Gives the raw interrupt state (prior to masking) of the EMCKFAIL interrupt	0

Name	Bit	Type	Description	Reset Value
LVDINT	[16]	R	Interrupt Level Detect of LVD Interrupt Gives the raw interrupt state (prior to masking) of the LVDINT interrupt This bit is set when LVDINTEN bit in CM_MR register is '1'.	0
LVDRS	[17]	R	Reset Level Detect Status Interrupt is not generated even if this bit is set to '1'. It just indicates that reset level, which is set in LVDRL bits of CM_MR register, is detected regardless of LVDRSTEN bit. This bit is same as LVDRS bit of CM_SR register. This bit is cleared by writing '1' into LVDRS bit of CM_ICR register.	0
CMDERR	[18]	R	Command Error Interrupt Gives the raw interrupt state (prior to masking) of the CMDERR interrupt	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

After Reset release, some bits (STABLE/ISCLK/ESCLK/IMCLK/EMCLK) will be set to '1'. Thus, it will be recommended to clear those bits above to use CM_ICR register just after Reset release.

7.6.1.15 CM_MISR (Masked Interrupt Status Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD													CMDERR	RSVD	LVDINT	EMCKFAIL	EMCKFAIL_END	ESCKFAIL	ESCKFAIL_END	RSVD				PLL	RSVD		STABLE	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Interrupt Source X	–	R	0 = Each interrupt doesn't occur 1 = Each interrupt occurs	–
EMCLK	[0]	R	External Main Clock Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the EMCLK interrupt	0
IMCLK	[1]	R	Internal Main Clock Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the IMCLK interrupt	0
ESCLK	[2]	R	External Sub Clock Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the ESCLK interrupt	0
ISCLK	[3]	R	Internal Sub Clock Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the ISCLK interrupt	0
STABLE	[4]	R	Clock (SYSCLK) Switching Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the STABLE interrupt	0
PLL	[7]	R	PLL Stable Interrupt Gives the unmasked interrupt state (after unmasking) of the PLL interrupt	0
ESCKFAIL_END	[12]	R	External Sub Clock Failure End Interrupt Gives the unmasked interrupt state (after unmasking) of the ESCKFAIL_END interrupt	0
ESCKFAIL	[13]	R	External Sub Clock Failure Interrupt Gives the unmasked interrupt state (after unmasking) of the ESCKFAIL interrupt	0
EMCKFAIL_END	[14]	R	External Main Clock Failure End Interrupt Gives the unmasked interrupt state (after unmasking) of the EMCKFAIL_END interrupt	0
EMCKFAIL	[15]	R	External Main Clock Failure Interrupt	0

Name	Bit	Type	Description	Reset Value
			Gives the unmasked interrupt state (after unmasking) of the EMCKFAIL interrupt	
LVDINT	[16]	R	Interrupt Level Detect of LVD Interrupt Gives the unmasked interrupt state (after unmasking) of the LVDINT interrupt	0
CMDERR	[18]	R	Command Error Interrupt Gives the unmasked interrupt state (after unmasking) of the CMDERR interrupt	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

7.6.1.16 CM_ICR (Interrupt Clear Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD													CMDERR	LVDRS	LVDINT	EMCKFAIL	EMCKFAIL_END	ESCKFAIL	ESCKFAIL_END	RSVD				PLL	RSVD		STABLE	ISCLK	ESCLK	IMCLK	EMCLK
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	R	R	R	R	W	R	R	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
EMCLK	[0]	W	External Main Clock Stable Interrupt 1 = Clears the EMCLK Stable Interrupt	0
IMCLK	[1]	W	Internal Main Clock Stable Interrupt 1 = Clears the IMCLK Stable Interrupt	0
ESCLK	[2]	W	External Sub Clock Stable Interrupt 1 = Clears the ESCLK Stable Interrupt	0
ISCLK	[3]	W	Internal Sub Clock Stable Interrupt 1 = Clears the ISCLK Stable Interrupt	0
STABLE	[4]	W	Clock (SYSCLK) Switching Stable Interrupt 1 = Clears the Switching Stable Interrupt	0
PLL	[7]	W	PLL Stable Interrupt 1 = Clears the PLL Stable Interrupt	0
ESCKFAIL_END	[12]	W	External Sub Clock Failure End Interrupt 1 = Clears the ESCKFAIL_END Interrupt	0
ESCKFAIL	[13]	W	External Sub Clock Failure Interrupt 1 = Clears the ESCKFAIL Interrupt	0
EMCKFAIL_END	[14]	W	External Main Clock Failure End Interrupt 1 = Clears the EMCKFAIL_END Interrupt	0
EMCKFAIL	[15]	W	External Main Clock Failure Interrupt 1 = Clears the EMCKFAIL Interrupt	0
LVDINT	[16]	W	Interrupt Level Detect of LVD Interrupt 1 = Clears the LVDINT Interrupt	0
LVDRS	[17]	W	Reset Level Detect of LVD 1 = Clears the LVDRS flag	0
CMDERR	[18]	W	Command Error Interrupt 1 = Clears the CMDERR Interrupt	0

On a write to '1', the corresponding interrupt in CM_RISR is cleared. A write of '0' has no effect.

7.6.1.17 CM_SR (Status Register)

- Address = Base Address + 0x0040, Reset Value = 0xYYA0_001F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYSRSTS	EMCMRSTS	ESCMRSTS	PORRSTS	WDTRSTS	LVDRSTS	NRSTS	SWRSTS	EMCM	EMCMRST	ESCM	ESCMRST	IDLESP	CMDERR	LVDRS	LVDINT	EMCKFAIL	EMCKFAIL_END	ESCKFAIL	ESCKFAIL_END	IDLEW	ISCLKS	PCLK	STCLK	PLL	RSVD	FWAKE	STABLE	ISCLK	ESCLK	IMCLK	EMCLK
X	X	X	X	X	X	X	X	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
R W	R W	R W	R W	R W	R W	R W	R W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EMCLK	[0]	R	External Main Clock Status 0 = EMCLK is disabled 1 = EMCLK is enabled and stabilized.	1
IMCLK	[1]	R	Internal Main Clock Status 0 = IMCLK is disabled 1 = IMCLK is enabled and stabilized.	1
ESCLK	[2]	R	External Sub-Clock Status 0 = ESCLK is disabled 1 = ESCLK is enabled and stabilized.	1
ISCLK	[3]	R	Internal Sub-Clock Status 0 = ISCLK is disabled 1 = ISCLK is enabled and stabilized.	1
STABLE	[4]	R	Clock source switching STABLE Status 1 = Clock source switching for SYSCLK is completed and stable status.	1
FWAKE	[5]	R	Fast Wake Up Control (Enable/Disable) Status 0 = Disable FWAKE 1 = Enable FWAKE (After wakeup, IMCLK is used as SYSCLK).	0
PLL	[7]	R	PLL Stable Interrupt 0 = PLL is disabled 1 = PLL is enabled and stabilized.	0
STCLK	[8]	R	STCLK Control Status in IDLE mode 0 = Disable STCLK in IDLE mode 1 = Enable STCLK in IDLE mode.	0
PCLK	[9]	R	PCLK Control Status in IDLE mode 0 = Disable PCLK in IDLE mode 1 = Enable PCLK in IDLE mode.	0
ISCLKS	[10]	R	ISCLK (Internal Sub Clock) status in STOP mode	0

Name	Bit	Type	Description	Reset Value
			0 = Disable ISCLK in STOP mode 1 = Enable ISCLK in STOP mode.	
IDLEW	[11]	R	Using/Ignore NISR register for interrupt or wakeup source. 0 = Ignore NISR register 1 = Using NISR register	0
ESCKFAIL_END	[12]	R	External Sub Clock Failure End Status 0 = No end of the external sub-clock oscillator failure has been detected. 1 = At least one end of the external sub-clock oscillator clock failure has been detected. This bit is cleared by writing 1' into ESCKFAIL_END bit of CM_ICR register	0
ESCKFAIL	[13]	R	External Sub Clock Fail Status 0 = The external sub-clock oscillator failure does not occur. 1 = The external sub-clock oscillator failure has been detected This bit will be cleared when ESCKFAIL_END or reset is occurred.	0
EMCKFAIL_END	[14]	R	External Main Clock Failure End Status 0 = No end of the external main oscillator failure has been detected. 1 = At least one end of the external main oscillator clock failure has been detected. This bit is cleared by writing 1' into EMCKFAIL_END bit of CM_ICR register	0
EMCKFAIL	[15]	R	External Main Clock Fail Status 0 = The external main oscillator failure does not occur. 1 = The external main oscillator failure has been detected. This bit will be cleared when EMCKFAIL_END or reset is occurred.	0
LVDINT	[16]	R	LVD Interrupt Status 0 = LVD interrupt voltage level is not detected. 1 = LVD interrupt voltage level is detected.	0
LVDRS	[17]	R	LVD Reset Level Detect Status (same as LVDRS in CM_RISR register) 0 = LVD reset voltage level is not detected. 1 = LVD reset voltage level is detected.	0
CMDERR	[18]	R	Command Error Interrupt 1 = Clears the CMDERR Interrupt	0
IDLESP	[19]	R	STOP IVC Control Status in IDLE mode 0 = Not use STOP IVC in IDLE mode. 1 = Only use STOP IVC in IDLE mode	0
ESCMRST	[20]	R	External Sub Clock Monitor Reset function Enable/Disable Status 0 = Disable reset function by clock fail mode	0

Name	Bit	Type	Description	Reset Value
			1 = Enable reset function by clock fail mode	
ESCM	[21]	R	External Sub Clock Monitor Fail function Enable/Disable Status 0 = Disable clock fail mode 1 = Enable clock fail mode	1
EMCMRST	[22]	R	External Main Clock Monitor Reset function Enable/Disable Status 0 = Disable reset function by clock fail mode 1 = Enable reset function by clock fail mode	0
EMCM	[23]	R	External Main Clock Monitor Fail function Enable/Disable Status 0 = Disable clock fail mode 1 = Enable clock fail mode	1
SWRSTS	[24]	RW	System(Chip) Reset from the software reset (SWRST in CM_SRR register) 0 = Software reset is not occurred. 1 = The last reset is caused by Software reset. This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	Y
NRSTS	[25]	RW	System(Chip) Reset by External Reset Pin This bit tells the reset source that generates reset signal pin. 0 = External Pin reset is not occurred. 1 = External Pin reset (nRESET) is occurred. This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	Y
LVDRSTS	[26]	RW	System(Chip) Reset by LVD Control Bit When LVDRST bit in CM_MR register is enabled and LVD reset condition has occurred 0 = LVD reset is not occurred. 1 = LVD reset is occurred. On a write to 1, this bit is cleared.	Y
WDTRSTS	[27]	RW	System(Chip) Reset by Watchdog timer 0 = Watchdog timer reset is not occurred. 1 = The last reset is caused by Watchdog timer reset. This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	Y
PORRSTS	[28]	RW	System(Chip) Reset from power on reset 0 = Power on reset is occurred. 1 = Power on reset is occurred. On a write to 1, this bit is cleared.	Y
ESCMRSTS	[29]	RW	System(Chip) Reset from External Sub Clock Monitor fail 0 = External Sub-Clock Monitor fail reset is not occurred 1 = The last reset is caused by the external Sub-clock monitor fail reset.	Y

Name	Bit	Type	Description	Reset Value
			This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	
EMCMRSTS	[30]	RW	System(Chip) Reset from External Main Clock Monitor fail 0 = External Main-Clock Monitor fail reset is not occurred 1 = The last reset is caused by the external Main-clock monitor fail reset. This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	Y
SYSRSTS	[31]	RW	System(Chip) Reset from CPU request 0 = CPU reset request is not occurred 1 = The last reset is caused by CPU request. This bit will be cleared when PORRSTS or LVDRSTS is occurred or written to '1'.	Y

CM_SR[31:24] bits are read/write. And each bit of CM_SR[31:24] must be cleared with writing '1' after checking status of each Reset.

NOTE: Command Error Event

1. When user disables IMCLK/ISCLK while the clock monitor function (EMCM/ESCM) is enabled. User should disable after disable the clock monitor function.
2. When user enables the clock monitor function (EMCM/ESCM), while the IMCLK/ISCLK is disabled.
3. When user chooses the disabled clock for FRT, STT, LCD. User should select the enabled and stabled clock source for each IP (FRT, STT, or LCD)

7.6.1.18 CM_SCDR (System Clock (SYSCLK) Divider Register)

- Address = Base Address + 0x0044, Reset Value = 0x0000_0007

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

Name	Bit	Type	Description	Reset Value
SDIV[2:0]	[2:0]	RW	SYSCLK divider This field selects the division ratio for SYSCLK. **NOTE:** The clock quality checker only works in the PLL mode, when the PLL is locked	0x7
SDIVKEY[15:0]	[31:16]	W	Key for write access into the CM_SCDR Any write in CM_SCDR register bits is only effective if the SDIVKEY is equal to 0xACDC.	0x0000

SYSCLK	Description	4MHz	8MHz	12MHz	75MHz
SDIV[2:0]	Division Ratio				
000	1	4	8	12	75
001	2	2	4	6	37.5
010	3	1.333333	2.666667	4	25
011	4	1	2	3	18.75
100	5	0.8	1.6	2.4	15
101	6	0.666667	1.333333	2	12.5
110	7	0.571429	1.142857	1.714286	10.714286
111	8	0.5	1	1.5	9.375

7.6.1.19 CM_PCDR (Peripheral Clock (PCLK) Divider Register)

- Address = Base Address + 0x0048, Reset Value = 0x0000_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
PDIVKEY[15:0]																RSVD												PDIV[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PDIV[3:0]	[3:0]	RW	SYSCLK divider for PCLK	0x0
PDIVKEY[15:0]	[31:16]	W	Key for write access into the CM_PCDR Any write in CM_PCDR register bits is only effective if the PDIVKEY is equal to 0xA3C5.	0x0000

PDIV[3:0]	Division Ratio	Frequency
0000	1	SYSCLK
0001	2	SYSCLK/2
001X	4	SYSCLK/4
01XX	8	SYSCLK/8
1XXX	16	SYSCLK/16

NOTE: 'X' means any value

7.6.1.20 CM_FCDR (FRT Clock Divider Register)

- Address = Base Address + 0x004C, Reset Value = 0x0000_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
FDIVKEY[15:0]																RSVD									MDIV[2:0]			NDIV[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
NDIV	[3:0]	RW	–	0x0
MDIV	[6:4]	RW	–	0x0
FDIVKEY[15:0]	[31:16]	W	Key for write access into the CM_FCDR Any write in CM_FCDR register bits is only effective if the FDIVKEY is equal to 0xC3DC.	0x0000

- FRT Clock = (Selected Clock / MDIV[2:0]) / NDIV[3:0])

MDIV[2:0]	Division Ratio	NDIV[3:0]	Division Ration
000	1	0000	1
001	2	0001	2
010	3	001X	4
011	4	01XX	8
100	5	1XXX	16
101	6	–	–
110	7	–	–
111	8	–	–

7.6.1.21 CM_STCDR (STT Clock Divider Register)

- Address = Base Address + 0x0050, Reset Value = 0x0000_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
STDIVKEY[15:0]																RSVD									CDIV[2:0]			DDIV[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DDIV[3:0]	[3:0]	RW	–	0x0
CDIV[2:0]	[6:4]	RW	–	0x0
STDIVKEY[15:0]	[31:16]	W	Key for write access into the CM_SCDR Any write in CM_SCDR register bits is only effective if the STDIVKEY is equal to 0xDC3C.	0x0000

- STT Clock (STTCLK) = (Selected Clock / CDIV[2:0]) / DDIV[3:0])

CDIV[2:0]	Division Ratio	DDIV[3:0]	Division Ration
000	1	0000	1
001	2	0001	2
010	3	001X	4
011	4	01XX	8
100	5	1XXX	16
101	6	–	–
110	7	–	–
111	8	–	–

7.6.1.22 CM_LCDR (LCD Clock Divider Register)

- Address = Base Address + 0x0054, Reset Value = 0x0000_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
LDIVKEY[15:0]																RSVD									JDIV[2:0]			KDIV[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
KDIV[3:0]	[3:0]	RW	–	0x0
JDIV[2:0]	[6:4]	RW	–	0x0
LDIVKEY[15:0]	[31:16]	W	Key for write access into the CM_LCDR Any write in CM_LCDR register bits is only effective if the LDIVKEY is equal to 0x3CCD.	0x0000

- LCD Clock = (Selected Clock / JDIV[2:0]) / KDIV[3:0]

JDIV[2:0]	Division Ratio	KDIV[3:0]	Division Ratio
000	1	0000	1
001	2	0001	2
010	3	0010	4
011	4	0011	8
100	5	0100	16
101	6	0101	32
110	7	0110	64
111	8	0111	128
–	–	1000	256
–	–	1001	512
–	–	1010	1024
–	–	1011	2048
–	–	1100	4096
–	–	Others	Not used

7.6.1.23 CM_PSTR (PLL Stabilization Time Register)

- Address = Base Address + 0x0058, Reset Value = 0x0000_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
PLLSKEY[15:0]																RSVD					PST[10:0]										
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	1	0	1	0	1	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PST[10:0]	[10:0]	RW	PLL stabilization time PST register value = (PLL stabilization time / (EMCLK period × 256)) Typically, PLL stabilization time must be longer than 300 us.	0x154
PLLSKEY[15:0]	[31:16]	W	Key for write access into the CM_PSTR register Any write in the CM_PSTR register bits will only be effective if the PLLSKEY field is equal to 0x59C1.	0x0000

NOTE: This register will be writable access only when the PLL is disabled, otherwise the register will be only readable.

7.6.1.24 CM_PDPR (PLL Divider Parameters Register)

- Address = Base Address + 0x005C, Reset Value = 0x0008_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
PLLKEY[7:0]								RSVD		PLLCUS[2:0]			RSVD	PLLPOST[1:0]		RSVD		PLLPRE[5:0]						PLLMUL[7:0]							
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	R	R	R W	R W	R W	R	R W	R W	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PLLMUL[7:0]	[7:0]	RW	PLL Multiplier (0x01 ~ 0xFF, 1 ~ 255)	0x00
PLLPRE[5:0]	[13:8]	RW	PLL Pre-divider (0x01 ~ 0x3F, 1 ~ 63)	0x00
PLLPOST[1:0]	[17:16]	RW	PLL Post-scaler (0x0 ~ 0x3, 0 ~ 3) FOUT (PLL output frequency) = (M x FIN) / ((P + 2) x (2^S)) (where, FIN = frequency of EMCLK)	00
PLLCUS[2:0]	[21:19]	RW	Charge Pump Current Selection	001'b
PLLKEY[7:0]	[31:24]	W	PLL Parameter Control Register Key Value Key for write access into the CM_PSTR register Any write in CM_PDPR register bits is only effective if the PLLKEY is equal to 0xC1.	0x00

NOTE: This register is write-accessible only if the PLL is disabled. Otherwise, this register is read-accessible.

7.6.1.25 CM_EMSTR (External Main Clock Stabilization Time Register)

- Address = Base Address + 0x0068, Reset Value = 0x0000_2710

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

Name	Bit	Type	Description	Reset Value
EMST[15:0]	[15:0]	RW	External Main Clock Oscillator Stabilization Time Control Field This clock source is IMCLK. The number of external main oscillator clock is defined as a stabilization time counter for the main oscillator. The default value is 0x2710 (10,000) corresponding to 10ms with internal main oscillator clock divided by 20. Stabilization Time = EMST[15:0] x (IMCLK/20) period For example, if IMCLK = 20MHz, 10ms= 0x2710 x 1us	0x2710
EMSKEY[15:0]	[31:16]	W	Key for write access into the CM_EMSTR register Any write in CM_EMSTR register bits is only effective if the EMSKEY is equal to 0xFA4B.	0x0000

7.6.1.26 CM_ESSTR (External Sub Clock Stabilization Time Register)

- Address = Base Address + 0x006C, Reset Value = 0x0000_FFFF

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

Name	Bit	Type	Description	Reset Value
ESST[15:0]	[15:0]	RW	External Sub-Clock Oscillator Stabilization Time Control Field This clock source is ISCLK. The number of an internal 32.768KHz oscillator clock is defined as a stabilization time counter for the external 32.768KHz oscillator. The default value is 0xFFFF corresponding to 2sec with the internal sub oscillator clock (ISCLK) 2 seconds = 0xFFFF x 1/ISCLK = 65,536 x 1/32,768 = 2s	0xFFFF
ESSKEY[15:0]	[31:16]	W	Key for write access into the CM_ESSTR register Any write in CM_ESSTR register bits is only effective if the ESSKEY is equal to 0xAFB4.	0x0000

7.6.1.27 CM_BTCDR (Basic Timer Clock Divider Register)

- Address = Base Address + 0x0070, Reset Value = 0x0000_000X

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

Name	Bit	Type	Description	Reset Value
BTCDIV[3:0]	[3:0]	RW	Basic Timer Clock Divider Control Field The number of basic timer clock divider value is defined.	XXXX'b
BTCDKEY[15:0]	[31:16]	W	Key for write access into the CM_BTCDR register Any write in CM_BTCDR register bits is only effective if the BTCDKEY is equal to 0x3569.	0x0000

At first in bear chip status, all bits in SO_CSR (Refer to flash memory chapter) are '1'. That means BT clock will be 20MHz IMCLK as SYSCLK and BT Clock divider will be 4096. So Reset release time by BT will be 256 counting time with 20MHz/ 4096, 52.428ms.

BT timer is also used at wake-up from stop, because the basic timer controls a reset release timer at reset or wakeup from STOP. Before entering STOP mode, user should write the target value on BT count and BT clock divider.

User can change the clock source and divider using smart option re-programming. According to the programmed smart option value, after reset, this register’s value is decided.

When BT Timer’s 8th bit sets to ‘1’, 256 counts, system reset or wakeup signal is released.

BTCDIV[3:0]	Divider Ratio	BTCDIV[3:0]	Divider Ratio
1111	4096	1000	32
1110	2048	0111	16
1101	1024	0110	8
1100	512	0101	4
1011	256	0100	2
1010	128	0011	1
1001	64	Others	Not used

7.6.1.28 CM_BTR (Basic Timer Register)

- Address = Base Address + 0x0074, Reset Value = 0x0000_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
RSVD																BTCV[15:0]															
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	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
BTCV	[15:0]	RW	Basic Timer Count Value	0x0100

7.6.1.29 CM_WCR0/1/2/3 (Wakeup Control Register 0/1/2/3)

7.6.1.29.1 CM_WCR0 (Wakeup Control Register 0)

- Address = Base Address + 0x0078, Reset Value = 0x0000_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
WEN3	EDGE3	RSVD	WSRC3[4:0]					WEN2	EDGE2	RSVD	WSRC2[4:0]					WEN1	EDGE1	RSVD	WSRC1[4:0]					WEN0	EDGE0	RSVD	WSRC[4:0]				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W

7.6.1.29.2 CM_WCR1 (CM Wakeup Control Register 1)

- Address = Base Address + 0x007C, Reset Value = 0x0000_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
WEN7	EDGE7	RSVD	WSRC7[4:0]					WEN6	EDGE6	RSVD	WSRC6[4:0]					WEN5	EDGE5	RSVD	WSRC5[4:0]					WEN4	EDGE4	RSVD	WSRC[4:0]				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W

7.6.1.29.3 CM_WCR2 (CM Wakeup Control Register 2)

- Address = Base Address + 0x0080, Reset Value = 0x0000_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
WEN11	EDGE11	RSVD	WSRC11[4:0]					WEN10	EDGE10	RSVD	WSRC10[4:0]					WEN9	EDGE9	RSVD	WSRC9[4:0]					WEN8	EDGE8	RSVD	WSRC8[4:0]				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W

CM_WCR3 (CM Wakeup Control Register 3)

- Address = Base Address + 0x0084, Reset Value = 0x0000_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
WEN15	EDGE15	RSVD	WSRC15[4:0]					WEN14	EDGE14	RSVD	WSRC14[4:0]					WEN13	EDGE13	RSVD	WSRC13[4:0]					WEN12	EDGE12	RSVD	WSRC12[4:0]				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
WSRCx[4:0]	[x]	RW	Wake-Up Source Selection Field	0x00
EDGEx	[x]	RW	Edge Type Selection Bit 0 = Rising edge trigger selected (for Event and Interrupt) 1 = Falling edge trigger selected (for Event and Interrupt)	0
WENx	[x]	RW	Wake-Up Enable/Disable Control Bit 0 = The edge trigger selected by EDGEx bit disable 1 = The edge trigger selected by EDGEx bit enable	0

NOTE: Any configuration in this register must be done before the entry of STOP/IDLE mode.
To enable an event or interrupt, at least either of corresponding edge or level must be configured.

Table 7-11 The Wake-up Sources And Pin Assignment

WSRCx[4:0]	Wakeup Source	Pin Information
00000	EXI0	P0.12 / SEG5 / TCLK5 / EXI0
00001	EXI1	P0.13 / SEG6 / TCLK4 / EXI1
00010	EXI2	P0.16 / SEG9 / TCLK3 / EXI2
00011	EXI3	P0.17 / SEG10 / TCLK2 / EXI3
00100	EXI4	P0.26 / SEG19 / USARTCLK3 / EXI4
00101	EXI5	P1.8 / SEG33 / USARTCLK2 / EXI5
00110	EXI6	P1.22 / ADTRG0 / EXI6
00111	EXI7	P2.0 / AIN10 / OP3_O / EXI7
01000	EXI8	P2.1 / AIN11 / OP3_N / EXI8
01001	EXI9	P2.13 / AIN06 / OP4_O / EXI9
01010	EXI10	P2.14 / AIN07 / OP4_N / EXI10
01011	EXI11	P2.15 / AIN08 / OP4_P / EXI11
01100	EXI12	P2.26 / SCL1 / EXI12
01101	EXI13	P2.27 / SDA1 / EXI13
01110	EXI14	P0.3 / COM0 / TPWM7 / EXI14
01111	EXI15	P0.4 / COM1 / TCAP6 / EXI15
10000	FRT	–
10001	USARTRX0	P0.0 / VLCD1 / CLKOUT / USARTRXD0 P1.30 / OP2_N / USARTRXD0
10010	USARTRX1	P0.21 / SEG14 / TCAP1 / USARTRXD1 P1.23 / OP0_O / USARTRXD1
10011	USARTRX2	P1.9 / SEG34 / USARTRXD2 / PWM3
10100	USARTRX3	P0.27 / SEG20 / USARTRXD3
10101	CANRX0	P0.19 / SEG12 / TPWM2 / CANRX0 P1.27 / OP1_N / CANRX0 / SSPCLK1
10110	CANRX1	P1.6 / SEG31 / CANRX1 / SSPCLK0
10111	SPI0	P0.10 / SEG3 / TCLK7 / SSPMISO0 @ Master Mode P1.5 / SEG30 / SDA1 / SSPMISO0 @ Master Mode P0.11 / SEG4 / TCLK6 / SSPMOSI0 @ Slave Mode P1.4 / SEG29 / SCL1 / SSPMOSI0 @ Slave Mode
11000	SPI1	P1.1 / SEG26 / SSPMISO1 @ Master Mode P1.26 / OP1_O / SCL0 / SSPMISO1 @ Master Mode P1.0 / SEG25 / SSPMOSI1 @ Slave Mode P1.25 / OP0_P / SDA0 / SSPMOSI1 @ Slave Mode
others	Reserved	

7.6.1.30 CM_WIMSCR (Wakeup Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0088, Reset Value = 0x0000_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
RSVD																WI15	WI14	WI13	WI14	WI11	WI10	WI9	WI8	WI7	WI6	WI5	WI4	WI3	WI2	WI1	WI0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
WIx	[x]	RW	0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0

7.6.1.31 CM_WRISR (Wakeup Raw Interrupt Status Register)

- Address = Base Address + 0x008C, Reset Value = 0x0000_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
RSVD																WI15	WI14	WI13	WI14	WI11	WI10	WI9	WI8	WI7	WI6	WI5	WI4	WI3	WI2	WI1	WI0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
WIx	[x]	R	0 = Each interrupt does not occur. (prior to unmasking) 1 = Each interrupt is occurred. (prior to unmasking)	0

On a read this register gives the current raw status value of the corresponding interrupt prior to unmasking. A write has no effect.

7.6.1.32 CM_WMISR (Wakeup Masked Interrupt Status Register)

- Address = Base Address + 0x0090, Reset Value = 0x0000_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
RSVD																WI15	WI14	WI13	WI14	WI11	WI10	WI9	WI8	WI7	WI6	WI5	WI4	WI3	WI2	WI1	WI0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
WIx	[x]	R	CM_WMISR = CM_WIMSCR & CM_WRISR 0 = Source X interrupt does not occur 1 = Source X interrupt occurs	0

On a read this register gives the current unmasked status value of the corresponding interrupt. A write has no effect.

7.6.1.33 CM_WICR (Wakeup Interrupt Clear Register)

- Address = Base Address + 0x0094, Reset Value = 0x0000_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
RSVD																WI15	WI14	WI13	WI14	WI11	WI10	WI9	WI8	WI7	WI6	WI5	WI4	WI3	WI2	WI1	WI0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
WIx	[x]	W	0 = No Effect 1 = Clear the external wakeup interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

7.6.1.34 CM_NISR0/1 (NVIC Interrupt Status Register 0/1)

7.6.1.34.1 CM_NISR0 (NVIC Interrupt Status Register 0)

- Address = Base Address + 0x0098, Reset Value = 0x0000_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
NVIC31	NVIC30	NVIC29	NVIC28	NVIC27	NVIC26	NVIC25	NVIC24	NVIC23	NVIC22	NVIC21	NVIC20	NVIC19	NVIC18	NVIC17	NVIC16	NVIC15	NVIC14	NVIC13	NVIC12	NVIC11	NVIC10	NVIC9	NVIC8	NVIC7	NVIC6	NVIC5	NVIC4	NVIC3	NVIC2	NVIC1	NVIC0
0	0	0	0	0	0	0	0	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

7.6.1.34.2 CM_NISR1 (NVIC Interrupt Status Register 1)

- Address = Base Address + 0x009C, Reset Value = 0x0000_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
NVIC63	NVIC62	NVIC61	NVIC60	NVIC59	NVIC58	NVIC57	NVIC56	NVIC55	NVIC54	NVIC53	NVIC52	NVIC51	NVIC50	NVIC49	NVIC48	NVIC47	NVIC46	NVIC45	NVIC44	NVIC43	NVIC42	NVIC41	NVIC40	NVIC39	NVIC38	NVIC37	NVIC36	NVIC35	NVIC34	NVIC33	NVIC32
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
NVICx	[x]	RW	Control Interrupt to NVIC of Cortex-M3 0 = NVICx interrupt does not occur. 1 = NVICx interrupt occurs. This register will be in effect when IDLEW bit of CM_SR is set to '1'. If IDLEW bit is set to '0', Interrupt signal will be handed over to NVIC regardless of this register. To generate interrupt, values of Interrupt Set Register of CPU have to be copied to this register when IDLEW bit is set to '1' (0XE000E100 value to CM_NISR0, 0xE000E104 to CM_NISR1)	0

8 Data Flash Controller

8.1 Overview

The S3FM02G has an on-chip data flash internally. The memory flash size is 16KBytes.

8.1.1 Features

- Flash memory size: 16 Kbytes
- Read access in 8, 16 or 32 bits
- Program Size: 8, 16 or 32 bits
- Page Size: 256 Bytes (32 words)
- Sector Size: 1 Kbytes
- Support page, sector and entire data flash erase
- Data Flash Access Time: Max. 100ns(AHB Bus Clock, HCLK)
- Standby: used to decrease the power consumption when the flash is not used.

8.2 Functional Description

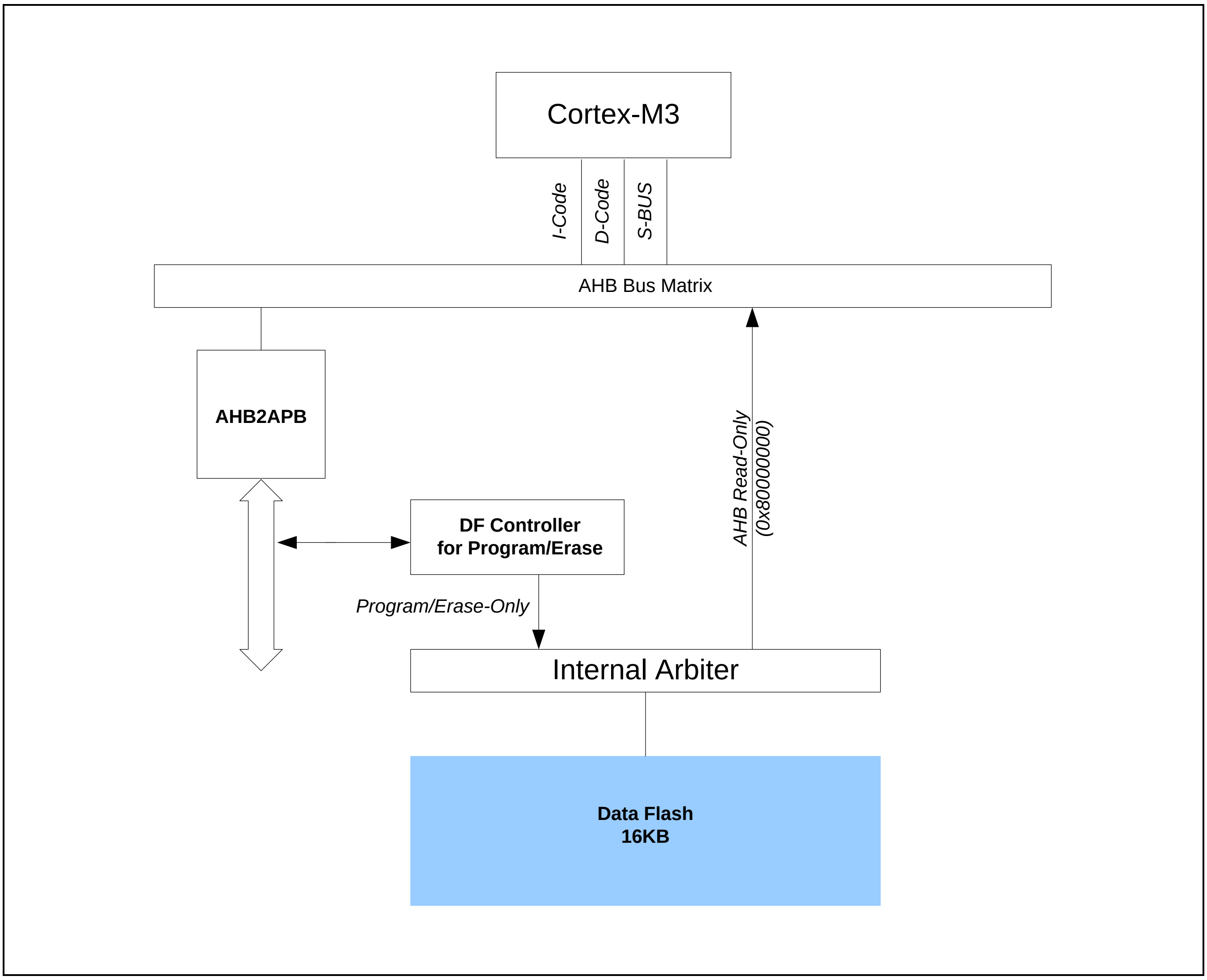
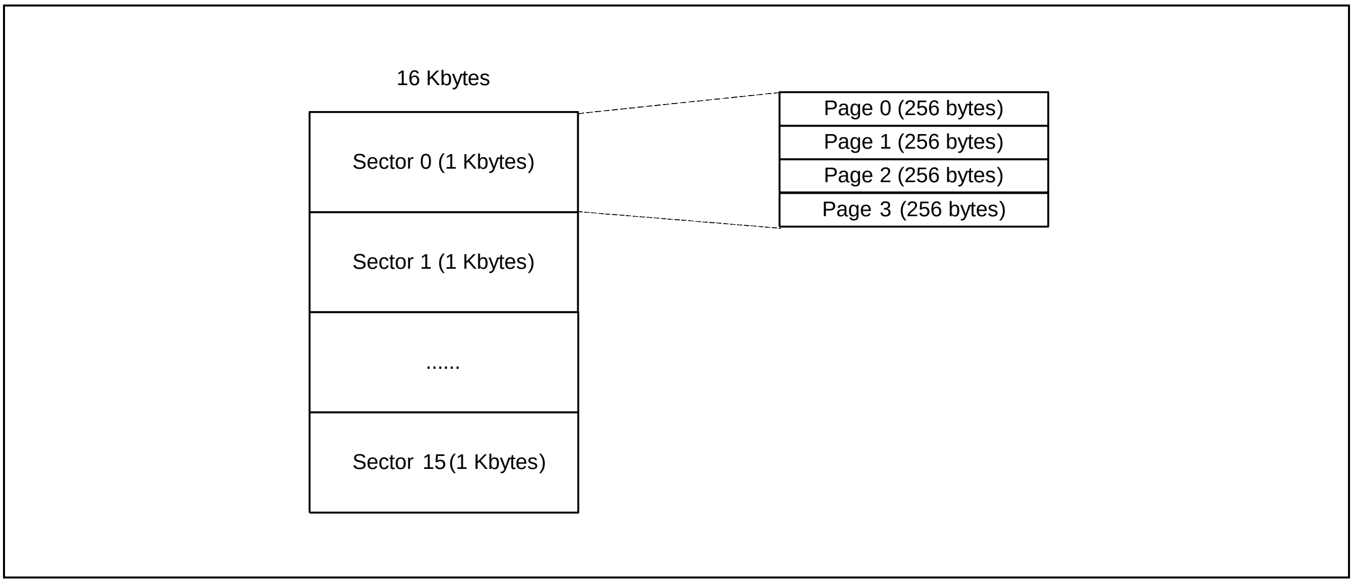


Figure 8-1 DFLASH Block Diagram

8.2.1 Organization

The 16KBytes data flash ROM consists of 16 sectors. Each sector consists of 4 pages defined as 256byte size.

The internal data flash controller allows to read or write a data of a data flash memory. The access can be done by 8, 16 or 32 bits.



8.2.1.1 Address Alignment

To set an address value in DF_AR register, abide by the following rules.

8.2.1.1.1 Program

When programming data flash, user can select one as the ADDRESS from 0x0000 to 0x3FFF by setting DF_AR, in 16Kbytes range.

8.2.1.1.2 Page Erase and Sector Erase

To erase any sector or page, user can select base address by setting DF_AR, in 16Kbytes range.

8.2.1.2 Programming Method

User can program (write) the data in a data flash memory with several programming methods.

- User program mode
- Program through JTAG interface
- Programming with UART interface.
- Programming with Flash Writing Tool (Serial interface)

8.2.2 Operation

8.2.2.1 Read Operation

Read access can be done in 8, 16 or 32 bits.

8.2.2.2 Normal Program Operation

The normal program operation writes one byte or half-word or word to the target address selected by DF_AR register. At first, user should write the value 0xA5A5A5A5 into the DF_KEY register for program. Next user should select a target address by writing to DF_AR register. The operation can execute when CMD[2:0] field should be written 001'b or 010'b or 011'b and set START bit. User can check by END bit for checking the operation is completed or not.

During a normal program operation, the busy bit in the DF_SR is set to high and automatically cleared at the end of the operation.

8.2.2.3 Page Erase Operation

When erasing a page, the lower 8-bits of address should be '0', because the size of a page is 256Bytes. User can select one as the **PAGE_ORDER** from 0 to 63, among 64 pages.

A Page has 256bytes Size.

Page Number	Page Address
0	0x8000 0000
1	0x8000 0100
2	0x8000 0200
3	0x8000 0300
4	0x8000 0400
5	0x8000 0500
6	0x8000 0600
7	0x8000 0700
8	0x8000 0800
…	…
60	0x8000 3CFF
61	0x8000 3DFF
62	0x8000 3EFF
63	0x8000 3FFF

At first, user should write the value 0xA5A5A5A5 into the DF_KEY register. Next user should select a target page to be erased by setting DF_AR register. The operation can execute when CMD[2:0] field should be written 100'b and set START bit. User can check by END bit, if whether the operation is completed or not.

8.2.2.4 Sector Erase Operation

A sector has 1Kbytes size. When erasing a sector, the lower 10-bits of address should be ‘0’, because the size of a sector is 1KBytes. At first, user should write the value 0xA5A5A5A5 into the DF_KEY register. Next user should select a target sector to be erased by setting DF_AR register. The operation can execute when CMD[2:0] field should be written 101’b and set START bit. User can check by END bit, if whether the operation is completed or not.

8.2.2.5 Entire Data Flash Erase Operation

At first, user should write the value 0xA5A5A5A5 into the DF_KEY register. In case of entire data flash erase, it doesn’t need to select the address and data. The operation can execute when CMD[2:0] field should be written 110’b and set START bit. User can check by END bit, if whether the operation is completed or not.

8.2.2.6 Flow Chart - Normal Program

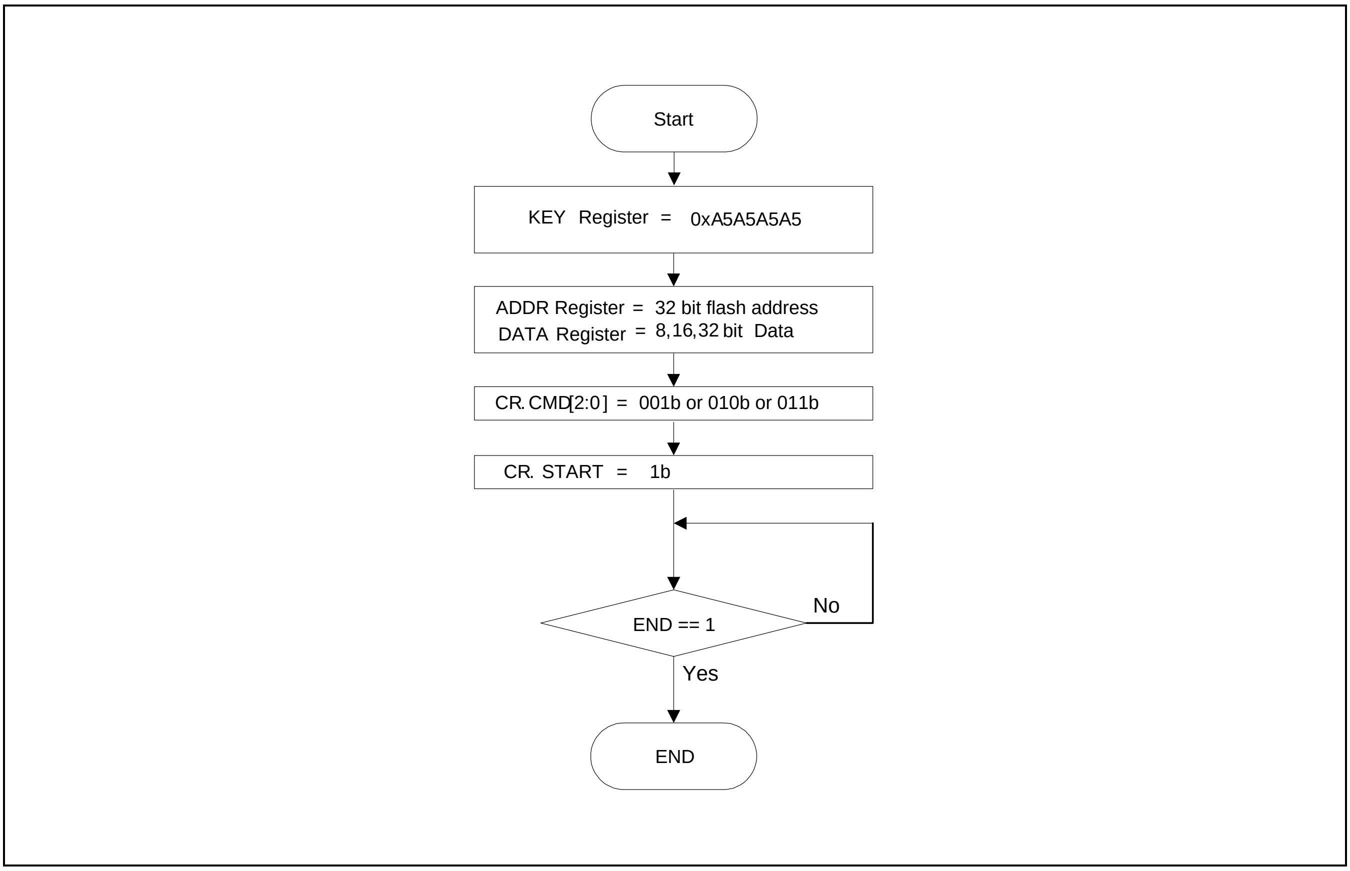

Start
KEY Register = 0xA5A5A5A5
ADDR Register = 32 bit flash address
DATA Register = 8,16,32 bit Data
CR. CMD[2:0] = 001b or 010b or 011b
CR. START = 1b
END == 1
No
Yes
END

Figure 8-2 Normal Program Flow Chart

8.2.2.7 Flow Chart – Page Erase

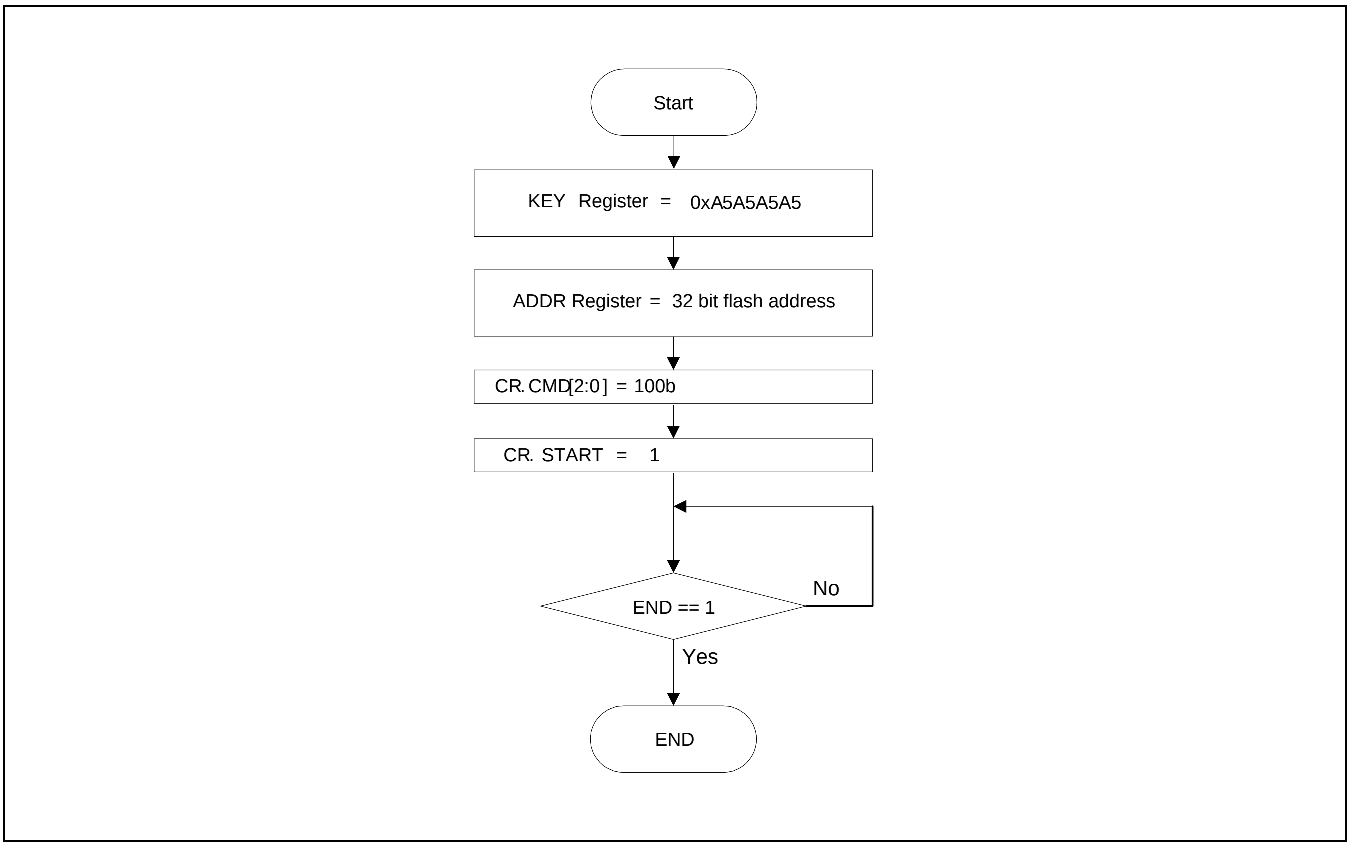

Start
KEY Register = 0xA5A5A5A5
ADDR Register = 32 bit flash address
CR. CMD[2:0] = 100b
CR. START = 1
END == 1
No
Yes
END

Figure 8-3 Page Erase Flow Chart

8.2.2.8 Flow Chart - Sector Erase

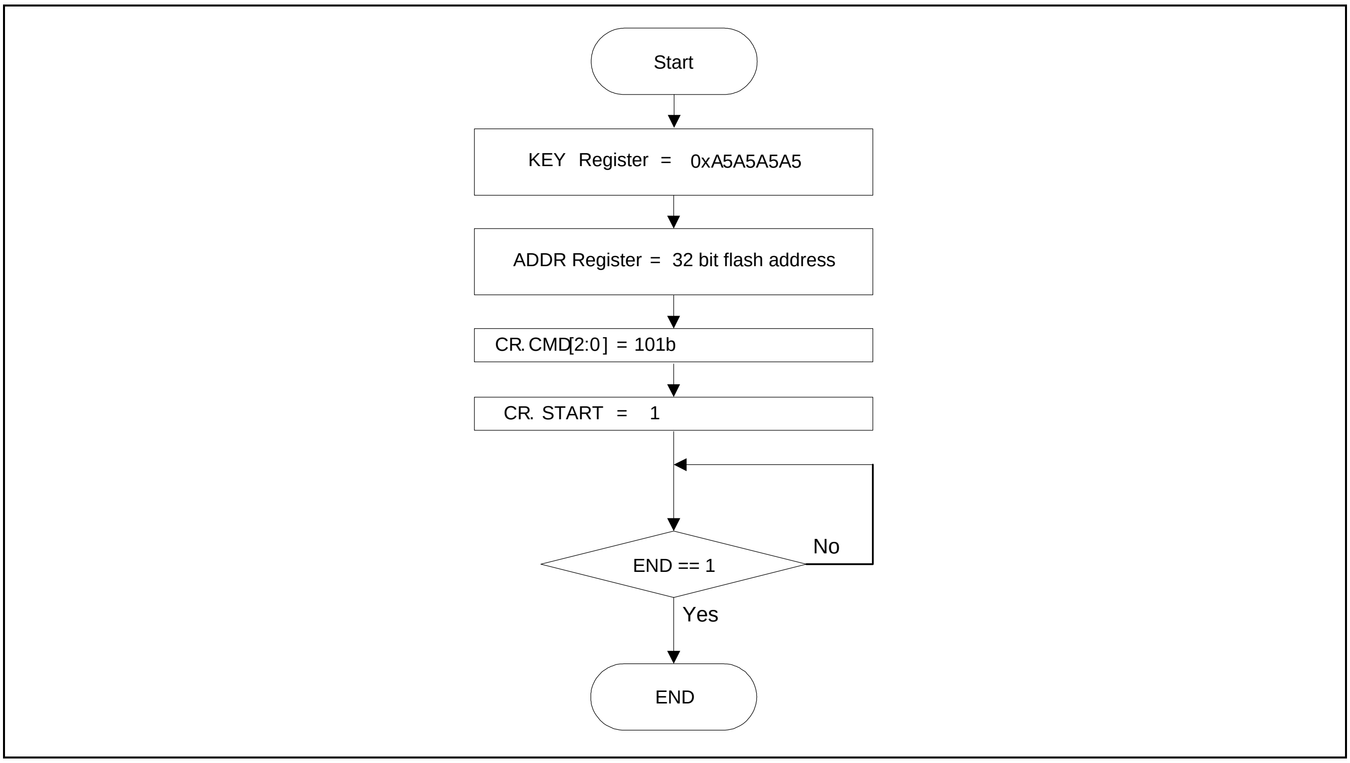

Start
KEY Register = 0xA5A5A5A5
ADDR Register = 32 bit flash address
CR. CMD[2:0] = 101b
CR. START = 1
END == 1
No
Yes
END

Figure 8-4 Sector Erase Flow Chart

8.2.2.9 Flow Chart – Entire Data Flash Erase

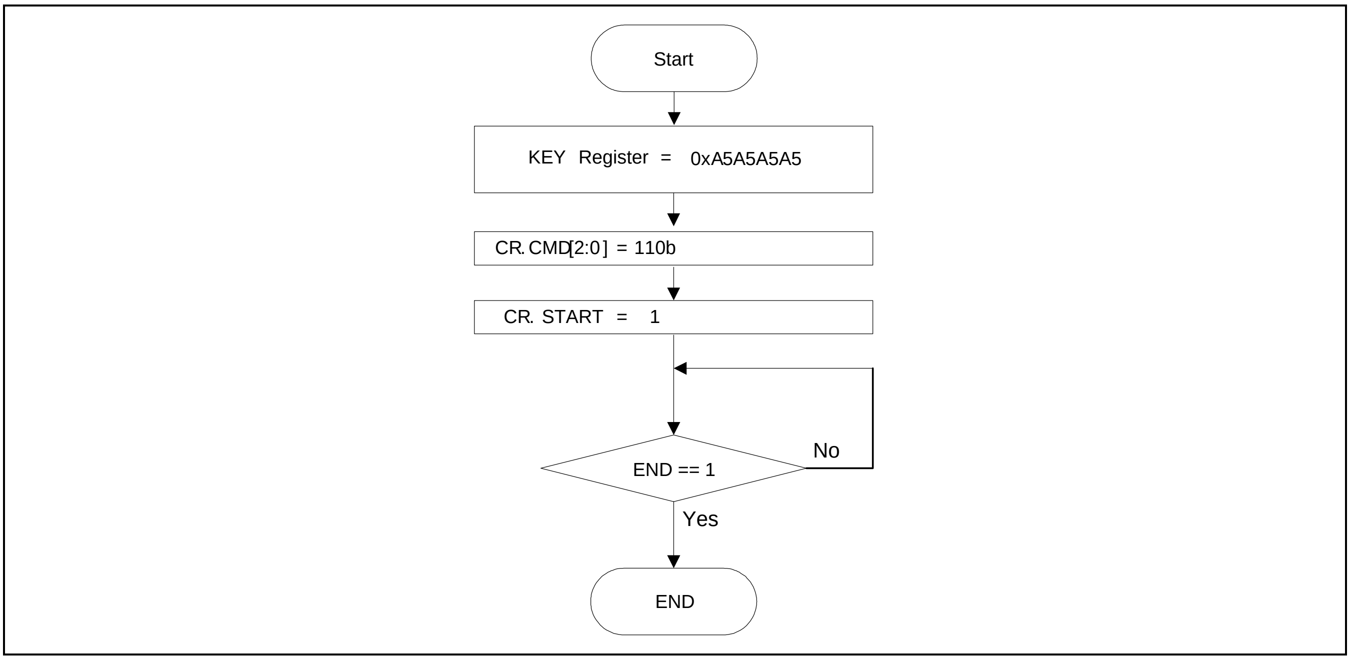


Figure 8-5 Entire Data Flash Erase Flow Chart

8.2.2.10 Error Case 0

When Code0 is not finished about writing and erasing and other program or erase operation is executed, error case 0 is occurred.

8.2.2.11 Error Case 1

When Code0 executes undefined command, error case 1 is occurred.

8.2.2.12 Error Case 2

When Code0 executes programming or erasing to protected area, error case 2 is occurred.

8.2.3 Data Protection

To protect data on data flash memory, user can do it by a protection control register. Protection can be selected partially. To set the protection bit, user should write with a specific key value, 0xA53C.

8.3 Register Description

8.3.1 Register Map Summary

- Base Address: 0x4001_1000

Register	Offset	Description	Reset Value
DF_IDR	0x000	ID Register	0x0000_2213
DF_CEDR	0x004	Clock Enable/Disable Register	0x0000_0000
DF_SRR	0x008	Software Reset Register	0x0000_0000
DF_CR	0x00C	Control Register	0x0000_0000
DF_MR	0x010	Mode Register	0x0000_0000
DF_IMSCR	0x014	Interrupt Mask Set/Clear Register	0x0000_0000
DF_RISR	0x018	Raw Interrupt Status Register	0x0000_0000
DF_MISR	0x01C	Masked Interrupt Status Register	0x0000_0000
DF_ICR	0x020	Interrupt Clear Register	0x0000_0000
DF_AR	0x024	Address Register	0x0000_0000
DF_DR	0x028	Data Register	0x0000_0000
DF_KR	0x02C	Key Register	0x0000_0000
DF_PCR	0x030	Protection Control Register	0x0000_0000

8.3.1.1 DF_IDR (D-Flash ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0000_2213

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

Name	Bit	Type	Description	Reset Value
IDCODE[25:0]	[25:0]	R	Identification Code Register This field stores the ID code for the corresponding IP.	0x0000_2213

8.3.1.2 DF_CEDR (D-Flash Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CKEN	[0]	RW	Clock Enable Bit 0 = Disable Data Flash Clock 1 = Enables Data Flash Clock Data Flash software reset dose not affect CKEN bit status.	0

8.3.1.3 DF_SRR (D-Flash Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset 0 = No effect 1 = Perform Software Reset operation	0

8.3.1.4 DF_CR (D-Flash Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																								CMD[2:0]				RSVD	DFSTABLE	DFEN	START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
START	[0]	RW	Operation(note) Start Bit After start and finish, selected operation command and START bit will be clear automatically. 0 = No effect 1 = Start	0
DFEN	[1]	RW	Data Flash Enable/Disable Control Bit 0 = Disable data flash for a power consumption. (Standby Mode) 1 = Enable data flash	0
DFSTABLE	[2]	RW	Data Flash Stable Bit 0 = Data flash is not stable. 1 = Data flash is stable. **NOTE:** After DFEN, 100us is needed for data flash stabilization. Data flash must be read or written only if DFSTABLE is 1.	0
CMD[3:0]	[7:4]	RW	Flash Program/Erase Command Field 000 = No Effect 001 = select ‘Byte Program’ Command 010 = select ‘Half-word Program’ Command 011 = select ‘Word Program’ Command 100 = select ‘Page Erase’ Command 101 = select ‘Sector Erase’ Command 110 = select ‘Entire Data Flash Erase’ Command 111 = Prohibited	0x0

NOTE: S3FM02G supports the following program.

Command	Description
NP	Normal Program
PE	Page Erase
SE	Sector Erase
EDFE	Entire Data Flash Erase

The DF_CR can determine the program / erase operation. In user program mode, data flash controller can support program, page erase, sector erase and entire data flash erase. Among 4 operations, only one operation must be selected.

8.3.1.5 DF_MR (D-Flash Mode Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																												WAIT[3:0]			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
WAIT[3:0]	[3:0]	RW	Data Flash Access Wait Cycle 0000 = 0 Wait Cycle during the Data Flash Read Access 0001 = 1 Wait Cycle during the Data Flash Read Access 0010 = 2 Wait Cycle during the Data Flash Read Access … 1110 = 14 Wait Cycle during the Data Flash Read Access 1111 = 15 Wait Cycle during the Data Flash Read Access	0x0

NOTE: WAIT[3:0] must be set the value when HCLK is over 10MHz (100ns: Max. Data Flash Access Time).
This equation: `(WAIT[3:0] + 1) * 10MHz(100ns) ≥ period of HCLK.`
for example: `HCLK = 75MHz, (Wait[3:0] + 1) * 10MHz ≥ 75MHz, Wait[3:0] = 7`

8.3.1.6 DF_IMSCR (D-Flash Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
END	[0]	RW	END Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	0
ERRn	[10:8]	RW	ERR Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	000'b

8.3.1.7 DF_RISR (D-Flash Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
END	[0]	R	END Interrupt Mask Status 0 = Disabled 1 = Enabled	0
ERRn	[10:8]	R	ERR Interrupt Mask Status 0 = Disabled 1 = Enabled	000'b

8.3.1.8 DF_MISR (D-Flash Masked Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
END	[0]	R	END Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	0
ERRn	[10:8]	R	ERR Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	000'b

8.3.1.9 DF_ICR (D-Flash Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
END	[0]	W	END Interrupt Clear Bit 0 = No effect 1 = Clear Interrupt	0
ERRn	[10:8]	W	ERR Interrupt Clear Bit 0 = No effect 1 = Clear Interrupt	000'b

8.3.1.10 DF_AR (D-Flash Address Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
ADDR[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADDR[31:0]	[31:0]	RW	Data Flash Memory Address to program(Normal) or erase(Page/Sector/Entire Data Flash) PF_AR [31:0] ← Address to be selected in data flash memory **NOTE:** For word write, ADDR[1:0] is no effect. For half word write, ADDR[0] is no effect.	0x0000_0000

8.3.1.11 DF_DR (D-Flash Data Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
DATA[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA[31:0]	[31:0]	RW	Data to write on the target address of the data flash memory. **NOTE:** For byte write, only DATA[7:0] is valid. For half word write, only DATA[15:0] is valid.	0x0000_0000

8.3.1.12 DF_KR (D-Flash Key Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
KEY[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
KEY[31:0]	[31:0]	W	Key register value for program, erase, protection operation To program any data into the flash memory by the user program mode, a specific key register with 0xA5A5A5A5 is required to prevent flash data from being destroyed under undesired situations. **NOTE:** The DF_KR register will be cleared automatically just after the completion of erase or program.	0x00000000

8.3.1.13 DF_PCR (D-Flash Protection Control Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
PRKEY[15:0]																WEPR15	WEPR14	WEPR13	WEPR12	WEPR11	WEPR10	WEPR9	WEPR8	WEPR7	WEPR6	WEPR5	WEPR4	WEPR3	WEPR2	WEPR1	WEPR0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
WEPRn	[15:0]	RW	Write/Erase Protection Region Selection Bit Protection can be selected partially. 0 = Protection is disabled. 1 = Protection is enabled. **NOTE:** WEPRn is matched with sector n.	0x0000
PRKEY	[31:16]	W	To set the protection bit, user should write with a specific key value, 0xA53C.	0x0000

9 Direct Memory Access (DMA) Controller

9.1 Overview

This section describes DMA operation.

The on-chip DMA controller can be started by two ways. One is by software, which is achieved by programming DMA internal register by CPU through system bus. The other is by DMA request from I/O devices such as USART0/1/2/3, SPI0/1, IIC0/1, and ADC0/1 which is achieved by DMA request and acknowledge mechanism between I/O device and DMA.

DMA operation can also be stopped and restarted by software. The CPU can recognize when a DMA operation has been completed by software polling or by a DMA interrupt request. The DMA controller can increase or decrease source or destination addresses and conduct 8-bit (byte), 16-bit (half-word) or 32-bit (word) data transfers.

- Both source and destination are in the system bus
  - (ex, memory to memory transfer. But, Flash memory can only be source)
- Source is in the system bus while destination is in the peripheral bus
  - (ex, memory to an I/O device transfer)
- Source is in the peripheral bus while destination is in the system bus
  - (ex, I/O device to memory transfer)
- Both source and destination are in the peripheral bus
  - (ex, I/O device to I/O device transfer)

9.2 Functional Description

9.2.1 Block Diagram

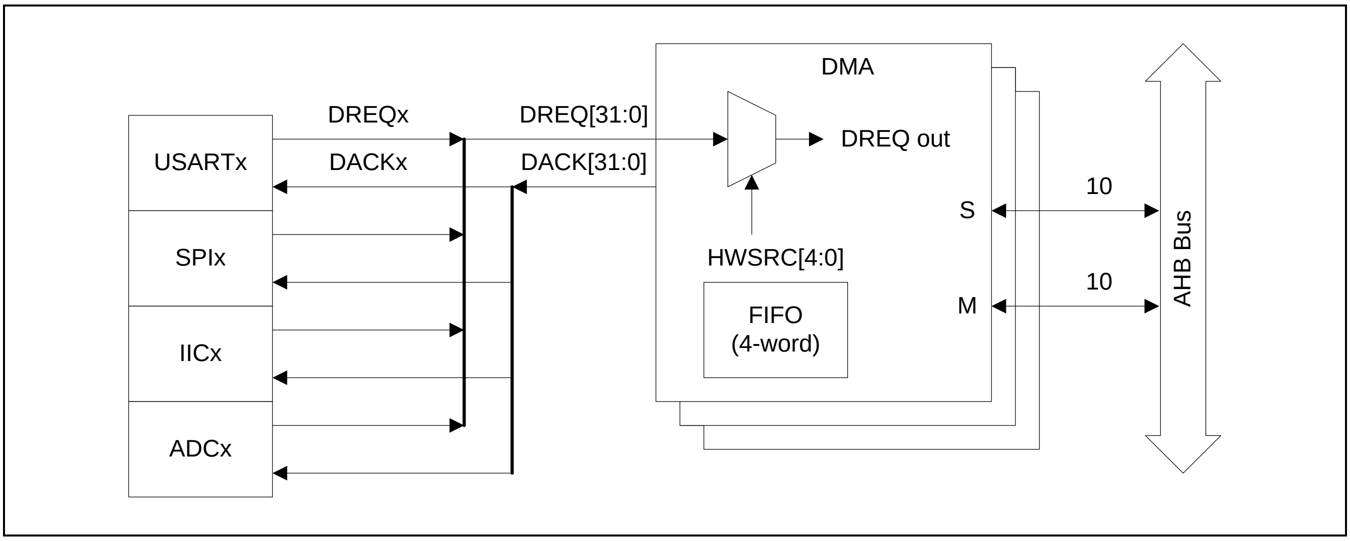


Figure 9-1 DMA Block Diagram

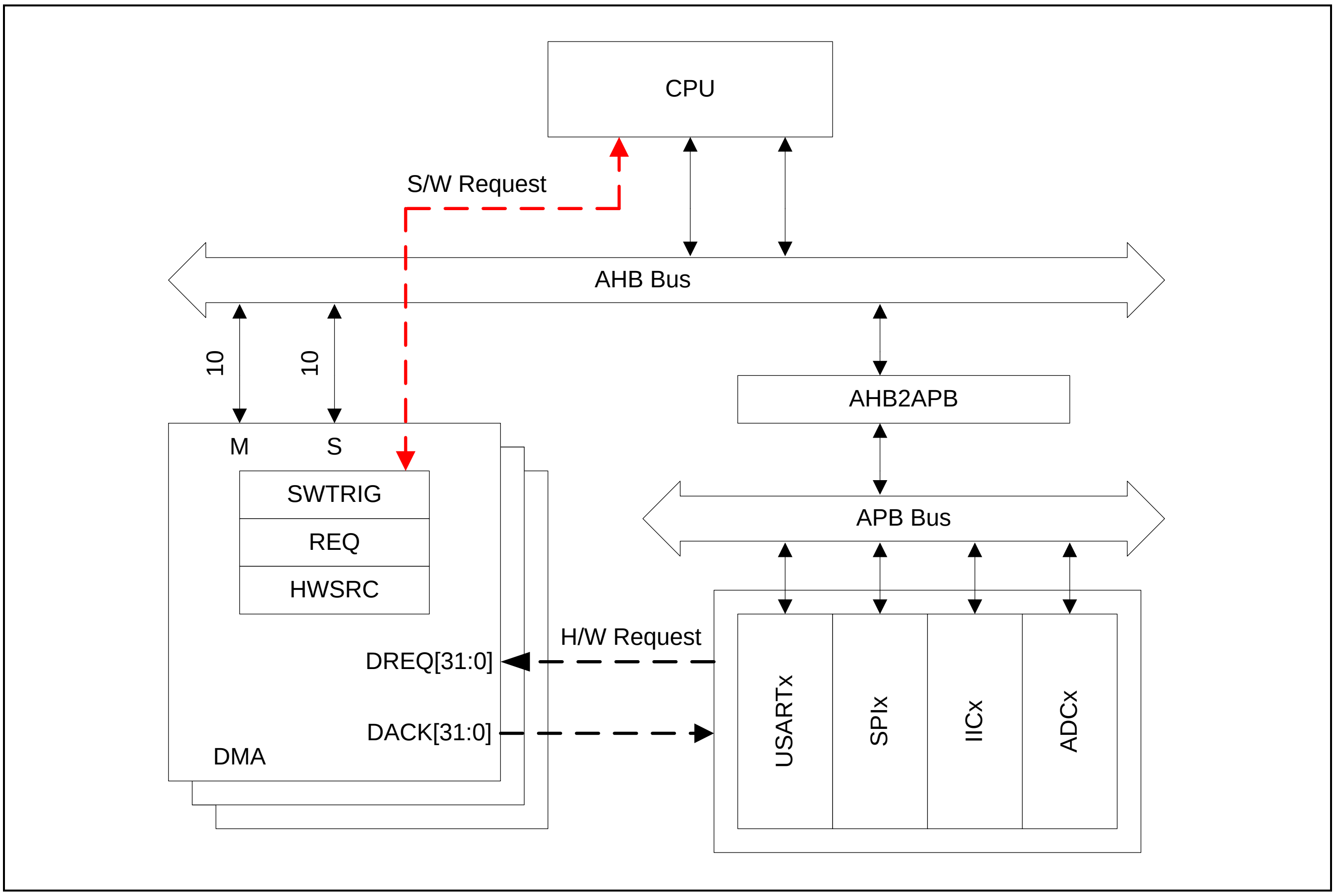


Figure 9-2 DMA Request

9.2.2 General Operation

The DMA operation can be summarized as follows:

- DMA transfers
- Starting/Ending DMA transfers

The priority of channel is the same the number of channel. That means channel 0 has the highest priority and channel 9’s priority is the lowest. So if user wants to use USART_TX with the highest priority, user selects the channel 0 and fills out HWSRC[4:0] with USART_TX value.

9.2.2.1 DMA Transfer

The DMA (Direct Memory Access) transfers data directly between source and destination. The source or the destination is SRAM, USART, SPI or IIC. Flash memory and ADC can only be source.

A channel is programmed by writing the DMA control registers, which contain control information such as source address, destination address, and the amount of data.

DMA operation is triggered by two methods. One is S/W request by CPU, and the other is H/W request by I/O device. The following I/O devices can send H/W request to DMA: USART0/1/2/3, SPI0/1, IIC0/1 and ADC. Each device is internally connected to the DMA to request DMA service.

9.2.2.2 Starting/Ending DMA Transfers

DMA starts to transfer data after the DMA receive request from USART0/1/2/3, SPI0/1, IIC0/1, ADC or software. When the entire data programmed in DMACNT has been transferred, the DMA become idle. If user wants to perform another transfer, the DMA must be reprogrammed. When the same transfer is performed again, the DMA must be reprogrammed.

Before the start of DMA operation, DMA internal registers such as source/destination address, control, and transfer count register should be programmed.

In case of S/W request, the DMA operation will start as soon as both CHEN and SWTRIG bit in DMA_MTRx (Mask Trigger Register) register set to 1.

In case of H/W request, the DMA waits until desired I/O device sends H/W request after CHEN bit in DMA_MTRx (Mask Trigger Register) register set to 1. Only after receiving the H/W request, the DMA operation will start based on DMA configurations.

In DMA, there are the following H/W request sources: USART0/1/2/3, SPI0/1, IIC0/1, and ADC. The H/W request source should be set only one I/O device at a time. H/W requests from other I/O devices are ignored.

The DMA request mode can be configured by HWSRC[4:0] bits of DMA_RSRx (Request Selection Register) register.

9.2.2.3 S/W Request Operation

The details of DMA S/W request operation is as follows:

- State-1: As an initial state, it waits for both CHEN and SWTRIG bits in DMA_MTRx register set to 1.
- State-2: In this state, current transfer counter (CURR_HTC[11:0] and CURR_LTC[11:0]) is loaded from the HTC[11:0] and LTC[11:0] and SWTRIG value is automatically cleared.
- State-3: In this state, the atomic operation of DMA handled by sub-FSM (Finite State Machine) is initiated. The sub-FSM reads the data from the source address and then writes it to destination address. In this operation, data size (byte, half-word, or word) and transfer size (single or burst) are considered. This operation is repeated until the current transfer counter(CURR_HTC[11:0] x CURR_LTC[11:0]) reaches 0 in Continuous Mode(SMODE set to 1),, while performed only once in a Single Service Mode. The main FSM counts down the current transfer counter when the sub_FSM finishes each atomic operation.

9.2.2.4 H/W Request Operation

The details of DMA H/W request operation can be explained based-on the following 4-state FSM (finite state machine):

- State-1: As an initial state, it waits for CHEN bit in DMA_MTRx register set to 1
- State-2: In this state, it waits for DMA REQ from desired I/O device. Desired I/O device is determined by HWSRC[4:0] bits of DMA_RSRx register
- State-3: In this state, DMA ACK becomes high and current transfer counter (CURR_HTC[11:0] and CURR_LTC[11:0]) is loaded from the HTC[11:0] and LTC[11:0].
  Note that DMA ACK becomes high and remains high until it becomes low later.
- State-4: In this state, the atomic operation of DMA handled by sub-FSM is initiated. The sub-FSM reads the data from the source address and then writes it to destination address. In this operation, data size (byte, half, or word) and transfer size (single or burst) are considered. This operation is repeated until the current transfer counter(CURR_LTC[11:0]) reaches 0 in Continuous Mode(SVMODE set to 1), while performed until only once in a Single Service Mode. The main FSM (this FSM) counts down the current transfer counter when the sub-FSM finishes each atomic operation.
  In addition, this main FSM asserts the LTCINT REQ signal when CURR_LTC[11:0] becomes 0. It asserts the TCINT REQ signal when CURR_HTC[11:0] and CURR_LTC[11:0] become 0. In addition, the DMA ACK becomes low if one of the following conditions are met.
  - CURR_LTC[11:0] becomes 0 in continuous mode
  - Atomic operation finishes in the single service mode

Note that in the single service mode, these three states (State-2 to State-4) of main FSM are performed once, then stops, and waits for another DMA REQ. And if another DMA REQ occurs, all the three states are repeated. Therefore, DMA ACK is asserted and then de-asserted for each atomic transfer. In contrast, in the whole service mode, main FSM waits at state-3 until CURR_HTC[11:0] and CURR_LTC[11:0] become 0. Therefore, DMA ACK is asserted during all the transfers and then de-asserted when the transfer counter reaches 0.

However, LTCINT REQ is asserted only if CURR_LTC[11:0] becomes 0 regardless of the service mode (single service mode or continuous mode). TCINT REQ is asserted when CURR_HTC[11:0] and CURR_LTC[11:0] become 0.

The DMA also provides a temporary buffer which allows multiple transfers to enhance the bus utilization as well as transfer speed. Specifically, it has a 4-word FIFO-type buffer to support the 4-word burst transfer during DMA operation. For example, the DMA operation between memories can be done by a 4-word burst write followed by a 4-word burst read.

9.2.3 Operation Mode

9.2.3.1 Single Service Mode

The single service mode needs DMA request and acknowledge handshake every atomic DMA read and write operation.

When the DMA request signal goes high, the DMA controller indicates the acknowledge to the I/O device by asserting the DMA acknowledge signal high. During the first high level period of the DMA acknowledge signal, a DMA read cycle will be initiated. After the DMA read cycle, the next DMA write cycle follows.

The DMA ACK signal is asserted until the atomic operation (i.e., read followed by write operation) is finished.

The TCINT REQ signal is asserted only if current transfer counter (CURR_HTC[11:0] & CURR_LT[11:0]) becomes 0. The LTCINT REQ is asserted when CURR_LTC[11:0] becomes 0.

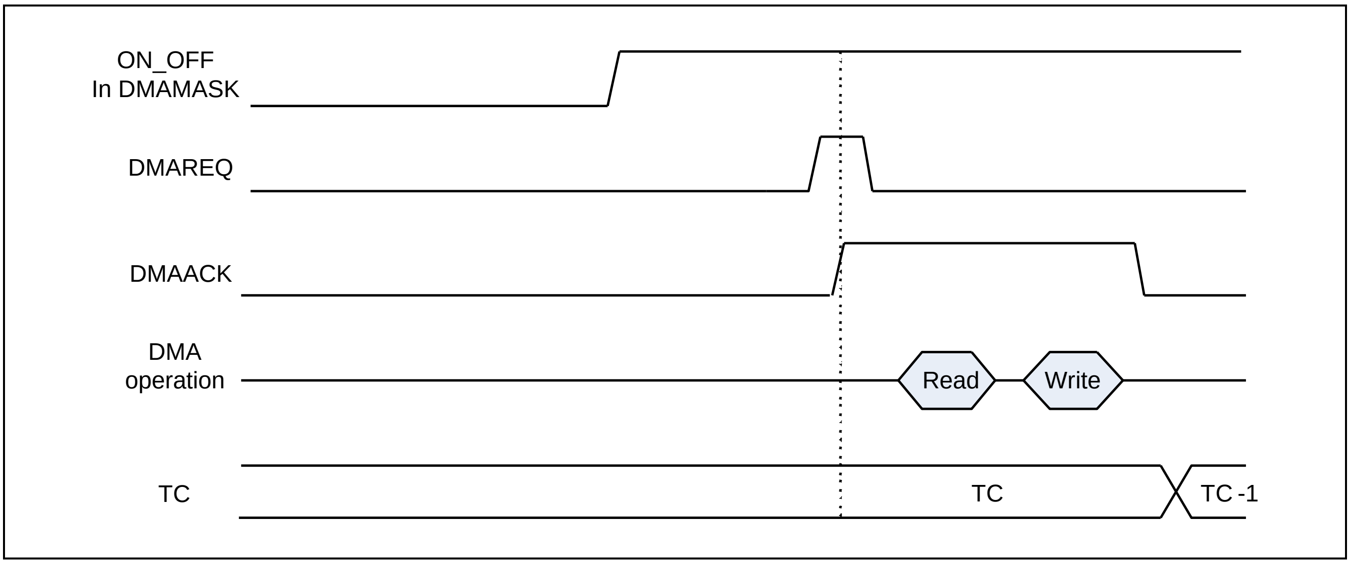


Figure 9-3 Single Transfer in Single Service Mode

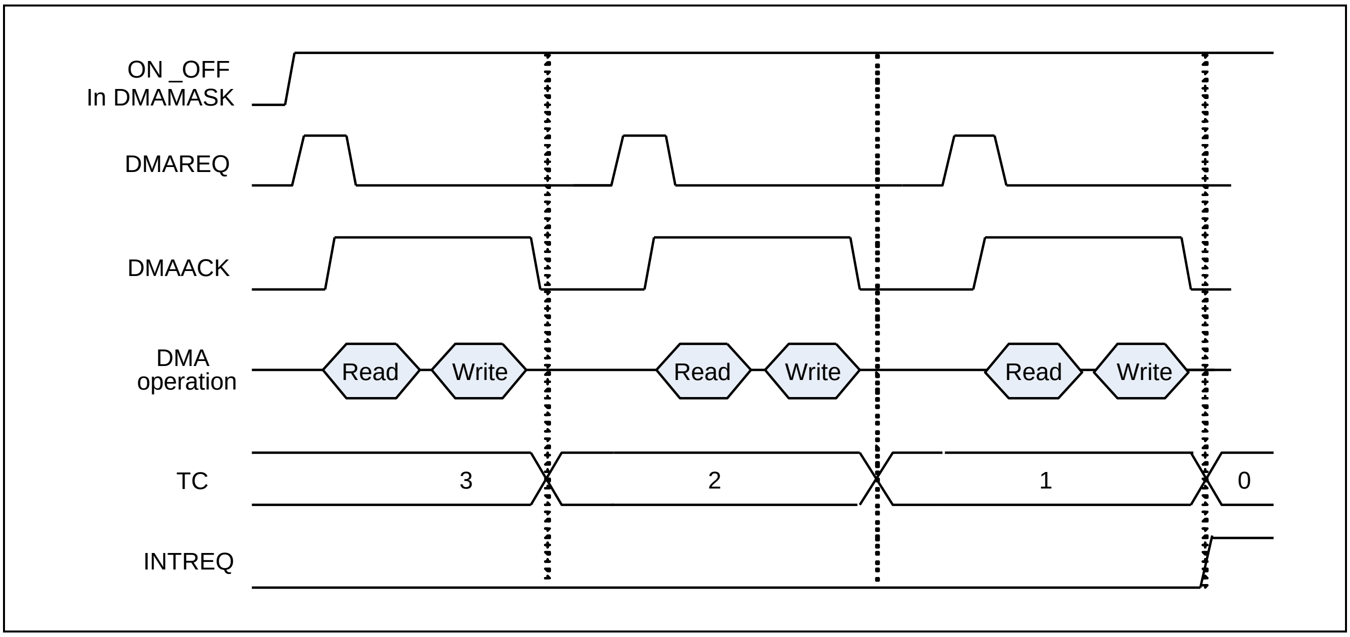


Figure 9-4 Sequential Transfer in Single Mode

9.2.3.2 Continuous mode

The continuous mode means that the specified number of DMA operations, i.e., number of DMA operations based on transfer count, will be initiated by a single activation of DMA request, and will proceed without further activation of DMA requests. The below figure shows how the continuous mode proceeds. The DMAACK signal remains high until the end of whole DMA operations.

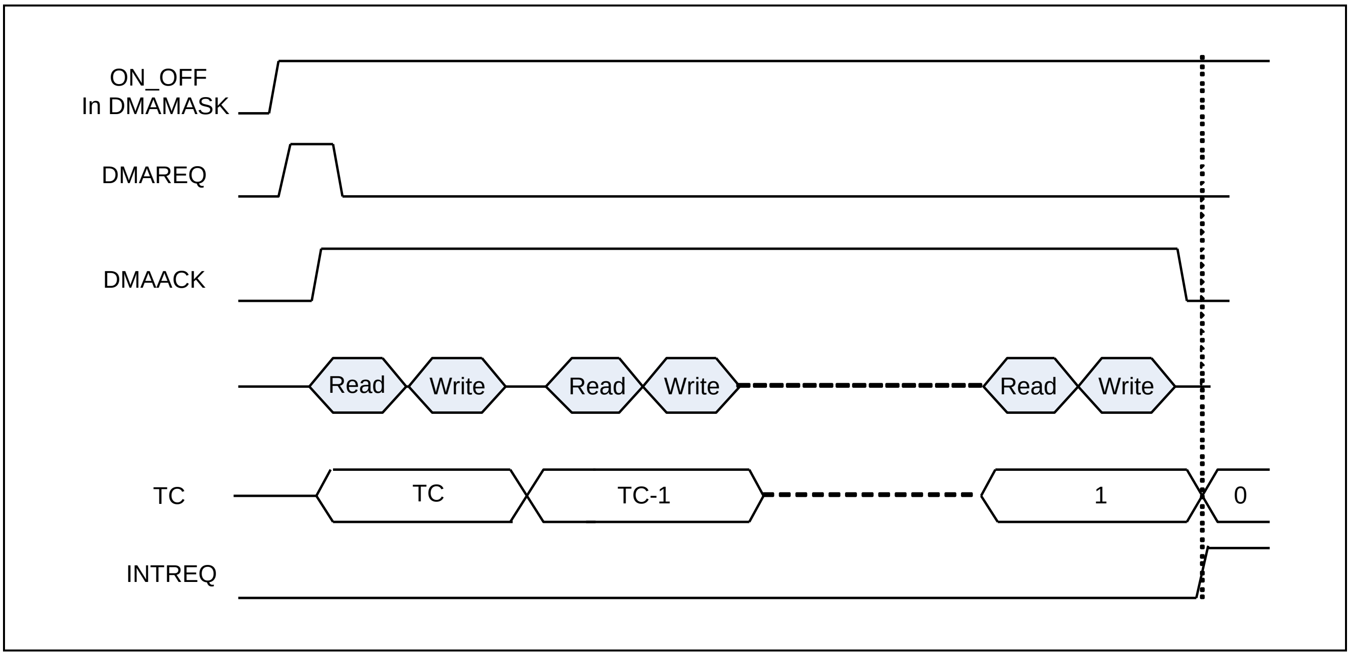


Figure 9-5 Continuous Service Mode

9.2.4 Data Transfer Size

There are two types of DMA transfer size (Single transfer size, Burst transfer size). Unlike DMA request/ acknowledge protocol, the DMA transfer size defines the number of read/write per unit transfer as shown in the following table.

Table 9-1 DMA Control Special Function Registers

DMA Transfer Size	Read/Write
Single transfer	1 unit read, then 1 unit write
Burst transfer	4 unit burst read, 4 unit burst write

NOTE: Unit is Byte(8bit), Half Word(16bit), or Word(32bit)

9.2.4.1 Single Transfer Size

The single transfer mode means that the paired DMA read/write cycle is performed per each DMA request as shown in *Figure 22-1*.

9.2.4.2 Burst (4-unit) Transfer Size

The burst (4-unit) transfer mode means that the successive 4-unit DMA read cycles is followed by the successive 4-unit DMA write cycles as shown in *Figure 9-6*.

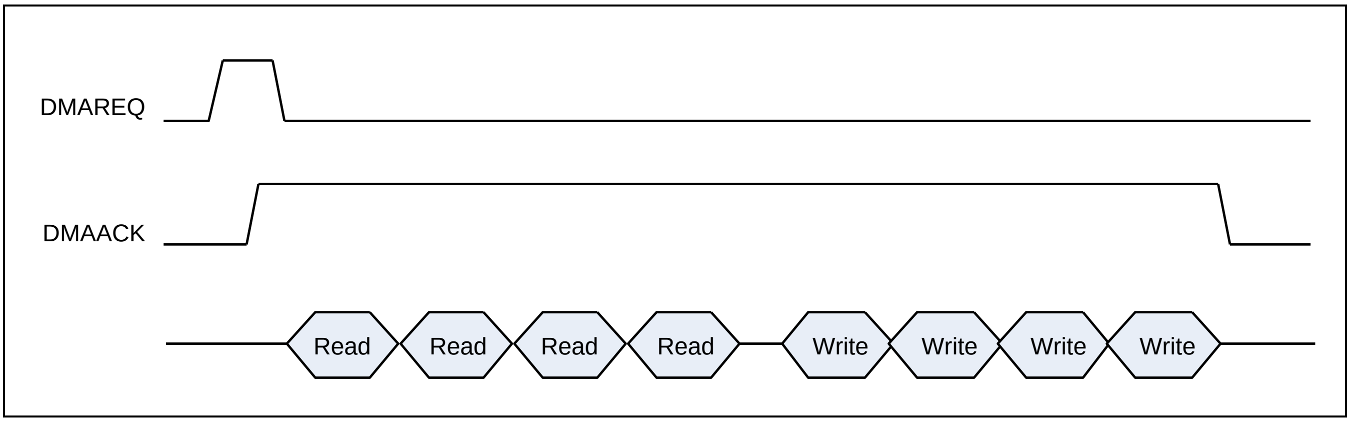


Figure 9-6 Burst (4-unit) Transfer Size

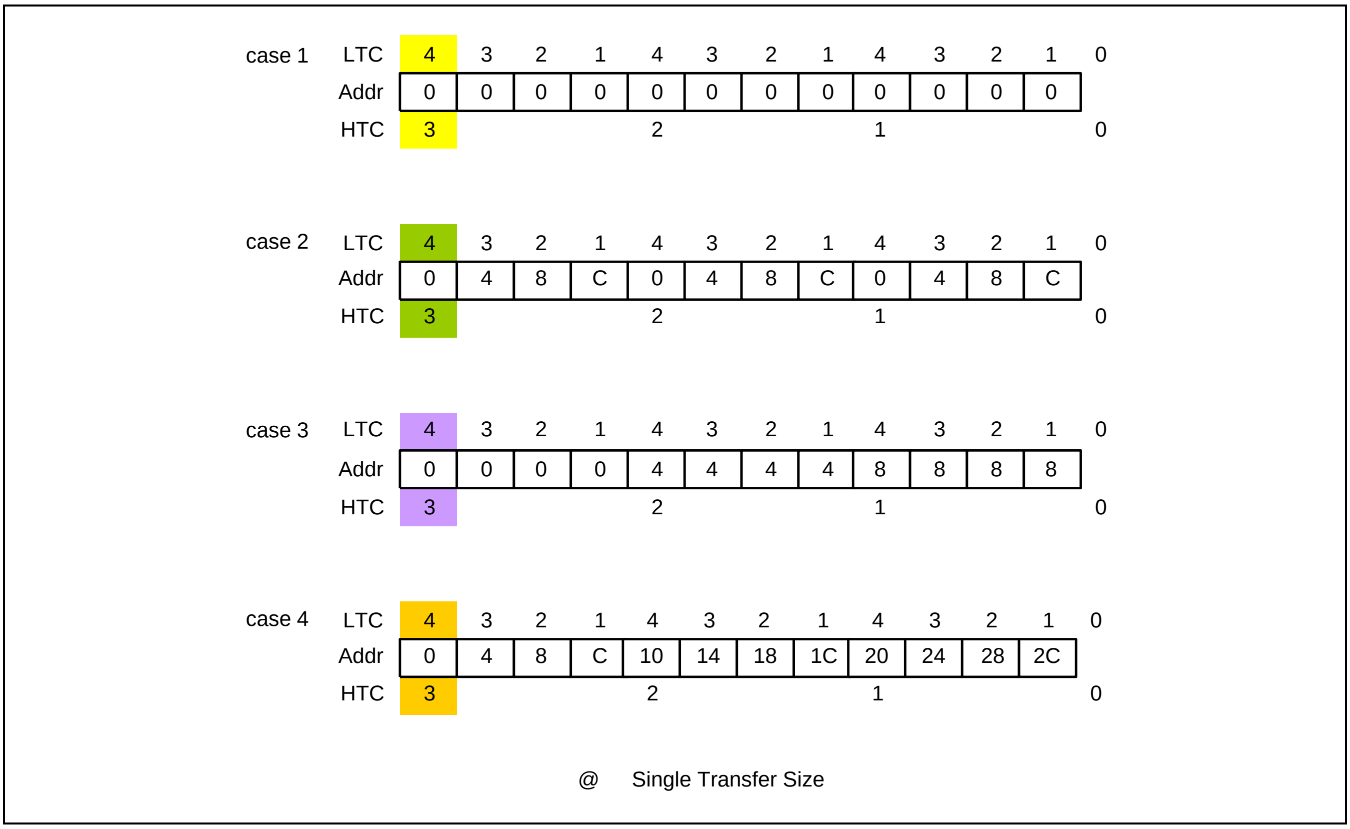
9.2.5 Transfer Count

Transfer count consists of HTC[11:0] and LTC[11:0]. HTC is a high transfer count. LTC is a low transfer count. To use DMA transfer, HTC[11:0] and LTC[11:0] can have from 1 to 4095. If either HTC[11:0] or LTC[11:0] is a zero, CHEN bit in a mask trigger register won’t be set to ‘1’.

Assume that followings: Initial address is ‘0x0’. Data size is the word. HTC has ‘3’, and LTC has ‘4’. In the figure, address, Addr, is expressed without ‘0x’. The ‘Addr’ can be the source and destination source.

Table 9-2 The Cases by the Address Control

Case	HTC Address Control	Low Address Control
1	Fix	Fix
2	Fix	Increment
3	Increment	Fix
4	Increment	Increment



case 1														
	LTC	4	3	2	1	4	3	2	1	4	3	2	1	0
	Addr	0	0	0	0	0	0	0	0	0	0	0	0	
	HTC	3				2				1				0

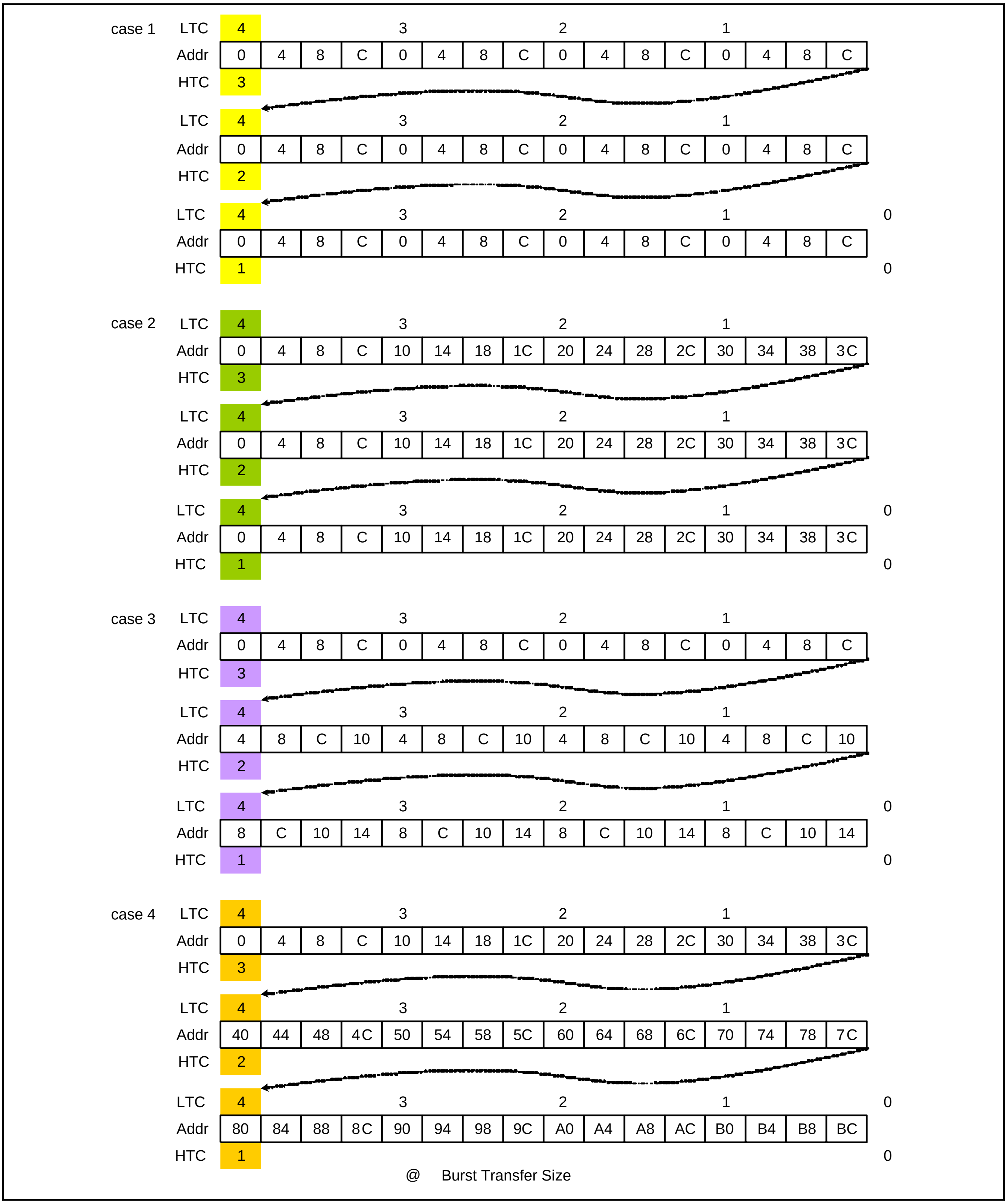
case 2														
	LTC	4	3	2	1	4	3	2	1	4	3	2	1	0
	Addr	0	4	8	C	0	4	8	C	0	4	8	C	
	HTC	3				2				1				0

case 3														
	LTC	4	3	2	1	4	3	2	1	4	3	2	1	0
	Addr	0	0	0	0	4	4	4	4	8	8	8	8	
	HTC	3				2				1				0

case 4														
	LTC	4	3	2	1	4	3	2	1	4	3	2	1	0
	Addr	0	4	8	C	10	14	18	1C	20	24	28	2C	
	HTC	3				2				1				0

@ Single Transfer Size

Figure 9-7 The Relationship Between Transfer Count and Address



case 1	LTC	4				3				2				1				
	Addr	0	4	8	C	0	4	8	C	0	4	8	C	0	4	8	C	
	HTC	3																
	LTC	4				3				2				1				
	Addr	0	4	8	C	0	4	8	C	0	4	8	C	0	4	8	C	
	HTC	2																
	LTC	4				3				2				1				0
	Addr	0	4	8	C	0	4	8	C	0	4	8	C	0	4	8	C	
	HTC	1																0
case 2	LTC	4				3				2				1				
	Addr	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	
	HTC	3																
	LTC	4				3				2				1				
	Addr	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	
	HTC	2																
	LTC	4				3				2				1				0
	Addr	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	
	HTC	1																0
case 3	LTC	4				3				2				1				
	Addr	0	4	8	C	0	4	8	C	0	4	8	C	0	4	8	C	
	HTC	3																
	LTC	4				3				2				1				
	Addr	4	8	C	10	4	8	C	10	4	8	C	10	4	8	C	10	
	HTC	2																
	LTC	4				3				2				1				0
	Addr	8	C	10	14	8	C	10	14	8	C	10	14	8	C	10	14	
	HTC	1																0
case 4	LTC	4				3				2				1				
	Addr	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	
	HTC	3																
	LTC	4				3				2				1				
	Addr	40	44	48	4C	50	54	58	5C	60	64	68	6C	70	74	78	7C	
	HTC	2																
	LTC	4				3				2				1				0
	Addr	80	84	88	8C	90	94	98	9C	A0	A4	A8	AC	B0	B4	B8	BC	
	HTC	1																0

@ Burst Transfer Size

Figure 9-8 The Relationship Between Transfer Count and Address

9.2.6 Operation Sequence

9.2.6.1 Software Sequence

- DMA Operation Condition Setting
  - Set source address and control (DMA_ISRx, DMA_ISCRx)
  - Set destination address and control (DMA_IDRx, DMA_IDCRx)
  - Set DMA control register (Interrupt, operation mode, transfer width, burst mode, transfer count)

- DMA Request Set, DMA Enable, and DMA Start
  - Set DMA request selection register (REQ = 0 (S/W))
  - Set DMA Enable and Start (DMA_MTRx: CHEN = 1, SWTRIG = 1)
  - When SWTRIG is set to ‘1’, the DMA will start operation based on DMA configurations described in 1. The SWTRIG is cleared automatically.
  - **Operation Mode (Single service Mode)**
    - After the DMA finishes atomic transaction (single or burst of length four) and decrease CURR_LTC[11:0]) by 1, DMA stops and waits until another SWTRIG is set to ‘1’
    - Repeat a until CURR_LTC[11:0] reaches to zero. When CURR_LTC[11:0] is zero, DMA decreases CURR_HTC[11:0] by 1.
    - Repeat a & b until CURR_HTC[11:0] and CURR_LTC[11:0] reach to zero.
  - **Operation Mode (Continuous service Mode)**
    - The DMA finishes whole DMA transactions (CURR_LTC[11:0] × atomic transaction) without further DMA request and decrease CURR_HTC[11:0] by 1, DMA stops and waits until another SWTRIG is set to ‘1’.
    - Repeat a until CURR_HTC[11:0] reaches to zero.

9.2.6.2 Hardware Request Sequence

- DMA Operation Condition Setting
  - Set source address and control (DMA_ISRx, DMA_ISCRx)
  - Set destination address and control (DMA_IDRx, DMA_IDCRx)
  - Set DMA control register (Interrupt, operation mode, transfer width, burst mode, transfer count)

- DMA Request Set, DMA Enable
  - Set DMA request selection register
    (DMA_RSRx: HWSRC[4:0](5’b00000(USART) ~ 5’b11000(ADC01)),REQ(1))
  - We can select desired I/O device by setting HWSRC[4:0]. H/W requests from other devices are ignored.
  - Set DMA Enable (DMA_MTRx: CHEN = 1)

- DMA Start
  - When the DMA receives H/W request from desired I/O device, the DMA operation will start automatically based on DMA configurations described in 1.
  - **Operation Mode (Single service Mode)**
    - After the DMA finishes atomic transaction (single or burst of length four) and decrease CURR_LTC[11:0]) by 1, DMA stops and waits until another H/W request from desired I/O device
    - Repeat a until CURR_LTC[11:0] reaches to zero. When CURR_LTC[11:0] is zero, DMA decreases CURR_HTC[11:0] by 1.
    - Repeat a & b until CURR_HTC[11:0] and CURR_LTC[11:0] reach to zero.
  - **Operation Mode (Continuous Service Mode)**
    - The DMA finishes whole DMA transactions (CURR_LTC[11:0] × atomic transaction) without further DMA request and decrease CURR_HTC[11:0] by 1, DMA stops and waits until another SWTRIG is set to ‘1’.
    - Repeat a until CURR_HTC[11:0] reaches to zero.

9.3 Register Description

9.3.1 Register Map Summary

- Base Address 0x400F_0000

Register	Offset	Description	Reset Value
DMA_ISRx	0x0000+0x080∗x (NOTE)	Channel x Initial Source Register	0x0000_0000
DMA_ISCRx	0x0004+0x080∗x (NOTE)	Channel x Initial Source Control Register	0x0000_0000
DMA_IDRx	0x0008+0x080∗x (NOTE)	Channel x Initial Destination Register	0x0000_0000
DMA_IDCRx	0x000C+0x080∗x (NOTE)	Channel x Initial Destination Control Register	0x0000_0000
DMA_CRx	0x0010+0x080∗x (NOTE)	Channel x Control Register	0x0000_0000
DMA_SRx	0x0014+0x080∗x (NOTE)	Channel x Status Register	0x0000_0000
DMA_CSRx	0x0018+0x080∗x (NOTE)	Channel x Current Source Register	0x0000_0000
DMA_CDRx	0x001C+0x080∗x (NOTE)	Channel x Current Destination Register	0x0000_0000
DMA_MTRx	0x0020+0x080∗x (NOTE)	Channel x Mask Trigger Register	0x0000_0000
DMA_RSRx	0x0024+0x080∗x (NOTE)	Channel x Request Selection Register	0x0000_0000
RSVD	0x0028+0x080∗x – 0x007C+0x080∗x (NOTE)	Reserved	–
RSVD	0x0300–0x04FC	Reserved	–
DMA_IDR	0x0500	DMA ID Register	0x00A2_0018
DMA_SRR	0x0504	DMA Software Reset Register	0x0000_0000
DMA_CESR	0x0508	DMA Channel Enable Status Register	0x0000_0000
DMA_ISR	0x050C	DMA Interrupt Status Register	0x0000_0000
DMA_ICR	0x0510	DMA Interrupt Clear Register	0x0000_0000

NOTE:

1. ‘x’ is the corresponding channel of DMA, from 0 to 9.
2. Channel Register Start Address Ch0: 0x000 Ch1: 0x080 Ch2: 0x100 Ch3: 0x180 Ch4: 0x200 Ch5: 0x280 Ch6: 0x300 Ch7: 0x380 Ch8: 0x400 Ch9: 0x480
3. The priority of channel is the same the number of channel. That means channel 0 has the highest priority and channel 9’s priority is the lowest. If two more DMA triggers occur at the same time, the DMA operation will transfer from channel with the lowest number.

9.3.1.1 DMA_ISRx (DMA Channel x Initial Source Register)

- Address = Base Address + 0x0000, Reset Value = 0x0000_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
SADDR																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
SADDR	[31:0]	RW	DMA Initial Source Address Field.	0x0000_0000

9.3.1.2 DMA_ISCRx (DMA Channel x Initial Source Control Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																														HINC	LINC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
LINC	[0]	RW	Source Address Increment Bit for Low Transfer Count. This bit is used to select whether or not the address is increased. 0 = Increment 1 = Fixed If it is 0, the address is increased by its data size after each transfer depending to the low transfer count in burst and single transfer mode. If it is 1, the address is not changed after the transfer.	0
HINC	[1]	RW	Source Address Increment Bit for High Transfer Count. This bit is used to select whether or not the address is increased. 0 = Increment 1 = Fixed If it is 0, the address is increased by its data size after each transfer depending to the high transfer count in burst and single transfer mode. If it is 1, the address is not changed after the transfer.	0

9.3.1.3 DMA_IDRx (DMA Channel x Initial Destination Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
DADDR																															
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
DADDR	[31:0]	RW	Destination Address Field. These bits are the base address (start address) of destination for the transfer.	0x0000_0000

9.3.1.4 DMA_IDCRx (DMA Channel x Initial Destination Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																														HINC	LINC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
LINC	[0]	RW	Destination Address Increment Bit for Low Transfer Count. This is used to select the address increment. 0 = Increment 1 = Fixed If it is 0, the address is increased by its data size after each transfer depending to the low transfer count in burst and single transfer mode. If it is 1, the address is not changed after the transfer.	0
HINC	[1]	RW	Destination Address Increment Bit for High Transfer Count. This is used to select the address increment. 0 = Increment 1 = Fixed If it is 0, the address is increased by its data size after each transfer depending to the high transfer count in burst and single transfer mode. If it is 1, the address is not changed after the transfer.	0

9.3.1.5 DMA_CRx (DMA Channel x Control Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD	TCINT	LTCINT	TSIZE	SMODE	RELOAD	DSIZE		HTC												LTC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	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

Name	Bit	Type	Description	Reset Value
LTC	[11:0]	RW	Low Transfer Count Field. Initial low transfer count (or transfer beat).	0x000
HTC	[23:12]	RW	High Transfer Count Field. Initial high transfer count (or transfer beat). Note: The count value for transfer operation is decided by HTC[11:0] × LTC[11:0]. The transfer operation will end when both CURR_HTC[11:0] and CURR_LTC[11:0] count value are '0'.	0x000
DSIZE	[25:24]	RW	Data Size Selection Field. Data size to be transferred. 00 = Byte 01 = Half word 10 = Word 11 = Reserved	00'b
RELOAD	[26]	RW	Auto Reload Enable/Disable Control Bit. Set the reload on/off option. 0 = Auto reload is performed when a current value of transfer count becomes 0 (i.e., all the required transfers are performed). Auto reload is useful for repetition of same pattern of DMA operation loading same address and count. 1 = DMA channel is turned off when a current value of transfer count becomes 0. The channel on/off bit(CHEN) is set to 0 to prevent unintended further start of new DMA operation.	0
SMODE	[27]	RW	Service Mode Selection Bit. Select the service mode between single service mode and whole service mode for DMA operation. 0 = Single Service Mode is selected in which after each atomic transfer (single or burst of length four) DMA stops and waits for another DMA request.	0

Name	Bit	Type	Description	Reset Value
			1 = Continuous Mode is selected in which one request gets atomic transfers to be repeated until the transfer count reaches to 0. Here, note that even in the whole service mode, DMA releases the bus after each atomic transfer and then tries to re-get the bus to prevent starving of other bus masters.	
TSIZE	[28]	RW	Transfer Size. Select the transfer size of an atomic transfer (i.e., transfer performed at each time DMA owns the bus before releasing the bus). 0 = A unit transfer of which size is defined by DSIZE is performed. 1 = A burst transfer of which size is 4 times of DSIZE is performed.	0
LTCIT	[29]	RW	Low Transfer Count Interrupt Enable/Disable Control Bit. Enable/Disable the interrupt setting for DMA 0 = Low transfer count zero interrupt is disabled. User has to look the transfer count in the status register by using bit polling method. 1 = Interrupt request is generated when all the low transfer is done (i.e., CURR_LTC[11:0] becomes 0).	0
TCIT	[30]	RW	Transfer Count Interrupt Enable/Disable Control Bit Enable/Disable the interrupt setting for DMA 0 = DMA transfer count interrupt is disabled. User has to look the transfer count in the status register by using bit polling method. 1 = Interrupt request is generated when all transfer is done (i.e., CURR_LTC[11:0] × CURR_LTC[11:0] becomes 0).	0

To do transfer operation by DMA, both HTC[11:0] and LTC[11:0] should be not ‘0’.

9.3.1.6 DMA_SRx (DMA Channel x Status Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
LTCST	RSVD							CURR_HTC												CURR_LTC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CURR_LTC	[11:0]	R	Current Low Transfer Count Value. Current value of low transfer count. That transfer count is initially set to the value of LTC[11:0] field in DMA_CR register and decreased by one at the end of every atomic transfer. If CURR_HTC[11:0] is not ‘0’ and CURR_LTC[11:0] is ‘1’, after a transfer, the value of CURR_HTC[11:0] is decreased and CURR_LTC[11:0] is initialized as the same value with LTC[11:0] value in a control register.	0x000
CURR_ HTC	[23:12]	R	Current High Transfer Count Value. Current value of high transfer count. That transfer count is initially set to the value of HTC[11:0] field in DMA_CRx register and decreased at the end of the transfer by count value defined LTC[11:0] field in DMA_CR. The count value for transfer operation is decided by HTC[11:0] x LTC[11:0]. The transferring will be finished when both HTC[11:0] and LTC[11:0] count value are ‘0’. At that time ‘STOP’ bit DMA_MTR register will be set ‘1’.	0x000
LTCST	[31]	R	Low Transfer Counter Status Bit. It tells the busy status of transfer operation by low transfer count value. When the trigger event is asserted and start a DMA operation, LTCST bit sets to ‘1’ (BUSY). 0 = It indicates that DMA controller is not busy. Although this bit is ‘0’, if both CURR_LTC[11:0] and CURR_HTC[11:0] are not ‘0’, CHEN bit in the mask trigger register can this bit will be ‘0’. 1 = It indicates that DMA controller is busy by low transfer count.	0

This register can use for polling the status of TCIT, LTCIT status under a disabled interrupt condition.

9.3.1.7 DMA_CSRx (DMA Channel x Current Source Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
CURR_SADDR																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CURR_SADDR	[31:0]	R	Current Source Address Field. Current source address from which data being transferring currently.	0x0000_0000

9.3.1.8 DMA_CDRx (DMA Channel x Current Destination Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
CURR_DADDR																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CURR_DADDR	[31:0]	R	Current Destination Address Field. Current destination address to which data is being transferred currently.	0x0000_0000

9.3.1.9 DMA_MTRx (DMA Channel x Mask Trigger Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																													STOP	CHEN	SWTRIG
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
SWTRIG	[0]	RW	DMA Software Request Trigger. Trigger DMA channel in S/W request mode. This bit should be set to ‘1’ to start DMA operation. When DMA operation starts, this bit is cleared automatically. 0 = No effect 1 = It requests a DMA operation to this controller.	0
CHEN	[1]	RW	DMA channel on/off Bit. 0 = DMA channel is turned off. (DMA request to this channel is ignored.) 1 = DMA channel is turned on and the DMA request is handled. This bit should be set to ‘1’ to start DMA operation. This bit is automatically set to off (‘0’) if we set the DMA_CRx[26] bit to “no auto reload” and/or STOP bit of DMA_MTRx to “stop”. When DMA_CRx[26] bit is "no auto reload", this bit becomes 0 when both CURR_LTC[11:0] and CURR_HTC[11:0] reaches 0. If the STOP bit is 1, this bit becomes 0 as soon as the current atomic transfer finishes. **NOTE:** This bit should not be changed manually during DMA operations (i.e., this has to be changed only by using DMA_CRx[26] or STOP bit.)	0
STOP	[2]	RW	DMA Operation Stop Bit. 0 = No effect 1 = DMA stops as soon as the current atomic transfer ends. If there is no current running atomic transfer, DMA stops immediately. The value of CURR_HTC[11:0] × CURR_LTC[11:0], CURR_SADDR[31:0], and CURR_DADDR[31:0] will have ‘0’. **NOTE:** Due to possible current atomic transfer, “stop” may take several cycles. The finish of “stopping” operation	0

Name	Bit	Type	Description	Reset Value
			(i.e., actual stop time) can be detected by waiting until the channel on/off bit (CHEN) is set to off. This stop is "actual stop".	

NOTE: You can freely change the values of ISRx , IDRx registers, and HTC[11:0] and LTC[11:0] field of DMA_CRx register. Those changes take effect only after the finish of current transfer (i.e., when CURR_HTC[11:0] and CURR_LTC[11:0] become '0'. On the other hand, any change made to other registers and/or fields takes immediate effect. Therefore, be careful in changing those registers and fields.

9.3.1.10 DMA_RSRx (DMA Channel x Request Selection Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																										HWSRC					REQ
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>REQ</td><td>[0]</td><td>RW</td><td>Request Type Selection Bit.
Select the DMA source between software (S/W request mode) and hardware (H/W request mode).
0 = S/W request mode is selected and DMA is triggered by setting SWTRIG bit of DMA_MTRx register.
1 = DMA source selected by HWSRC[4:0] is used to trigger the DMA operation.</td><td>0</td></tr>
<tr><td>HWSRC</td><td>[5:1]</td><td>RW</td><td>DMA Hardware Source Selection Field.
Select DMA request source for each DMA. DMA H/W request source is below table. These bits have meanings if and only if H/W request mode is selected by REQ[0].
<table>
<tr><th>Peripheral</th><th>Binary</th><th>Decimal</th><th>Peripheral</th><th>Binary</th><th>Decimal</th></tr>
<tr><td>USART0_TX</td><td>00000</td><td>0</td><td>SPI1_TX</td><td>01100</td><td>12</td></tr>
<tr><td>USART0_RX</td><td>00001</td><td>1</td><td>SPI1_RX</td><td>01101</td><td>13</td></tr>
<tr><td>USART1_TX</td><td>00010</td><td>2</td><td>RSVD</td><td>01110~</td><td>14~17</td></tr>
<tr><td>USART1_RX</td><td>00011</td><td>3</td><td>IIC0_TX</td><td>10010</td><td>18</td></tr>
<tr><td>USART2_TX</td><td>00100</td><td>4</td><td>IIC0_RX</td><td>10011</td><td>19</td></tr>
<tr><td>USART2_RX</td><td>00101</td><td>5</td><td>IIC1_TX</td><td>10100</td><td>20</td></tr>
<tr><td>USART3_TX</td><td>00110</td><td>6</td><td>IIC1_RX</td><td>10101</td><td>21</td></tr>
<tr><td>USART3_RX</td><td>00111</td><td>7</td><td>ADC00</td><td>10110</td><td>22</td></tr>
<tr><td>RSVD</td><td>01000~</td><td>8~9</td><td>ADC1</td><td>10111</td><td>23</td></tr>
<tr><td>SPI0_TX</td><td>01010</td><td>10</td><td>ADC01</td><td>11000</td><td>24</td></tr>
<tr><td>SPI0_RX</td><td>01011</td><td>11</td><td>RSVD</td><td>11001~</td><td>25~31</td></tr>
</table></td><td>00000'b</td></tr>
</table>

9.3.1.11 DMA_IDR (DMA ID Register)

- Address = Base Address + 0x0500, Reset Value = 0x00A2_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
RSVD						IDCODE																									
0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Code Register. This field stores the ID code for the corresponding IP.	0x00A2_0018

9.3.1.12 DMA_SRR (DMA Software Reset Register)

- Address = Base Address + 0x0504, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset 0 = No effect 1 = Global software reset	0

9.3.1.13 DMA_CESR (DMA Channel Enable Status Register)

- Address = Base Address + 0x0508, Reset Value = 0x0000_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
RSVD																						CH9EN	CH8EN	CH7EN	CH6EN	CH5EN	CH4EN	CH3EN	CH2EN	CH1EN	CH0EN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHxEN	[x]	R	Channel x enable status bit (‘x’ is from 0 to 9). 0 = Channel x is disabled. 1 = Channel x is enabled.	0

9.3.1.14 DMA_ISR (DMA Interrupt Status Register)

- Address = Base Address + 0x050C, Reset Value = 0x0000_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
RSVD						CH9_TCIT	CH8_TCIT	CH7_TCIT	CH6_TCIT	CH5_TCIT	CH4_TCIT	CH3_TCIT	CH2_TCIT	CH1_TCIT	CH0_TCIT	RSVD						CH9_LTCIT	CH8_LTCIT	CH7_LTCIT	CH6_LTCIT	CH5_LTCIT	CH4_LTCIT	CH3_LTCIT	CH2_LTCIT	CH1_LTCIT	CH0_LTCIT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CHx_LTCIT	[x]	R	Channel x low transfer count interrupt status bit ('x' is from 0 to 9). 0 = Low Transfer Count zero interrupt doesn't occur. User has to look the transfer count in the status register by using a bit polling method. 1= Interrupt request is generated when all the low transfer is done (i.e., CURR_LTC[11:0] becomes 0). This bit can be set to '1', when LTCINT in CRx register is enabled (set to '1') and LTCINT REQ is asserted.	0
CHx_TCIT	[x+16]	R	Channel x transfer count interrupt status bit ('x' is from 0 to 9). 0 = DMA Transfer Count interrupt doesn't occur. User has to look the transfer count in the9-29status register by using a bit polling method. 1 = Interrupt request is generated when all transfer is done (i.e., Both CURR_LTC[11:0] and CURR_LTC[11:0] become 0). This bit can be set to '1', when TCINT in CRx register is enabled (set to '1') and TCINT REQ is asserted.	0

User can use this register when LTCIT or TCIT in a control register sets to '1', because the enabled interrupt (event) is only updated.
User has to look the transfer count in the channel x status register by using a bit polling method.
If user checks with the status polling, not interrupt, user should check DMA_SRx register. Also when RELOAD bit in a control register is '1' (disable auto-reload), user can use CHEN bit in DMA_MTRx.

9.3.1.15 DMA_ICR (DMA Interrupt Clear Register)

- Address = Base Address + 0x0510, Reset Value = 0x0000_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
RSVD																						CH9_IT	CH8_IT	CH7_IT	CH6_IT	CH5_IT	CH4_IT	CH3_IT	CH2_IT	CH1_IT	CH0_IT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
CHx_IT	[x]	W	Channel x Interrupt clear bit (‘x’ is from 0 to 9). 0 = No effect 1 = Clear Channel x interrupt (CHx_TCIT and CHx_LTCIT)	0

10 Encoder Counter

10.1 Overview

The encoder counter block can be used for measuring position and speed.

10.1.1 Features

- Three input signals: PHASEA, PHASEB and PHASEZ
- Two 16-bit up/down counters: Position Counter (PCR) and Speed Counter (SPCR)
- Capture function support for slow rotating: Phase a Capture (PACDR) and Phase B Capture (PBCDR)
- Filter in the PHASEZ and edge selector for PHASEZ

10.1.2 Pin Description

Table 10-1 ENC Pin Description

Pin Name	Function	I/O Type	Comments
PHASEA[1:0]	Phase A input	I	–
PHASEB[1:0]	Phase B input	I	–
PHASEZ[1:0]	Phase Z input	I	–

10.2 Functional Description

10.2.1 Block Diagram

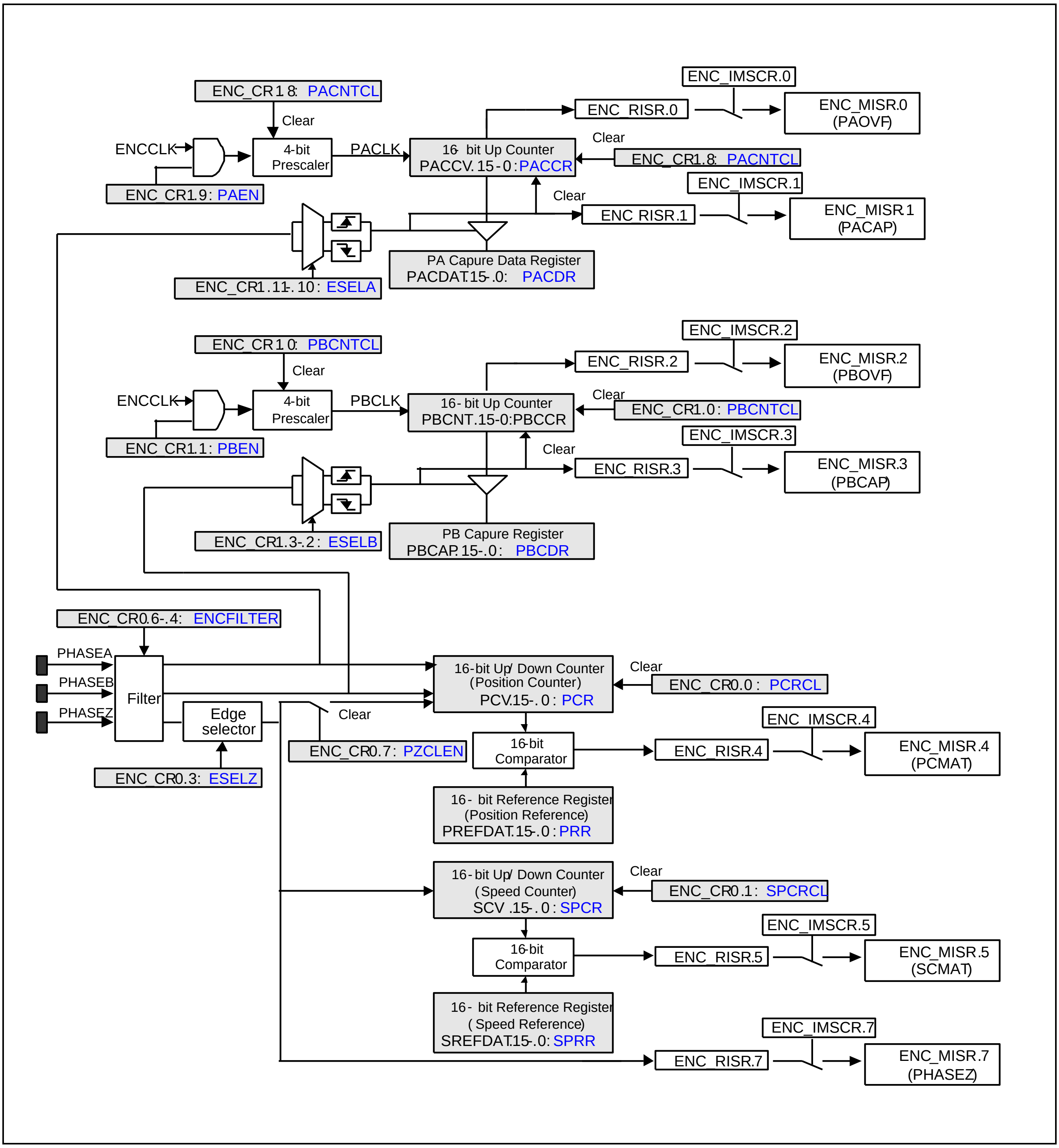


Figure 10-1 ENC Block Diagram

10.2.2 Operation

10.2.2.1 Function Description

To measure position and speed the encoder counter has the three input signals, PHASEA, PHASEB and PHASEZ. The difference of phase between phase A and phase B pulse is 90°. The input of PHASEZ is one-pulse signal to be generated at specific position 1 cyclic.

10.2.2.2 Position Counter Operation

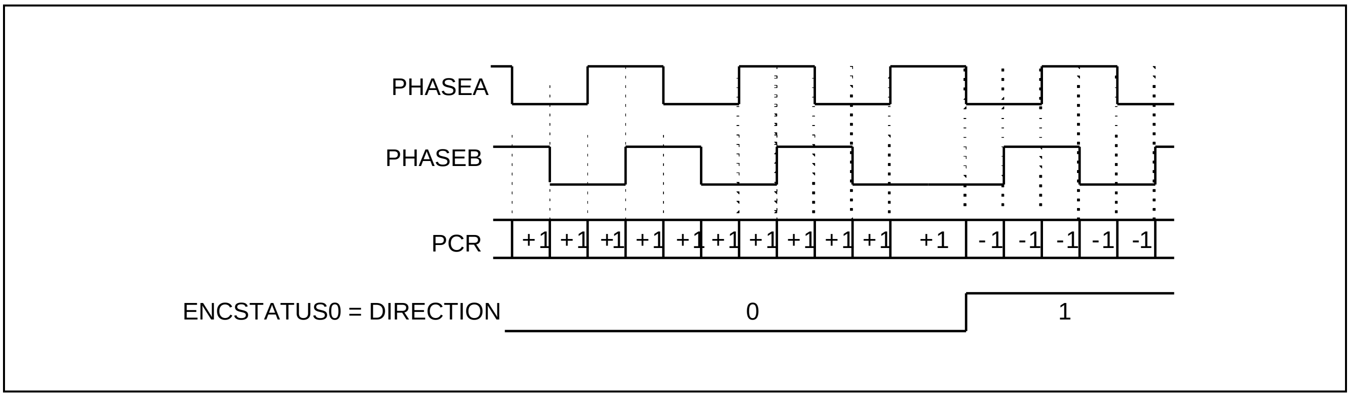


Figure 10-2 Position Counter Operation

- **Direction of Rotation**

When DIRECTION bit is ‘0’, the counter value of PCR increases. On the other hands, when DIRECTION bit is ‘1’, PCR decreases. Position counter is an up and down counter. The DIRECTION bit status and counting direction are decided which phase signal between PHASE A and PHASE B is leading.

NOTE: Although the PBEN and PAEN bit are ‘0’, disable, if inserted any signal into PHASE A, PHASE B input port and CAP_A0/A1 and CAP_B0/B1 interrupt are unmask, those interrupts occur until interrupts mask or non-signal.

10.3 Register Description

10.3.1 Register Map Summary

- Base Address: 0x400C_0000
- Base Address: 0x400C_1000

Register	Offset	Description	Reset Value
ENC_IDR	0x000	ENC ID Register	0x0001_0011
ENC_CEDR	0x004	ENC Clock Enable/Disable Register	0x0000_0000
ENC_SRR	0x008	ENC Software Reset Register	0x0000_0000
ENC_CR0	0x00C	ENC Control Register 0	0x0000_0000
ENC_CR1	0x010	ENC Control Register 1	0x0000_0000
ENC_SR	0x014	ENC Status Register	0x0000_0000
ENC_IMSCR	0x018	ENC Interrupt Mask Set and Clear Register	0x0000_0000
ENC_RISR	0x01C	ENC Raw Interrupt Status Register	0x0000_0000
ENC_MISR	0x020	ENC Masked Interrupt Status Register	0x0000_0000
ENC_ICR	0x024	ENC Interrupt Clear Register	0x0000_0000
ENC_PCR	0x028	ENC 16bit Position Counter Register	0x0000_0000
ENC_PRR	0x02C	ENC 16bit Position Reference Register	0x0000_0000
ENC_SPCR	0x030	ENC 16bit Speed Counter Register	0x0000_0000
ENC_SPRR	0x034	ENC 16bit Speed Reference Register	0x0000_0000
ENC_PACCR	0x038	ENC 16bit Phase A Capture Counter Register	0x0000_0000
ENC_PACDR	0x03C	ENC 16bit Phase A Capture Data Register	0x0000_0000
ENC_PBCCR	0x040	ENC 16bit Phase B Capture Counter Register	0x0000_0000
ENC_PBCDR	0x044	ENC 16bit Phase B Capture Data Register	0x0000_0000

NOTE: The ENC_PCR, ENC_SPCR are 2's complement. The range of PCR and SPCR are -2^{15} ~ $(+2^{15}-1)$.

10.3.1.1 ENC_IDR (ENC ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_0011

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Code Register. This field stores the ID code for the corresponding IP.	0x0001_0011

10.3.1.2 ENC_CEDR (ENC Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	W	Clock Enable. 0 = ENC Clock Disable 1 = ENC Clock Enable	0
DBGEN	[31]	W	Debug Enable Bit. 0 = Disable debug mode ENC is not halted during processor debug mode. 1 = Enable debug mode ENC is halted during processor debug mode.	0

10.3.1.3 ENC_SRR (ENC Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No effect 1 = Perform ENC Software Reset	0

10.3.1.4 ENC_CR 0(ENC Control Register 0)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																					ENCCLKSEL			PZCLEN	ENCFILTER			ESELZ	ENCEN	SPCRCL	PCRCL
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PCRCL	[0]	RW	Position Counter Register (PCR) Clear. This bit can clear the content of Position Counter register. 0 = No effect 1 = Clear the counter register **NOTE:** This bit is auto-clear bit.	0
SPCRCL	[1]	RW	Speed Counter (SPCR) Clear. This bit can clear the content of Speed Counter register. 0 = No effect 1 = Clear the counter register **NOTE:** This bit is auto-clear bit.	0
ENCEN	[2]	RW	Encoder Counter Block Enable. This field can determine the encoder counter block enable/disable. 0 = Disable encoder counter block 1 = Enable encoder counter block	0
ESELZ	[3]	RW	Phase Z Edge Selection. This field can determine the edge selection for phase Z. 0 = Falling edge is selected for PHASEZ 1 = Rising edge is selected for PHASEZ	0
ENCFILTER	[6:4]	RW	Filter Clock Selection of Encoder counter. This field can determine the filter clock selection of encoder counter. 000 = ENCCLK 001 = ENCCLK /2 010 = ENCCLK /4 011 = ENCCLK /8 100 = ENCCLK /16 101 = ENCCLK /32 110 = ENCCLK /64 111 = ENCCLK /128	000'b

Name	Bit	Type	Description	Reset Value
			NOTE: Note: Only 5 times same level in a row is recognized as effective signal.	
PZCLEN	[7]	RW	Position counter (ENC_PCR) clear enable by Phase Z. This field can determine PCNT clear enable by Phase Z. 0 = Enable 1 = Disable	0
ENCCLKSEL	[10:8]	RW	Encoder Counter Clock (ENCCLK) Selection. This field can determine the encoder counter clock selection. 000 = PCLK 001 = PCLK/2 010 = PCLK/4 011 = PCLK/8 100 = PCLK/16 101 = PCLK/32 110 = PCLK/64 111 = PCLK/128	000'b

NOTE: PCRCL and SPCRCL are auto clear bits.
Position counter can be cleared by PZCLEN bit only when ENCEN is set to 1. But, position counter can be cleared by SPCRCL and PCRNTCL even though ENCEN bit is set to 0.

10.3.1.5 ENC_CR 1(ENC Control Register1)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																PRESCALEA				ESELA		PAEN	PACCRCL	PRESCALEB				ESELB		PBEN	PBCCRCL
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PBCCRCL	[0]	RW	PBCCRCL: Phase B Counter Register (PBCCR) Clear. This bit can clear the content of phase B counter register. 0 = No effect 1 = Clear the counter register Note: This bit is auto-clear bit.	0
PBEN	[1]	RW	PBEN: Phase B Enable. This bit can enable or disable of phase B functionality. 0 = Disable(Stop) 1 = Enable(Start)	0
ESELB	[3:2]	RW	ESELB[1:0]: Phase B Capture Operating Mode Selection. This field can determine the edge selection for capture. 00 = Capture on falling edge of PHASEB 01 = Capture on rising edge of PHASEB 1x = Capture on both edges of PHASEB	00'b
PRESCALEB	[7:4]	RW	PRESCALEB[3:0]: Phase B Pre-scale Bits. This bit can set the pre-scale value for Phase B. PBCLK = ENCCLK / 2 ^ PRESCALEB	0000'b
PACCRCL	[8]	RW	PACCRCL: Phase A Counter Register (PACCR) Clear. This bit can clear the content of phase A counter register. 0 = No effect 1 = Clear the counter register Note: This bit is auto-clear bit.	0
PAEN	[9]	RW	PAEN: Phase A Enable. This bit can enable or disable of phase A Functionality. 0 = Disable(Stop) 1 = Enable(Start)	0
ESELA	[11:10]	RW	ESELA[1:0]: Phase A Capture Operating Mode Selection. This field can determine the edge selection for capture. 00 = Capture on falling edge of PHASEA	00'b

Name	Bit	Type	Description	Reset Value
			01 = Capture on rising edge of PHASEA 1x = Capture on both edges of PHASEA	
PRESCALEA	[15:12]	RW	PRESCALEA[3:0]: Phase A Pre-scale Bits. This bit can set the pre-scale value for Phase A. PACLK = ENCCLK / 2 ^ PRESCALEA	0000'b

NOTE: PACCRCL and PBCCRCL are auto clear bits.
Phase A and Phase B counter can be cleared by PACCRCL and PBCCRCL even though ENCEN bit is set to 0.

10.3.1.6 ENC_SR (ENC Status Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																								UFSCNT	OFSCNT	UFPCNT	OFPCNT	PASTAT	PBSTAT	GLITCH	DIRECTION
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R	R	R W	R

Name	Bit	Type	Description	Reset Value
DIRECTION	[0]	R	Direction of Motor Rotation Bit. 0 = Clockwise – The value of PCNT is increased. 1 = Counter-clockwise – The value of PCNT is decreased. **NOTE:** This bit is read-only bit.	0
GLITCH	[1]	RW	Glitch detection of Phase A, Phase B and Phase Z. This bit can notify the glitch detection in the PHASEA, PHASEB or PHASEZ pin. Read) 0 = Glitch is not occurred 1 = Glitch is occurred Write) 0 = GLITCH bit is cleared 1 = No effect **NOTE:** Glitch is detected according to the checking whether if 5 times same level in a row is recognized as effective signal.	0
PBSTAT	[2]	R	Phase B Status Bit. 0 = Low level 1 = High level **NOTE:** This bit is read only bit.	0
PASTAT	[3]	R	Phase A Status Bit. 0 = Low level 1 = High level **NOTE:** This bit is read only bit.	0
OFPCNT	[4]	RW	Overflow of PCNT. This bit can notify the overflow of PCNT counter. Read) 0 = Overflow is not occurred 1 = Overflow is occurred	0

Name	Bit	Type	Description	Reset Value
			Write) 0 = OFPCNT bit is cleared 1 = No effect	
UFPCNT	[5]	RW	Underflow of PCNT. This bit can notify the underflow of PCNT counter. Read) 0 = Underflow is not occurred 1 = Underflow is occurred Write) 0 = UFPCNT bit is cleared 1 = No effect	0
OFSCNT	[6]	RW	Overflow of SCNT. This bit can notify the overflow of SCNT counter. Read) 0 = Overflow is not occurred 1 = Overflow is occurred Write) 0 = OFSCNT bit is cleared 1 = No effect	0
UFSCNT	[7]	RW	Underflow of SCNT. This bit can notify the underflow of SCNT counter. Read) 0 = Underflow is not occurred 1 = Underflow is occurred Write) 0 = UFSCNT bit is cleared 1 = No effect	0

NOTE:

1. ENC_SR.0, ENC_SR.2 and ENC_SR.3 are read only bits.
2. ENC_SR.5 – .4 are cleared automatically by PHASEZ and ENC_CR0.0.
3. ENC_SR.7 – .6 are cleared automatically by ENC_CR0.1.

10.3.1.7 ENC_IMSCR (ENC Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																								PHASEZ	RSVD	SCMAT	PCMAT	PBCAP	PBOVF	PACAP	PAOVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PAOVF	[0]	RW	Phase A Counter Overflow Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PACAP	[1]	RW	Phase A Capture Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PBOVF	[2]	RW	Phase B Counter Overflow Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PBCAP	[3]	RW	Phase B Capture Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PCMAT	[4]	RW	Position Counter Match Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
SCMAT	[5]	RW	Speed Counter Match Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PHASEZ	[7]	RW	PHASEZ: Phase Z Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

10.3.1.8 ENC_RISR (ENC Raw Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																								PHASEZ	RSVD	SCMAT	PCMAT	PBCAP	PBOVF	PACAP	PAOVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PAOVF	[0]	R	Phase A Counter Overflow Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PAOVF interrupt.	0
PACAP	[1]	R	Phase A Capture Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PACAP interrupt.	0
PBOVF	[2]	R	Phase B Counter Overflow Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PBOVF interrupt.	0
PBCAP	[3]	R	Phase B Capture Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PBCAP interrupt.	0
PCMAT	[4]	R	Position Counter Match Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PCMAT interrupt.	0
SCMAT	[5]	R	Speed Counter Match Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SCMAT interrupt.	0
PHASEZ	[7]	R	Phase Z Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PHASEZ interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

10.3.1.9 ENC_MISR (ENC Masked Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																								PHASEZ	RSVD	SCMAT	PCMAT	PBCAP	PBOVF	PACAP	PAOVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PAOVF	[0]	R	Phase A Counter Overflow Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PAOVF interrupt.	0
PACAP	[1]	R	Phase A Capture Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PACAP interrupt.	0
PBOVF	[2]	R	Phase B Counter Overflow Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PBOVF interrupt.	0
PBCAP	[3]	R	Phase B Capture Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PBCAP interrupt.	0
PCMAT	[4]	R	Position Counter Match Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PCMAT interrupt.	0
SCMAT	[5]	R	Speed Counter Match Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the SCMAT interrupt.	0
PHASEZ	[7]	R	Phase Z Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PHASEZ interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

10.3.1.10 ENC_ICR (ENC Interrupt Clear Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																								PHASEZ	RSVD	SCMAT	PCMAT	PBCAP	PBOVF	PACAP	PAOVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
PAOVF	[0]	W	Phase A Counter Overflow Interrupt Clear. 0 = No effect 1 = Clears the PAOVF interrupt	0
PACAP	[1]	W	Phase A Capture Interrupt Clear. 0 = No effect 1 = Clears the PACAP interrupt	0
PBOVF	[2]	W	Phase B Counter Overflow Interrupt Clear. 0 = No effect 1 = Clears the PBOVF interrupt	0
PBCAP	[3]	W	Phase B Capture Interrupt Clear. 0 = No effect 1 = Clears the PBCAP interrupt	0
PCMAT	[4]	W	Position Counter Match Interrupt Clear. 0 = No effect 1 = Clears the PCMAT interrupt	0
SCMAT	[5]	W	Speed Counter Match Interrupt Clear. 0 = No effect 1 = Clears the SCMAT interrupt	0
PHASEZ	[7]	W	Phase Z Interrupt Clear. 0 = No effect 1 = Clears the PHASEZ interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

10.3.1.11 ENC_PCR (ENC Position Counter Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																PCV															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PCV	[15:0]	RW	Position Counter Value. This field contains the current position counter value.	0x0000

10.3.1.12 ENC_PRR (ENC Position Reference Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																PREFDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PREFDAT	[15:0]	RW	Position Counter Reference Data Value. This field can determine the reference value for position counter.	0x0000

10.3.1.13 ENC_SCR (ENC Speed Counter Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																SCV															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
SCV	[15:0]	RW	Speed Counter Value. This field contains the current speed counter value.	0x0000

10.3.1.14 ENC_ SRR (ENC Speed Reference Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																SREFDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
SREFDAT	[15:0]	RW	Speed Counter Reference Data Value. This field can determine the reference value for the speed counter.	0x0000

10.3.1.15 ENC_ PACCR (ENC Phase A Capture Counter Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																PACCV															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PACV	[15:0]	RW	Phase A Capture Counter Value. This field contains the Phase A counter value.	0x0000

10.3.1.16 ENC_ PACDR (ENC Phase A Capture Data Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																PACDA															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PACV	[15:0]	RW	Phase A Capture Data Value. This field can determine the Phase A reference value.	0x0000

10.3.1.17 ENC_ PBCCR (ENC Phase B Capture Counter Register)

- Address = Base Address + 0x0040, Reset Value = 0x0000_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
RSVD																PBCCV															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PBCV	[15:0]	RW	Phase B Counter Value. This field contains the Phase B counter value.	0x0000

10.3.1.18 ENC_ PBCDR (ENC Phase B Capture Data Register)

- Address = Base Address + 0x0044, Reset Value = 0x0000_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
RSVD																PBCDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PBCDAT	[15:0]	RW	Phase B Capture Data Value. This field can determine the Phase B reference value.	0x0000

11 Free Running Timer

11.1 Overview

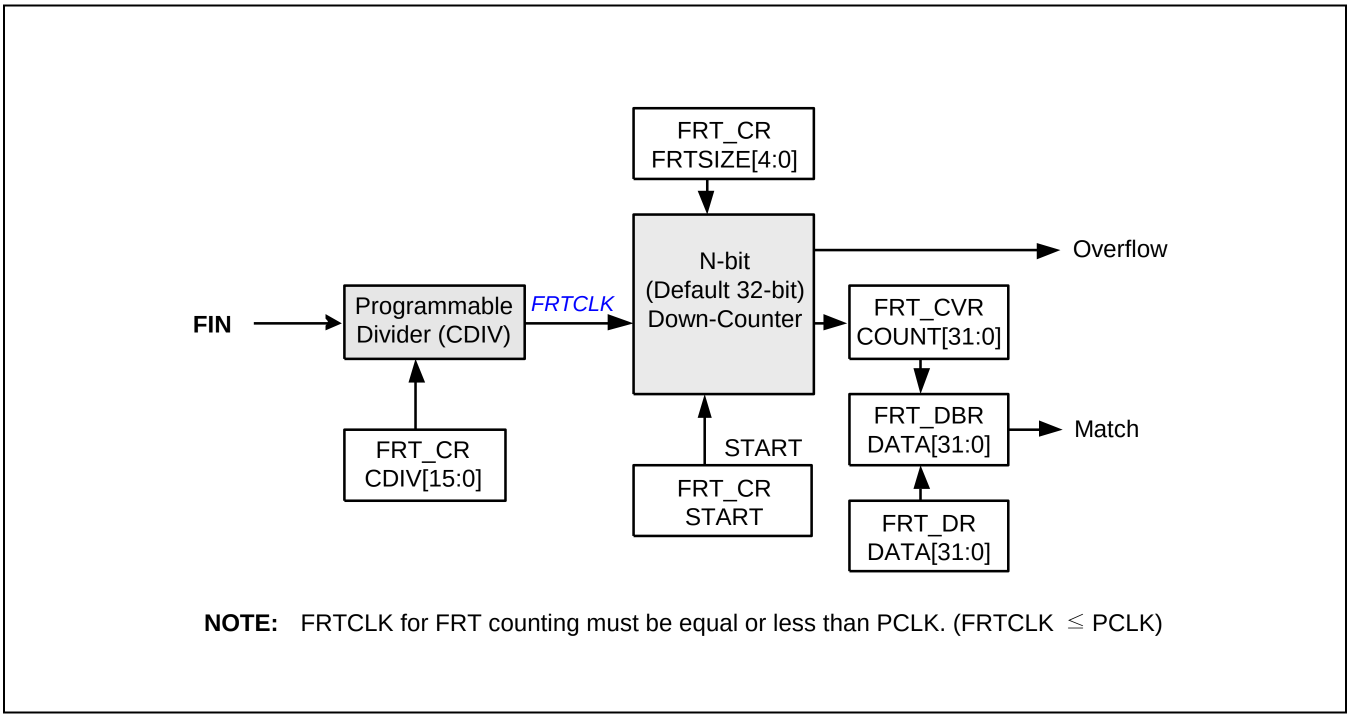
This chapter describes a free running timer (FRT). FRT supports general timer functions, a match and overflow generation. The count size is configurable up to 32-bit timer. FRT can operate using ISCLK (Internal Sub-Clock) in stop mode especially.

11.1.1 Features

- 32-bit free running timer
- Configurable count size, up to 32-bit counting
- Support a divider to generate the count clock
- Internal interrupts are generated on the occurrence of
  - Counter overflow (OVF)
  - Match to data buffer register (MATCH)

11.2 Functional Description

11.2.1 Block Diagram



NOTE: FRTCLK for FRT counting must be equal or less than PCLK. (FRTCLK ≤ PCLK)

Figure 11-1 Free Running Timer Block Diagram

NOTE: FRTCLK for FRT counting must be equal or less than PCLK. (FRTCLK ≤ PCLK)

11.2.2 Timer Count Size

FRTSIZE[4:0] in FRT_CR register determines a count size for FRT. If the value of FRTSIZE[4:0] is n, FRT will become a (n+1) bit timer. In other words, this timer can count from 1 to $2^{(n+1)}$-1.

For example, if you want the FRT act as 20-bit Free Running Timer, then please set FRTSIZE[4:0] to 0x13. It is possible to be up to 32-bit timer.

11.2.3 Timer Clock

11.2.3.1 Clock Source

FRT can use one of 5 different clocks (EMCLK, IMCLK, ESCLK, ISCLK, and PLLCLK). FRT clock is supported from a clock manager. The clock manager can generate an input clock (FIN) by FRTCLK[2:0] in CM_MR register and MDIV[2:0]/NDIV[2:0] in CM_FCDR register.

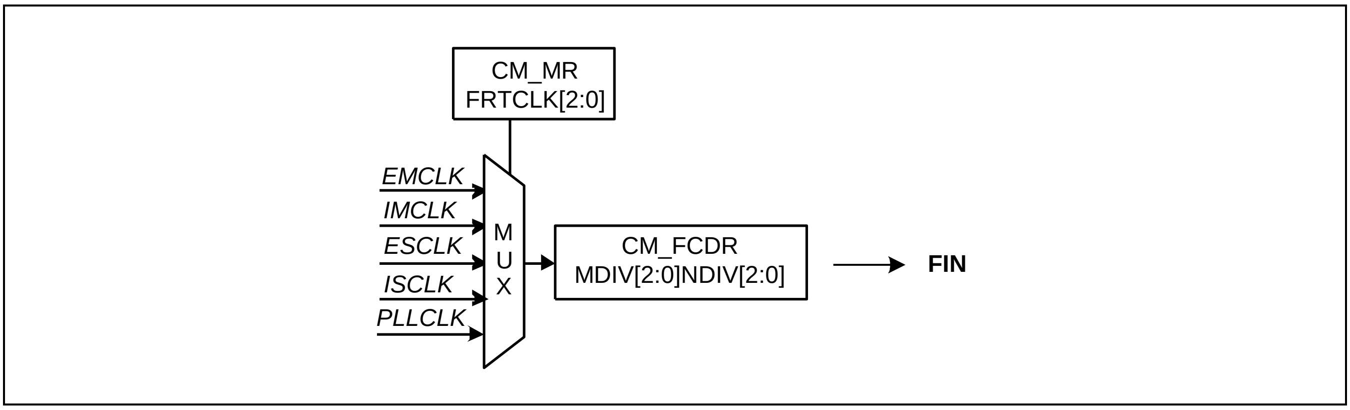


Figure 11-2 FRT Clock Source

11.2.3.2 Count Clock

The count clock based on FIN is determined by CDIV[15:0] in FRT_CR register finally. The frequency of the counting clock is FRTCLK determined by FIN and the internal clock divider (CDIV).

FRTCLK = FIN/(CDIV[15:0]+1)

Counter Resolution = 1/FRTCLK

Maximal Count Duration (seconds) = (2(FRTSIZE[4:0]+1)–1)/FRTCLK where FRTCLK is in Hz.

NOTE: FRTCLK for FRT counting must be equal or less than PCLK. (FRTCLK ≤ PCLK)

11.2.4 Count and Data Register

The current counter value can be read in FRT_CVR register. FRT_DBR has the current value to generate a match signal. When a value is loaded in from FRT_DR to FRT_DBR register and timer is started by START bit of FRT_CR register, the counter starts down-counting until the counter reaches zero. If counter meets the value of FRT_DBR during down-counting, FRT generates a match signal. FRT_DR will be copied (updated) the FRT_DBR at the time to start a timer, a match event/interrupt or overflow event/interrupt.

11.2.5 Interrupt

There are 2 types of an interrupt, MATCH and OVF.

11.2.5.1 Match Interrupt

A match signal can be generated when the counter value is identical to the value written to the timer data register, FRT_DBR.

11.2.5.2 Overflow Interrupt

The timer can run up to the overflow of counter value and generate an overflow interrupt, also. After the overflow of counter value, the counter value will be counted from 0, again.

11.2.5.3 Interruption Handling

- Interrupt Service Routine (ISR) Entry and call C function.
- Read FRT_IMSR and verify the source of the interrupt.
- Clear the corresponding interrupt at peripheral level by writing in the FRT_ICR
- Interrupt treatment
- ISR Exit.

11.3 Register Description

11.3.1 Register Map Summary

- Base Address: 0x4003_1000

Register	Offset	Description	Reset Value
FRT_IDR	0x000	ID Register	0x0001_0018
FRT_CEDR	0x004	Clock Enable/Disable Register	0x0000_0001
FRT_SRR	0x008	Software Reset Register	0x0000_0000
FRT_CR	0x00C	Control Register	0x0000_1F01
FRT_SR	0x010	Status Register	0x0000_0000
FRT_IMSCR	0x014	Interrupt Masked Set/Clear Register	0x0000_0001
FRT_RISR	0x018	Raw Interrupt Status Register	0x0000_0000
FRT_MISR	0x01C	Masked Interrupt Status Register	0x0000_0000
FRT_ICR	0x020	Interrupt Clear Register	0x0000_0000
FRT_DR	0x024	Data Register	0x0000_0000
FRT_DBR	0x028	Data Buffer Register	0x0000_0000
FRT_CVR	0x02C	Counter Value Register	0x0000_0000

11.3.1.1 FRT_IDR (Free Running Timer ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_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
RSVD						IDCODE																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_0018

11.3.1.2 FRT_CEDR (Free Running Timer Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_0001

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

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	CLKEN. 0 = Counter Clock is disabled 1 = Counter Clock is enabled	1
DBGEN	[31]	RW	DBGEN. 0 = Disable debug mode FRT is not halted during processor debug mode. 1 = Enable debug mode FRT is halted during processor debug mode.	0

11.3.1.3 FRT_SRR (Free Running Timer Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No Effect 1 = Free Running Timer Software Reset	0

11.3.1.4 FRT_CR (Free Running Timer Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_1F01

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

Name	Bit	Type	Description	Reset Value
START	[0]	RW	START/STOP Control Bit. 0 = Free Running Timer Stop 1 = Free Running Timer Start	1
FRTSIZE	[12:8]	RW	Free Running Timer Bit Size Selection Field. Free Running Timer Bit-size mask. Default = 32-bit Free Running Timer mode. **NOTE:** If you want the FRT act as 8-bit Free. Running Timer, then please set FRTSIZE = 0x07. If you want the FRT act as 16-bit Free Running Timer, then please set FRTSIZE = 0x0F.	0x1F
CDIV	[31:16]	RW	Clock Divider Field. Free Running Timer Clock Divider. The Frequency of Free Running Timer = Source Clock/(CKDIV+1).	0x0000

NOTE: FRT timer clock can be one among 5 clock sources, ISCLK, ESCLK, IMCLK, EMCLK, or PLLCLK. (See 1.2.3 Timer Clock) It must be equal or less than PCLK. (FRT_CLK ≤ PCLK).

11.3.1.5 FRT_SR (Free Running Timer Status Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
SWRST	[0]	R	Software Release Status Bit. User can check the end of software reset operation using SWRST bit in FRT_SR (Status Register). SWRST bit in FRT_SR (Status Register) will set to '1 when starting a software reset. After ending a software reset, this bit will be cleared (0) automatically.	0

11.3.1.6 FRT_IMSCR (Free Running Timer Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_0001

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

Name	Bit	Type	Description	Reset Value
OVF	[0]	RW	Overflow Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	1
MATCH	[2]	RW	Match Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

11.3.1.7 FRT_RISR (Free Running Timer Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																													MATCH	RSVD	OVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
OVF	[0]	R	Overflow Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the OVR interrupt.	0
MATCH	[2]	R	Match Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the OVR interrupt. Match interrupt will occur when FRT_DATAR is the same FRT_CVR.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

11.3.1.8 FRT_MISR (Free RunningTimer Masked Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																													MATCH	RSVD	OVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
OVF	[0]	R	Overflow Masked Interrupt State. Gives the masked interrupt status (After masking) of the OVF interrupt.	0
MATCH	[2]	R	Match Masked Interrupt State. Gives the masked interrupt status (After masking) of the MATCH interrupt. Match interrupt will occur when FRT_DATAR is the same FRT_CVR.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.
FRT_MISR = FRT_IMSCR & FRT_RISR

11.3.1.9 FRT_ICR (Free Running Timer Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																													MATCH	RSVD	OVF
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R	W

Name	Bit	Type	Description	Reset Value
OVF	[0]	W	Overflow Interrupt Clear. 0 = No effect 1 = Clears the OVF interrupt	–
MATCH	[2]	W	Match Interrupt Clear. 0 = No effect 1 = Clears the MATCH interrupt	–

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

11.3.1.10 FRT_DR (Free Running Timer Data Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
DATA																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA	[31:0]	RW	DATA for Match. This register is for next match. This value will be copied the DBR (Data buffer register) at start timer, match event/interrupt, overflow event/interrupt.	0x0000_0000

11.3.1.11 FRT_DBR (Free Running Timer Data Buffer Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
DATA																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DATA	[31:0]	R	DATA for Match. A match interrupt will occur when FRT_DBR[DATA] = FRT_CVR[COUNT]. This value will come from the DR (Data register).	0x0000_0000

11.3.1.12 FRT_CVR (Free Running Timer Counter Value Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
COUNT																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
COUNT	[31:0]	R	COUNT Value. This register shows the current count value. If FRTSIZE is 0x0F (16-Bit FRT mode), COUNT[15:0] is valid and COUNT[31:16] will be read as 0x0000.	0x0000_0000

12 General Purpose IO (GPIO)

12.1 Overview

This chapter describes the configuration of general purpose I/O such as input, output, peripheral, etc. All I/O lines in the microcontroller are unified in the GPIO controller. All I/O lines are controlled by this module. The GPIO controller also provides interrupt signals to the Interrupt Controller. Each of the general purpose I/O pins (GPIOs) is capable of controlling the following features.

- Port 0 Group: 32 input/output ports, P0.0 ~ P0.31
- Port 1 Group: 32 input/output ports, P1.0 ~ P1.31
- Port 2 Group: 30 input/output ports, P2.0 ~ P2.27
- Port 3 Group: 10 input/output ports, P3.0 ~ P3.9

Each port can be easily configured by software to meet the various configuration of target system and design requirement. You should define the functionality of port before the start application program. If you do not want to use the function for multiplexed pins, these pin can be configured as simple I/O port.

12.1.1 Features

- Clock Supply Enable / Disable
  - Configuration of GPIO requires a clock supply
  - Clock supply can be disabled to optimize the current consumption
- Output (data direction) Enable / Disable / Status monitoring
- Output Data Set / Clear / Data Status monitoring
- Software Reset (SWRST) function
  - Set to default configuration
- GPIO Interrupt
  - Mask Set and Clear / Raw Interrupt Status / Masked Interrupt Status / Clear Interrupt

12.1.2 Pin Description

Table 12-1 GPIO Pin Description

Pin Name	Function	I/O Type	Active Level	Status @Reset
P0[31:0]	General Purpose Input / Output 0	I/O	–	GPIO input
P1[31:0]	General Purpose Input / Output 1	I/O	–	GPIO input
P2[27:0]	General Purpose Input / Output 2	I/O	–	GPIO input
P3[9:0]	General Purpose Input / Output 3	I/O	–	Debug Port

12.2 Functional Description

12.2.1 Block Diagram

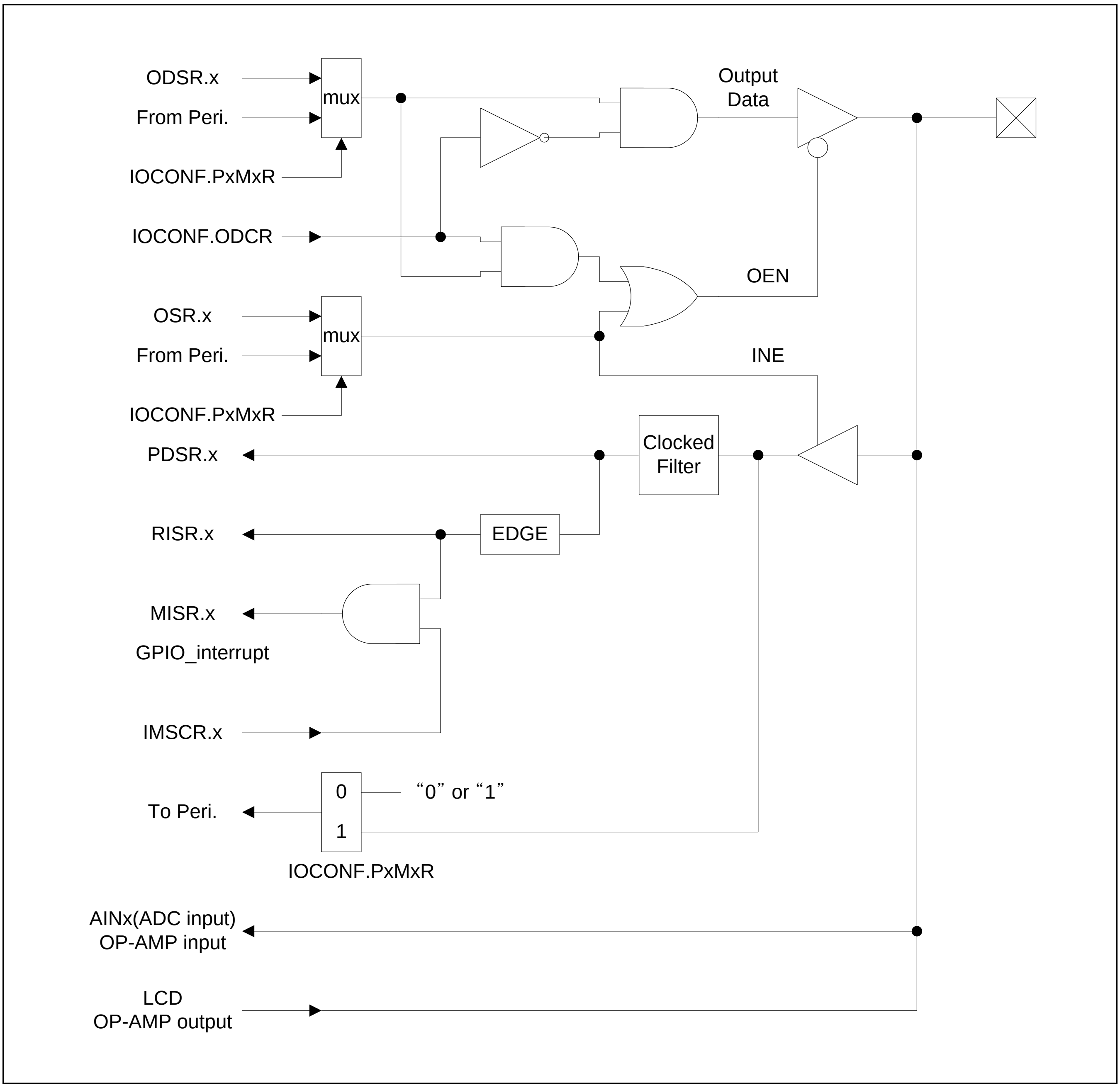


Figure 12-1 GPIO Block Diagram

12.2.2 Operation

12.2.2.1 Functional Description

Pins have their multiplexed functions from 0 to 3. That function is defined by IOCONF. GPIO should be configured when multiplexed pins is defined as general purpose I/O pins (GPIO) by IOCONF. Therefore, it is necessary to control IOCONF before using the GPIO.

Each of 32 bits of control registers such as enable, disable and status registers is corresponding to individual port or I/O ports. The I/O port can be individually configured or the whole port can be configured by writing a value to the 32 bits register.

Prior to any configuration on I/O ports, it is necessary to enable the clock supply to the I/O ports. The following register are responsible for clock supply

- Clock Enable/Disable Register (GPIO_CEDR)

It is also possible to reset each of general purpose I/O blocks and recall the default values. For example, the PIO0 can be reset by means of the software reset register (GPIO_SRR).

- Software Reset Register (GPIO_SRR)

12.2.2.2 Input Configuration

When the input functions of I/O port is used, the output function of I/O port are disabled. To configure the ports as input, the following registers shall be configured. The default value is to enable the input function.

- Output Disable Register (GPIO_ODR)
- Pin Data Status Register (GPIO_PDSR)

The input values can be read by GPIO_PDSR for each I/O port.

12.2.2.3 Output Configuration

The data direction of I/O ports shall be configured as output and data can be written to the ports. The following registers are used to configure the data direction, enable / disable / monitor, and data value.

- Output Enable Register (GPIO_OER)
- Write Output Data Register (GPIO_WODR)
- Set Output Data Register (GPIO_SODR)
- Clear Output Data Register (GPIO_CODR)
- Output Data Status Register (GPIO_ODSR)

GPIO_WODR is to set the data value to I/O ports in accordance with the register value (either high or low level). On the contrary, GPIO_SODR and GPIO_CODR set or clear the data value of I/O ports according to the register value. In other words, GPIO_SODR sets the I/O ports to high level when values are written to the register. GPIO_CODR sets the I/O ports to low level when values are written to the register. GPIO_ODSR can read the current data value of each pin.

12.2.3 Operation Mode

GPIO has different configurations and behaviors according to the operation modes

12.2.3.1 Normal Mode

In normal mode, GPIO is powered, and is to fully operate as well as configurable. GPIO control block is clocked by PCLK.

12.2.3.2 Low Power Modes

In **Idle mode**, GPIO is powered and clocked by PCLK. It is fully configurable. It is also allowed to disconnect the clock in order to reduce the current consumption. Upon exit from Idle mode, the configuration and states of I/Os are not changed.

In **Stop mode**, the clock is disconnected, but the power is still maintained. Thus, the configuration and states of I/Os are preserved by itself. Upon exit from Stop mode, the configuration and states of I/Os are not changed.

12.2.4 Interrupt

Each GPIO controller block also provides an internal interrupt signal. Each GPIO can be programmed to generate an interrupt when a level change occurs (rising / falling edge).

The interrupt occurs regardless of the port configuration and data direction. That is, an interrupt occurs when an edge transition occurs on a pin while the port is configured for the peripherals or output. It detects any edge transition.

- Interrupt Mask Set and Clear Register (GPIO_IMSCR)
- Raw Interrupt Status Register (GPIO_RISR)
- Masked Interrupt Status Register (GPIO_MISR)
- Interrupt Clear Register (GPIO_ICR)

The interrupt for a pin is enabled or disabled by the corresponding bits in the GPIO_IMSCR.

When an edge transition occurs on a pin, the corresponding bit in the GPIO_RISR (Raw Interrupt Status) register is set to 1. If a bit in GPIO_IMSCR register is set to 1, this means an interrupt for the pin was enabled.

The GPIO interrupt is cleared when a 1 is set to a corresponding bit of GPIO_ICR register.

Note that these interrupts are different types of interrupt from the external interrupt in Clock manager (CM).

12.3 Register Description

12.3.1 Register Map Summary

- Port 0 Base Address: 0x4005_0000
- Port 1 Base Address: 0x4005_1000
- Port 2 Base Address: 0x4005_2000
- Port 3 Base Address: 0x4005_3000

Register	Offset	Description	Reset Value
GPIO_IDR	0x0000	ID-Code Register	0x0001_0020
GPIO_CEDR	0x0004	Clock Enable Disable Register	0x0000_0000
GPIO_SRR	0x0008	Software Reset Register	0x0000_0000
GPIO_IMSCR	0x000C	Interrupt Mask Set Clear Register	0x0000_0000
GPIO_RISR	0x0010	Raw Interrupt Status Register	0x0000_0000
GPIO_MISR	0x0014	Masked Interrupt Status Register	0x0000_0000
GPIO_ICR	0x0018	Interrupt Clear Register	0x0000_0000
GPIO_OER	0x001C	Output Enable Register	0x0000_0000
GPIO_ODR	0x0020	Output Disable Register	0x0000_0000
GPIO_OSR	0x0024	Output Status Register	0x0000_0000
GPIO_WODR	0x0028	Write Output Data Register	0x0000_0000
GPIO_SODR	0x002C	Set Output Data Register	0x0000_0000
GPIO_CODR	0x0030	Clear Output Data Register	0x0000_0000
GPIO_ODSR	0x0034	Output Data Status Register	0x0000_0000
GPIO_PDSR	0x0038	Pin Data Status Register	0x0000_0000

NOTE: GPIO2 has only 28 ports (P2.0 ~ P2.27), GPIO3 has only 10 ports (P3.0 ~ P3.9)

12.3.1.1 GPIO_IDR (GPIO ID Code Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_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
RSVD						IDCODE																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Code Register This field stores the ID code for the corresponding IP.	0x0001_0020

12.3.1.2 GPIO_CEDR (GPIO Clock Enable Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable 0 = GPIO Clock Disable 1 = GPIO Clock Enable CLKEN does not affected by Software Reset (GPIO_SRR)	0

12.3.1.3 GPIO_SRR (GPIO Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No Effect 1 = Perform GPIO Software Reset and auto-cleared	0

12.3.1.4 GPIO_IMSCR (GPIO Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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

Name	Bit	Type	Description	Reset Value
Px	[x]	RW	Port x Interrupt Mask 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt) Interrupt occurs when a logic level change is detected on the corresponding pin	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

12.3.1.5 GPIO_RISR (GPIO Raw Interrupt Status Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Px	[x]	R	Port x Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the Px interrupt	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking.

12.3.1.6 GPIO_MISR (GPIO Masked Interrupt Status Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Px	[x]	R	Port x Masked Interrupt Status Gives the masked interrupt status(after masking) of the Px interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

GPIO_MISR = GPIO_IMSCR & GPIO_RISR

12.3.1.7 GPIO_ICR (GPIO Interrupt Clear Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Interrupt Clear 0 = No effect 1 = Clear Px interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

12.3.1.8 GPIO_OER (GPIO Interrupt Clear Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Output Enable Bit 0 = No effect 1 = Enable the GPIO output (data direction) on the corresponding pin	0

12.3.1.9 GPIO_ODR (GPIO Output Disable Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Output Enable Bit (Input Enable) 0 = No effect 1 = Disable the PIO output (data direction) on the corresponding pin, In other word, this is to enable GPIO input on the corresponding pin.	0

12.3.1.10 GPIO_OSR (GPIO Output Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Px	[x]	R	Port x Data Direction(Input or Output) Status 0 = The corresponding GPIO is input on this line 1 = The corresponding GPIO is output on this line	0

12.3.1.11 GPIO_WODR (GPIO Write Output Data Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Output Data Control Bit 0 = The output data on the corresponding pins is ‘0’, low level. 1 = The output data on the corresponding pins is ‘1’, high level. The purpose of this register is same with GPIO_SODR (Set Output Data Register) and GPIO_CODR (Clear Output Data Register). But, all output data (1 and 0) is affected at the same time. This function is different from that of GPIO_SODR and GPIO_CODR	0

12.3.1.12 GPIO_SODR (GPIO Set Output Data Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Output Data Set (1) 0 = No effect 1 = GPIO output data on the corresponding pin is set, high level.	0

12.3.1.13 GPIO_CODR (GPIO Clear Output Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
Px	[x]	W	Port x Output Data Clear (0) 0 = No effect 1 = GPIO output data on the corresponding pin is cleared, low level.	0

12.3.1.14 GPIO_ODSR (GPIO Output Data Status Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Px	[x]	R	Port x Output Data Status 0 = The output data for the corresponding GPIO is programmed to ‘0’, low level. 1 = The output data for the corresponding GPIO is programmed to ‘1’, high level.	0

12.3.1.15 GPIO_PDSR (GPIO Pin Data Status Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16	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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
Px	[x]	R	Port x Pin Status 0 = The real level for the corresponding pin is at logic “0” 1 = The real level for the corresponding pin is at logic “1”	0

13 Inter-Integrated Circuit (IIC)

13.1 Overview

The I2C (Inter-Integrated Circuit) bus is a two-wire synchronous serial interface consisting of one data (SDA) and one clock (SCL). Each device connected to the bus is software addressable by a unique address and simple relationships exist at all times. The lines SDA and SCL are bi-directional lines connected to a positive supply voltage via a pull-up resistor. The output stages of devices connected to the bus must have an open-drain in order to perform the wired-AND function.

It is a true multi-master bus including collision detection and arbitration to prevent data corruption if two or more masters simultaneously initiate data transfer. Clock synchronization is performed using the wired-AND connection of I2C interfaces to the SCL lines.

The I2C interface can operate in fast mode or in normal mode. This allows baud rates from 0 to 400Kbit/s (fast mode) or from 0 to 100Kbit/s (normal mode). The device supports four modes: Master Transmitter, Master Receiver, slave Transmitter, slave Receiver. The device allows 7-bits addressing or 10–bits addressing and detection of its own address and General Call address (slave mode).

13.1.1 Features

- Multi-Master Bus
- Serial, 8-bit Oriented and Bi-directional Data Transfers
- 100Kbit/s in Standard Mode and up to 400Kbit/s in Fast mode
- Dedicated DMA channel

13.1.2 Pin Description

Table 13-1 Inter-Integrated Circuit (IIC) Pin Description

Pin Name	Function	I/O Type	Active Level	Comments
SDA[1:0]	Serial Data Line	I/O	High	–
SCL[1:0]	Serial Clock Line	I/O	High	–

13.2 Functional Description

13.2.1 Block Diagram

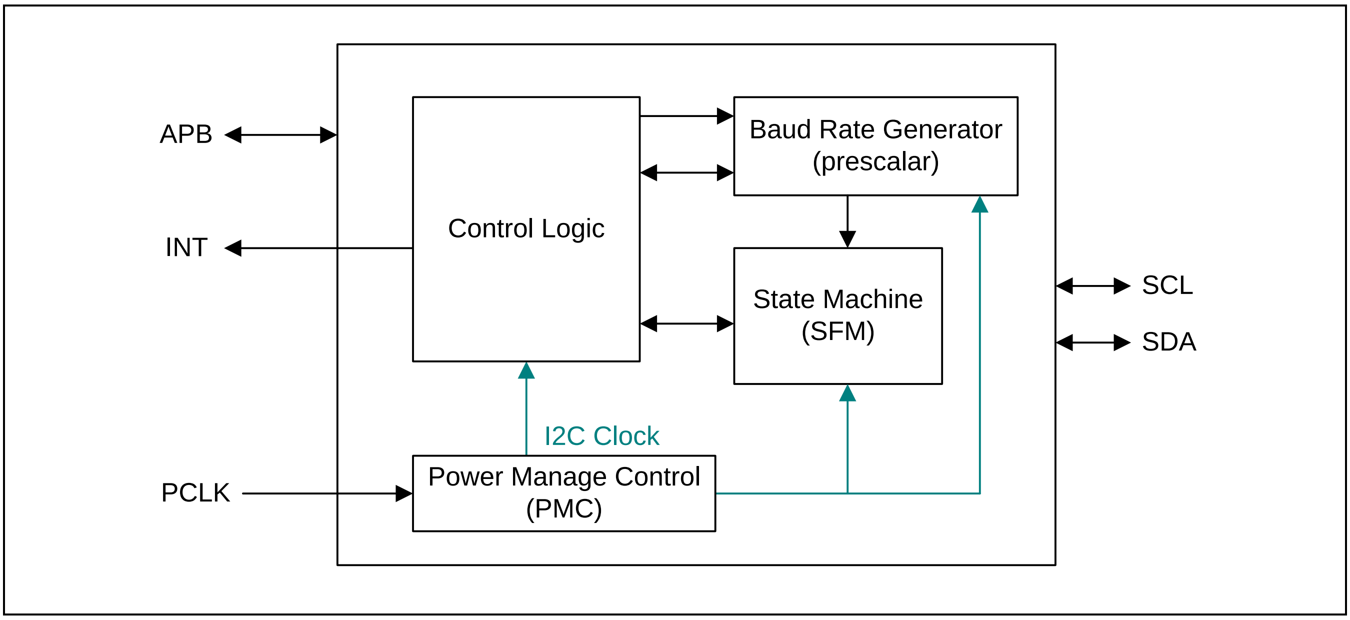


Figure 13-1 Inter-Integrated Circuit (IIC) Block Diagram

13.2.2 Functional Operation

13.2.2.1 I2C Bus Concept

13.2.2.1.1 General Description

Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. Each device is recognized by a unique address – whether it’s a microcontroller, LCD driver, memory or keyboard interface - and can operate as either a transmitter or receiver, depending on the function of the device. Obviously an LCD driver is only a receiver, whereas a memory can both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

The I2C bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually micro-controllers, let’s consider the case of a data transfer between two microcontrollers connected to the I2C bus. This highlights the master-slave and receiver-transmitter relationships to be found on the I2C bus. It should be noted that these relationships are not permanent, but only depend on the direction of data transfer at that time. The transfer of data would proceed as follows:

- Suppose microcontroller A wants to send information to microcontroller B
  - Microcontroller A (Master) addresses microcontroller B (Slave).
  - Microcontroller A (Master-Transmitter) sends data to microcontroller B (Slave).
  - Microcontroller A terminates the transfer.

- If microcontroller A wants to receive information from microcontroller B
  - Microcontroller A (Master) addresses microcontroller B (Slave).
  - Microcontroller A (Master-Receiver) receives data from microcontroller B (Slave-Transmitter).
  - Microcontroller A terminates the transfer.

Even in this case, the master (Microcontroller A) generates the timing and terminates the transfer.

The possibility of connecting more than one microcontroller to the I2C bus means that more than one master could try to initiate a data transfer at the same time. To avoid the chaos that might ensue from such an event – an arbitration procedure has been developed. This procedure relies on the wired-AND connection of all I2C interfaces to the I2C bus.

If two or more masters try to put information onto the bus, the first to produce a ’one’ when the other produces a ’zero’ will lose the arbitration. The clock signals during arbitration are a synchronized combination of the clocks generated by the masters using the wired-AND connection to the SCL.

Generation of clock signals on the I2C bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master can only be altered when they are stretched by a slow-slave device holding-down the clock line or by another master when arbitration occurs.

13.2.2.1.2 General Characteristics

Both SDA and SCL are bi-directional lines, connected to a positive supply voltage via a pull-up. When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-drain in order to perform the wired-AND function. Data on the I2C bus can be transferred at a rate up to 100Kbit/s in the standard mode, or up to 400Kbit/s in the fast mode. The number of interfaces connected to the bus is solely dependent on the bus capacitance limit of 400pF.

The table below shows some examples of configuration for some pre-defined baud rates, depending on the I2C clock, FAST mode and PRV field (in I2C_MR register):

Table 13-2 Examples of Baud Rate Configuration

I2C Clock	PRV	Baud Rate	FAST	% Error
75	777	96000	0	–0.03%
	596	125000	1	0.00%
	387	192000	1	0.10%
	191	384000	1	–0.16%
72	746	96000	0	0.00%
	572	125000	1	0.00%
	371	192000	1	0.00%
	184	384000	1	0.27%
60	621	96000	0	0.00%
	476	125000	1	0.00%
	309	192000	1	0.16%
	152	384000	1	–0.16%
50	517	96000	0	0.03%
	396	125000	1	0.00%
	256	192000	1	–0.16%
	126	384000	1	–0.16%
40	413	96000	0	0.08%
	316	125000	1	0.00%
	204	192000	1	–0.16%
	100	384000	1	–0.16%
36	371	96000	0	0.00%
	284	125000	1	0.00%
	184	192000	1	0.27%
	90	384000	1	0.27%
30	309	96000	0	0.16%
	236	125000	1	0.00%
	152	192000	1	–0.16%
	74	384000	1	–0.16%

I2C Clock	PRV	Baud Rate	FAST	% Error
20	204	96000	0	–0.16%
	156	125000	1	0.00%
	100	192000	1	–0.16%
	48	384000	1	–0.16%
18	184	96000	0	0.27%
	140	125000	1	0.00%
	90	192000	1	0.27%
	43	384000	1	0.27%
37.5	387	96000	0	0.10%
	296	125000	1	0.00%
	191	192000	1	–0.16%
	94	384000	1	0.35%
18.75	191	96000	0	–0.16%
	146	125000	1	0.00%
	94	192000	1	0.35%
	45	384000	1	0.35%
10	100	96000	0	–0.16%
	76	125000	1	0.00%
	48	192000	1	–0.16%
	22	384000	1	–0.16%
9.375	94	96000	0	0.35%
	71	125000	1	0.00%
	45	192000	1	0.35%
4.6875	45	96000	0	0.35%

13.2.2.2 Bit Transfer

Due to the variety of different technology devices (CMOS, NMOS, bipolar) which can be connected to the I2C bus, the levels of the logical ’0’ (LOW) and ’1’ (HIGH) are not fixed and depend on the associated level of VDD. One clock pulse is generated for each data bit transferred.

13.2.2.2.1 Data Validity

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW.

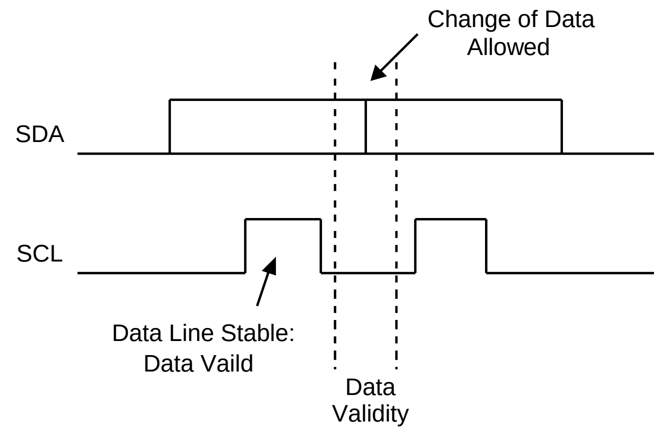


Figure 13-2 Data Validity

13.2.2.2.2 START and STOP Conditions

Within the procedure of the I2C bus, unique situations arise which are defined as START and STOP conditions. A HIGH to LOW transition on the SDA line while SCL is HIGH is one such unique case. This situation indicates a START condition.

A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition. START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition.

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

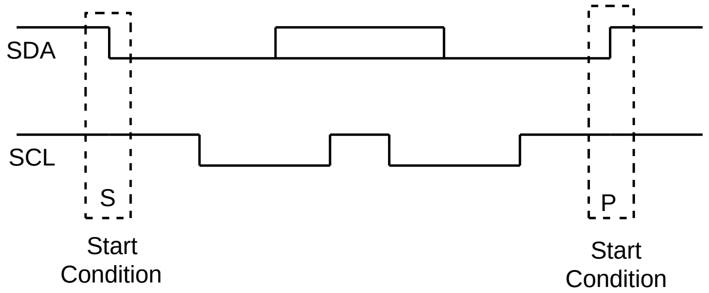


Figure 13-3 Start and Stop Conditions

13.2.2.3 Transferring Data

13.2.2.3.1 Byte Format

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 has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first. If a receiver can't receive 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 LOW to force the transmitter into a wait state. Data transfer then continues when the receiver is ready for another byte of data and releases clock line SCL.

In some cases, it's permitted to use a different format from the I2C-bus format (for CBUS compatible devices for example). A message which starts with such an address can be terminated by generation of a STOP condition, even during the transmission of a byte. In this case, no acknowledge is generated.

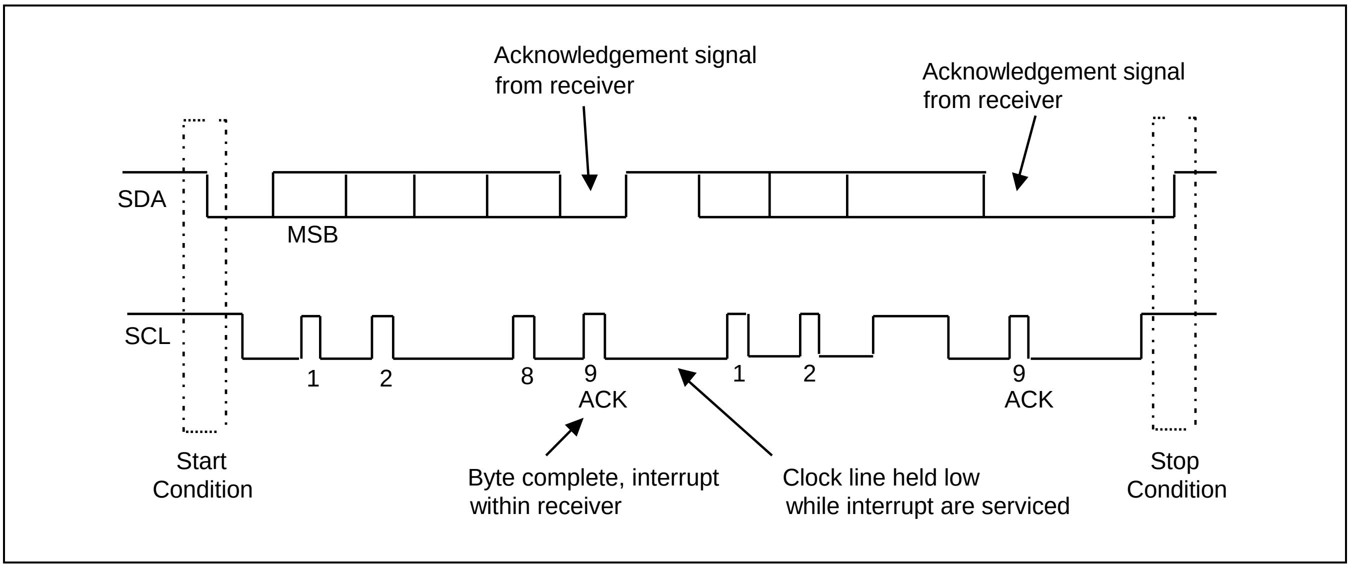

Acknowledgement signal from receiver
Acknowledgement signal from receiver
SDA
MSB
SCL
1
2
8
9
ACK
1
2
9
ACK
Start Condition
Byte complete, interrupt within receiver
Clock line held low while interrupt are serviced
Stop Condition

Figure 13-4 Data Transfer on the I2C Bus

13.2.2.3.2 Acknowledge

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

The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. Of course, set-up and hold times must also be taken into account.

Usually, a receiver which has been addressed is obliged to generate an Acknowledge after each byte has been received, except when the message starts with a CBUS address.

When a slave-receiver doesn't acknowledge the slave address (for example, it's unable to receive because it's performing some real-time function), the data line must be left HIGH by the slave. The master can then generate a STOP condition to abort the transfer.

If a slave-receiver does acknowledge the slave address but, some time later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave generating the not acknowledge on the first byte to follow. The slave leaves the data line HIGH and the master generates the STOP condition.

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

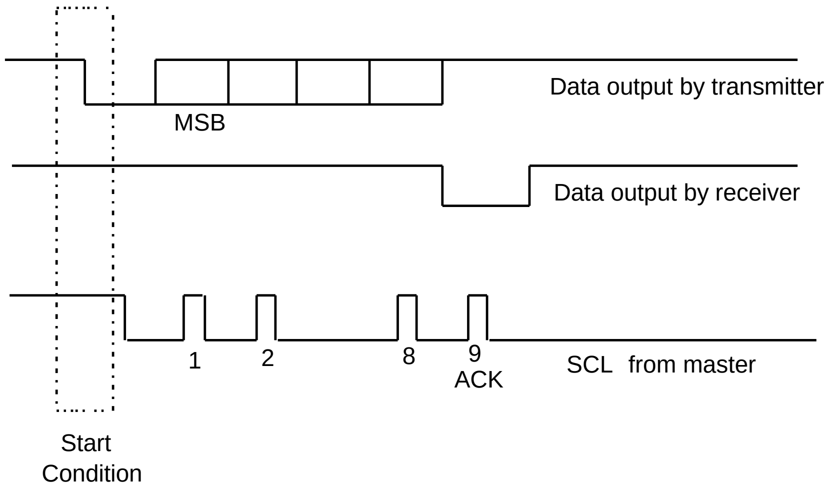


Figure 13-5 Acknowledge

13.2.2.4 Arbitration on Clock Generation

13.2.2.4.1 Synchronization

All masters generate their own clock on the SCL line to transfer messages on the I2C-bus. Data is only valid during the HIGH 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 a HIGH to LOW transition on the SCL line will cause the devices concerned to start counting off their LOW period and, once a device clock has gone LOW, it will hold the SCL line in that state until the clock HIGH state is reached. However, the LOW to HIGH transition of this clock may not change the state of the SCL line if another clock is still within its LOW period. The SCL line will therefore be held LOW by the device with the longest LOW period. Devices with shorter LOW periods enter a HIGH wait-state during this time. When all devices concerned have counted off their LOW period, the clock line will be released and go HIGH. There will then be no difference between the device clocks and the state of the SCL line, and all the devices will start counting their HIGH periods. The first device to complete its HIGH period will again pull the SCL line LOW.

In this way, a synchronized SCL clock is generated with its LOW period determined by the device with the longest clock LOW period, and its HIGH period determined by the one with the shortest clock HIGH period.

13.2.2.4.2 Arbitration

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

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

Arbitration can continue for many bits. Its first stage is comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with comparison of the data. Because address and data information on the I2C-bus is used for arbitration, no information is lost during this process.

A master which 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’s possible that the winning master is trying to address it. The losing master must therefore switch over immediately to its slave-receiver mode.

Since control of the I2C-bus is decided solely on the address and data sent by competing masters, there is no central master, nor any order of priority on the bus.

Special attention must be paid if, during a serial transfer, the arbitration procedure is still in progress at the moment when a repeated START condition or a STOP condition is transmitted to the I2C-bus. If it’s possible for such a situation to occur, the masters involved must send this repeated START condition or STOP condition at the same position in the format frame. In other words, arbitration isn’t allowed between:

- A Repeated START condition and a data bit
- A STOP condition and a data bit
- A Repeated START condition and a STOP condition

13.2.2.4.3 Use of the Clock Synchronizing Mechanism as a Handshake

In addition to being used during the arbitration procedure, the clock synchronization mechanism can be used to enable receivers to cope with fast data transfers, on either a byte level or a bit level.

On the byte level, a device may be able to receive bytes of data at a fast rate, but needs more time to store a received byte or prepare another byte to be transmitted. Slaves can then hold the SCL line LOW after reception and acknowledgement of a byte to force the master into a wait state until the slave is ready for the next byte transfer in a type of handshake procedure.

On the bit level, a device such as a microcontroller without, or with only a limited hardware I2C interface on-chip can slow down the bus clock by extending each clock LOW period. The speed of any master is thereby adapted to the internal operating rate of this device.

13.2.2.4.4 Formats with 7-bits Addresses

After the START condition (S), a slave address is sent. This address is 7 bits long followed by an eighth bit which is a data direction bit (R/W) – a ’zero’ indicates a transmission (WRITE), a ’one’ indicates a request for data (READ). A data transfer is always terminated by a STOP condition (P) generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition (Sr) and address another slave without first generating a STOP condition. Various combinations of read/write formats are then possible within such a transfer.

Possible data transfer formats are:

- Master-transmitter transmits to slave-receiver. The transfer direction is not changed
- Master reads slave immediately after first byte.

At the moment of the first acknowledge, the master-transmitter becomes a master-receiver and the slave-receiver becomes a slave-transmitter. This Acknowledge is still generated by the slave.

The STOP condition is generated by the master.

- Combined formats. During a change of direction within a transfer, the START condition and the slave address are both repeated, but with the R/W bit reversed. If a master receiver sends a repeated START condition, it has previously sent a not acknowledge (A).

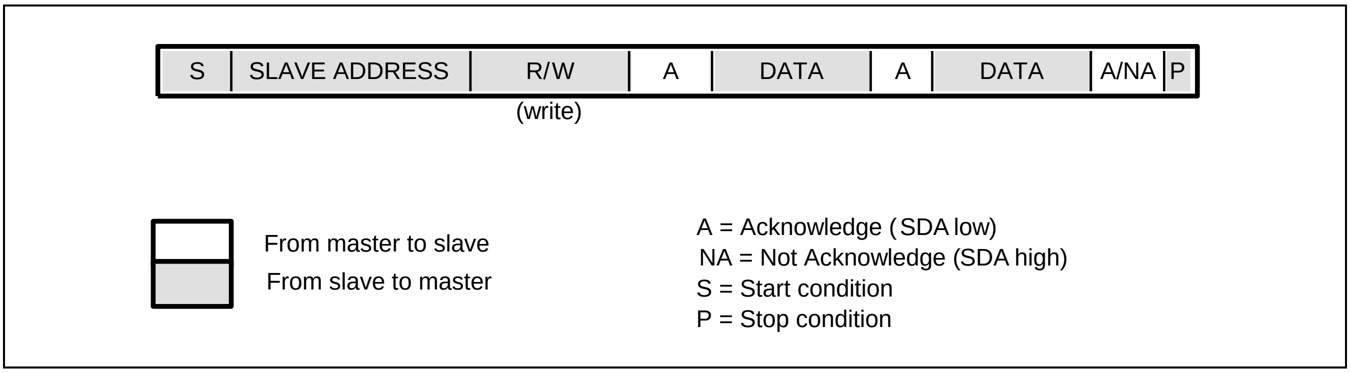


Figure 13-6 A Master-Transmitter Addresses a Slave

13.2.2.5 7-Bits Addressing

The addressing procedure for the I2C-bus is such that the first byte after the START condition usually determines which slave will be selected by the master. The exception is the 'general call' address which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge. However, devices can be made to ignore this address. The second byte of the general call address then defines the action to be taken.

13.2.2.5.1 Definition of Bits in the First Byte

The first seven bits of the first byte make up the slave address. The eighth bit is the LSB (least significant bit). It determines the direction of the message. A 'zero' in the least significant position of the first byte means that the master will write information to a selected slave. A 'one' in this position means that the master will read information from the slave.

When an address is sent, each device in a system compares the first seven bits after the START condition with its address. If they match, the device considers itself addressed by the master as a slave-receiver or slave-transmitter, depending on the R/W bit.

A slave address can be made-up of a fixed and a programmable part. Since it's likely that there will be several identical devices in a system, the programmable part of the slave address enables the maximum possible number of such devices to be connected to the I2C-bus. The number of programmable address bits of a device depends on the number of pins available. For example, if a device has 4 fixed and 3 programmable address bits, a total of 8 identical devices can be connected to the same bus.

The I2C-bus committee coordinates allocation of I2C addresses.

Two groups of eight addresses (0000XXX and 1111XXX) are reserved for the purposes shown in *Table 13-3*. The bit combination 11110XX of the slave address is reserved for 10-bit addressing.

Table 13-3 Definition of Bytes in First Byte

Slave Address	RNW bit	Description
0000 000	0	General call address
0000 000	1	START byte [1]
0000 001	X	CBUS address [2]
0000 010	X	Reserved for different bus format [3]
0000 011	X	Reserved for future purposes
0000 1XX	X	HS-mode master code
1111 1XX	X	Reserved for future purposes
1111 0XX	X	10–bit slave addressing

NOTE:

1. No device is allowed to acknowledge at the reception of the START byte.
2. The CBUS address has been reserved to enable the inter-mixing of CBUS compatible and I2C -bus compatible devices in the same system. I2C -bus compatible devices are not allowed to respond on reception of this address.
3. The address reserved for a different bus format is included to enable I2C and other protocols to be mixed. Only I2C-bus compatible devices that can work with such formats and protocols are allowed to respond to this address.

The general call address is for addressing every device connected to the I2C-bus. However, if a device doesn't need any of the data supplied within the general call structure, it can ignore this address by not issuing an acknowledgement. If a device does require data from a general call address, it will acknowledge this address and behave as a slave-receiver.

The second and following bytes will be acknowledged by every slave-receiver capable of handling this data.

A slave which cannot process one of these bytes must ignore it by not acknowledging. The meaning of the general call address is always specified in the second byte.

There are two cases to consider:

- When the least significant bit B is a 'zero'
- When the least significant bit B is a 'one'

When bit B is a 'zero'; the second byte has the following definition:

- 00000110 (H'06'). Reset and write programmable part of slave address by hardware. On receiving this 2-byte sequence, all devices designed to respond to the general call address will reset and take in the programmable part of their address. Precautions have to be taken to ensure that a device is not pulling down the SDA or SCL line after applying the supply voltage, since these low levels would block the bus.
- 00000100 (H'04'). Write programmable part of slave address by hardware. All devices which define the programmable part of their address by hardware (and which respond to the general call address) will latch this programmable part at the reception of this two byte sequence. The device will not reset.
- 00000000 (H'00'). This code is not allowed to be used as the second byte.

The remaining codes have not been fixed and devices must ignore them.

When bit B is a 'one'; the 2-byte sequence is a 'hardware general call'. This means that the sequence is transmitted by a hardware master device, such as a keyboard scanner, which cannot be programmed to transmit a desired slave address. Since a hardware master doesn't know in advance to which device the message has to be transferred, it can only generate this hardware general call and its own address - identifying itself to the system.

The seven bits remaining in the second byte contain the address of the hardware master. This address is recognized by an intelligent device (e.g. a microcontroller) connected to the bus which will then direct the information from the hardware master. If the hardware master can also act as a slave, the slave address is identical to the master address.

In some systems, an alternative could be that the hardware master transmitter is set in the slave-receiver mode after the system reset.

In this way, a system configuring master can tell the hardware master-transmitter (which is now in slave-receiver mode) to which address data must be sent. After this programming procedure, the hardware master remains in the master-transmitter mode.

13.2.2.5.2 Start Byte

Microcontrollers can be connected to the I2C -bus in two ways. A microcontroller with an on-chip hardware I2C-bus interface can be programmed to be only interrupted by requests from the bus. When the device doesn't have such an interface, it must constantly monitor the bus via software. Obviously, the more times the microcontroller monitors, or polls the bus, the less time it can spend carrying out its intended function. There is therefore a speed difference between fast hardware devices and a relatively slow microcontroller which relies on software polling.

In this case, data transfer can be preceded by a start procedure which is much longer than normal.

The start procedure consists of:

- A START condition (S)
- A START byte (00000001)
- An Acknowledge clock pulse (ACK)
- A Repeated START condition (Sr)

After the START condition S has been transmitted by a master which requires bus access, the START byte (00000001) is transmitted. Another microcontroller can therefore sample the SDA line at a low sampling rate until one of the seven zeros in the START byte is detected. After detection of this LOW level on the SDA line, the microcontroller can switch to a higher sampling rate to find the repeated START condition Sr which is then used for synchronization.

A hardware receiver will reset on receipt of the repeated START condition Sr and will therefore ignore the START byte.

An acknowledge-related clock pulse is generated after the START byte. This is present only to conform with the byte handling format used on the bus. No device is allowed to acknowledge the START byte.

13.2.3 Extensions to the I2C-bus Specification

The I2C -bus with a data transfer rate of up to 100kbit/s and 7-bit addressing has now been in existence for more than ten years with an unchanged specification. The concept is accepted world-wide as a de facto standard and hundreds of different types of I2C-bus compatible ICs are available. The I2C-bus specification is now extended with the following two features:

- A Fast-Mode which allows a fourfold increase of the bit rate to 0 to 400kbit/s.
- 10-bit Addressing which allows the use of up to 1024 additional addresses.

There are two reasons for these extensions to the I2C-bus specification:

- New applications will need to transfer a larger amount of serial data and will therefore demand a higher bit rate than 100kbit/s. Improved IC manufacturing technology now allows a fourfold speed increase without increasing the manufacturing cost of the interface circuitry.
- Most of the 112 addresses available with the 7-bit addressing scheme have been issued more than once. To prevent problems with the allocation of slave addresses for new devices, it is desirable to have more address combinations. About a tenfold increase of the number of available addresses is obtained with the new 10-bit addressing.

All new devices with an I2C-bus interface are provided with the fast-mode. Preferably, they should be able to receive and/or transmit at 400kbit/s. The minimum requirement is that they can synchronize with a 400kbit/s transfer; they can then prolong the LOW period of the SCL signal to slow down the transfer. Fast-mode devices must be downward-compatible which means that they must still be able to communicate with 0 to 100kbit/s devices in a 0 to 100kbit/s I2C-bus system.

Obviously, devices with a 0 to 100kbit/s I2C-bus interface cannot be incorporated in a fast-mode I2C-bus system because, since they cannot follow the higher transfer rate, unpredictable states of these devices would occur.

Slave devices with a fast-mode I2C-bus interface can have a 7-bit or a 10-bit slave address. However, a 7-bit address is preferred because it is the cheapest solution in hardware and it results in the shortest message length. Devices with 7-bit and 10-bit addresses can be mixed in the same I2C-bus system regardless of whether it is a 0 to 100kbit/s standard-mode system or a 0 to 400kbit/s fast-mode system. Both existing and future masters can generate either 7-bit or 10-bit addresses.

13.2.4 1 Fast Mode

In the fast-mode of the I2C-bus, the protocol, format, logic levels and maximum capacitive load for the SDA and SCL lines quoted in the previous I2C-bus specification are unchanged. Changes to the previous I2C-bus specification are:

- The maximum bit rate is increased to 400kbit/s
- Timing of the serial data (SDA) and serial clock (SCL) signals has been adapted. There is no need for compatibility with other bus systems such as CBUS because they cannot operate at the increased bit rate.
- The inputs of fast-mode devices must incorporate spike suppression and a Schmitt trigger at the SDA and SCL inputs.
- The output buffers of fast-mode devices must incorporate slope control of the falling edges of the SDA and SCL signals.
- If the power supply to a fast-mode device is switched off, the SDA and SCL I/O pins must be floating so that they don’t obstruct the bus lines.
- The external pull-up devices connected to the bus lines must be adapted to accommodate the shorter maximum permissible rise time for the fast-mode I2C-bus. For bus loads up to 200pF, the pull-up device for each bus line can be a resistor; for bus loads between 200pF and 400pF, the pull-up device can be a current source (3mA max.) or a switched resistor.

13.2.4.1 10-Bits Addressing

The 10-bit addressing does not change the format in the I2C-bus specification. Using 10 bits for addressing exploits the reserved combination 1111XXX for the first seven bits of the first byte following a START (S) or repeated START (Sr) condition.

The 10-bit addressing does not affect the existing 7-bit addressing. Devices with 7-bit and 10-bit addresses can be connected to the same I2C-bus, and both 7-bit and 10-bit addressing can be used in a standard-mode system (up to 100kbit/s) or a fast-mode system (up to 400kbit/s).

Although there are eight possible combinations of the reserved address bits 1111XXX, only the four combinations 11110XX are used for 10-bit addressing. The remaining four combinations 11111XX are reserved for future I2C-bus enhancements.

A – Definition of bits in the first two bytes

The 10-bit slave address is formed from the first two bytes following a START condition (S) or a repeated START condition (Sr).

The first seven bits of the first byte are the combination 11110XX of which the last two bits (XX) are the two most-significant bits (MSBs) of the 10-bit address; the eighth bit of the first byte is the R/W bit that determines the direction of the message. A ’zero’ in the least significant position of the first byte means that the master will write information to a selected slave. A ’one’ in this position means that the master will read information from the slave.

If the R/W bit is ’zero’, then the second byte contains the remaining 8 bits (XXXXXXXX) of the 10-bit address. If the R/W bit is ’one’, then the next byte contains data transmitted from a slave to a master.

B – Formats with 10-bit Addresses

Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing. Possible data transfer formats are:

- Master-transmitter transmits to slave-receiver with a 10-bit slave address. The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first seven bits of the first byte of the slave address (11110XX) with its own address and tests if the eighth bit (R/W direction bit) is 0. It is possible that more than one device will find a match and generate an acknowledge (A1). All slaves that found a match will compare the eight bits of the second byte of the slave address (XXXXXXXX) with their own addresses, but only one slave will find a match and generate an acknowledge (A2). The matching slave will remain addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.
- Master-receiver reads slave- transmitter with a 10-bit slave address. The transfer direction is changed after the second R/W bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks if the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and tests if the eighth (R/W) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3.

The slave-transmitter remains addressed until it receives a STOP condition (P) or until it receives another repeated START condition (Sr) followed by a different slave address. After a repeated START condition (Sr), all the other slave devices will also compare the first seven bits of the first byte of the slave address (11110XX) with their own addresses and test the eighth (R/W) bit. However, none of them will be addressed because R/W = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

- Combined format. A master transmits data to a slave and then reads data from the same slave. The same master occupies the bus all the time. The transfer direction is changed after the second R/W bit – Combined format. A master transmits data to one slave and then transmits data to another slave. The same master occupies the bus all the time.
- Combined format. 10-bit and 7-bit addressing combined in one serial transfer. After each START condition (S), or each repeated START condition (Sr), a 10-bit or 7-bit slave address can be transmitted.

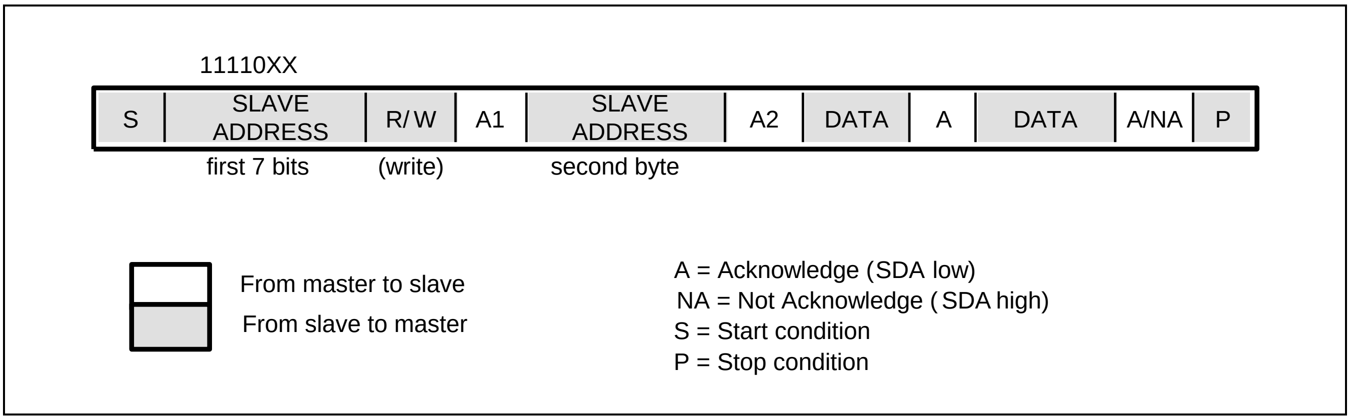


Figure 13-7 A Master-Transmitter Addresses a Slave-Receiver with a 10–bits Address

13.2.4.2 General Call Address and Start Byte

The 10-bit addressing procedure for the I2C-bus is such that the first two bytes after the START condition (S) usually determine which slave will be selected by the master. The exception is the ’general call’ address 00000000 (H’00’).

Slave devices with 10-bit addressing will react to a ’general call’ in the same way as slave devices with 7-bit addressing.

Hardware masters can transmit their 10-bit address after a ’general call’. In this case, the ’general call’ address byte is followed by two successive bytes containing the 10-bit address of the master-transmitter.

The START byte 00000001 (H’01’) can precede the 10-bit addressing in the same way as for 7-bit addressing.

13.3 Inter-Integrated Circuit Timing

Table 13-4 Timing Requirements

Parameter	Symbol	Standard-Mode I2C Bus		Fast-Mode I2C Bus		Unit
		Min	Max	Min	Max	
SCL clock frequency	FSCL	0	100	0	400	kHz
Bus free time between a STOP and START condition	TBUF	4.7	–	1.3	–	us
Hold time (repeated) START condition. After this period, the first clock pulse is generated	THD;STA	4.0	–	0.6	–	us
Low period of the SCL clock	TLOW	4.7	–	1.3	–	us
High period of the SCL clock	THIGH	4.0	–	0.6	–	us
Set-up time for a repeated START condition	TSU;STA	4.7	–	0.6	–	us
Data hold time	THD;DAT	0	–	0	0.9	us
Data set-up time	TSU;DAT	250	–	100	–	ns
Rise time for both SDA and SCL signals	Tr	–	1000	20+01Cb	300	ns
Fall time for both SDA and SCL signals	Tf	–	300	20+01Cb	300	ns
Set-up time for STOP condition	TSU;STO	4.0	–	0.6	–	us
Capacitive load for each bus line	Cb	–	400	–	400	pF

13.4 Register Description

13.4.1 Register Map Summary

- Base Address: 0x400A_0000
- Base Address: 0x400A_1000

Register	Offset	Description	Reset Value
I2C_IDR	0x0000	ID Register	0x0001_000E
I2C_CEDR	0x0004	Clock Enable/Disable Register	0x0000_0000
I2C_SRR	0x0008	Software Reset Register	0x0000_0000
I2C_CR	0x000C	Control Register	0x0000_0000
I2C_MR	0x0010	Mode Register	0x0000_01F4
I2C_SR	0x0014	Status Register	0x0000_00F8
I2C_IMSCR	0x0018	Interrupt Mask Set/Clear Register	0x0000_0000
I2C_RISR	0x001C	Raw Interrupt Status Register	0x0000_0000
I2C_MISR	0x0020	Masked Interrupt Status Register	0x0000_0000
I2C_ICR	0x0024	Interrupt Clear Register	0x0000_0000
I2C_SDR	0x0028	Serial Data Register	0x0000_0000
I2C_SSAR	0x002C	Serial Slave Address Register	0x0000_0000
I2C_HSDR	0x0030	Hold/Setup Delay Register	0x0000_0001
I2C_DMACR	0x0034	DMA Control Register	0x0000_0000

13.4.1.1 I2C_IDR (IIC ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_000E

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_000E

13.4.1.2 I2C_CEDR (IIC Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable Control Bit. 0 = Disable I2C Clock 1 = Enables I2C Clock I2C software reset dose not affect CLKEN bit status.	0

13.4.1.3 I2C_SRR (IIC Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No effect 1 = Perform I2C Software Reset operation	0

13.4.1.4 I2C_CR (IIC Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																							ENA	RSVD				STA	STO	AA	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	R	R	R	R	RW	RW	RW	R

Name	Bit	Type	Description	Reset Value
AA	[1]	RW	I2C Acknowledge. 0 = No acknowledge is returned (SDA remains high during the acknowledge SCL clock pulses). 1 = Acknowledge is returned when the “own slave address” is received (or the general call address has been received while bit GC of I2C_ADR is set) or when a data byte has been received while being in the receiver mode.	0
STO	[2]	RW	I2C Stop. 0 = Clear not to send a STOP condition on the bus. 1 = Generates a stop condition. When the STOP condition is detected on bus, the STO bit is automatically cleared. In slave mode, this bit is used to recover from a bus error. In such case, no STOP condition is transmitted, but the interface do as if a STOP condition has been received and switch to the ’not addressed’ slave mode (the STO bit is cleared by the interface).	0
STA	[3]	RW	I2C Start. 0 = Set to ’0’ put the Interface in slave mode. 1 = When set to ’1’, the interface enters in master mode, check the status of the I2C bus and generates a START condition if the bus is free. If the bus is not free, the Interface will wait until a STOP condition to generate a START condition (after a minimum time). Set to ’0’ put the Interface in slave mode.	0
ENA	[8]	RW	I2C Enable. 0 = Disables the I2C Interface (When the I2C interface is disabled, the SCL and SDA lines are disabled. No output is generated and no input is taken account). 1 = Enables the I2C Interface.	0

13.4.1.5 I2C_MR (IIC Mode Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_01F4

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

Name	Bit	Type	Description	Reset Value
PRV	[11:0]	RW	Pre-scalar value. These bits select the speed of the bus (FSCL). The value of PRV (pre-scaler Value) is used to generate the FSCL with the formula: FSCL = PCLK / (PRV+4)	0x1F4
FAST	[12]	RW	Fast mode. 0 = Disables FAST mode. Enables STANDARD mode. In this mode the high to low ratio is 1:1 and the maximum baud rate is 100kHz. 1 = Enables FAST mode. In this mode the high to low ratio is 2:3 and the maximum baud rate is 400kHz.	0

Table 13-5 Values of FSCL in kHz Depending of PRV, FAST and PCLK

PRV (Decimal)	FAST	PCLK (MHz)				
		10	20	25	40	50
500	0	19.8	39.7	49.6	80	99.2
400	0	24.7	49.5	61.8	99	–
250	0	39.3	78.7	98.4	–	–
200	0	49	98	–	–	–
125	0	77.5	–	–	–	–
125	1	77.5	155	194	310	387
100	1	96	192	240	400	–
62.5	1	150	300	375	–	–
50	1	185	370	–	–	–
25	1	344	–	–	–	–

NOTE:

1. The PCLK frequency must be at least 6 times greater than the faster FSCL frequency used on the I2C bus (SCL re-synchronization + state machine).
2. The PCLK frequency must be 6 times greater than the desired FSCL value programmed in PRV.
3. PRV value after reset is '500' (decimal).
4. It is not possible to write '000' (hexadecimal) to PRV.

13.4.1.6 I2C_SR (IIC Status Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_00F8

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

Name	Bit	Type	Description	Reset Value
SR	[7:3]	R	I2C status code. Interface Status code. There are 27 possible status codes. I2C_SR reset value is 0x000000F8. When the I2C_SR contains this status code, no relevant status information is available. All other status codes correspond to a defined state of the interface. When each of these states is entered the corresponding status code appears in this register and the bit SI of I2C_CR is set. All of these status codes, their meaning, the next action to be performed by the sequencer/controller (software) driving the interface and the next action the interface will perform, are described next page.	0x1F

Table 13-6 Master/Transmitter Mode Status Codes

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
0x0000_0008	START condition has been transmitted	x	0	x	0	Load I2C_DAT with slave address and R/W bit. Reset SI of I2C_CR.	Slave address and R/W will be transmitted. Wait ACK.
0x0000_0010	REPEAT START condition has been transmitted	x	0	x	0	Load I2C_DAT with slave address and R/W bit. Reset SI of I2C_CR.	Slave address and R/W will be transmitted. Wait ACK. If R/W is Read, interface switch into receiver mode.
0x0000_0018	Slave address and WRITE has been sent, ACK received	0	0	x	0	Load I2C_DAT with data byte. Reset SI of I2C_CR.	Data byte will be transmitted. Wait for ACK.
		1	0	x	0	Set STA of I2C_CR. Reset SI of I2C_CR.	Generates repeated START condition.
		0	1	x	0	Set STO of I2C_CR Reset SI of I2C_CR.	Generates STOP condition.
		1	1	x	0	Set STA and STO of I2C_CR. Reset SI of I2C_CR.	Generates STOP condition, then generates SART condition
0x0000_0020	Slave address and WRITE has been sent, No ACK received	As above.				As above.	As above.
0x0000_0028	Data byte transmitted, ACK received	As above.				As above.	As above.
0x0000_0030	Data byte transmitted, No ACK received	As above.				As above.	As above.
0x0000_0038	Arbitration lost in Slave address and R/W bit transmission or in data byte transmission	0	0	x	0	Reset SI of I2C_CR.	Release I2C bus, switch to slave mode.
		1	0	x	0	Set STA of I2C_CR. Reset SI of I2C_CR.	Wait until the I2C is free to generate a START condition.

Table 13-7 Master/Receiver Mode Status Codes

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
0x0000_0008	START condition has been transmitted	x	0	x	0	Load I2C_DAT with slave address and R/W bit. Reset SI of I2C_CR.	Slave address and R/W will be transmitted. Wait ACK.
0x0000_0010	REPEAT START condition has been transmitted	x	0	x	0	Load I2C_DAT with slave address and R/W bit. Reset SI of I2C_CR.	Slave address and R/W will be transmitted. Wait ACK. If R/W is Read, interface switch into receiver mode.
0x0000_0038	Arbitration lost in Slave address and R/W bit transmission	0	0	x	0	Reset SI of I2C_CR.	Release I2C bus, switch to slave mode.
		1	0	x	0	Set STA of I2C_CR. Reset SI of I2C_CR.	Wait until the I2C is free to generate a START condition.
0x0000_0040	Slave address and Read has been sent, ACK received	0	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		0	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0048	Slave address and Read has been sent, No ACK received	1	0	x	0	Set STA of I2C_CR Reset SI of I2C_CR.	Generates repeated START condition.
		0	1	x	0	Set STO of I2C_CR. Reset SI of I2C_CR.	Generates STOP condition.
		1	1	x	0	Set STA and STO of I2C_CR. Reset SI of I2C_CR.	Generates STOP condition, then generates START condition
0x0000_0050	Data byte received, ACK returned	0	0	1	0	Read data byte, reset SI of I2C_CR, set AA of I2C_CR.	New data byte will be received and ACK will be returned.
		0	0	0	0	Read data byte, reset SI of I2C_CR, reset AA of I2C_CR.	New data byte will be received and no ACK will be returned.
0x0000_0058	Data byte received, No ACK returned.	1	0	x	0	Read data byte, set STA of I2C_CR, reset SI of I2C_CR.	Generates repeated START condition.
		0	1	x	0	Read data byte, set STO of I2C_CR, reset SI of I2C_CR.	Generates STOP condition.
		1	1	x	0	Read data byte, set STO of I2C_CR, set	Generates STOP condition, then

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
						STA of I2C_CR, reset SI of I2C_CR.	generates START condition

Table 13-8 Slave/Receiver Mode Status Codes

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
0x0000_0060	Own Slave address +W received, ACK returned.	x	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		x	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0068	Arbitration lost in Slave address + R/W (in master mode); switch to slave mode, own slave address received; ACK has been returned.	x	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		x	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0070	General Call Address has been received; ACK has been returned.	x	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		x	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0078	Arbitration lost in Slave address + R/W (in master mode); switch to slave mode, general call address received; ACK has been returned.	x	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		x	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0080	Addressed with own address and W, data byte received, ACK returned.	x	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK will be returned.
		x	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, No ACK will be returned.
0x0000_0088	Addressed with own	0	0	0	0	Read data byte,	Switch to ’not

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
	address, data byte received, no ACK returned.					reset SI of I2C_CR. Reset AA of I2C_CR.	addressed’ slave mode. Own address and general call recognition disabled.
		0	0	1	0	Read data byte, reset SI of I2C_CR. Set AA of I2C_CR.	Switch to ’not addressed’ slave mode. Acknowledge own slave address and general call (if GC=’1’ of I2C_ADR).
		1	0	0	0	Read data byte, reset SI of I2C_CR. Reset AA of I2C_CR.Set STA of I2C_CR.	Switch to ’not addressed’ slave mode. Own address and general call recognition disabled. Start will be transmitted as soon as the bus becomes free.
		1	0	1	0	Read data byte, reset SI of I2C_CR. Set AA of I2C_CR. Set STA of I2C_CR.	Switch to ’not addressed’ slave mode. Acknowledge own slave address and general call (if GC=’1’ of I2C_ADR). Start will be transmitted as soon as the bus becomes free.
0x0000_0090	Addressed by general call address, data byte received, ACK returned.	x	0	1	0	Read data byte, reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be received, ACK returned.
		x	0	0	0	Read data byte, reset SI of I2C_CR. Reset AA of I2C_CR.	Data byte will be received, no ACK returned.
0x0000_0098	Addressed by general call address, data byte received, no ACK returned.	0	0	0	0	Read data byte, reset SI of I2C_CR. Reset AA of I2C_CR.	Switch to ’not addressed’ slave mode. Own address and general call recognition disabled.
		0	0	1	0	Read data byte, reset SI of I2C_CR. Set AA of I2C_CR.	Switch to ’not addressed’ slave mode. Acknowledge own slave address

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
							and general call (if GC='1' of I2C_ADR).
		1	0	0	0	Read data byte, reset SI of I2C_CR. Reset AA of I2C_CR. Set STA of I2C_CR	Switch to 'not addressed' slave mode. Own address and general call recognition disabled. Start will be transmitted as soon as the bus becomes free.
		1	0	1	0	Read data byte, reset SI of I2C_CR. Set AA of I2C_CR. Set STA of I2C_CR.	Switch to 'not addressed' slave mode. Acknowledge own slave address and general call (if GC='1' of I2C_ADR). Start will be transmitted as soon as the bus becomes free.
0x0000_00A0	A stop condition or repeated start condition has been received while still addressed as slave.	0	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	As Above.
		0	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	
		1	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR. Set STA of I2C_CR.	
		1	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR. Set STA of I2C_CR.	

Table 13-9 Slave/Transmitter Mode Status Codes

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
0x0000_00A8	Own Slave address + R received, ACK returned.	x	0	1	0	Write data byte. Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be transmitted.
		x	0	0	0	Write data byte. Reset SI of I2C_CR. Reset AA of I2C_CR.	Last data byte will be transmitted and slave will not further respond.
0x0000_00B0	Arbitration lost in slave address + R/W as master; own slave address has been received; ACK has been returned.	x	0	1	0	Write data byte. Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be transmitted.
		x	0	0	0	Write data byte. Reset SI of I2C_CR. Reset AA of I2C_CR.	Last data byte will be transmitted and slave will not further respond.
0x0000_00B8	Data has been transmitted; ACK has been received.	x	0	1	0	Write data byte. Reset SI of I2C_CR. Set AA of I2C_CR.	Data byte will be transmitted.
		x	0	0	0	Write data byte. Reset SI of I2C_CR. Reset AA of I2C_CR.	Last data byte will be transmitted and slave will not further respond.
0x0000_00C0	Data has been transmitted; No ACK has been received.	0	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	Switch to ’not addressed’ slave mode. Own address and general call recognition disabled.
		0	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	Switch to ’not addressed’ slave mode. Acknowledge own slave address and general call (if GC=’1’ of I2C_ADR).
		1	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR. Set STA of I2C_CR.	Switch to ’not addressed’ slave mode. Own address and general call recognition disabled. Start will be transmitted as soon as the bus becomes free.
		1	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR. Set STA of I2C_CR.	Switch to ’not addressed’ slave mode. Acknowledge

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
							own slave address and general call (if GC='1' of I2C_ADR). Start will be transmitted as soon as the bus becomes free.
0x0000_00C8	Last data has been transmitted, ACK received.	0	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR.	As Above.
		0	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR.	
		1	0	0	0	Reset SI of I2C_CR. Reset AA of I2C_CR. Set STA of I2C_CR.	
		1	0	1	0	Reset SI of I2C_CR. Set AA of I2C_CR. Set STA of I2C_CR.	

Table 13-10 Miscellaneous Status Codes

Status Code	Meaning	Next Action to be Done by Controller (Software)					Next Action Interface will Perform
		STA	STO	AA	SI	–	
0x0000_00F8	No revelant state information; SI='0' in I2C_CR	–	–	–	–	No Action.	Wait or Proceed current transfer.
0x0000_0000	Bus error due to an illegal START or STOP condition.	0	1	x	0	No Action.	Only internal hardware is affected. In all cases, the bus is released and STO is reset.

13.4.1.7 I2C_IMSCR (IIC Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																											SI	RSVD			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	R	R	R	R

Name	Bit	Type	Description	Reset Value
SI	[4]	RW	SI interrupt enable Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

13.4.1.8 I2C_RISR (IIC Raw Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																											SI	RSVD			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
SI	[4]	R	SI Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SI interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

13.4.1.9 I2C_MISR (IIC Masked Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																											SI	RSVD			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
SI	[4]	R	SI Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the SI interrupt. An interrupt has to be serviced. When the SI is '1', the i2c_int line is high, and the SCL line is pulled low. The transfer is Suspend until the SI bit is cleared.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect

13.4.1.10 I2C_ICR (IIC Interrupt Clear Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																											SI	RSVD			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	R	R	R	R

Name	Bit	Type	Description	Reset Value
SI	[4]	W	SI Interrupt Clear. 0 = No effect 1 = Clears the SI interrupt	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

13.4.1.11 I2C_SDR (IIC Serial Data Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																								DAT							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
DAT	[7:0]	RW	I2C Data. Contains the incoming byte (just received from the I2C-bus) in the receiver mode and the outgoing data (to be send to the I2C-bus) in transmitter mode. DAT is always shifted from right to left. It means, the first bit transmitted to the I2C-bus is the MSB in transmitter mode, and the MSB is the first bit received from the I2C-bus in the receiver mode. During a transmission (when the interface send data to the I2C-bus) while the data is shifted out, the value on the SDA line is shifted in. So in case of an arbitration lost, DAT will contains the correct data (the value read from the bus).	0x00

13.4.1.12 I2C_SSAR (IIC Serial Slave Address Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_0000

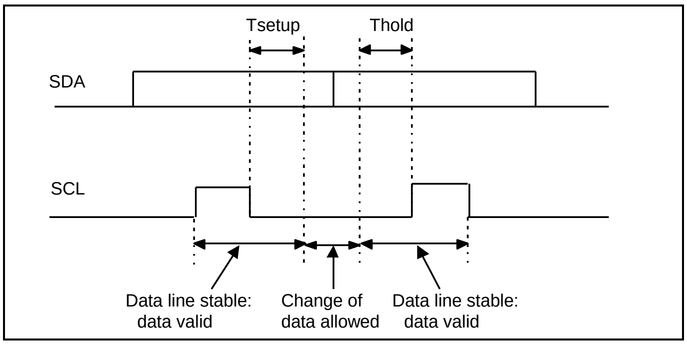
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								ADR							GC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
GC	[0]	RW	General Call. Enable the recognition of ’general call address’. When GC is set to ’1’, the interface will generate an interrupt when the ’general call address’ is recognized.	0
ADR	[7:1]	RW	I2C Address. Contains the 7-bit I2C address to which the interface will respond when programmed as a slave (transmitter or receiver).	0x00

13.4.1.13 I2C_HSDR (IIC Hold/Setup Delay Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_0001

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

Name	Bit	Type	Description	Reset Value
DL	[7:0]	RW	Hold/setup delay. Contains the value of the Hold/Setup delay that is calculated with the formula: THOLD = DL[7-0] × PCLK, TSETUP = DL[7-0] × PCLK  Tsetup Thold SDA SCL Data line stable: data valid Change of data allowed Data line stable: data valid **Figure 13-8 Hold and Setup Delays** **NOTE:** 1. DL value after reset is '1' (hexadecimal). 2. Setup delay (TSETUP) must be 250 ns (min in standard mode) or 100 ns (min in fast mode). 3. An I2C device must internally provide a hold time of at least 300 ns for the SDA signal. It is the user's responsibility to program the correct hold value to meet timing requirements of slow devices. 4. It is not possible to write '0' (hexadecimal) to DL.	0x01

13.4.1.14 I2C_DMACR (IIC DMA Control Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																														TXDMAE	RXDMAE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
RXDMAE	[0]	RW	DMA for the receive Enable/Disable Control Bit. 0 = Disable 1 = Enable	0
TXDMAE	[1]	RW	DMA for the transmit Enable/Disable Control Bit. 0 = Disable 1 = Enable	0

14 Inverter Motor Controller (IMC)

14.1 Overview

This inverter motor controller can be used for 3-phase inverter motor.

14.1.1 Features

- 3-phase PWM signal outputs: (PWMxU0, PWMxD0), (PWMxU1, PWMxD1), (PWMxU2, PWMxD2)
- Dead-time insertion
- 8 compare registers for IMC start trigger signal generation and interrupt
- High-Z output generation

14.1.2 Pin Description

Table 14-1 IMC Pin Description

Pin Name	Function	I/O Type	Comments
PWM[1:0]U[2:0]	PWM up side output for inverter motor	O	–
PWM[1:0]D[2:0]	PWM down side output for inverter motor	O	–
PWM[1:0]OFF	Input pin for PWM output off	I	–

14.2 Functional Description

14.2.1 Block Diagram

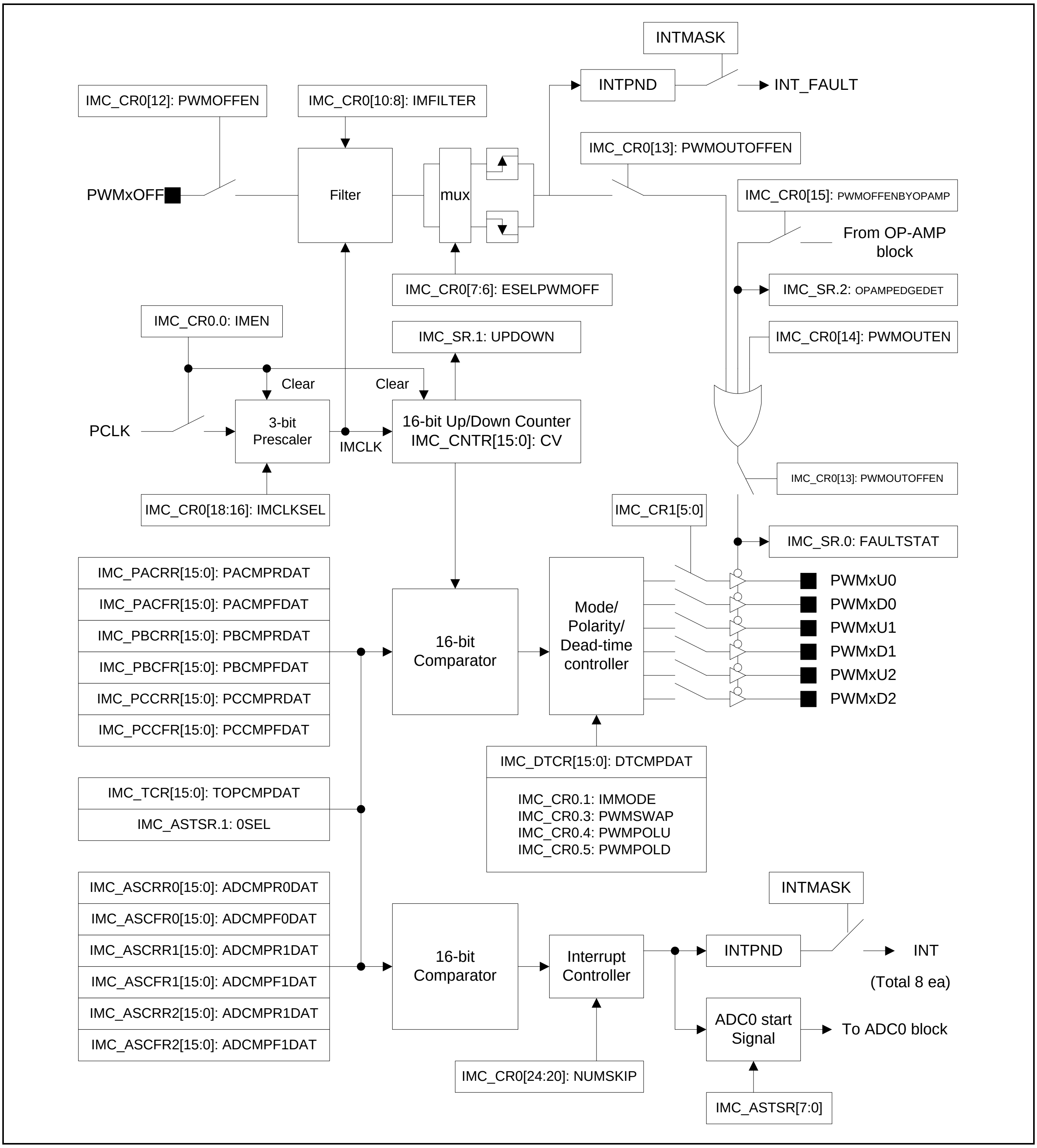


Figure 14-1 IMC Block Diagram

14.2.2 Operation

14.2.2.1 Tri-Angular Wave

Figure 14-2 describes IMC block's Tri-angular wave mode with the following example:

Example) High-Active Switch-ON Inverter Motor Control

- PWMPOLU is set to the value of "LOW START"
- PWMPOLD is set to the value of "HIGH START"
  - – IMC_TCR = 7, IMC_DTCR = 2, IMC_PACRR = 3, IMC_PACFR = 4, IMC_ASCRR0 = 6

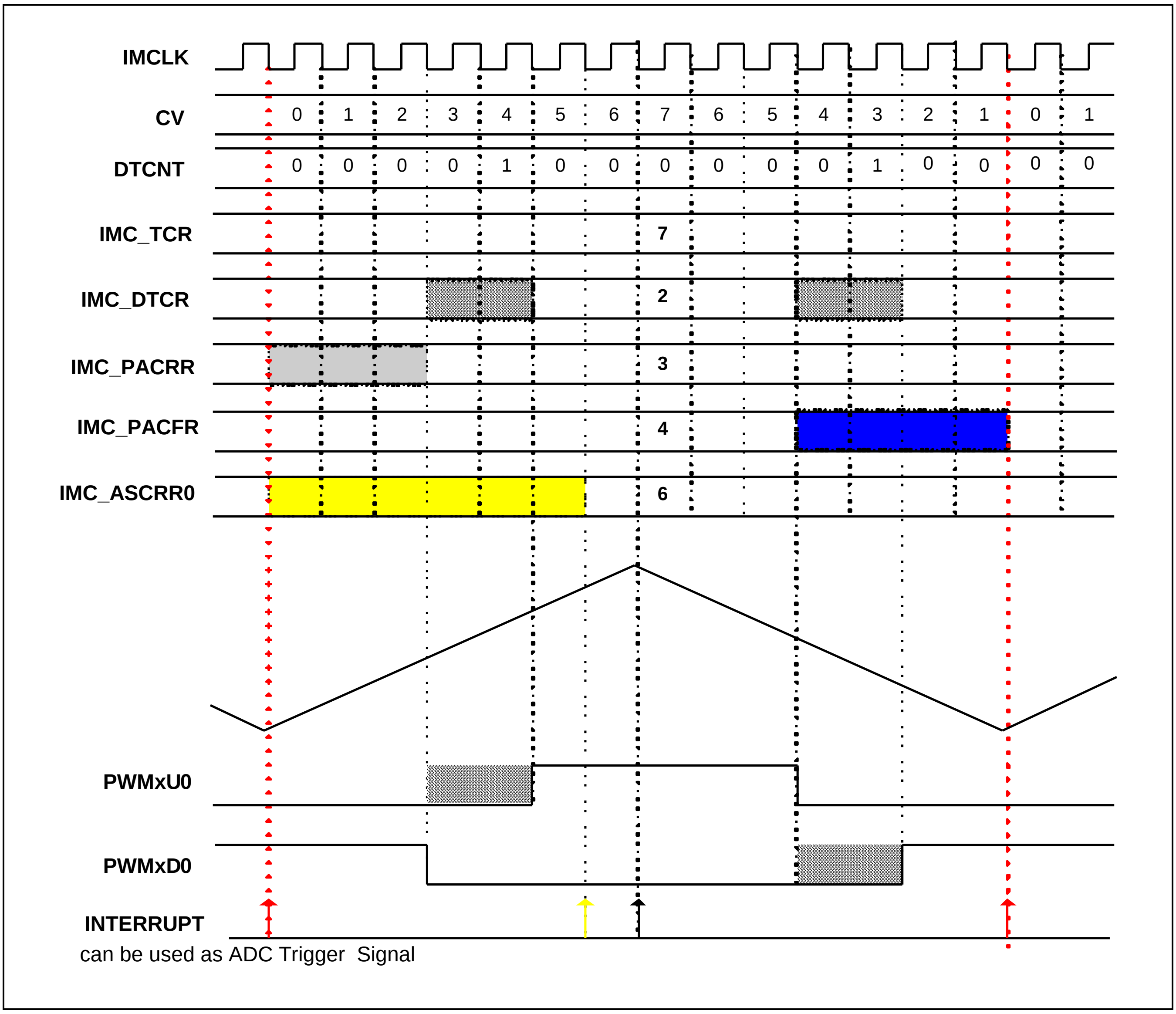


Figure 14-2 Tri-angular Wave Signal Generation

14.2.2.2 Saw-Tooth Wave

Figure 14-3 describes IMC block's Saw-tooth wave mode with the following example:

Example) High-Active Switch-ON Inverter Motor Control

- PWMPOLU is set to the value of "LOW START"
- PWMPOLD is set to the value of "HIGH START"
  - IMC_TCR = 7, IMC_DTCR = 2, IMC_PACRR = 3, IMC_ASCRR0 = 6

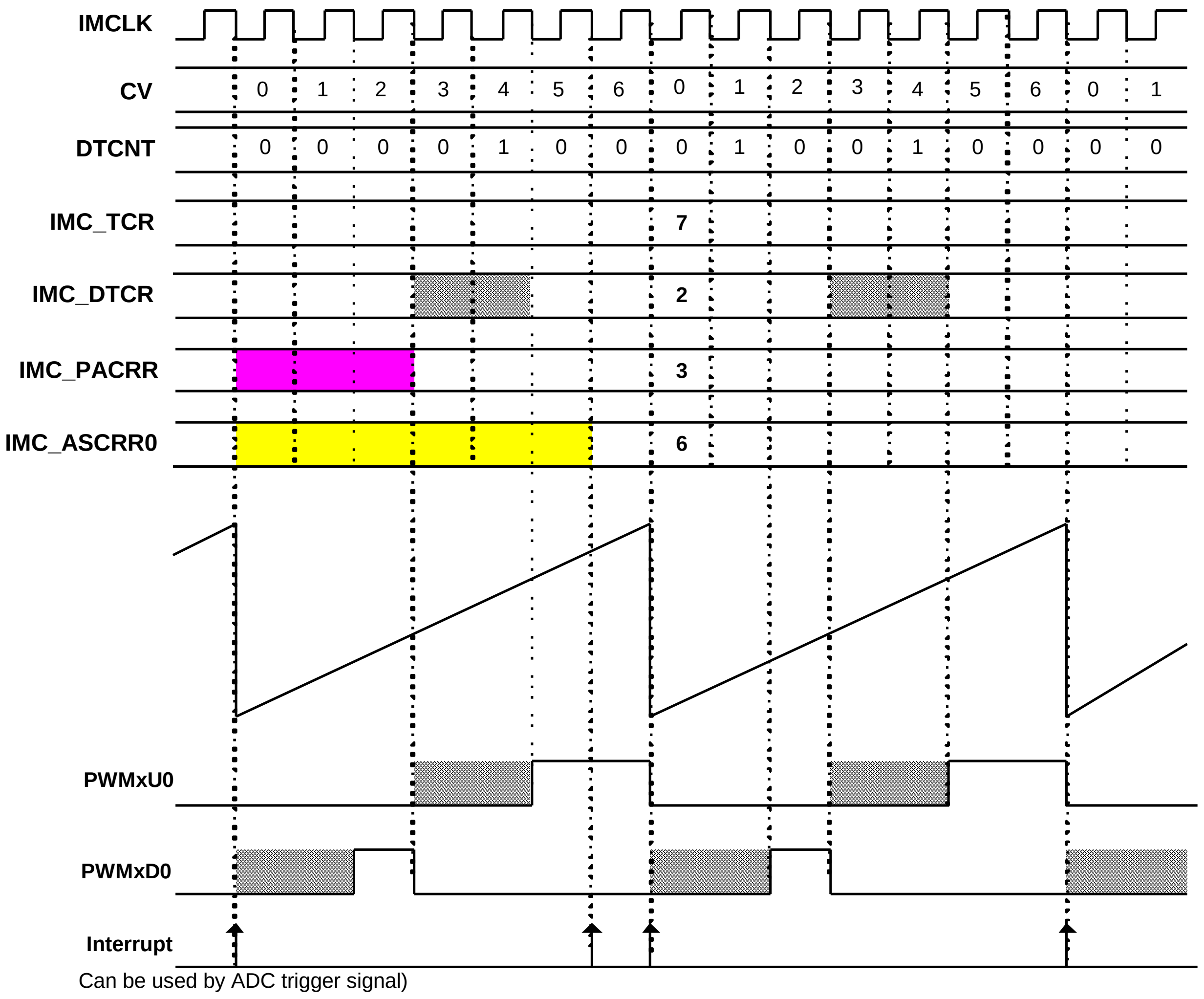


Figure 14-3 Saw-Tooth Wave Signal Generation

14.2.3 Phase Signal Generation

14.2.3.1 Tri-Angular Wave (IMMODE = 0)

- PWMSWAP = 0, PWMPOLU = 0 (Low start), PWMPOLD = 1 (High start)

These phase signals are used when switches of UP side and DOWN side are high active in inverter motor application. That means one pair switches of UP side and DOWN side don't have condition with high active at the same time. So dead time is inserted the following that.

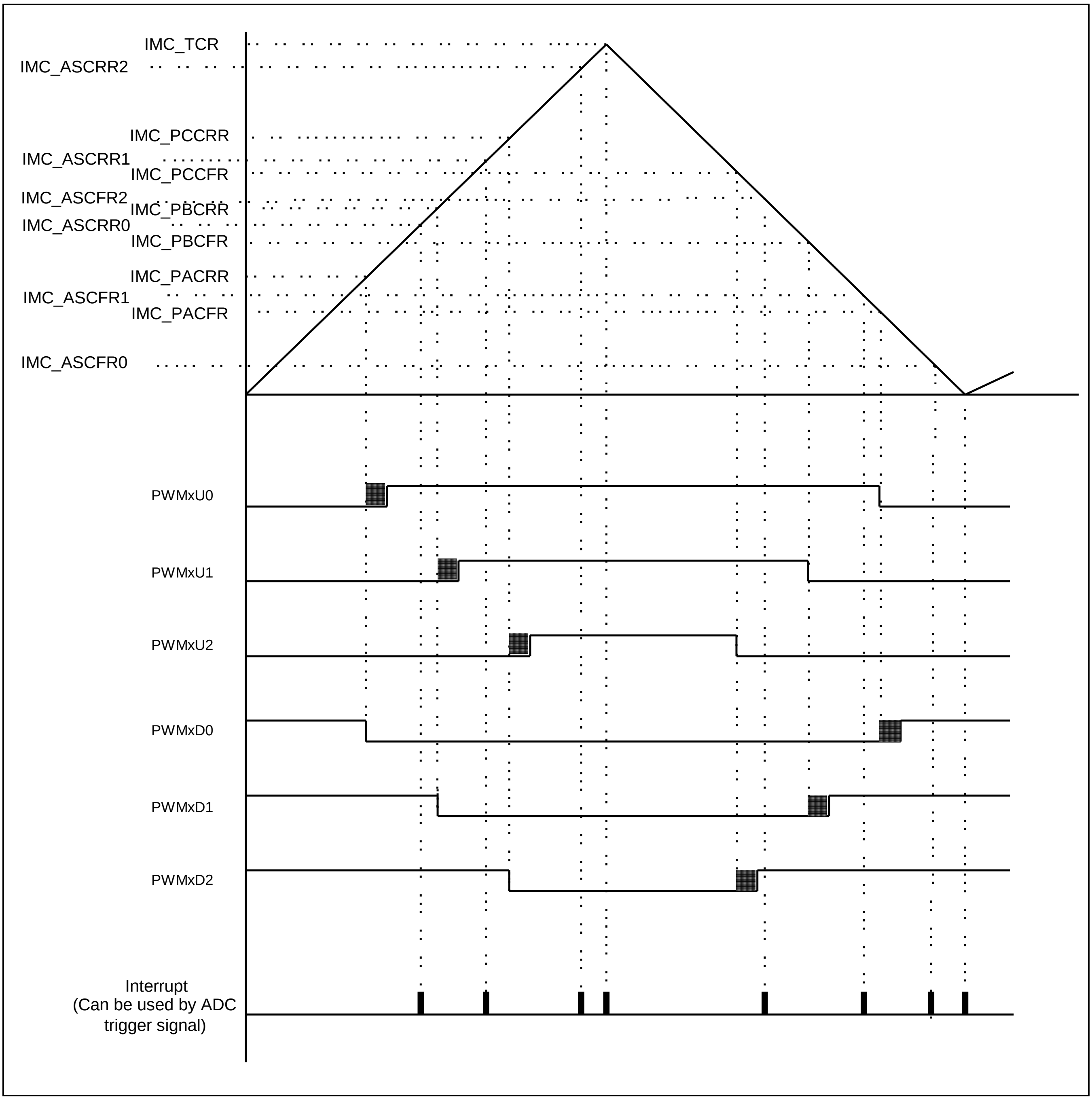


Figure 14-4 Tri-Angular Wave (No SWAP, A Low Start PWMxUy and High Start PWMxDy)

For 100% duty of upside, the rising/falling compare register must be set to '0'. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

The signal of PWM is described in the below figure. (Assumption: Duration of dead time is 2% duty.)

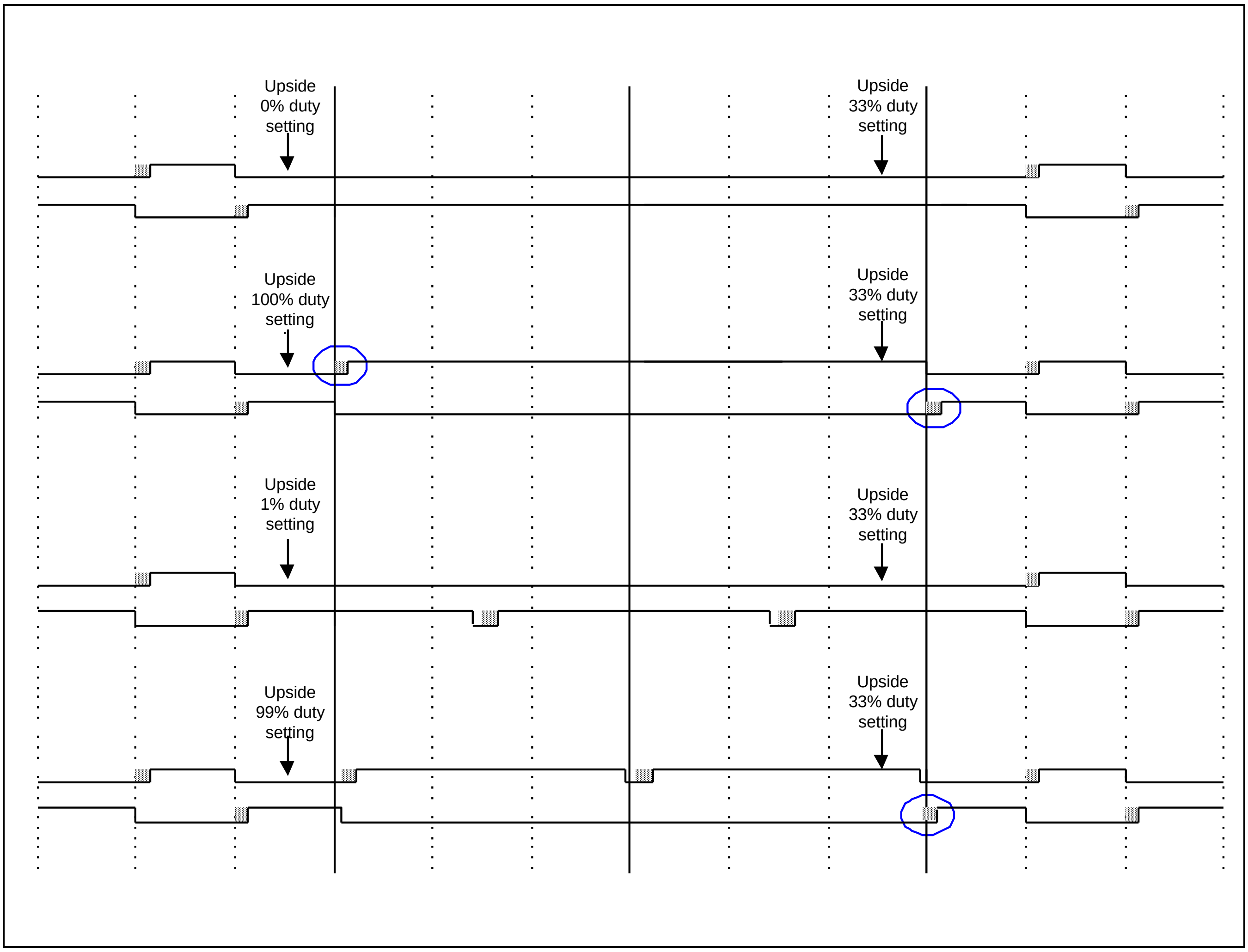


Figure 14-5 Tri-Angular Wave Duty (No SWAP, A Low Start PWMxUy and High Start PWMxDy)

14.2.3.2 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 1, PWMPOLU = 0 (Low start), PWMPOLD = 1 (High start)

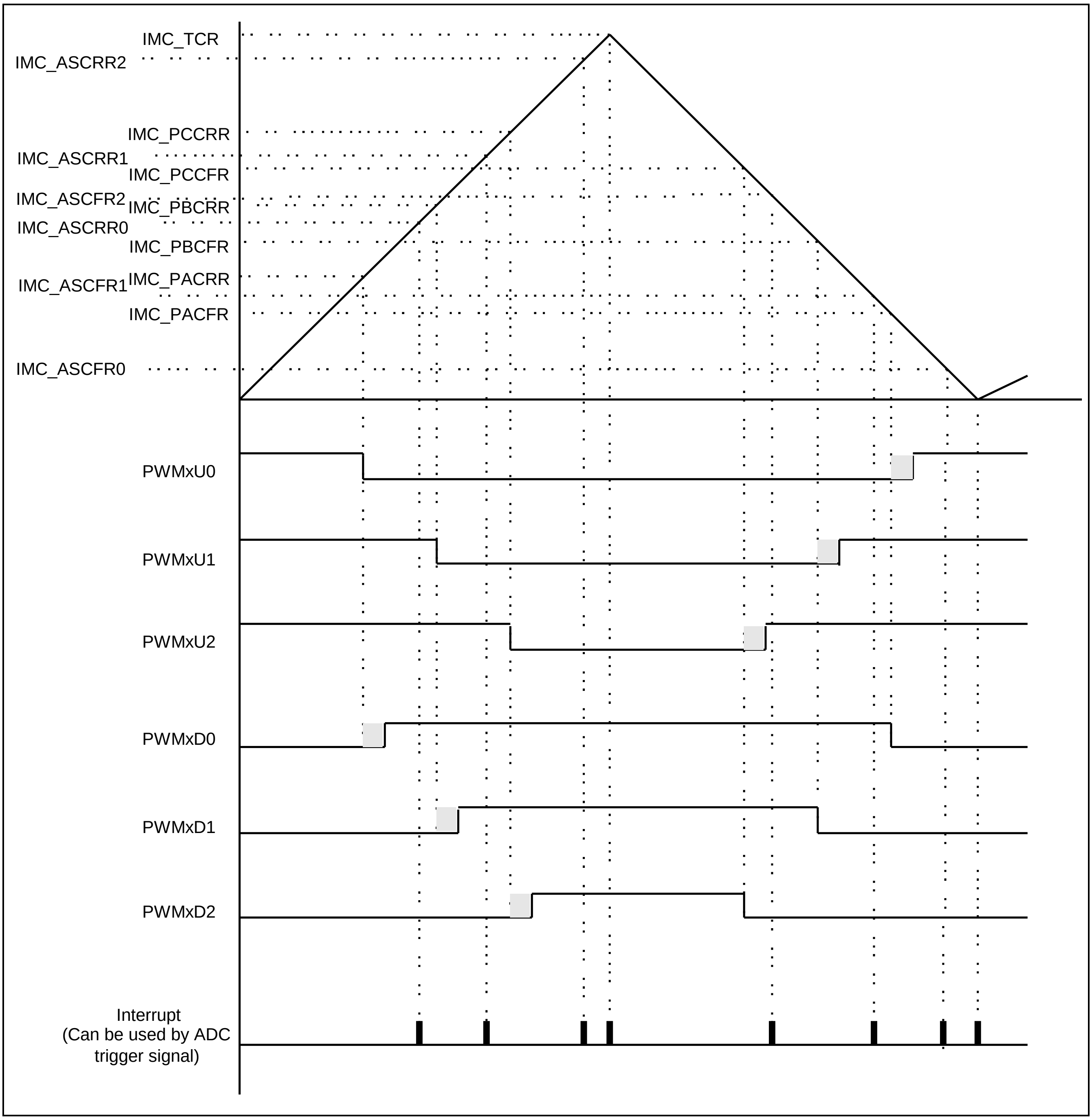


Figure 14-6 Tri-Angular Wave (SWAP, A Low Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switches of up side and down side are high active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

The signal of PWM is described in the below picture. (Assumption: Duration of dead time is 2% duty.)

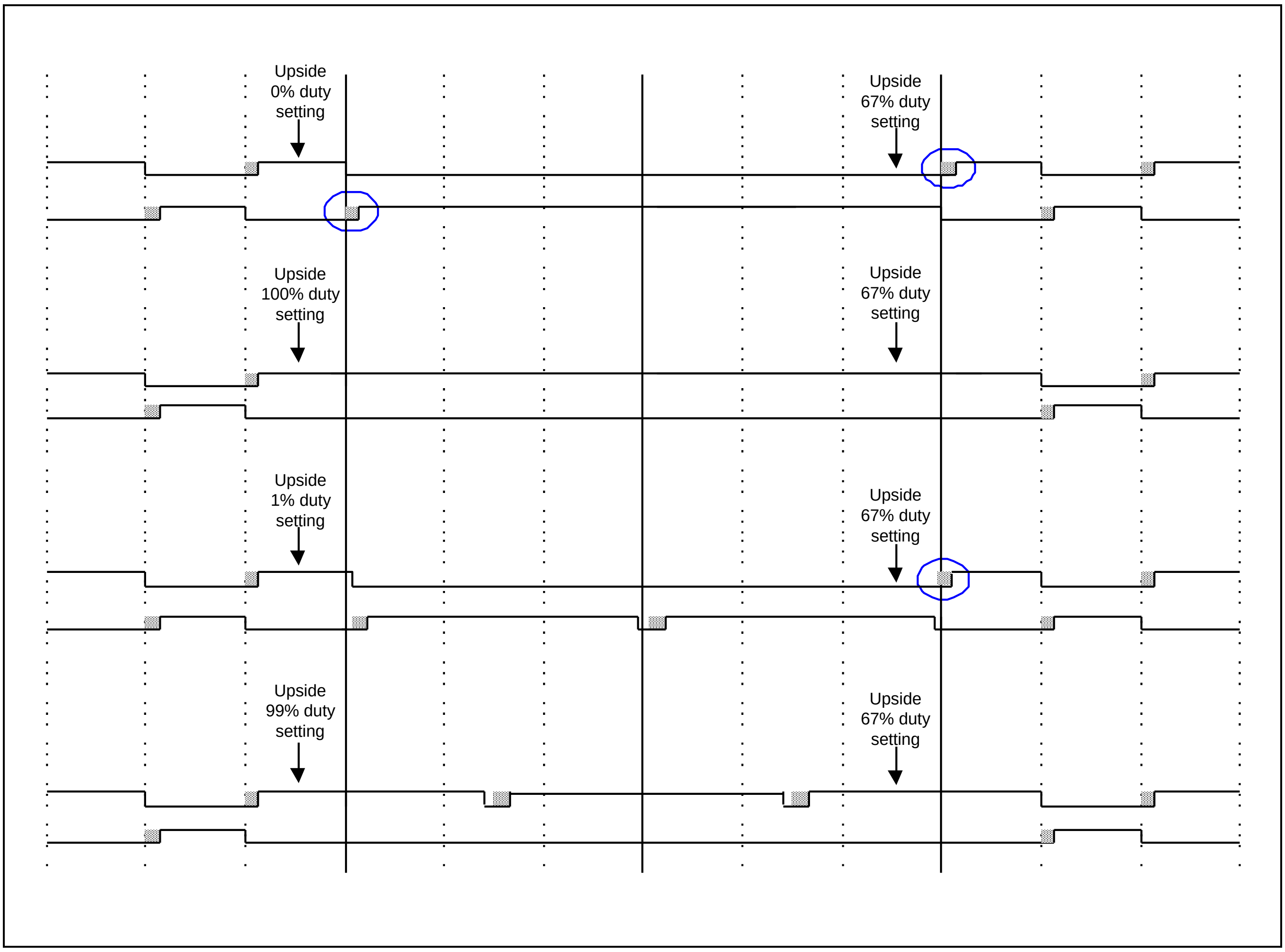


Figure 14-7 Tri-Angular Wave Duty (SWAP, A Low Start PWMxUy and High Start PWMxDy)

14.2.3.3 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 0, PWMPOLU = 0 (Low start), PWMPOLD = 0 (Low start)

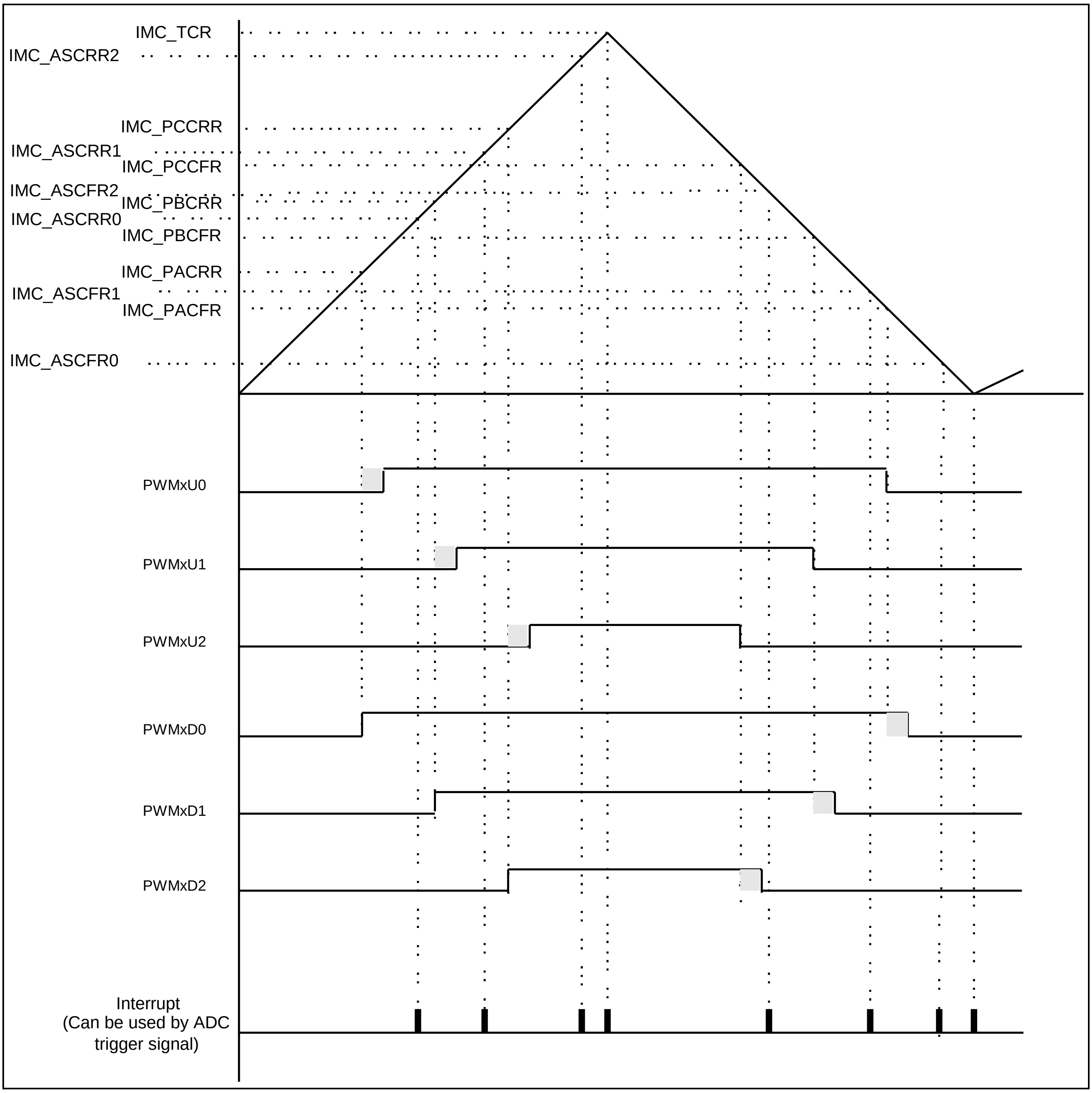


Figure 14-8 Tri-Angular Wave (No SWAP, A Low Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switch of up side is high active and switch of down side is low active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.4 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 1, PWMPOLU = 0 (Low start), PWMPOLD = 0 (Low start)

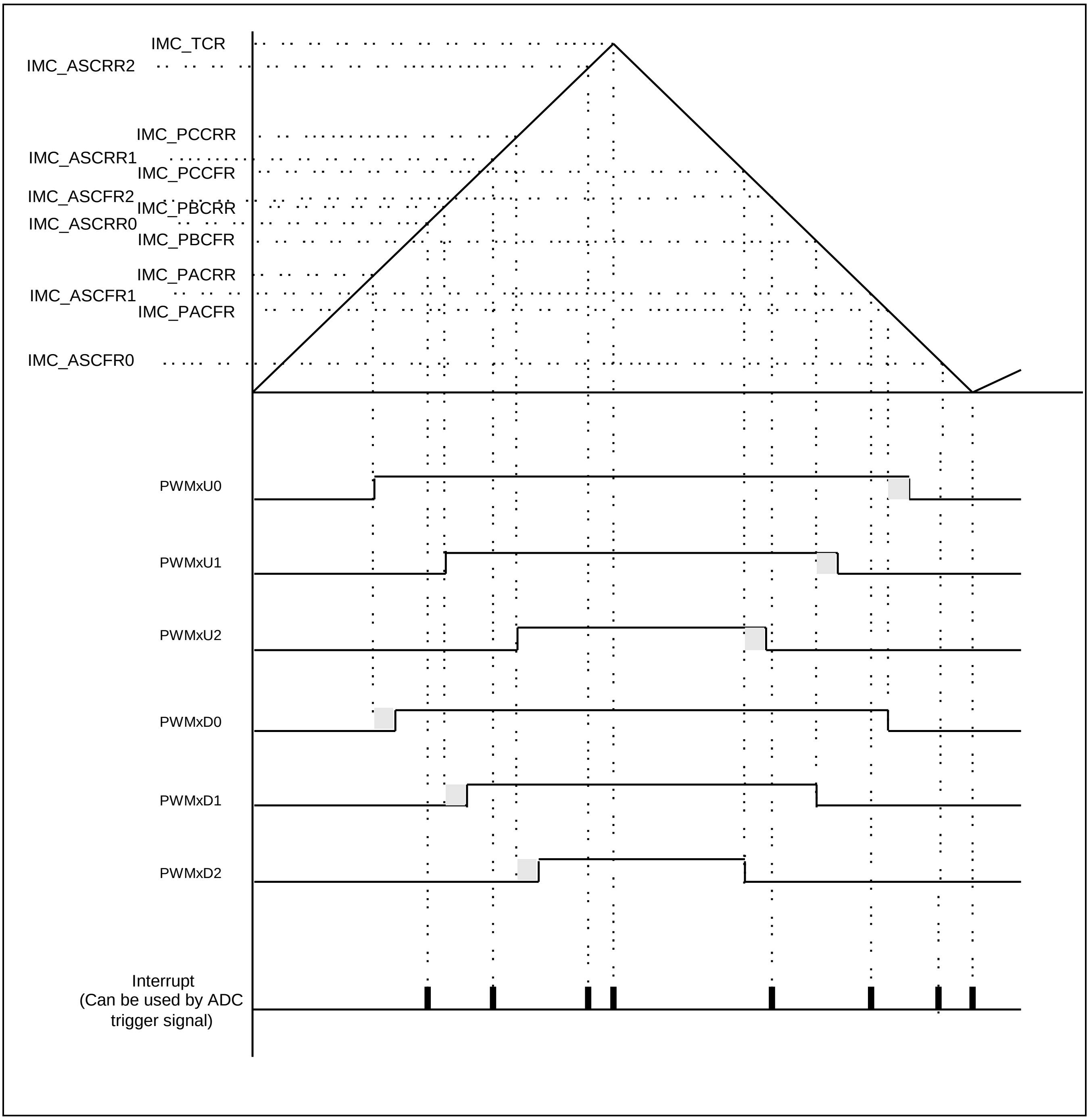


Figure 14-9 Tri-Angular Wave (SWAP, A Low Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switch of up side is low active and switch of down side is high active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.5 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 0, PWMPOLU = 1 (High start), PWMPOLD = 0 (Low start)

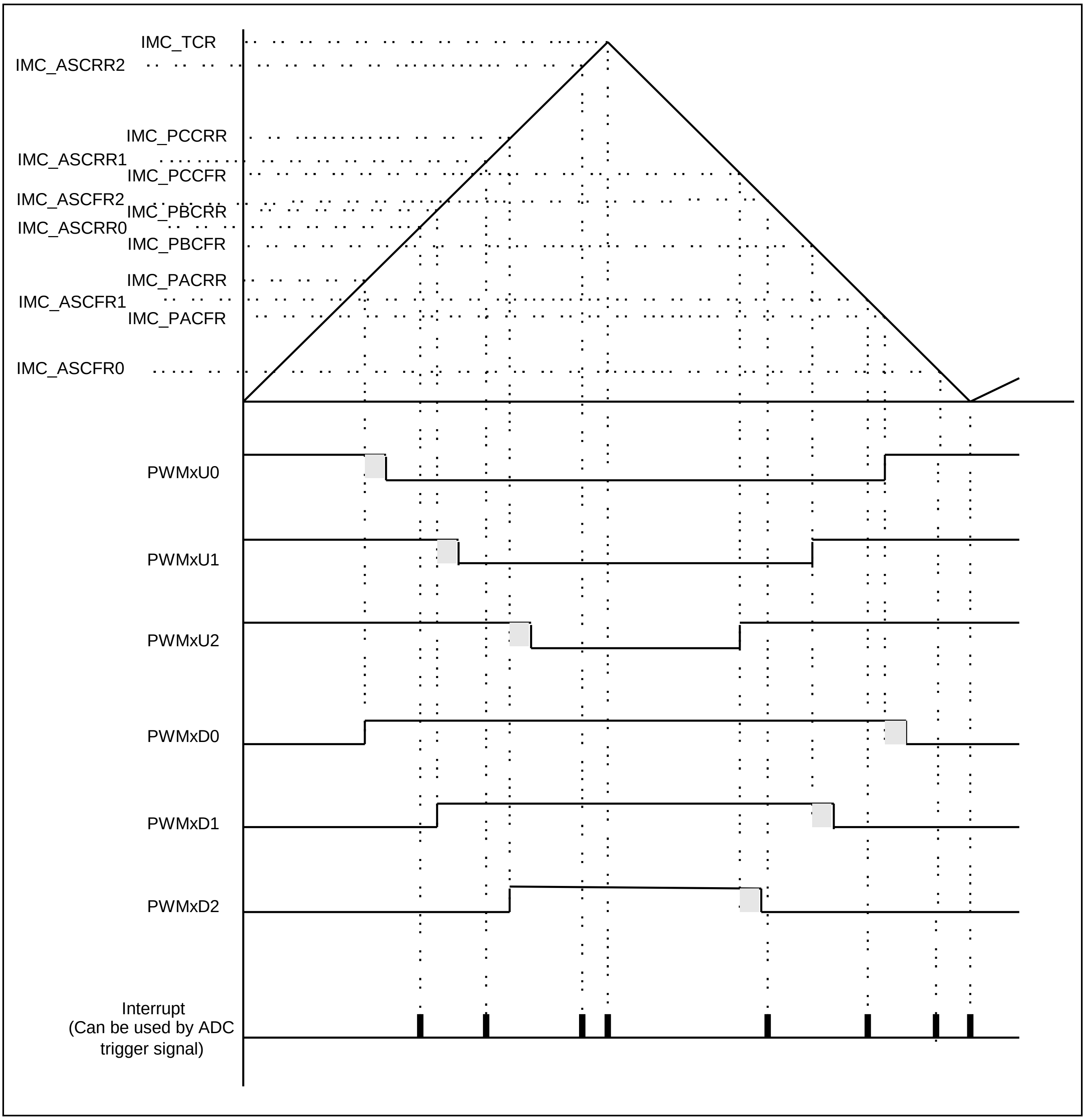


Figure 14-10 Tri-Angular Wave (No SWAP, A High Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switches of up side and down side are low active.
2. For 100% duty of upside, the rising/falling compare register must be set to '0'. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

The signal of PWM is described in the below picture. (Assumption: Duration of dead-time is 2% duty.)

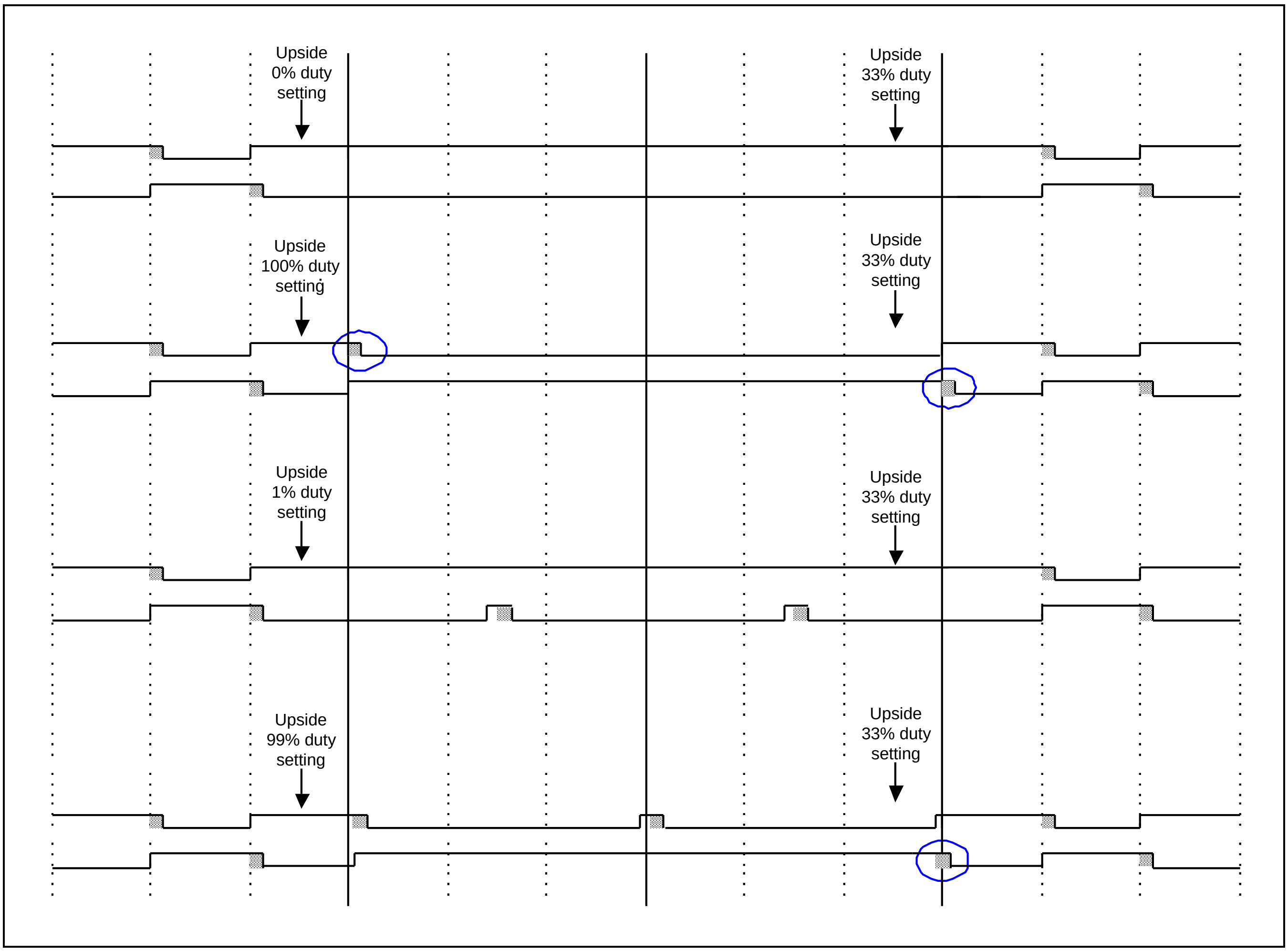


Figure 14-11 Tri-Angular Wave Duty (No SWAP, A High Start PWMxUy and Low Start PWMxDy)

14.2.3.6 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 1, PWMPOLU = 1 (High start), PWMPOLD = 0 (Low start)

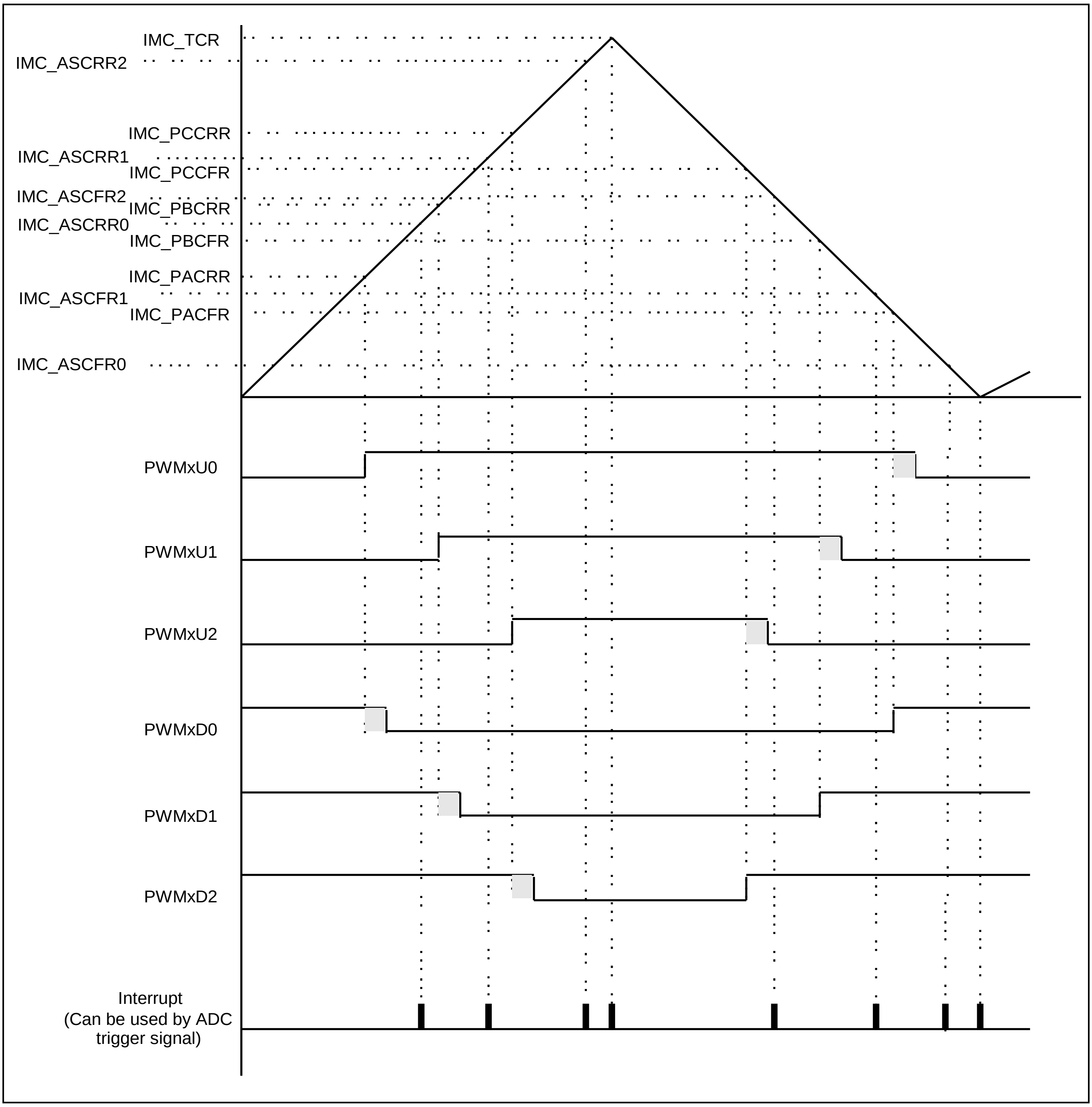


Figure 14-12 Tri-Angular Wave (SWAP, A High Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switches of up side and down side are low active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.7 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 0, PWMPOLU = 1 (High start), PWMPOLD = 1 (High start)

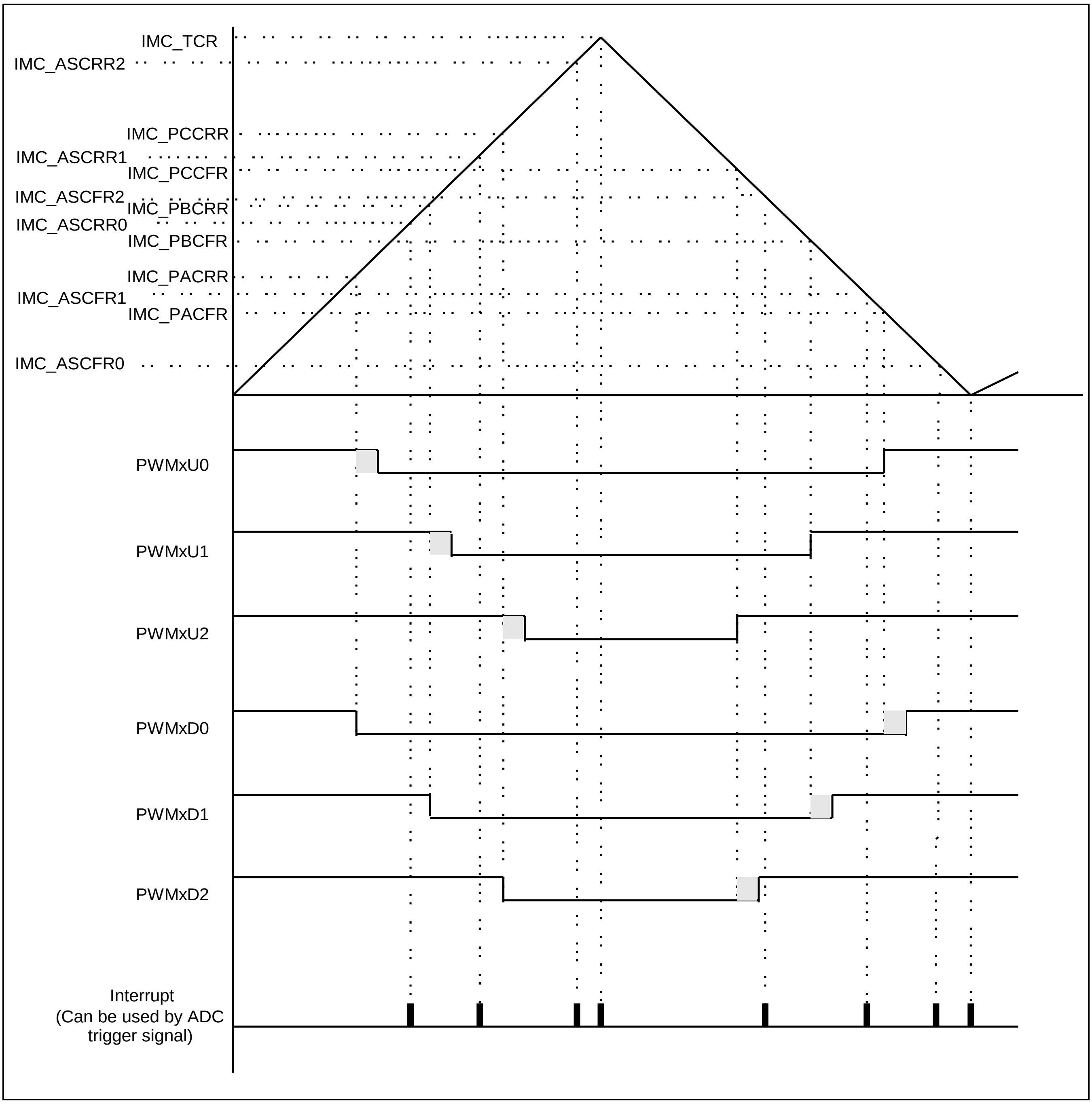


Figure 14-13 Tri-Angular Wave (No SWAP, A High Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switch of up side is low active and switch of down side is high active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.8 Tri-angular Wave (IMMODE = 0)

- PWMSWAP = 1, PWMPOLU = 1 (High start), PWMPOLD = 1 (High start)

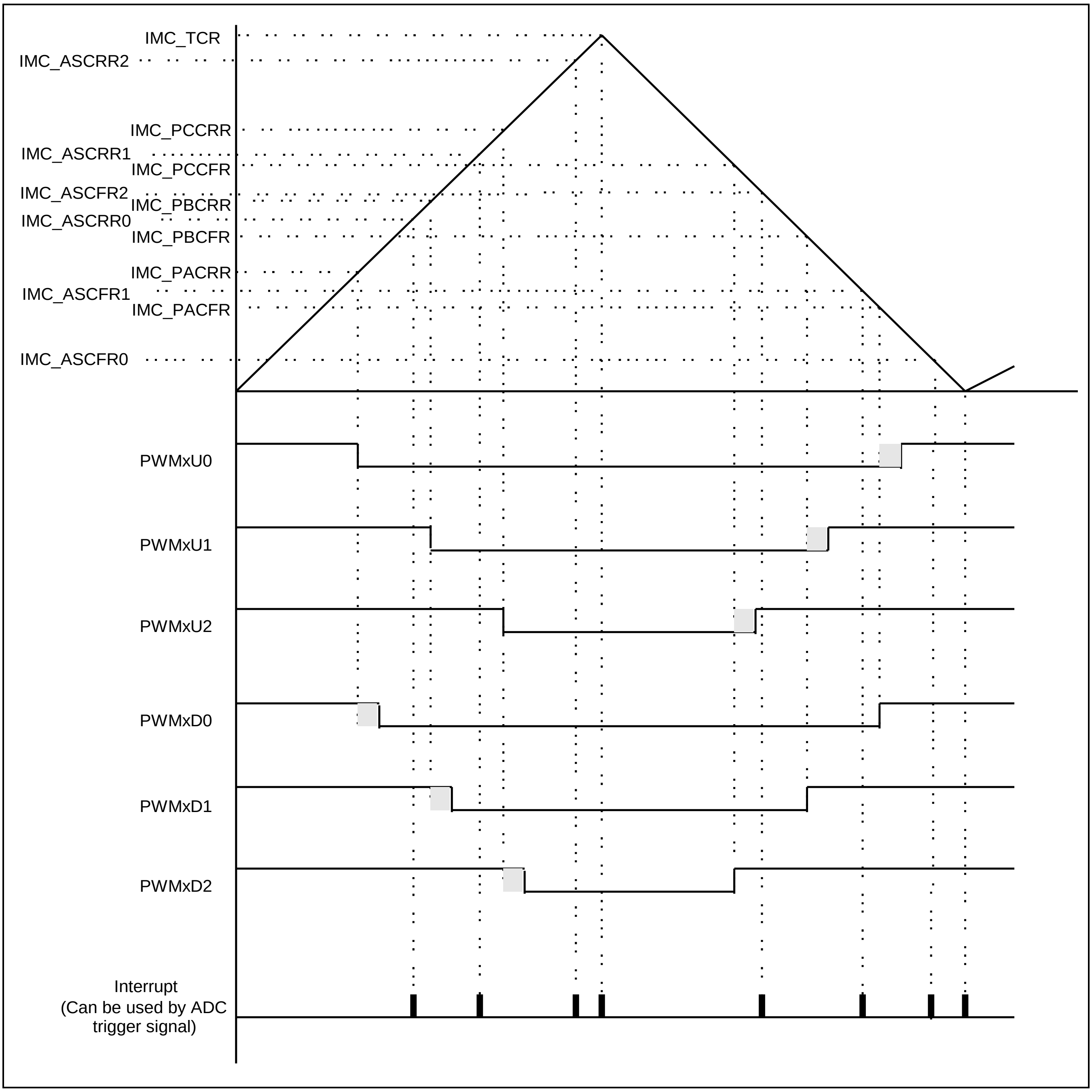


Figure 14-14 Tri-Angular Wave (SWAP, A High Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switch of up side is high active and switch of down side is low active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.9 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 0, PWMPOLU = 0 (Low start), PWMPOLD = 1 (High start)

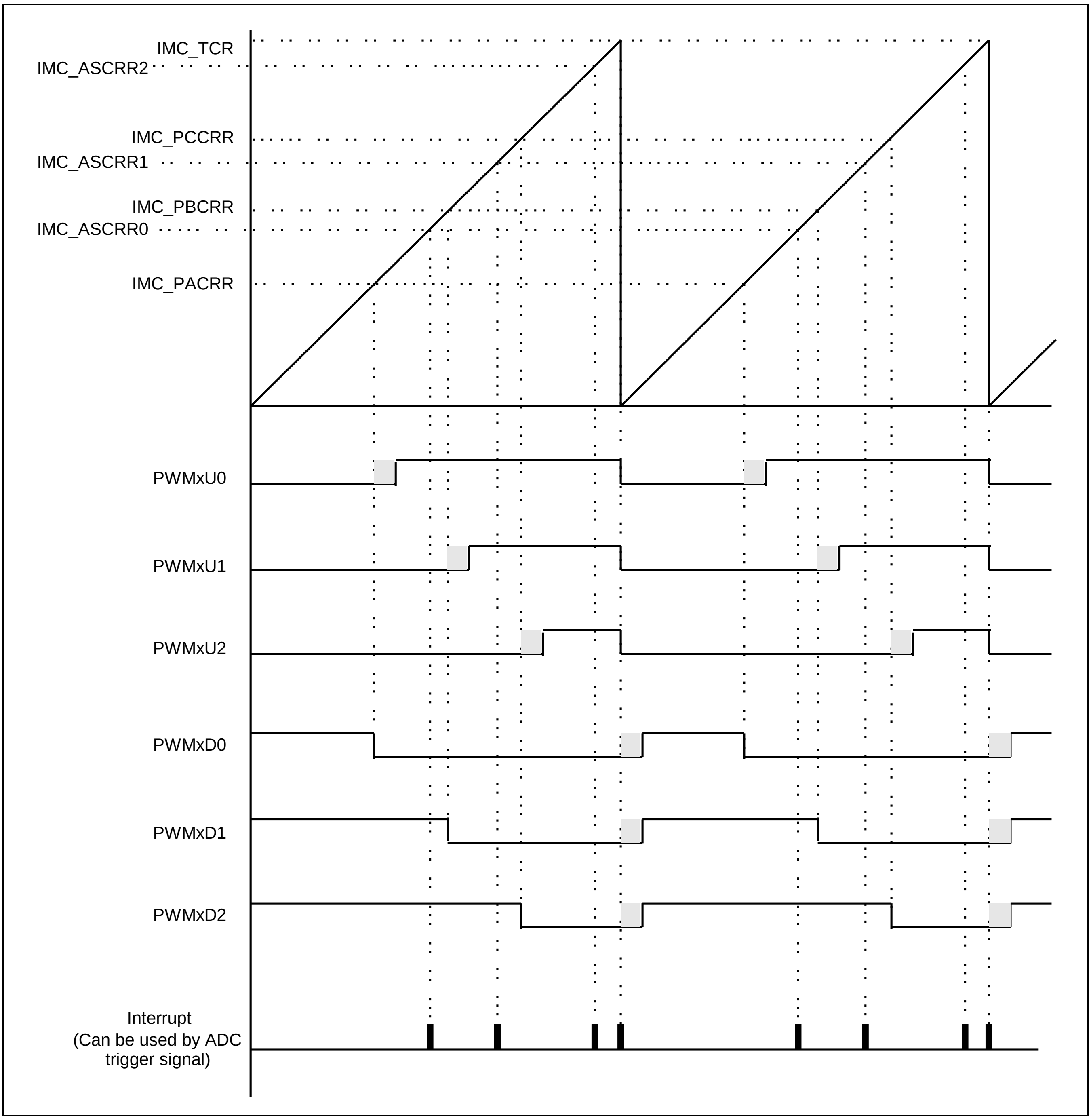


Figure 14-15 Saw-Tooth Wave (No SWAP, A Low Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switches of up side and down side are high active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

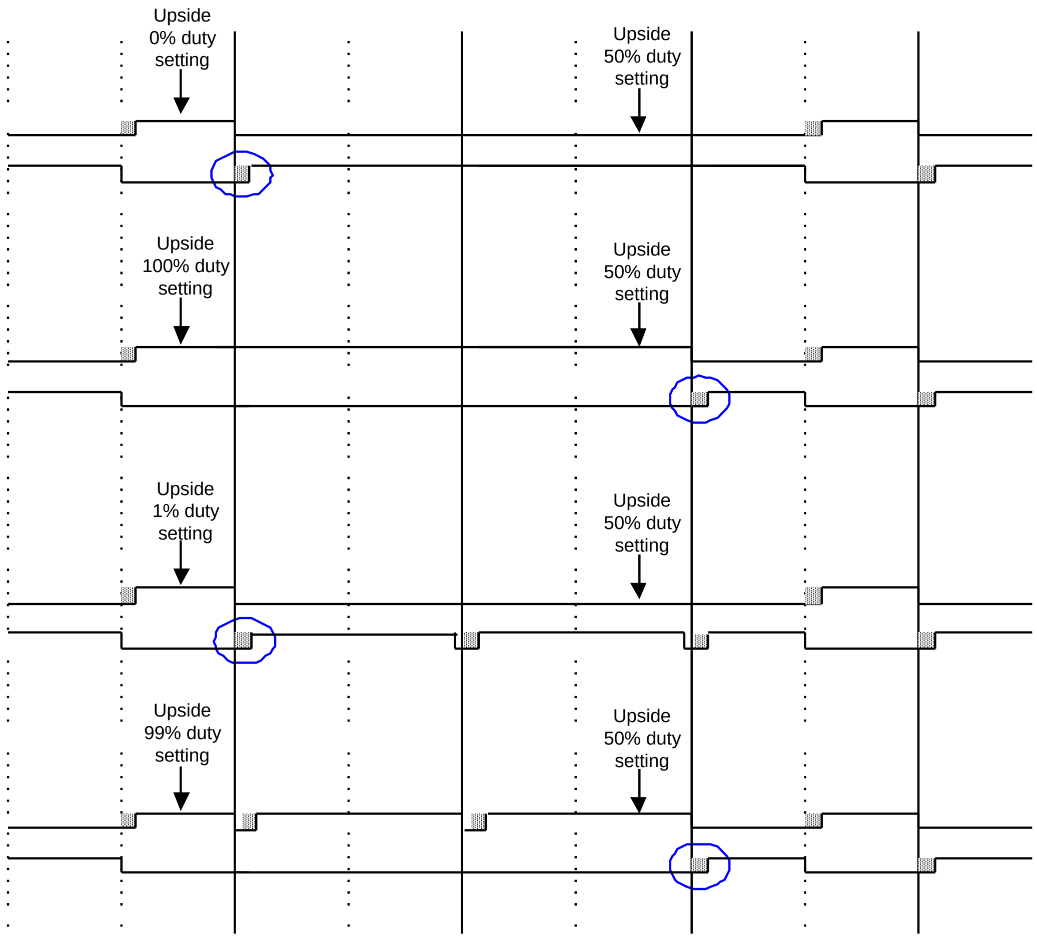


Figure 14-16 Saw-Tooth Wave Duty (No SWAP, A Low Start PWMxUy and High Start PWMxDy)

14.2.3.10 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 1, PWMPOLU = 0 (Low start), PWMPOLD = 1 (High start)

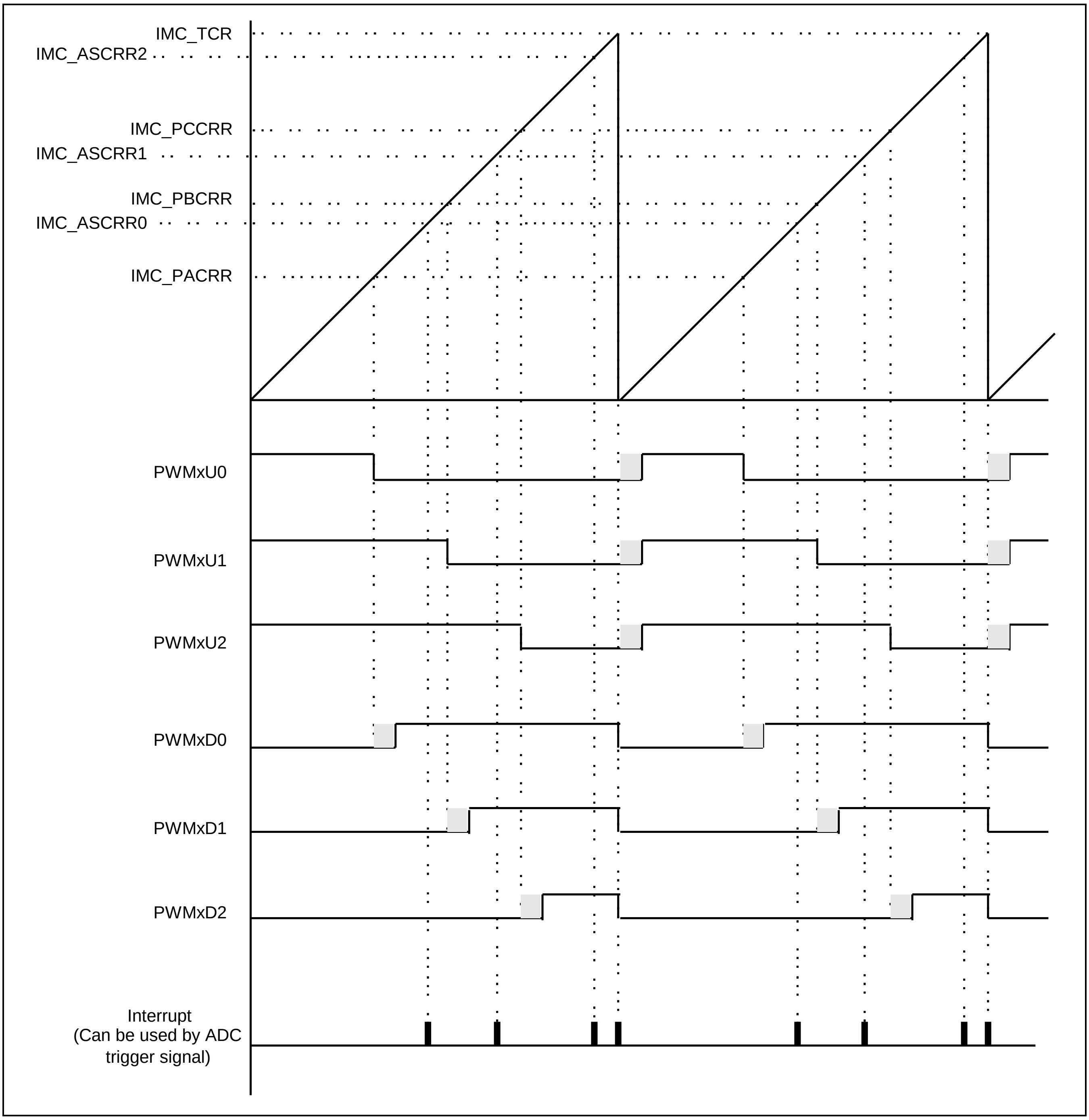


Figure 14-17 Saw-Tooth Wave (SWAP, A Low Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switches of up side and down side are high active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.11 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 0, PWMPOLU = 0 (Low start), PWMPOLD = 0 (Low start)

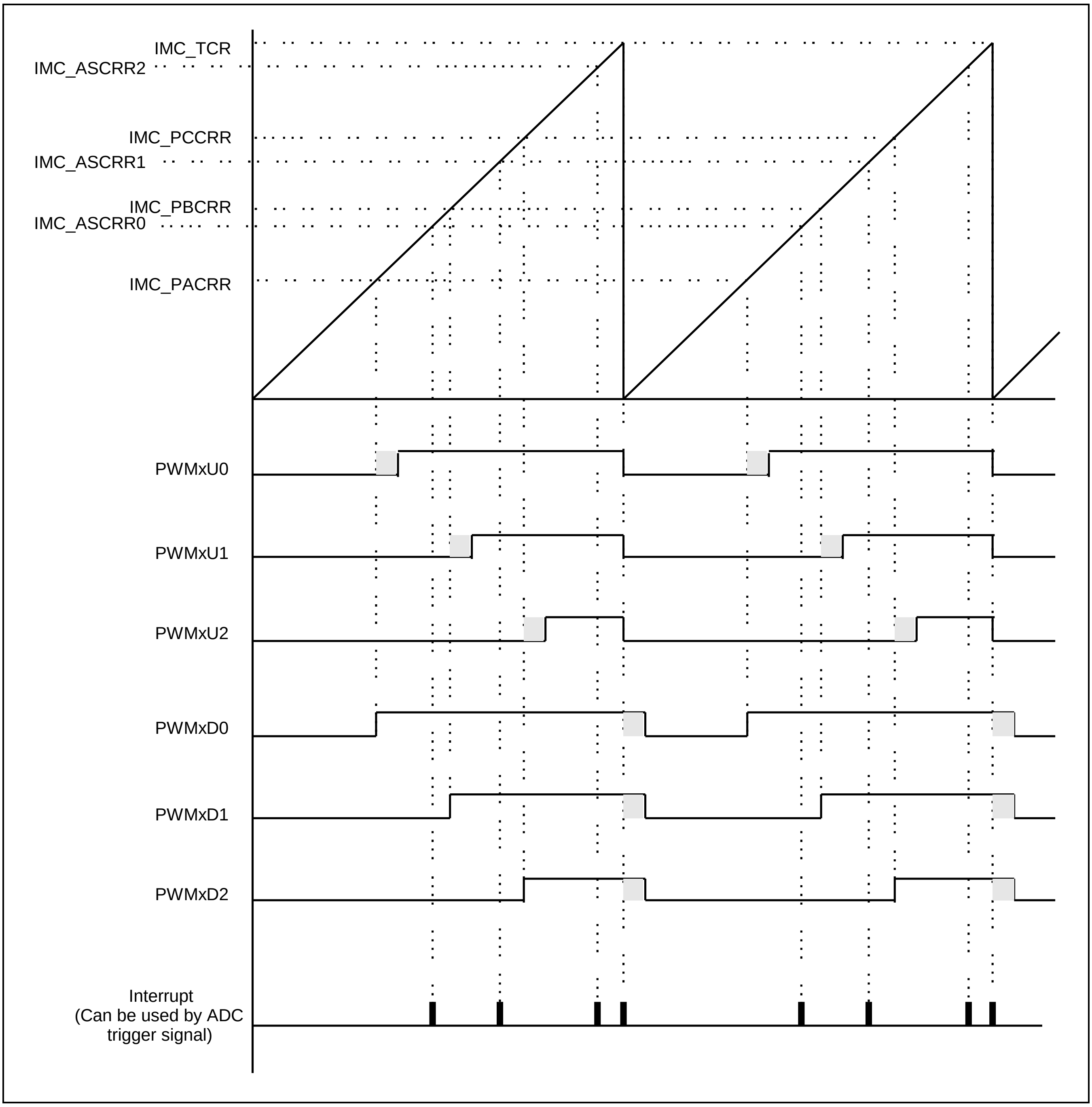


Figure 14-18 Saw-Tooth Wave (No SWAP, A Low Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switch of up side is high active and switch of down side is low active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.12 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 1, PWMPOLU = 0 (Low start), PWMPOLD = 0 (Low start)

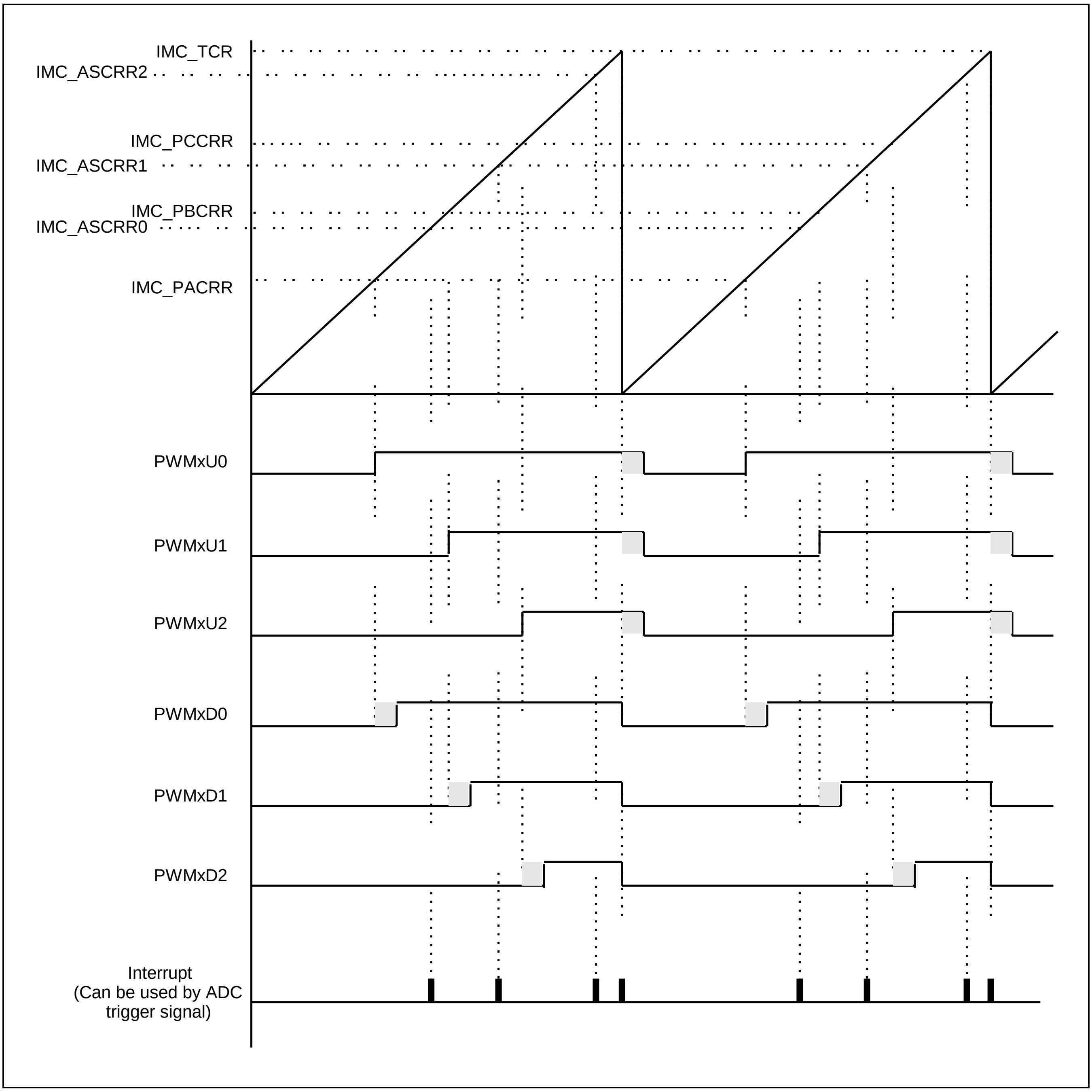


Figure 14-19 Saw-Tooth Wave (SWAP, A Low Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switch of up side is low active and switch of down side is high active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.13 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 0, PWMPOLU = 1 (High start), PWMPOLD = 0 (Low start)

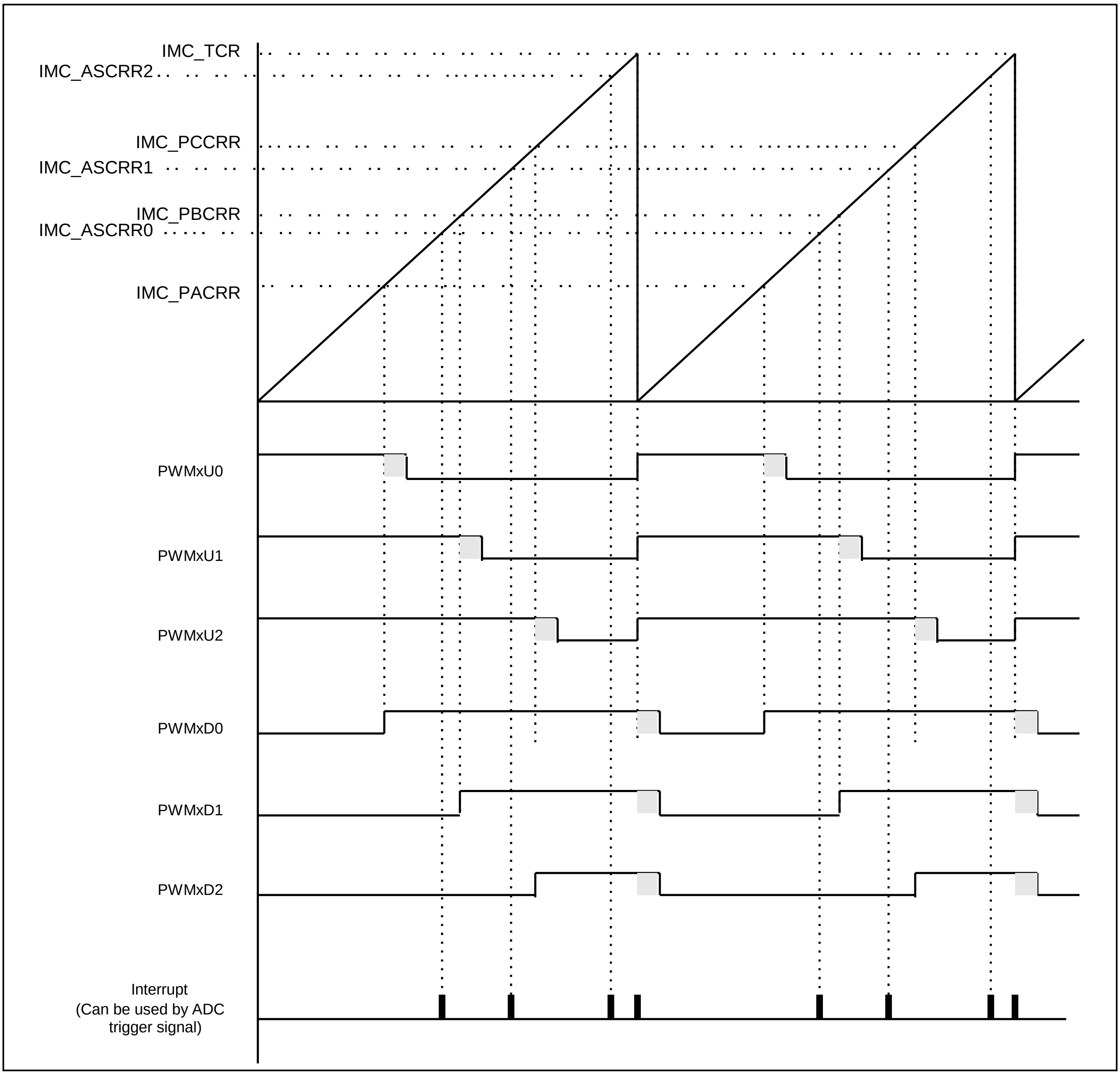


Figure 14-20 Saw-Tooth Save (No SWAP, A High Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switches of up side and down side are low active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.14 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 1, PWMPOLU = 1 (High start), PWMPOLD = 0 (Low start)

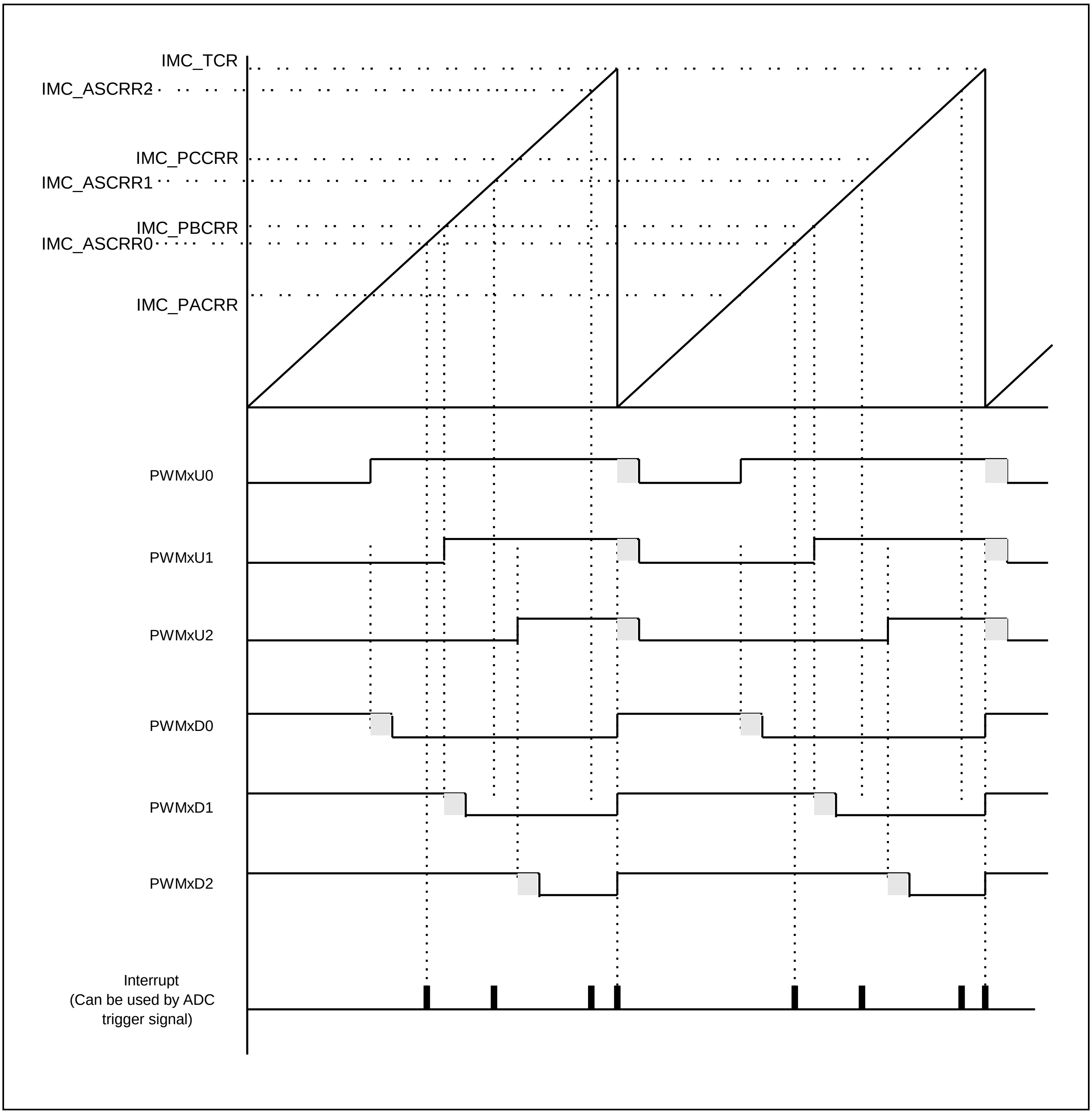


Figure 14-21 Saw-Tooth Wave (SWAP, A High Start PWMxUy and Low Start PWMxDy)

NOTE:

1. Switches of up side and down side are low active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.15 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 0, PWMPOLU = 1 (High start), PWMPOLD = 1 (High start)

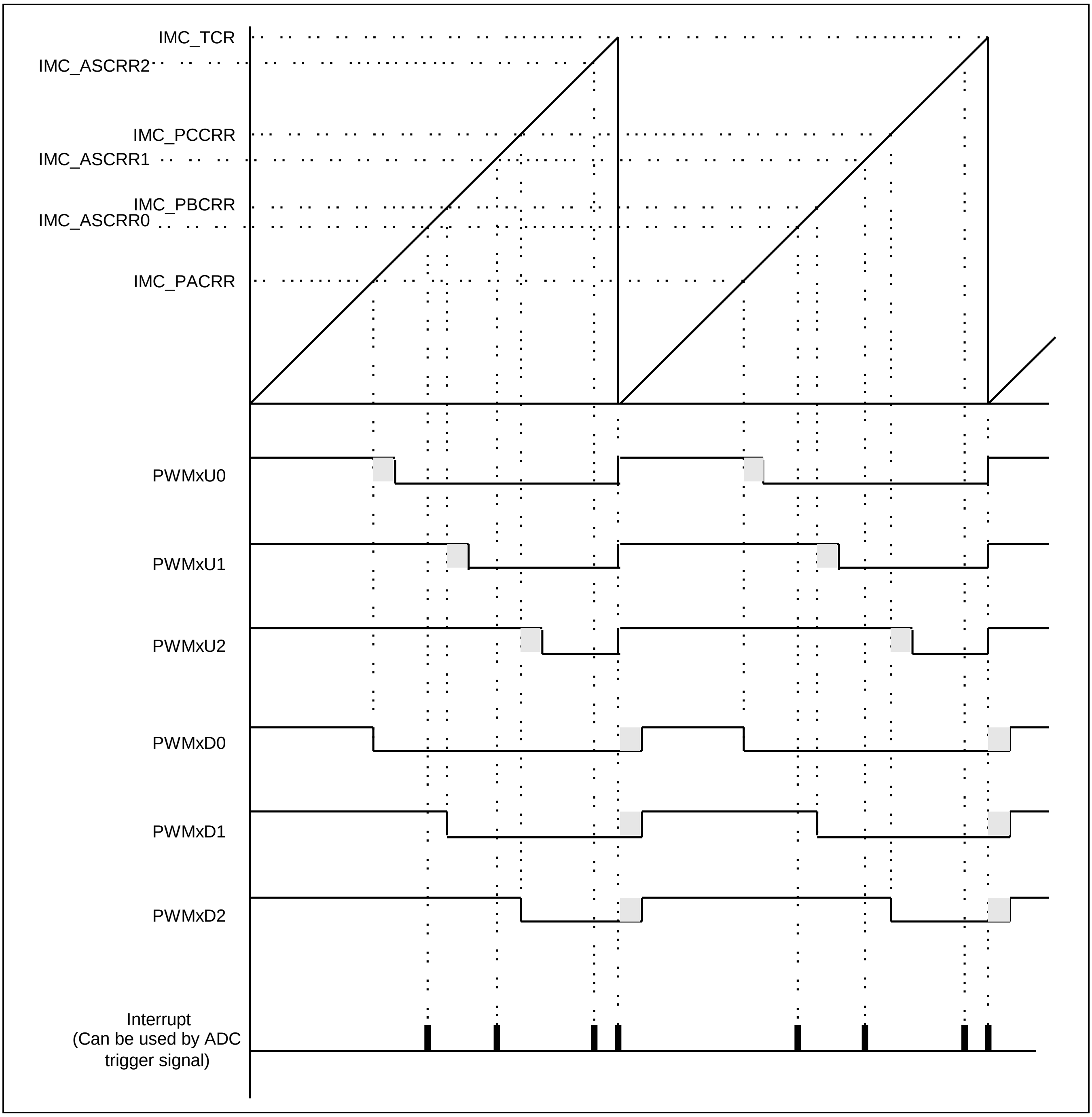


Figure 14-22 Saw-Tooth Wave (No SWAP, A High Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switch of up side is low active and switch of down side is high active.
2. For 100% duty of upside, the rising/falling compare register must be set to ‘0’. For 0% duty of upside, the rising compare register must be equal to TOPCMP value.

14.2.3.16 Saw-Tooth Wave (IMMODE = 1)

- PWMSWAP = 1, PWMPOLU = 1 (High start), PWMPOLD = 1 (High start)

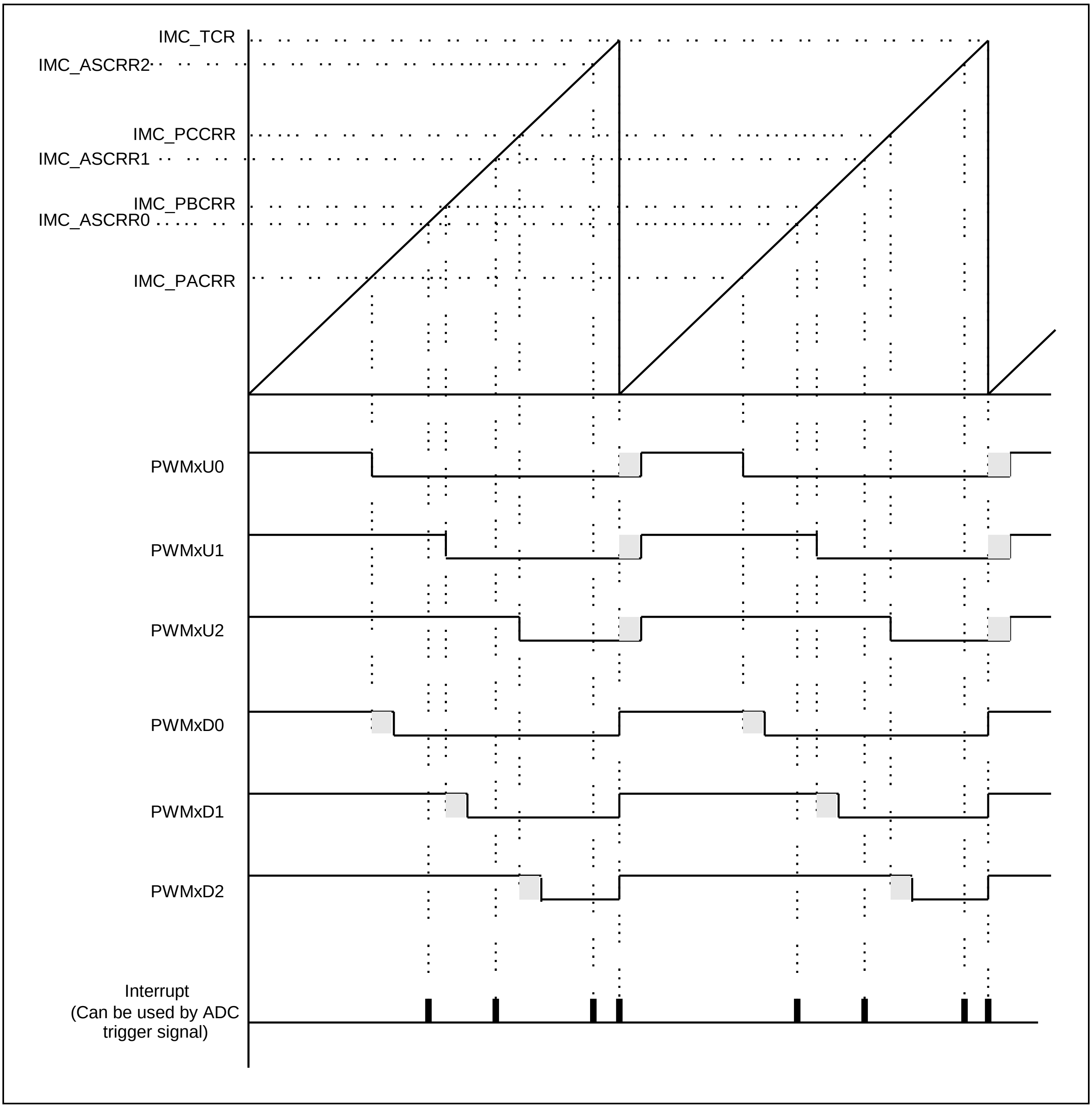


Figure 14-23 Saw-Tooth Wave (SWAP, A High Start PWMxUy and High Start PWMxDy)

NOTE:

1. Switch of up side is high active and switch of down side is low active.
2. For 0% duty of upside, the rising/falling compare register must be set to ‘0’. For 100% duty of upside, the rising compare register must be equal to TOPCMP value.

14.3 Register Description

14.3.1 Register Map Summary

- Base Address: 0x400B_0000
- Base Address: 0x400B_1000

Register	Offset	Description	Reset Value
IMC_IDR	0x0000	IMC ID Register	0x0001_0012
IMC_CEDR	0x0004	IMC Clock Enable/Disable Register	0x0000_0000
IMC_SRR	0x0008	IMC Software Reset Register	0x0000_0000
IMC_CR0	0x000C	IMC Control Register 0	0x0000_0000
IMC_CR1	0x0010	IMC Control Register 1	0x0000_0000
IMC_CNTR	0x0014	IMC Counter Register	0x0000_0000
IMC_SR	0x0018	IMC Status Register	0x0000_0000
IMC_IMSCR	0x001C	IMC Interrupt Mask Set/Clear Register	0x0000_0000
IMC_RISR	0x0020	IMC Raw Interrupt Status Register	0x0000_0000
IMC_MISR	0x0024	IMC Masked Interrupt Status Register	0x0000_0000
IMC_ICR	0x0028	IMC Interrupt Clear Register	0x0000_0000
IMC_TCR	0x002C	16-bit Top Compare Register	0x0000_0000
IMC_DTCR	0x0030	16-bit Dead-Time Control Register	0x0000_0000
IMC_PACRR	0x0034	IMC 16-bit Phase A Compare Rising Register	0x0000_0000
IMC_PBCRR	0x0038	IMC 16-bit Phase B Compare Rising Register	0x0000_0000
IMC_PCCRR	0x003C	IMC 16-bit Phase C Compare Rising Register	0x0000_0000
IMC_PACFR	0x0040	IMC 16-bit Phase A Compare Falling Register	0x0000_0000
IMC_PBCFR	0x0044	IMC 16-bit Phase B Compare Falling Register	0x0000_0000
IMC_PCCFR	0x0048	IMC 16-bit Phase C Compare Falling Register	0x0000_0000
IMC_ASTSR	0x004C	IMC ADC Start Trigger Selection Register	0x0000_0000
IMC_ASCRR0	0x0050	16-bit ADC Start Compare Rising Register0	0x0000_0000
IMC_ASCRR1	0x0054	16-bit ADC Start Compare Rising Register1	0x0000_0000
IMC_ASCRR2	0x0058	16-bit ADC Start Compare Rising Register2	0x0000_0000
IMC_ASCFR0	0x005C	16-bit ADC Start Compare Falling Register 0	0x0000_0000
IMC_ASCFR1	0x0060	16-bit ADC Start Compare Falling Register 1	0x0000_0000
IMC_ASCFR2	0x0064	16-bit ADC Start Compare Falling Register 2	0x0000_0000

14.3.1.1 IMC_IDR (IMC ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_0012

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register This field stores the ID code for the corresponding IP.	0x0001_0012

14.3.1.2 IMC_CEDR (IMC Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable Control Bit 0 = IMC Clock Disable 1 = IMC Clock Enable	0
DBGEN	[31]	RW	Debug enable bit 0 = IMC is not halted during processor debug mode. 1 = IMC is halted during processor debug mode.	0

14.3.1.3 IMC_SRR (IMC Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset 0 = No effect 1 = Perform IMC Software Reset operation	0

14.3.1.4 IMC_CR0 (IMC Control Register 0)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD				SYNCSEL		RSVD	NUMSKIP					RSVD	IMCLKSEL			PWMOUTOFFENBYOPAMP	PWMOUTEN	PWMOUTOFFEN	PWMOFFEN	RSVD	IMFILTER			ESELPWMOFF		PWMPOLD	PWMPOLU	PWMSWAP	WMODE	IMMODE	IMEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	RW	RW	R	RW	RW	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
IMEN	[0]	RW	Inverter motor block enable bit This field can determine the inverter motor block enable/disable. 0 = Disable inverter motor block 1 = Enable inverter motor block **NOTE:** If this bit is set to ‘0’, IMCNT is cleared to 0 automatically.	0
IMMODE	[1]	RW	Inverter Motor Mode Selection Bit 0 = Tri-angular shape 1 = Saw-tooth shape This bit can be changed only when IMC_CR0.0 is 0. If this bit is set to ‘1’, comparison with ‘0’ is no effect (INT_ZEROx will not be occurred.)	0
WMODE	[2]	RW	Write mode of compare register This field can determine the write mode of all compare register. 0 = Immediate write 1 = Synchronous write **NOTE:** In the synchronous write, if IMCNT equals to 0 or TOPCMP, compare registers including dead-time compare register which are written are updated simultaneously. Synchronous write is related to NUMSKIP. For example, if NUMSKIP is 30, synchronous write happens only one time in every 30 times. ADC trigger signal is the same situation.	0
PWMSWAP	[3]	RW	Swapping of PWMxUx and PWMxDx This field can determine swapping of PWMxUx and PWMxDx.	0

Name	Bit	Type	Description	Reset Value
			0 = No swap 1 = Swap **NOTE:** This bit can be changed only when IMCON.0 is 0.	
PWMPOLU	[4]	RW	Polarity of PWM in the PWMxU0/1/2 This field can determine the polarity of PWM signal in the PWMxU0/1/2. 0 = Low start 1 = High start	0
PWMPOLD	[5]	RW	Polarity of PWM in the PWMxD0/1/2 This field can determine the polarity of PWM signal in the PWMxD0/1/2. 0 = Low start 1 = High start	0
ESELPWMOFF	[7:6]	RW	PWMxOFF Active Selection This field can determine the active selection for PWM0OFF. 00 = Falling edge 01 = Rising edge 10 = Low level 11 = High level **NOTE:** These bits must be changed only when IMCON.0 is 0.	00'b
IMFILTER	[10:8]	RW	Filter Clock Selection of PWMxOFF pin This field can determine the filter clock selection of IMC. 000 = IMCLK 001 = IMCLK/2 010 = IMCLK /4 011 = IMCLK /8 100 = IMCLK /16 101 = IMCLK /32 110 = IMCLK /64 111 = IMCLK /128 **NOTE:** Only 6 times same level in a row is recognized as effective signal.	000'b
PWMOFFEN	[12]	RW	PWMxOFF enable bit This field can determine the PWMxOFF enable/disable. 0 = Disable fault detection of PWMxOFF 1 = Enable fault detection of PWMxOFF	0
PWMOUTOFFEN	[13]	RW	PWM output disable by PWMxOFF This field can determine the PWM output disable by PWMxOFF. 0 = Disable PWM output disable by PWMxOFF 1 = Enable PWM output disable by PWMxOFF **NOTE:** If this bit is set to ‘1’ and PWMxOFF condition is met, the PWM output goes to High-Z state.	0
PWMOUTEN	[14]	RW	PWM output enable bit This field can determine the PWM output signal	0

Name	Bit	Type	Description	Reset Value
			enable/disable. The PWM output goes to High-Z state if this bit is set to 1. This bit can be used in the debugging time. 0 = Enable PWM output signal 1 = Disable PWM output signal	
PWMOUTOFFENBYOPAMP	[15]	RW	PWM output off enable by OPAMP This field can determine the PWM output signal enable/disable by detecting edge in the OPAMP block. The PWM output goes to High-Z state if this bit is set to '1'. 0 = Enable PWM output signal 1 = Disable PWM output signal	0
IMCLKSEL	[18:16]	RW	Inverter Clock (IMCLK) Selection This field can determine the Inverter motor clock selection. 000 = PCLK 001 = PCLK /2 010 = PCLK /4 011 = PCLK /8 100 = PCLK /16 101 = PCLK /32 110 = PCLK /64 111 = PCLK /128 **NOTE:** The clock source of dead-time compare register is IMCLK.	000'b
NUMSKIP	[24:20]	RW	Numbers of skip for motor match interrupt This field can determine the number of skip for motor match interrupt and ADC trigger signal. The unit of skip is PWM full cycle. 00000 = No skip 00001 = 1 Time skip 00010 = 2 Times skip 00011 = 3 Times skip 00100 = 4 Times skip ... 11100 = 28 Times skip 11101 = 29 Times skip 11110 = 30 Times skip 11111 = 31 Times skip	0000'b
SYNCSEL	[27:26]	RW	Synchronous Write Selection This field can determine the timing of synchronous write. 00 = Synchronous write at counter matches ZERO and IMC_TCR. 01 = Synchronous write at counter matches ZERO. 10 = Synchronous write at counter matches TOPCMP. 11 = Should not be used.	00'b

NOTE: If IMEN is equal to 0, all PWM output (PWMxU/Dx) goes to High-Z state.
In the immediate write mode (WMODE == 0) and saw tooth mode, the compare register update operation is like below picture.

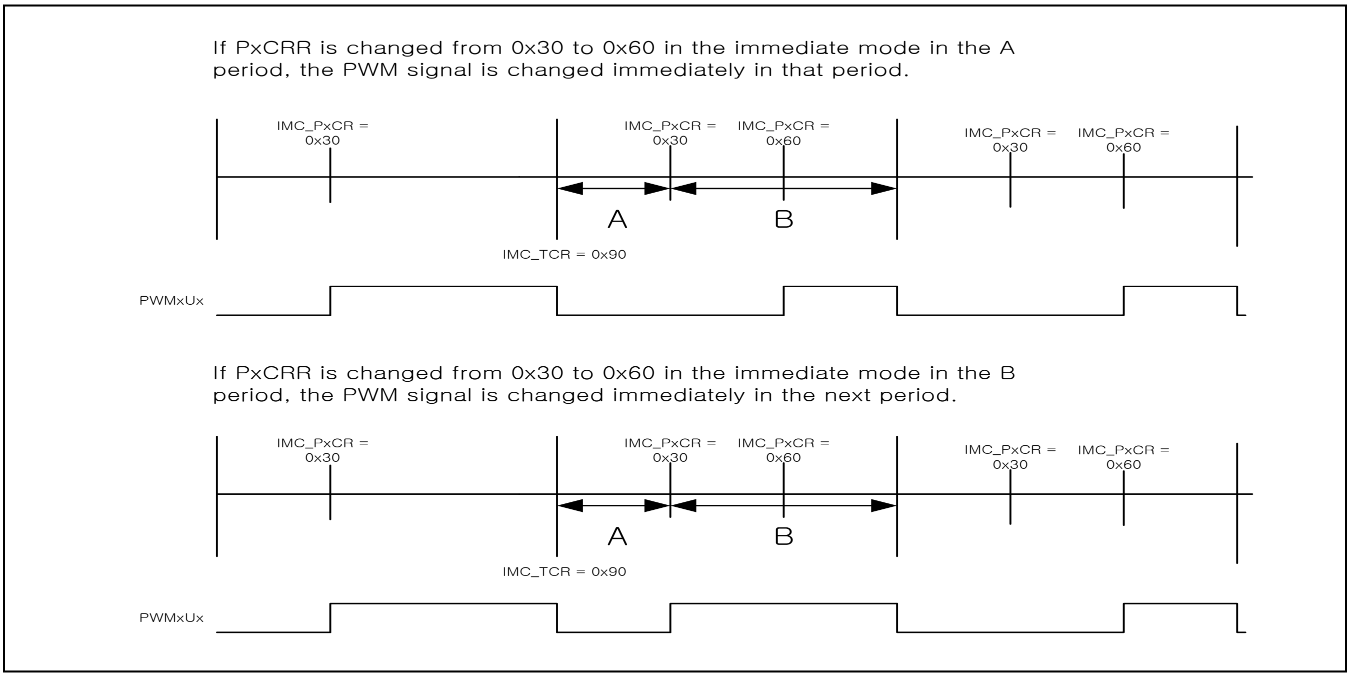

If PxCRR is changed from 0x30 to 0x60 in the immediate mode in the A period, the PWM signal is changed immediately in that period.
IMC_PxCR = 0x30
IMC_PxCR = 0x30
IMC_PxCR = 0x60
IMC_PxCR = 0x30
IMC_PxCR = 0x60
A
B
IMC_TCR = 0x90
PWMxUx
If PxCRR is changed from 0x30 to 0x60 in the immediate mode in the B period, the PWM signal is changed immediately in the next period.
IMC_PxCR = 0x30
IMC_PxCR = 0x30
IMC_PxCR = 0x60
IMC_PxCR = 0x30
IMC_PxCR = 0x60
A
B
IMC_TCR = 0x90
PWMxUx

Figure 14-24 Synchronous Write at Zero & IMC_TCR Match (SYNCSEL = 00'b, NUMSKIP = 00000'b)

NOTE: If WMODE is equal to 1 and NUMSKIP is equal to 0, the update of compare registers is like below picture. Because NUMSKIP is 0, the written compare registers are updated every IMC_TCR and 0 time.

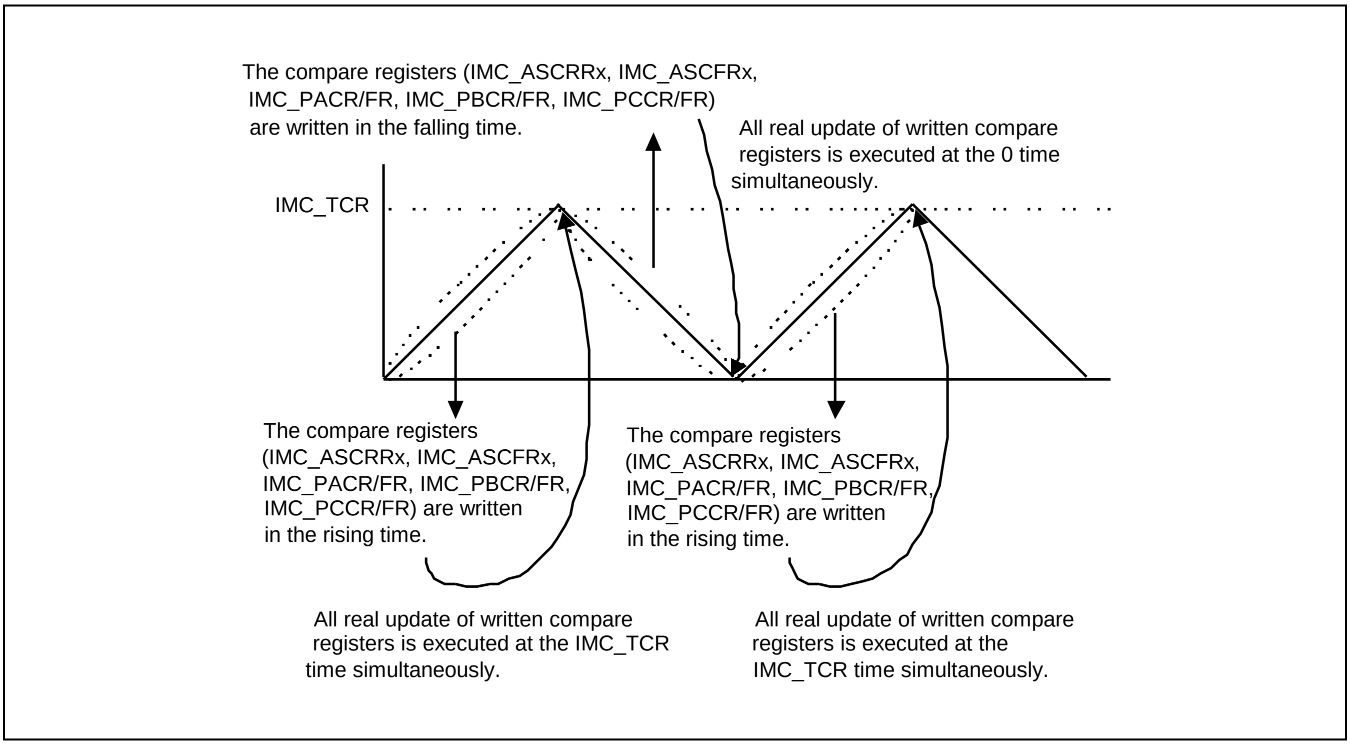

The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the falling time.
All real update of written compare registers is executed at the 0 time simultaneously.
IMC_TCR
The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the rising time.
The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the rising time.
All real update of written compare registers is executed at the IMC_TCR time simultaneously.
All real update of written compare registers is executed at the IMC_TCR time simultaneously.

Figure 14-25 Synchronous Write at Zero & IMC_TCR Match (SYNCSEL = 00'b, NUMSKIP = 00000'b)

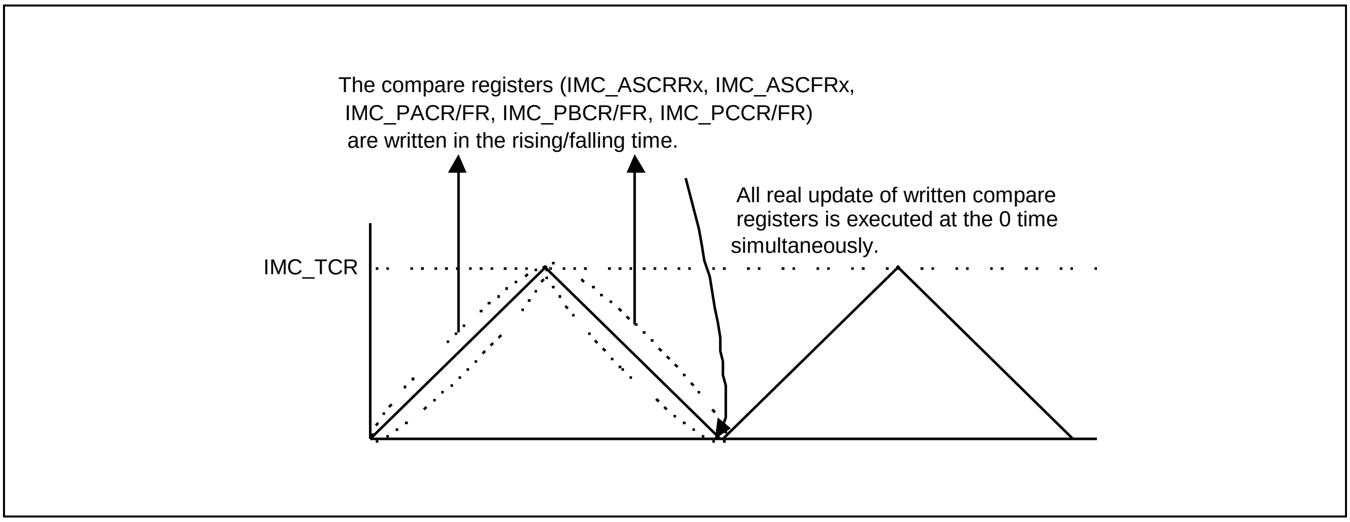

The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the rising/falling time.
All real update of written compare registers is executed at the 0 time simultaneously.
IMC_TCR

Figure 14-26 Synchronous Write at Zero Match (SYNCSEL = 01'b, NUMSKIP = 00000'b)

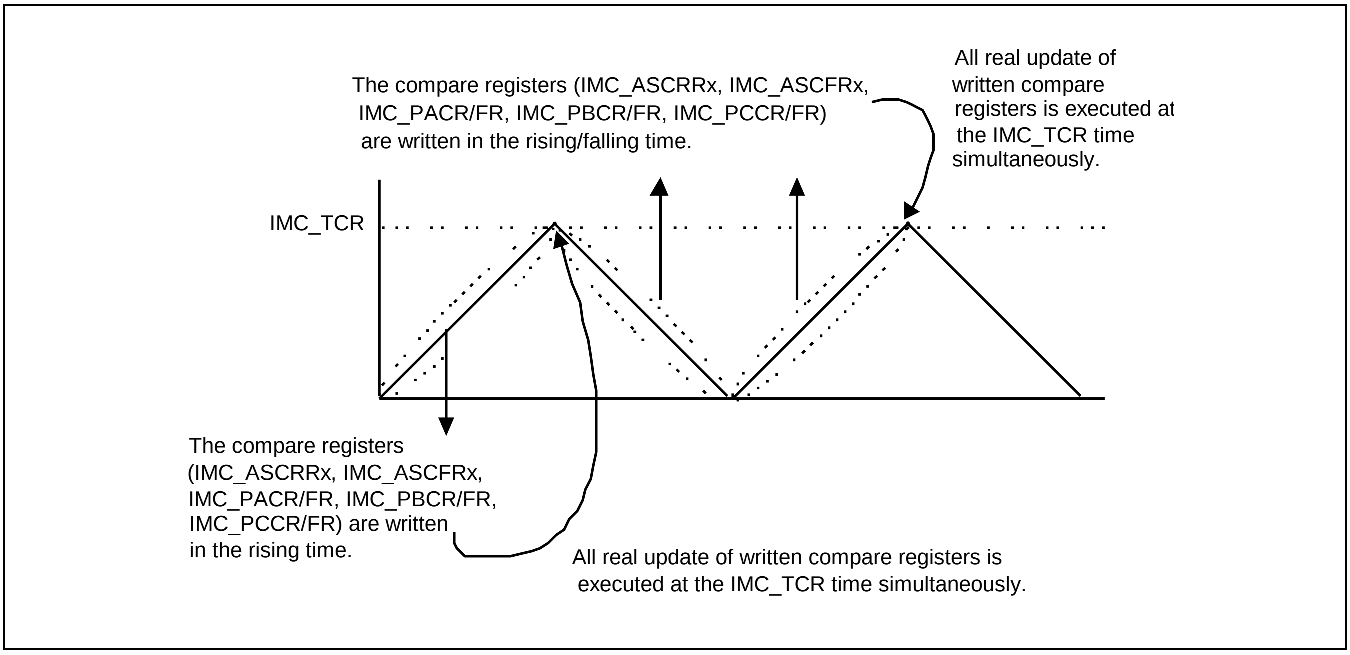

The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the rising/falling time.
All real update of written compare registers is executed at the IMC_TCR time simultaneously.
IMC_TCR
The compare registers (IMC_ASCRRx, IMC_ASCFRx, IMC_PACR/FR, IMC_PBCR/FR, IMC_PCCR/FR) are written in the rising time.
All real update of written compare registers is executed at the IMC_TCR time simultaneously.

Figure 14-27 Synchronous Write at IMC_TCR Match (SYNCSEL = 10'b, NUMSKIP = 00000'b)

NOTE: If WMODE is equal to 1 and NUMSKIP is equal to 1, the update of compare registers is like below picture. Because NUMSKIP is 1, the written compare registers are updated once per two IMC_TCR and 0 time. Second and fourth pulse is the skipped pulse.

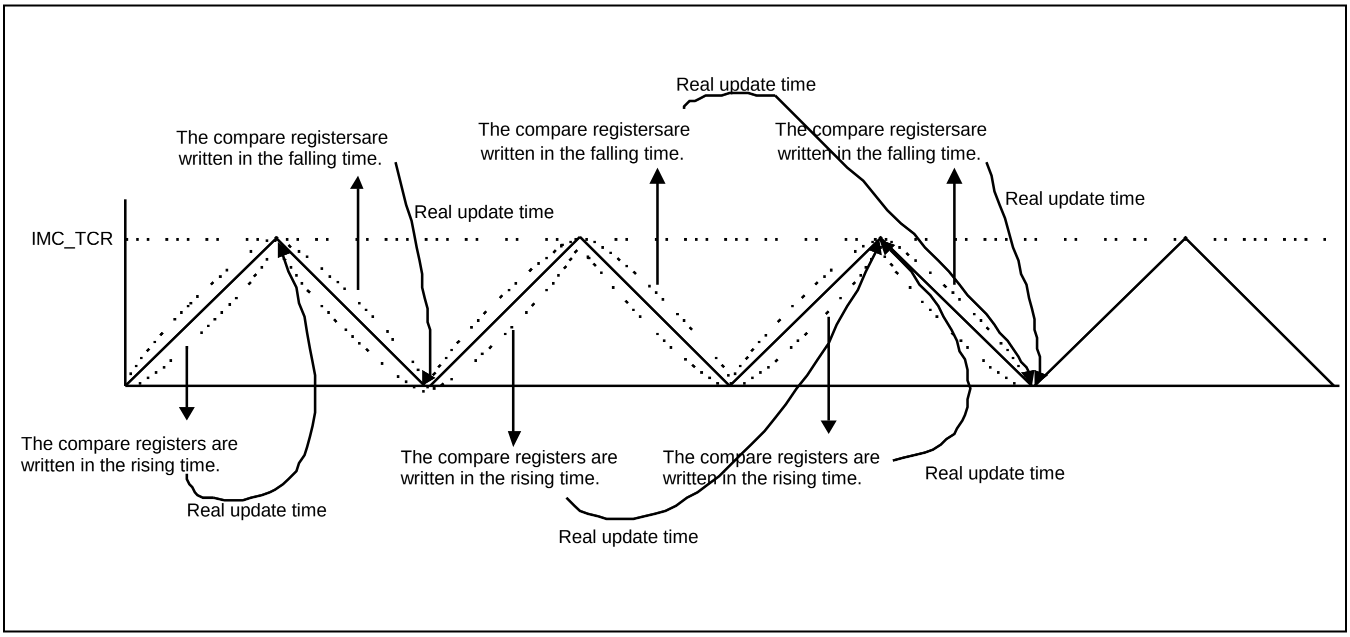

Real update time
The compare registersare written in the falling time.
The compare registersare written in the falling time.
The compare registersare written in the falling time.
Real update time
Real update time
IMC_TCR
The compare registers are written in the rising time.
The compare registers are written in the rising time.
The compare registers are written in the rising time.
Real update time
Real update time
Real update time

Figure 14-28 Synchronous Write at Zero & IMC_TCR Match (SYNCSEL = 00'b, NUMSKIP = 00001'b)

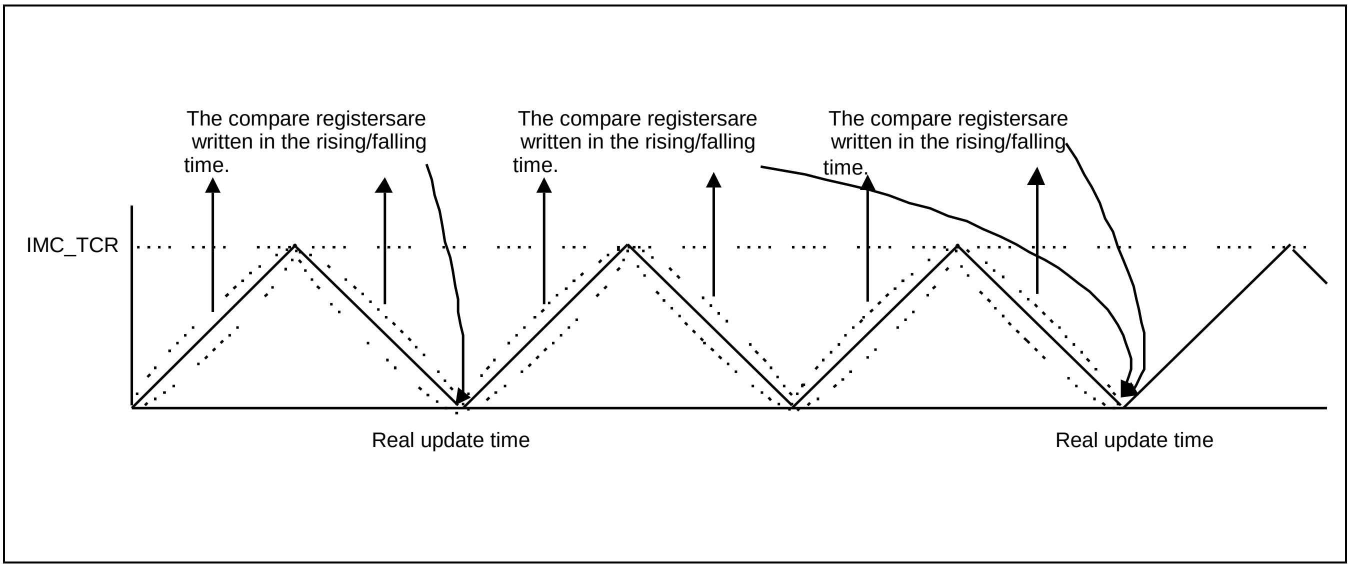

The compare registersare written in the rising/falling time.
The compare registersare written in the rising/falling time.
The compare registersare written in the rising/falling time.
IMC_TCR
Real update time
Real update time

Figure 14-30 Synchronous Write at Zero Match (SYNCSEL = 01'b, NUMSKIP = 00001'b)

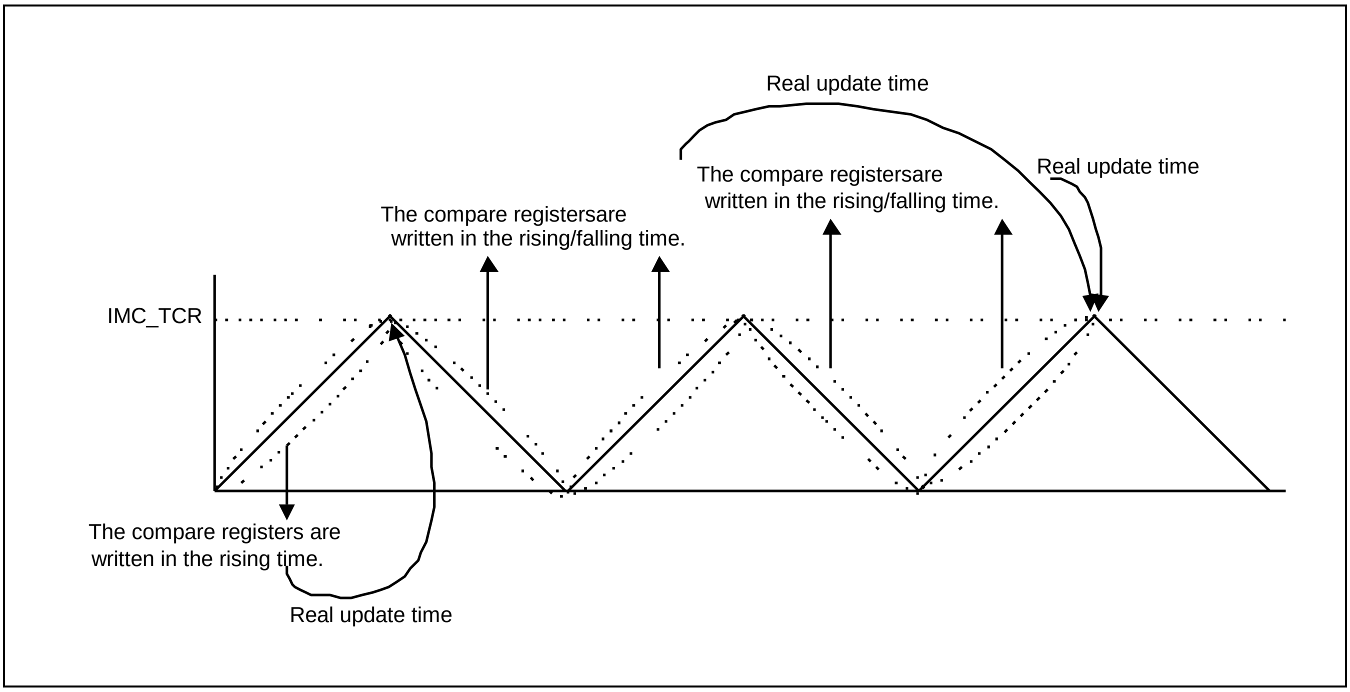

Real update time
The compare registersare written in the rising/falling time.
Real update time
The compare registersare written in the rising/falling time.
IMC_TCR
The compare registers are written in the rising time.
Real update time

Figure 14-29 Synchronous Write at IMC_TCR Match (SYNCSEL = 10'b, NUMSKIP = 00001'b)

NOTE: The value of NUMSKIP affects interrupt also. If IMC_ASCRRx/FRx is set to interrupt source and the value of NUMSKIP is 1, interrupt is not occurred in the second and fourth pulse.

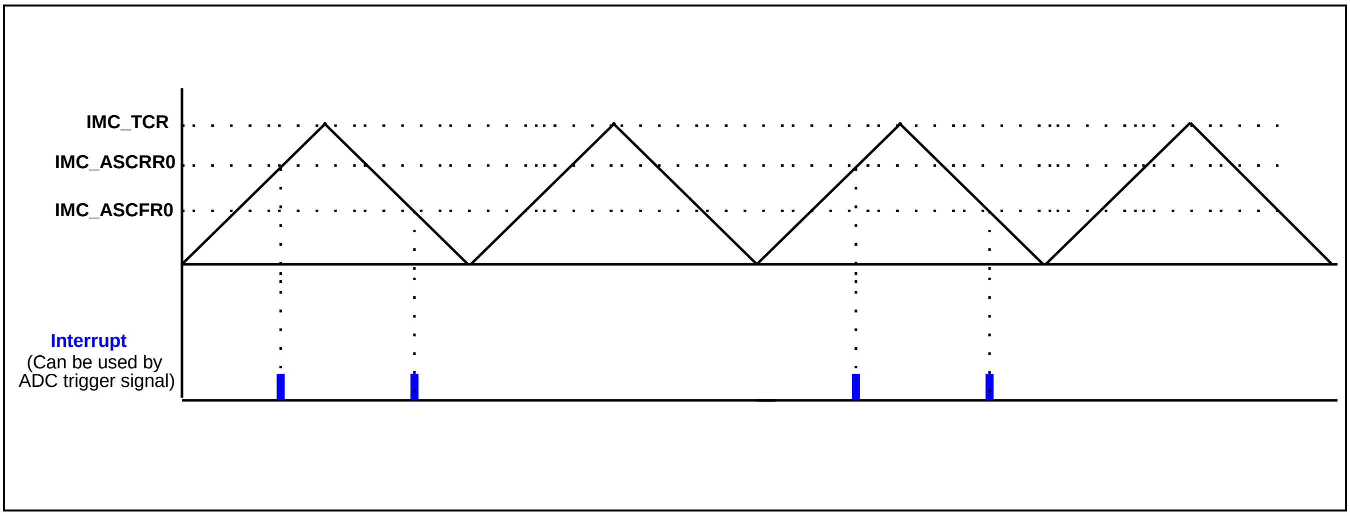


Figure 14-32 The Skip Control of ADC Trigger Signal Interrupt

14.3.1.5 IMC_CR1 (IMC Control Register 1)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD										PWMxU0DT	PWMxU1DT	PWMxU2DT	PWMxD0DT	PWMxD1DT	PWMxD2DT	RSVD		PWMxU0 LEVEL	PWMxU1 LEVEL	PWMxU2LEVEL	PWMxD0 LEVEL	PWMxD1 LEVEL	PWMxD2LEVEL	RSVD		PWMxU0EN	PWMxU1EN	PWMxU2EN	PWMxD0EN	PWMxD1EN	PWMxD2EN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R	R	R W	R W	R W	R W	R W	R W	R	R	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PWMxD2EN	[0]	RW	PWMxD2 PWM Output Enable Bit 0 = Enable PWM signal to PWMxD2 1 = Disable PWM signal to PWMxD2 → The level of PWMxD2 is determined by IMCON1.8	0
PWMxD1EN	[1]	RW	PWMxD1 PWM Output Enable Bit 0 = Enable PWM signal to PWMxD1 1 = Disable PWM signal to PWMxD1 → The level of PWMxD1 is determined by IMCON1.9	0
PWMxD0EN	[2]	RW	PWMxD0 PWM Output Enable Bit 0 = Enable PWM signal to PWMxD0 1 = Disable PWM signal to PWMxD0 → The level of PWMxD2 is determined by IMCON1.10	0
PWMxU2EN	[3]	RW	PWMxU2 PWM Output Enable Bit 0 = Enable PWM signal to PWMxU2 1 = Disable PWM signal to PWMxU2 → The level of PWMxU2 is determined by IMCON1.11	0
PWMxU1EN	[4]	RW	PWMxU1 PWM Output Enable Bit 0 = Enable PWM signal to PWMxU1 1 = Disable PWM signal to PWMxU1 → The level of PWMxU1 is determined by IMCON1.12	0
PWMxU0EN	[5]	RW	PWMxU0 PWM Output Enable Bit 0 = Enable PWM signal to PWMxU0 1 = Disable PWM signal to PWMxU0 → The level of PWMxU0 is determined by IMCON1.13	0
PWMxD2LEVEL	[8]	RW	PWMxD2 Output Level Selection Bit 0 = Low level 1 = High level	0
PWMxD1 LEVEL	[9]	RW	PWMxD1 Output Level Selection Bit 0 = Low level	0

Name	Bit	Type	Description	Reset Value
			1 = High level	
PWMxD0 LEVEL	[10]	RW	PWMxD0 Output Level Selection Bit 0 = Low level 1 = High level	0
PWMxU2LEVEL	[11]	RW	PWMxU2 Output Level Selection Bit 0 = Low level 1 = High level	0
PWMxU1 LEVEL	[12]	RW	PWMxU1 Output Level Selection Bit 0 = Low level 1 = High level	0
PWMxU0 LEVEL	[13]	RW	PWMxU0 Output Level Selection Bit 0 = Low level 1 = High level	0
PWMxD2DT	[16]	RW	PWMxD2 Dead-time Insert Bit: before PWM output disable by setting PWMxD2EN 0 = No insertion 1 = Insertion	0
PWMxD1DT	[17]	RW	PWMxD1 Dead-time Insert Bit: before PWM output disable by setting PWMxD1EN 0 = No insertion 1 = Insertion	0
PWMxD0DT	[18]	RW	PWMxD0 Dead-time Insert Bit: before PWM output disable by setting PWMxD0EN 0 = No insertion 1 = Insertion	0
PWMxU2DT	[19]	RW	PWMxU2 Dead-time Insert Bit: before PWM output disable by setting PWMxU2EN 0 = No insertion 1 = Insertion	0
PWMxU1DT	[20]	RW	PWMxU1 Dead-time Insert Bit: before PWM output disable by setting PWMxU1EN 0 = No insertion 1 = Insertion	0
PWMxU0DT	[21]	RW	PWMxU0 Dead-time Insert Bit: before PWM output disable by setting PWMxU0EN 0 = No insertion 1 = Insertion	0

NOTE: There is no releationship between level setting in the IMC_CR1 register and SWAP function in the IMC_CR0 register.

14.3.1.6 IMC_CNTR (IMC 16 Bit Counter Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																CV															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
CV	[15:0]	R	Count Value This field contains the current IMC's count value.	0x0000

14.3.1.7 IMC_SR (IMC Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																													OPAMPEDGEDET	UPDOWN	FAULTSTAT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	R	RW

Name	Bit	Type	Description	Reset Value
FAULTSTAT	[0]	RW	Status of PWM Output Signal 0 = Normal operating 1 = High-Z (Inverter motor block is operating but the status of PWM signal is High-Z. This bit can be set by fault detection of PWMxOFF pin or IMC_CR0.14.) **NOTE:** If this bit is written to 0 and IMC_CR0.14 is 0, inverter motor control signal is output to PWM output.	0
UPDOWN	[1]	R	Status of PWM counter (Read Only Bit) This bit can notify the status of PWM counter. 0 = Up counting 1 = Down counting **NOTE:** This bit is always ‘0’ in the saw-tooth mode.	0
OPAMPEDGEDET	[2]	RW	OPAMP edge detect This bit can notify the status of edge detection from OP-AMP block 0 = not detected 1 = Detected **NOTE:** This bit is retained if OP-AMP edge is detected. This bit can be cleared by the software clear.	0

14.3.1.8 IMC_IMSCR (IMC Interrupt Mask Set and Clear Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																		ADCFM2	ADCRM 2	ADCFM1	ADCRM 1	ADCFM0	ADCRM0	TOP	ZERO	RSVD					FAULT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
FAULT	[0]	RW	FAULT Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ZERO	[6]	RW	ZERO Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
TOP	[7]	RW	TOP Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCRM0	[8]	RW	ADC Compare0 Rising Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCFM0	[9]	RW	ADC Compare0 Falling Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCRM1	[10]	RW	ADC Compare1 Rising Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCFM1	[11]	RW	ADC Compare1 Falling Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCRM2	[12]	RW	ADC Compare2 Rising Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
ADCFM2	[13]	RW	ADC Compare2 Falling Match Interrupt Mask 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

14.3.1.9 IMC_RISR (IMC Raw Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																		ADCFM2	ADCRM2	ADCFM1	ADCRM1	ADCFM0	ADCRM0	TOP	ZERO	RSVD					FAULT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
FAULT	[0]	R	FAULT Raw Interrupt State Gives the raw interrupt state (prior to masking) of the FAULT interrupt	0
ZERO	[6]	R	ZERO Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ZERO interrupt	0
TOP	[7]	R	TOP Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the TOP interrupt	0
ADCRM0	[8]	R	ADC Compare0 Rising Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCRM0 interrupt	0
ADCFM0	[9]	R	ADC Compare0 Falling Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCFM0 interrupt	0
ADCRM1	[10]	R	ADC Compare1 Rising Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCRM1 interrupt	0
ADCFM1	[11]	R	ADC Compare1 Falling Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCFM1 interrupt	0
ADCRM2	[12]	R	ADC Compare2 Rising Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCRM2 interrupt	0
ADCFM2	[13]	R	ADC Compare2 Falling Match Raw Interrupt State Gives the raw interrupt state (prior to masking) of the ADCFM2 interrupt	0

NOTE: On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect. ZERO is set right after IMC enable.

14.3.1.10 IMC_MISR (IMC Masked Interrupt Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																		ADCFM2	ADCRM2	ADCFM1	ADCRM1	ADCFM0	ADCRM0	TOP	ZERO	RSVD					FAULT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
FAULT	[0]	R	FAULT Masked Interrupt State Gives the masked interrupt state (prior to masking) of the FAULT interrupt	0
ZERO	[6]	R	ZERO Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ZERO interrupt	0
TOP	[7]	R	TOP Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the TOP interrupt	0
ADCRM0	[8]	R	ADC Compare0 Rising Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCRM0 interrupt	0
ADCFM0	[9]	R	ADC Compare0 Falling Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCFM0 interrupt	0
ADCRM1	[10]	R	ADC Compare1 Rising Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCRM1 interrupt	0
ADCFM1	[11]	R	ADC Compare1 Falling Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCFM1 interrupt	0
ADCRM2	[12]	R	ADC Compare2 Rising Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCRM2 interrupt	0
ADCFM2	[13]	R	ADC Compare2 Falling Match Masked Interrupt State Gives the masked interrupt state (prior to masking) of the ADCFM2 interrupt	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

14.3.1.11 IMC_ICR (IMC Interrupt Clear Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																		ADCFM2	ADCRM2	ADCFM1	ADCRM1	ADCFM0	ADCRM0	TOP	ZERO	RSVD					FAULT
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
FAULT	[0]	W	FAULT Interrupt Clear 0 = No effect 1 = Clears the FAULT interrupt	0
ZERO	[6]	W	ZERO Match Interrupt Clear 0 = No effect 1 = Clears the ZERO interrupt	0
TOP	[7]	W	TOP Match Interrupt Clear 0 = No effect 1 = Clears the TOP interrupt	0
ADCRM0	[8]	W	ADC Compare0 Rising Match Interrupt Clear 0 = No effect 1 = Clears the ADCRM0 interrupt	0
ADCFM0	[9]	W	ADC Compare0 Falling Match Interrupt Clear 0 = No effect 1 = Clears the ADCFM0 interrupt	0
ADCRM1	[10]	W	ADC Compare1 Rising Match Interrupt Clear 0 = No effect 1 = Clears the ADCRM1 interrupt	0
ADCFM1	[11]	W	ADC Compare1 Falling Match Interrupt Clear 0 = No effect 1 = Clears the ADCFM1 interrupt	0
ADCRM2	[12]	W	ADC Compare2 Rising Match Interrupt Clear 0 = No effect 1 = Clears the ADCRM2 interrupt	0
ADCFM2	[13]	W	ADC Compare2 Falling Match Interrupt Clear 0 = No effect 1 = Clears the ADCFM2 interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

14.3.1.12 IMC_TCR (IMC 16 Bit Top Compare Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																TOPCMPDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
TOPCMPDAT	[15:0]	RW	This field can determine the TOP compare register value.	0x0000

The update of IMC_TCR can be executed only when IMC is disabled. (IMC_CR0.0 = 0)

Caution: IMC_PACRR/FR, IMC_PBCRR/FR and IMC_PCCRR/FR must be less than or equal to IMC_TCR. (IMC_PxCRR/FR <= IMC_TCR)

14.3.1.13 IMC_DTCR (16 Bit Dead-time Control Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																DTCMPDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DTCMPDAT	[15:0]	RW	This field can determine the Dead-time compare register value.	0x0000

NOTE: If ADC compare interrupt is used, ADCCMPR/Fx must be set to from 1 to TOPCMP –1.
(0 < ADCCMPR/Fx < TOPCMP)

14.3.1.14 IMC_PACRR (16 Bit Phase A Compare Rising Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																PACMPRDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PACMPRDAT	[15:0]	RW	This field can determine the Phase A compare register value at rising.	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.15 IMC_PBCRR (16 Bit Phase B Compare Rising Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																PBCMPRDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PBCMPRDAT	[15:0]	RW	This field can determine the Phase B compare register value at rising.	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.16 IMC_PCCRR (16 Bit Phase C Compare Rising Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																PCCMPRDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PCCMPRDAT	[15:0]	RW	This field can determine the Phase C compare register value at rising.	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.17 IMC_PACFR (16 Bit Phase A Compare Falling Register)

- Address = Base Address + 0x0040, Reset Value = 0x0000_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
RSVD																PACMPFDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PACMPFDAT	[15:0]	RW	This field can determine the Phase A compare register value at falling	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.18 IMC_PBCFR (16 Bit Phase B Compare Falling Register)

- Address = Base Address + 0x0044, Reset Value = 0x0000_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
RSVD																PBCMPFDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PBCMPFDAT	[15:0]	RW	This field can determine the Phase B compare register value at falling	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.19 IMC_PCCFR (16 Bit Phase C Compare Falling Register)

- Address = Base Address + 0x0048, Reset Value = 0x0000_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
RSVD																PCCMPFDAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PCCMPFDAT	[15:0]	RW	This field can determine the Phase C compare register value at falling	0x0000

NOTE: If ADC compare interrupt is used, IMC_PxCRR/F must be set to from 1 to IMC_TCR –1.
(0 < IMC_PxCRR/FR < IMC_TCR)

14.3.1.20 IMC_ASTSR (IMC ADC Start Signal Select Register)

- Address = Base Address + 0x004C, Reset Value = 0x0000_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
RSVD																								ADCMPF2SEL	ADCMPR2SEL	ADCMPF1SEL	ADCMPR1SEL	ADCMPF0SEL	ADCMPR0SEL	0SEL	TOPCMPSEL
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
TOPCMPSEL	[0]	RW	ADC start trigger signal by TOPCMP match This bit can determine whether TOPCMP match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
0SEL	[1]	RW	ADC start trigger signal by counter zero match This bit can determine whether 0 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
ADCMPR0SEL	[2]	RW	ADC start trigger signal by ADCCMPR0 match This bit can determine whether ADCCMPR0 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
ADCMPF0SEL	[3]	RW	ADC start trigger signal by ADCCMPF0 match This bit can determine whether ADCCMPF0 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
ADCMPR1SEL	[4]	RW	ADC start trigger signal by ADCCMPR1 match This bit can determine whether ADCCMPR1 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
ADCMPF1SEL	[5]	RW	ADC start trigger signal by ADCCMPF1 match This bit can determine whether ADCMPF1 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0

Name	Bit	Type	Description	Reset Value
ADCMPR2SEL	[6]	RW	ADC start trigger signal by ADCCMPR2 match This bit can determine whether ADCMPR2 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0
ADCMPF2SEL	[7]	RW	ADC start trigger signal by ADCCMPF2 match This bit can determine whether ADCMPF2 match of IMCNT is used for ADC trigger signal or not. 0 = Not selected 1 = Selected	0

NOTE:

1. ADC conversion must not be overlapped by setting appropriate value to each compare register.
2. The setting of this register bit doesn’t affect interrupt generation.
3. When IMC is in a saw-tooth wave mode, the values of ADCCMPF0SEL, ADCCMPF1SEL and ADC MPF2SEL bit do not effect in operation.

14.3.1.21 IMC_ASCRR0 (16 Bit ADC Start Compare Rising Register 0)

- Address = Base Address + 0x0050, Reset Value = 0x0000_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
RSVD																ADCMPR0DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADCMPR0DAT	[15:0]	RW	This field can determine the ADC compare register value at rising.	0x0000

14.3.1.22 IMC_ASCRR1 (16 Bit ADC Start Compare Rising Register 1)

- Address = Base Address + 0x0054, Reset Value = 0x0000_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
RSVD																ADCMPR1DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ADCMPR1DAT	[15:0]	RW	This field can determine the ADC compare register value at rising.	0x0000

14.3.1.23 IMC_ASCRR2 (16 Bit ADC Start Compare Rising Register 2)

- Address = Base Address + 0x0058, Reset Value = 0x0000_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
RSVD																ADCMPR2DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ADCMPR2DAT	[15:0]	RW	This field can determine the ADC compare register value at rising.	0x0000

14.3.1.24 IMC_ASCFR0 (16 Bit ADC Start Compare Falling Register 0)

- Address = Base Address + 0x005C, Reset Value = 0x0000_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
RSVD																ADCMPF0DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADCMPF0DAT	[15:0]	RW	This field can determine the ADC compare register value at falling.	0x0000

14.3.1.25 IMC_ASCFR1 (16 Bit ADC Start Compare Falling Register 1)

- Address = Base Address + 0x0060, Reset Value = 0x0000_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
RSVD																ADCMPF1DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADCMPF1DAT	[15:0]	RW	This field can determine the ADC compare register value at falling.	0x0000

14.3.1.26 IMC_ASCFR2 (16 Bit ADC Start Compare Falling Register 2)

- Address = Base Address + 0x0064, Reset Value = 0x0000_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
RSVD																ADCMPF2DAT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ADCMPF2DAT	[15:0]	RW	This field can determine the ADC compare register value at falling.	0x0000

15 Interrupt Controller

15.1 Overview

The NVIC and Cortex-M3 processor enable low latency interrupt processing and provide efficient service for late arriving interrupts. There are 60 individual interrupt vector sources including 16 Cortex-M3 own interrupt sources.

15.1.1 Features

- Cortex-M3 NVIC interface
- 60 interrupt vector sources(not including 16 Cortex-M3 own interrupt vector sources)
- 16 programmable priorities: from 0 to 63
- Dynamically reconfigurable interrupt priority, 16 Levels
- Low latency exception and interrupt handling
- Implement Cortex-M3 system control registers

The NVIC and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts. All interrupts including the core exceptions are managed by the NVIC. For more information on exceptions and NVIC programming see Chap 5 Exceptions & Chap 8 Nested Vectored Interrupt Controller of the *Cortex-M3*$^{\text{TM}}$ *Technical Reference Manual*.

15.2 Functional Description

15.2.1 Block Diagram

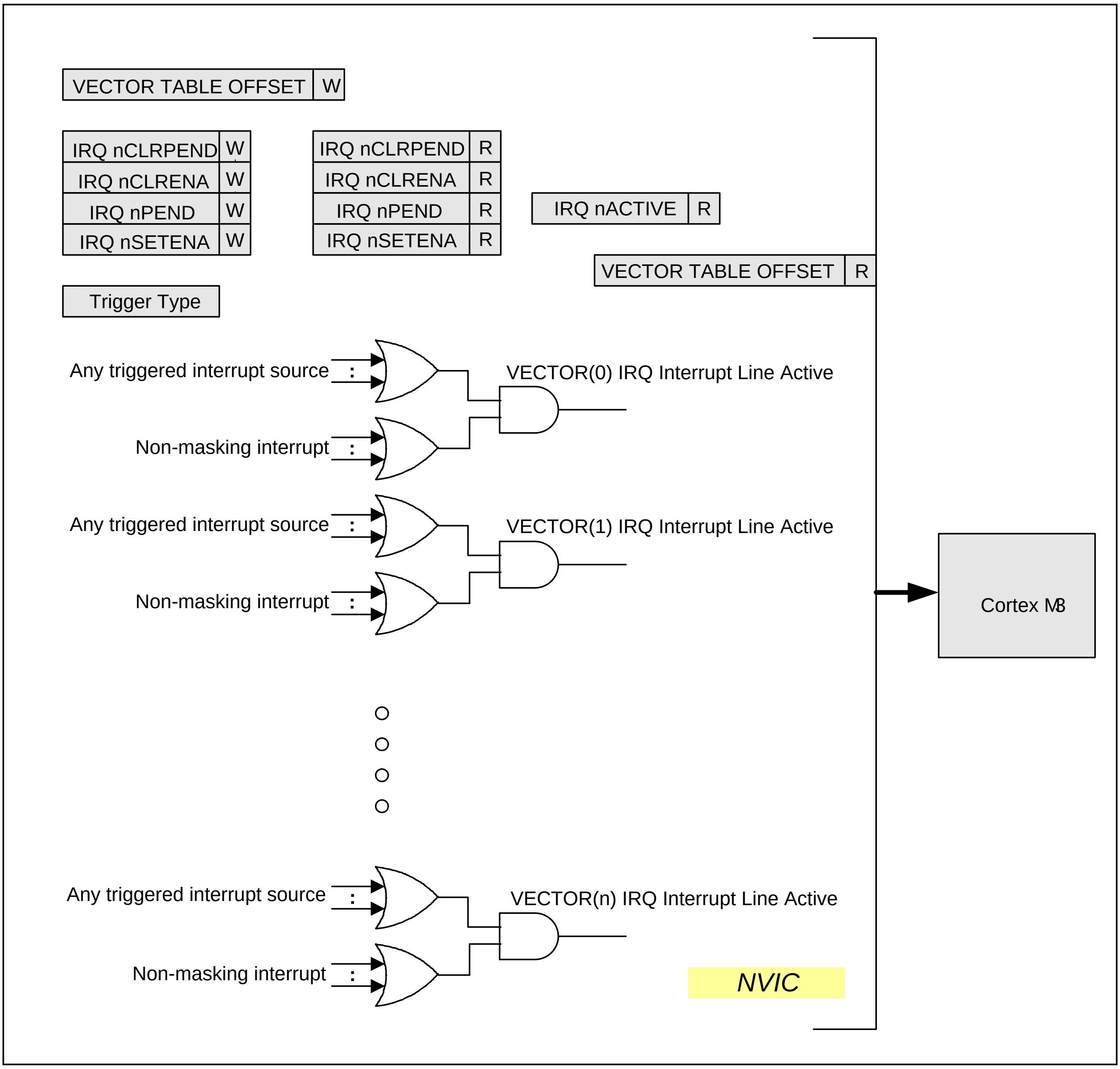


Figure 15-1 Interrupt Control Block Diagram

15.2.2 Interrupt Sources & Vectors

Interrupt vector table includes Cortex-M3 interrupt vectors and vectors for Sxxx interrupt sources.

Table 15-1 Core Exception Vector

Vector Category	Address	Vector	Description
Core Interrupt Vector	0x0000_0000	Reserved	Starting value of the MSP
	0x0000_0004	Reset	–
	0x0000_0008	NMI	Non-maskable interrupt
	0x0000_000C	HardFault	All class of fault
	0x0000_0010	MemManage	Memory Management Fault
	0x0000_0014	BusFault	Pre-fetch fault, memory access fault
	0x0000_0018	UsageFault	Undefined instruction or illegal state
	0x0000_001C	Reserved	–
	0x0000_0020	Reserved	–
	0x0000_0024	Reserved	–
	0x0000_0028	Reserved	–
	0x0000_002C	SVCall	System service call via SWI instruction
	0x0000_0030	Debug Monitor	Debug Monitor
	0x0000_0034	Reserved	–
	0x0000_0038	PendSV	Pendable request for system service
	0x0000_003C	SysTick	System tick timer
System Interrupt Vector	0x0000_0040 ~ 0x0000_013C	INTV0 ~ INTV63	Vectors for interrupt sources generated by peripherals and external input signals

15.2.3 System Interrupt Vector

Table 15-2 System Interrupt Vectors & Sources

Num.	Address	Vector	Interrupt Sources
0	0x0000_0040	WDT	Refer to Watch-dog Timer Interrupt Register
1	0x0000_0044	CM	Refer to Clock Manager Interrupt Register
2	0x0000_0048	PFC	Refer to Program Flash Controller Interrupt Register
3	0x0000_004C	DFC	Refer to Data Flash Controller Interrupt Register
4	0x0000_0050	DMA	Refer to DMA Controller Interrupt Register
5	0x0000_0054	FRT	Refer to Free Running Timer Interrupt Register
6	0x0000_0058	WSI0	Refer to Clock Manager Wakeup Source Interrupt 0
7	0x0000_005C	WSI1	Refer to Clock Manager Wakeup Source Interrupt 1
8	0x0000_0060	IMC0	Refer to Inverter Motor Controller 0 Interrupt Register
9	0x0000_0064	ENC0	Refer to Encoder Counter 0 Interrupt Register
10	0x0000_0068	IMC1	Refer to Inverter Motor Controller 1 Interrupt Register
11	0x0000_006C	ENC1	Refer to Encoder Counter 1 Interrupt Register
12	0x0000_0070	CAN0	Refer to CAN 0 Interrupt Register
13	0x0000_0074	USART0	Refer to USART 0 Interrupt Register
14	0x0000_0078	ADC0	Refer to ADC 0 Interrupt Register
15	0x0000_007C	ADC1	Refer to ADC 1 Interrupt Register
16	0x0000_0080	SSP0	Refer to SSP 0 Interrupt Register
17	0x0000_0084	I2C0	Refer to I2C 0 Interrupt Register
18	0x0000_0088	TC0	Refer to Timer/Counter 0 Interrupt Register
19	0x0000_008C	PWM0	Refer to PWM 0 Interrupt Register
20	0x0000_0090	WSI2	Refer to Clock Manager Wakeup Source Interrupt 2
21	0x0000_0094	WSI3	Refer to Clock Manager Wakeup Source Interrupt 3
22	0x0000_0098	TC1	Refer to Timer/Counter 1 Interrupt Register
23	0x0000_009C	PWM1	Refer to PWM 1 Interrupt Register
24	0x0000_00A0	USART1	Refer to USART 1 Interrupt Register
25	0x0000_00A4	SSP	Refer to SSP 1 Interrupt Register
26	0x0000_00A8	I2C1	Refer to I2C 1 Interrupt Register
27	0x0000_00AC	CAN1	Refer to CAN 1 Interrupt Register
28	0x0000_00B0	STT	Refer to Stamp Timer Interrupt Register
29	0x0000_00B4	USART2	Refer to USART 2 Interrupt Register
30	0x0000_00B8	TC2	Refer to Timer/Counter 2 Interrupt Register
31	0x0000_00BC	TC3	Refer to Timer/Counter 3 Interrupt Register
32	0x0000_00C0	PWM2	Refer to PWM 2 Interrupt Register
33	0x0000_00C4	WSI4	Refer to Clock Manager Wakeup Source Interrupt 4
34	0x0000_00C8	WSI5	Refer to Clock Manager Wakeup Source Interrupt 5

Num.	Address	Vector	Interrupt Sources
35	0x0000_00CC	PWM3	Refer to PWM 3 Interrupt Register
36	0x0000_00D0	USART3	Refer to USART 3 Interrupt Register
37	0x0000_00D4	GPIO0	Refer to GPIO interrupt Register
38	0x0000_00D8	GPIO1	Refer to GPIO interrupt Register
39	0x0000_00DC	TC4	Refer to Timer/Counter 4 Interrupt Register
40	0x0000_00E0	WSI6	Refer to Clock Manager Wakeup Source Interrupt 6
41	0x0000_00E4	WSI7	Refer to Clock Manager Wakeup Source Interrupt 7
42	0x0000_00E8	PWM4	Refer to PWM 4 Interrupt Register
43	0x0000_00EC	TC5	Refer to Timer/Counter 5 Interrupt Register
44	0x0000_00F0	WSI8	Refer to Clock Manager Wakeup Source Interrupt 8
45	0x0000_00F4	WSI9	Refer to Clock Manager Wakeup Source Interrupt 9
46	0x0000_00F8	WSI10	Refer to Clock Manager Wakeup Source Interrupt 10
47	0x0000_00FC	WSI11	Refer to Clock Manager Wakeup Source Interrupt 11
48	0x0000_0100	PWM5	Refer to PWM 5 Interrupt Register
49	0x0000_0104	TC6	Refer to Timer/Counter 6 Interrupt Register
50	0x0000_0108	WSI12	Refer to Clock Manager Wakeup Source Interrupt 12
51	0x0000_010C	WSI13	Refer to Clock Manager Wakeup Source Interrupt 13
52	0x0000_0110	WSI14	Refer to Clock Manager Wakeup Source Interrupt 14
53	0x0000_0114	WSI15	Refer to Clock Manager Wakeup Source Interrupt 15
54	0x0000_0118	PWM6	Refer to PWM 6 Interrupt Register
55	0x0000_011C	TC7	Refer to Timer/Counter 7 Interrupt Register
56	0x0000_0120	PWM7	Refer to PWM 7 Interrupt Register
57	0x0000_0124	GPIO2	Refer to GPIO interrupt Register
58	0x0000_0128	GPIO3	Refer to GPIO interrupt Register
59	0x0000_012C	OPAMP	OP-AMP interrupt Register
60	–	–	Reserved
61	–	–	Reserved
62	–	–	Reserved
63	–	–	Reserved

For more information on interrupt sources according each vector, see each chapters describing their function.

16 I/O Configuration (IOCONF)

16.1 Overview

This chapter describes the configuration of specific function pins mapped each I/O pin. The I/O configuration module has been designed to handle specific I/O configuration features and devices special functions.

16.1.1 Features

- Configuration of function pin.
  - Selectable one among 4 functions
    - Function 0(GPIO), Function 1, Function 2, or Function 3
- Configuration of Pull-Up Register (PUCR)
- Configuration of Open-Drain (ODCR)

16.2 Functional Description

16.2.1 General Description

The peripherals have their own dedicated pins multiplexed with general purpose I/O pin (GPIO) and other peripheral pins. User should configure the corresponding pin for target peripheral's function. A pin can be defined as one function pin among maximum 4 functions. The first function of pin will be general purpose I/O (GPIO).

I/O configuration block can control enable/disable for each pin's pull-up resistor and open-drain.

16.2.2 Peripheral Configuration

Table 16-1 GPIO 0 Function Mode Configuration

Pin Num.	Function Number	F0 (00)	F1 (01)	F2 (10)	F3 (11)
20	IO0_0_FSEL[1:0]	P0.0 (B)	VLCD1 (O)	CLKOUT (O)	***USARTRXD0*** (I)
21	IO0_1_FSEL[1:0]	P0.1 (B)	VLCD2 (O)	–	***USARTTXD0*** (O)
22	IO0_2_FSEL[1:0]	P0.2 (B)	VLCD3 (O)	TCAP7 (I)	***USARTCLK0*** (B)
23	IO0_3_FSEL[1:0]	P0.3 (B)	COM0 (O)	TPWM7 (O)	EXI14 (I)
24	IO0_4_FSEL[1:0]	P0.4 (B)	COM1 (O)	TCAP6 (I)	EXI15 (I)
25	IO0_5_FSEL[1:0]	P0.5 (B)	COM2 (O)	TPWM6 (O)	–
26	IO0_6_FSEL[1:0]	P0.6 (B)	COM3 (O)	TCAP5 (I)	–
27	IO0_7_FSEL[1:0]	P0.7 (B)	SEG0 (O)	TPWM5 (O)	ADTRG1 (I)
28	IO0_8_FSEL[1:0]	P0.8 (B)	SEG1 (O)	TCAP4 (I)	***SSPFSS0*** (B)
29	IO0_9_FSEL[1:0]	P0.9 (B)	SEG2 (O)	TPWM4 (O)	***SSPCLK0*** (B)
30	IO0_10_FSEL[1:0]	P0.10 (B)	SEG3 (O)	TCLK7 (I)	***SSPMISO0*** (B)
31	IO0_11_FSEL[1:0]	P0.11 (B)	SEG4 (O)	TCLK6 (I)	***SSPMOSI0*** (B)
32	IO0_12_FSEL[1:0]	P0.12 (B)	SEG5 (O)	TCLK5 (I)	EXI0 (I)
33	IO0_13_FSEL[1:0]	P0.13 (B)	SEG6 (O)	TCLK4 (I)	EXI1 (I)
34	IO0_14_FSEL[1:0]	P0.14 (B)	SEG7 (O)	TCAP3 (I)	***SCL0*** (B)
35	IO0_15_FSEL[1:0]	P0.15 (B)	SEG8 (O)	TPWM3 (O)	***SDA0*** (B)
36	IO0_16_FSEL[1:0]	P0.16 (B)	SEG9 (O)	TCLK3 (I)	EXI2 (I)
37	IO0_17_FSEL[1:0]	P0.17 (B)	SEG10 (O)	TCLK2 (I)	EXI3 (I)
38	IO0_18_FSEL[1:0]	P0.18 (B)	SEG11 (O)	TCAP2 (I)	***CANTX0*** (O)
39	IO0_19_FSEL[1:0]	P0.19 (B)	SEG12 (O)	TPWM2 (O)	***CANRX0*** (I)
40	IO0_20_FSEL[1:0]	P0.20 (B)	SEG13 (O)	TCLK1 (I)	***USARTCLK1*** (B)
41	IO0_21_FSEL[1:0]	P0.21 (B)	SEG14 (O)	TCAP1 (I)	***USARTRXD1*** (I)
42	IO0_22_FSEL[1:0]	P0.22 (B)	SEG15 (O)	TPWM1 (O)	***USARTTXD1*** (O)
43	IO0_23_FSEL[1:0]	P0.23 (B)	SEG16 (O)	PWM0 (O)	–
44	IO0_24_FSEL[1:0]	P0.24 (B)	SEG17 (O)	PWM1 (O)	–
45	IO0_25_FSEL[1:0]	P0.25 (B)	SEG18 (O)	PWM2 (O)	–
48	IO0_26_FSEL[1:0]	P0.26 (B)	SEG19 (O)	USARTCLK3 (B)	EXI4 (I)
49	IO0_27_FSEL[1:0]	P0.27 (B)	SEG20 (O)	USARTRXD3 (I)	–
50	IO0_28_FSEL[1:0]	P0.28 (B)	SEG21 (O)	USARTTXD3 (O)	–
51	IO0_29_FSEL[1:0]	P0.29 (B)	SEG22 (O)	PHASEA1 (I)	–
52	IO0_30_FSEL[1:0]	P0.30 (B)	SEG23 (O)	PHASEB1 (I)	–
53	IO0_31_FSEL[1:0]	P0.31 (B)	SEG24 (O)	PHASEZ1 (I)	–

Table 16-2 GPIO 1 Function Mode Configuration

Pin Num.	Function Number	F0 (00)	F1 (01)	F2 (10)	F3 (11)
54	IO1_0_FSEL[1:0]	P1.0 (B)	SEG25 (O)	***SSPMOSI1*** (B)	–
55	IO1_1_FSEL[1:0]	P1.1 (B)	SEG26 (O)	***SSPMISO1*** (B)	–
56	IO1_2_FSEL[1:0]	P1.2 (B)	SEG27 (O)	***SSPCLK1*** (B)	–
57	IO1_3_FSEL[1:0]	P1.3 (B)	SEG28 (O)	***SSPFSS1*** (B)	–
58	IO1_4_FSEL[1:0]	P1.4 (B)	SEG29 (O)	***SCL1*** (B)	***SSPMOSI0*** (B)
59	IO1_5_FSEL[1:0]	P1.5 (B)	SEG30 (O)	***SDA1*** (B)	***SSPMISO0*** (B)
60	IO1_6_FSEL[1:0]	P1.6 (B)	SEG31 (O)	***CANRX1*** (I)	***SSPCLK0*** (B)
61	IO1_7_FSEL[1:0]	P1.7 (B)	SEG32 (O)	***CANTX1*** (O)	***SSPFSS0*** (B)
62	IO1_8_FSEL[1:0]	P1.8 (B)	SEG33 (O)	USARTCLK2 (B)	EXI5 (I)
63	IO1_9_FSEL[1:0]	P1.9 (B)	SEG34 (O)	USARTRXD2 (I)	PWM3 (O)
64	IO1_10_FSEL[1:0]	P1.10 (B)	SEG35 (O)	USARTTXD2 (O)	PWM4 (O)
65	IO1_11_FSEL[1:0]	P1.11 (B)	SEG36 (O)	TCLK0 (I)	PWM5 (O)
66	IO1_12_FSEL[1:0]	P1.12 (B)	SEG37 (O)	TCAP0 (I)	PWM6 (O)
67	IO1_13_FSEL[1:0]	P1.13 (B)	SEG38 (O)	TPWM0 (O)	PWM7 (O)
68	IO1_14_FSEL[1:0]	P1.14 (B)	SEG39 (O)	***USARTCLK1*** (B)	NMI (I)
69	IO1_15_FSEL[1:0]	P1.15 (B)	PWM1D2 (O)	–	–
70	IO1_16_FSEL[1:0]	P1.16 (B)	PWM1U2 (O)	–	–
71	IO1_17_FSEL[1:0]	P1.17 (B)	PWM1D1 (O)	–	–
72	IO1_18_FSEL[1:0]	P1.18 (B)	PWM1U1 (O)	–	–
73	IO1_19_FSEL[1:0]	P1.19 (B)	PWM1D0 (O)	–	–
74	IO1_20_FSEL[1:0]	P1.20 (B)	PWM1U0 (O)	–	–
75	IO1_21_FSEL[1:0]	P1.21 (B)	PWM1OFF (I)	–	–
76	IO1_22_FSEL[1:0]	P1.22 (B)	ADTRG0 (I)	EXI6 (I)	–
79	IO1_23_FSEL[1:0]	P1.23 (B)	OP0_O (O)	***USARTRXD1*** (I)	–
80	IO1_24_FSEL[1:0]	P1.24 (B)	OP0_N (I)	***USARTTXD1*** (O)	–
81	IO1_25_FSEL[1:0]	P1.25 (B)	OP0_P (I)	***SDA0*** (B)	***SSPMOSI1*** (B)
82	IO1_26_FSEL[1:0]	P1.26 (B)	OP1_O (O)	***SCL0*** (B)	***SSPMISO1*** (B)
83	IO1_27_FSEL[1:0]	P1.27 (B)	OP1_N (I)	***CANRX0*** (I)	***SSPCLK1*** (B)
84	IO1_28_FSEL[1:0]	P1.28 (B)	OP1_P (I)	***CANTX0*** (O)	***SSPFSS1*** (B)
85	IO1_29_FSEL[1:0]	P1.29 (B)	OP2_O (O)	***USARTCLK0*** (B)	ADTRG1
86	IO1_30_FSEL[1:0]	P1.30 (B)	OP2_N (I)	***USARTRXD0*** (I)	–
87	IO1_31_FSEL[1:0]	P1.31 (B)	OP2_P (I)	***USARTTXD0*** (O)	–

Table 16-3 GPIO 2 Function Mode Configuration

Pin Num.	Function Number	F0 (00)	F1 (01)	F2 (10)	F3 (11)
88	IO2_0_FSEL[1:0]	P2.0 (B)	AIN10 (I)	OP3_O (O)	EXI7 (I)
89	IO2_1_FSEL[1:0]	P2.1 (B)	AIN11 (I)	OP3_N (I)	EXI8 (I)
90	IO2_2_FSEL[1:0]	P2.2 (B)	AIN12 (I)	OP3_P (I)	–
91	IO2_3_FSEL[1:0]	P2.3 (B)	AIN13 (I)	–	–
92	IO2_4_FSEL[1:0]	P2.4 (B)	AIN14 (I)	–	–
93	IO2_5_FSEL[1:0]	P2.5 (B)	AIN15 (I)	–	–
94	IO2_6_FSEL[1:0]	P2.6 (B)	AIN16 (I)	–	–
95	IO2_7_FSEL[1:0]	P2.7 (B)	AIN17 (I)	–	–
102	IO2_8_FSEL[1:0]	P2.8 (B)	AIN01 (I)	–	–
103	IO2_9_FSEL[1:0]	P2.9 (B)	AIN02 (I)	–	–
104	IO2_10_FSEL[1:0]	P2.10 (B)	AIN03 (I)	–	–
105	IO2_11_FSEL[1:0]	P2.11 (B)	AIN04 (I)	–	–
106	IO2_12_FSEL[1:0]	P2.12 (B)	AIN05 (I)	–	–
107	IO2_13_FSEL[1:0]	P2.13 (B)	AIN06 (I)	OP4_O (O)	EXI9 (I)
108	IO2_14_FSEL[1:0]	P2.14 (B)	AIN07 (I)	OP4_N (I)	EXI10 (I)
109	IO2_15_FSEL[1:0]	P2.15 (B)	AIN08 (I)	OP4_P (I)	EXI11 (I)
112	IO2_16_FSEL[1:0]	P2.16 (B)	PWM0OFF (I)	–	–
113	IO2_17_FSEL[1:0]	P2.17 (B)	PWM0U0 (O)	–	–
114	IO2_18_FSEL[1:0]	P2.18 (B)	PWM0D0 (O)	–	–
115	IO2_19_FSEL[1:0]	P2.19 (B)	PWM0U1 (O)	–	–
116	IO2_20_FSEL[1:0]	P2.20 (B)	PWM0D1 (O)	–	–
117	IO2_21_FSEL[1:0]	P2.21 (B)	PWM0U2 (O)	–	–
118	IO2_22_FSEL[1:0]	P2.22 (B)	PWM0D2 (O)	–	–
119	IO2_23_FSEL[1:0]	P2.23 (B)	PHASEA0 (I)	–	–
120	IO2_24_FSEL[1:0]	P2.24 (B)	PHASEB0 (I)	–	–
121	IO2_25_FSEL[1:0]	P2.25 (B)	PHASEZ0 (I)	–	–
122	IO2_26_FSEL[1:0]	P2.26 (B)	***SCL1*** (B)	EXI12 (I)	–
123	IO2_27_FSEL[1:0]	P2.27 (B)	***SDA1*** (B)	EXI13 (I)	–

Table 16-4 GPIO 3 Function Mode Configuration

Pin Num.	Function Number	F0 (00)	F1 (01)	F2 (10)	F3 (11)
124	IO3_0_FSEL[1:0]	P3.0 (B)	TRACED3 (O)	–	–
125	IO3_1_FSEL[1:0]	P3.1 (B)	TRACED2 (O)	–	–
126	IO3_2_FSEL[1:0]	P3.2 (B)	TRACED1 (O)	–	–
127	IO3_3_FSEL[1:0]	P3.3 (B)	TRACED0 (O)	–	–
128	IO3_4_FSEL[1:0]	P3.4 (B)	TRACECLK (O)	–	–
1	IO3_5_FSEL[1:0]	P3.5 (B)	nTRST (I)	–	–
2	IO3_6_FSEL[1:0]	P3.6 (B)	TDO (O) TRACESWO (O)	–	–
3	IO3_7_FSEL[1:0]	P3.7 (B)	TDI (I)	–	–

4	IO3_8_FSEL[1:0]	P3.8 (B)	TMS(I)
			SWDIO(B)
				–	–

5	IO3_9_FSEL[1:0]	P3.9 (B)	TCK(I) SWCLK (I)	–	–

16.3 Register Description

16.3.1 Register Map Summary

- Base Address: 0x4005_8000

Register	Offset	Description	Reset Value
IOCONF_P0MLR	0x0000	PORT 0 Mode Low Register	0x0000_0000
IOCONF_P0MHR	0x0004	PORT 0 Mode High Register	0x0000_0000
IOCONF_P0PUR	0x0008	PORT 0 Pull-Up Control Register	0x0000_0000
IOCONF_P0ODCR	0x000C	PORT 0 Open-Drain Control Register	0x0000_0000
IOCONF_P1MLR	0x0010	PORT 1 Mode Low Register	0x0000_0000
IOCONF_P1MHR	0x0014	PORT 1 Mode High Register	0x0000_0000
IOCONF_P1PUR	0x0018	PORT 1 Pull-Up Control Register	0x0000_0000
IOCONF_P1ODCR	0x001C	PORT 1 Open-Drain Control Register	0x0000_0000
IOCONF_P2MLR	0x0020	PORT 2 Mode Low Register	0x0000_0000
IOCONF_P2MHR	0x0024	PORT 2 Mode High Register	0x0000_0000
IOCONF_P2PUR	0x0028	PORT 2 Pull-Up Control Register	0x0000_0000
IOCONF_P2ODCR	0x002C	PORT 2 Open-Drain Control Register	0x0000_0000
IOCONF_P3MLR	0x0030	PORT 3 Mode Low Register	0x0005_5400
RSVD	0x0034	Reserved	–
IOCONF_P3PUR	0x0038	PORT 3 Pull-Up Control Register	0x0000_03A0
IOCONF_P3ODCR	0x003C	PORT 3 Open-Drain Control Register	0x0000_0000

16.3.1.1 IOCONF_P0MLR (PORT 0 Mode Low Register)

- Address = Base Address + 0x0000, Reset Value = 0x0000_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
IO0_15_FSEL		IO0_14_FSEL		IO0_13_FSEL		IO0_12_FSEL		IO0_11_FSEL		IO0_10_FSEL		IO0_9_FSEL		IO0_8_FSEL		IO0_7_FSEL		IO0_6_FSEL		IO0_5_FSEL		IO0_4_FSEL		IO0_3_FSEL		IO0_2_FSEL		IO0_1_FSEL		IO0_0_FSEL	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO0_n_FSEL[1:0]	[31:0]	RW	GPIO0.n Function Selection (n = 0 ~ 15) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-1* for more detail	0x0000_0000

16.3.1.2 IOCONF_P0MHR (PORT 0 Mode High Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
IO0_31_FSEL		IO0_30_FSEL		IO0_29_FSEL		IO0_28_FSEL		IO0_27_FSEL		IO0_26_FSEL		IO0_25_FSEL		IO0_24_FSEL		IO0_23_FSEL		IO0_22_FSEL		IO0_21_FSEL		IO0_20_FSEL		IO0_19_FSEL		IO0_18_FSEL		IO0_17_FSEL		IO0_16_FSEL	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO0_n_FSEL[1:0]	[31:0]	RW	GPIO0.n Function Selection (n = 16 ~ 31) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-1* for more detail	0x0000_0000

16.3.1.3 IOCONF_P0PUR (PORT 0 Pull-up Control Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
IO0_31_PUEN	IO0_30_PUEN	IO0_29_PUEN	IO0_28_PUEN	IO0_27_PUEN	IO0_26_PUEN	IO0_25_PUEN	IO0_24_PUEN	IO0_23_PUEN	IO0_22_PUEN	IO0_21_PUEN	IO0_20_PUEN	IO0_19_PUEN	IO0_18_PUEN	IO0_17_PUEN	IO0_16_PUEN	IO0_15_PUEN	IO0_14_PUEN	IO0_13_PUEN	IO0_12_PUEN	IO0_11_PUEN	IO0_10_PUEN	IO0_9_PUEN	IO0_8_PUEN	IO0_7_PUEN	IO0_6_PUEN	IO0_5_PUEN	IO0_4_PUEN	IO0_3_PUEN	IO0_2_PUEN	IO0_1_PUEN	IO0_0_PUEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO0_n_PUEN	[31:0]	RW	GPIO0.n Pull-Up Enable/Disable (n = 0 ~ 31) 0 = Disable Pull-Up resistor 1 = Enable Pull-Up resistor	0x0000_0000

16.3.1.4 IOCONF_P0ODCR (PORT 0 Open-Drain Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
IO0_31_ODEN	IO0_30_ODEN	IO0_29_ODEN	IO0_28_ODEN	IO0_27_ODEN	IO0_26_ODEN	IO0_25_ODEN	IO0_24_ODEN	IO0_23_ODEN	IO0_22_ODEN	IO0_21_ODEN	IO0_20_ODEN	IO0_19_ODEN	IO0_18_ODEN	IO0_17_ODEN	IO0_16_ODEN	IO0_15_ODEN	IO0_14_ODEN	IO0_13_ODEN	IO0_12_ODEN	IO0_11_ODEN	IO0_10_ODEN	IO0_9_ODEN	IO0_8_ODEN	IO0_7_ODEN	IO0_6_ODEN	IO0_5_ODEN	IO0_4_ODEN	IO0_3_ODEN	IO0_2_ODEN	IO0_1_ODEN	IO0_0_ODEN
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
IO0_n_ODEN	[31:0]	RW	GPIO0.n Open-Drain Enable/Disable (n = 0 ~ 31) 0 = Disable Open-Drain 1 = Enable Open-Drain	0x0000_0000

If Open-Drain is enabled on corresponding pin, it drives only “LOW” level. A pull-up resistor needs to ensure the level of the pin (HIGH) when it is not drive

16.3.1.5 IOCONF_P1MLR (PORT 1 Mode Low Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
IO1_15_FSEL		IO1_14_FSEL		IO1_13_FSEL		IO1_12_FSEL		IO1_11_FSEL		IO1_10_FSEL		IO1_9_FSEL		IO1_8_FSEL		IO1_7_FSEL		IO1_6_FSEL		IO1_5_FSEL		IO1_4_FSEL		IO1_3_FSEL		IO1_2_FSEL		IO1_1_FSEL		IO1_0_FSEL	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO0_n_FSEL[1:0]	[31:0]	RW	GPIO1.n Function Selection (n = 0 ~ 15) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-2* for more detail	0x0000_0000

16.3.1.6 IOCONF_P1MHR (PORT 1 Mode High Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
IO1_31_FSEL		IO1_30_FSEL		IO1_29_FSEL		IO1_28_FSEL		IO1_27_FSEL		IO1_26_FSEL		IO1_25_FSEL		IO1_24_FSEL		IO1_23_FSEL		IO1_22_FSEL		IO1_21_FSEL		IO1_20_FSEL		IO1_19_FSEL		IO1_18_FSEL		IO1_17_FSEL		IO1_16_FSEL	
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
IO1_n_FSEL[1:0]	[31:0]	RW	GPIO1.n Function Selection (n = 16 ~ 31) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-2* for more detail	0x0000_0000

16.3.1.7 IOCONF_P1PUR (PORT 1 Pull-up Control Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
IO1_31_PUEN	IO1_30_PUEN	IO1_29_PUEN	IO1_28_PUEN	IO1_27_PUEN	IO1_26_PUEN	IO1_25_PUEN	IO1_24_PUEN	IO1_23_PUEN	IO1_22_PUEN	IO1_21_PUEN	IO1_20_PUEN	IO1_19_PUEN	IO1_18_PUEN	IO1_17_PUEN	IO1_16_PUEN	IO1_15_PUEN	IO1_14_PUEN	IO1_13_PUEN	IO1_12_PUEN	IO1_11_PUEN	IO1_10_PUEN	IO1_9_PUEN	IO1_8_PUEN	IO1_7_PUEN	IO1_6_PUEN	IO1_5_PUEN	IO1_4_PUEN	IO1_3_PUEN	IO1_2_PUEN	IO1_1_PUEN	IO1_0_PUEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO1_n_PUEN	[31:0]	RW	GPIO**1**.n Pull-Up Enable/Disable (n = 0 ~ 31) 0 = Disable Pull-Up resistor 1 = Enable Pull-Up resistor	0x0000_0000

16.3.1.8 IOCONF_P1ODCR (PORT 1 Open-Drain Control Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
IO1_31_ODEN	IO1_30_ODEN	IO1_29_ODEN	IO1_28_ODEN	IO1_27_ODEN	IO1_26_ODEN	IO1_25_ODEN	IO1_24_ODEN	IO1_23_ODEN	IO1_22_ODEN	IO1_21_ODEN	IO1_20_ODEN	IO1_19_ODEN	IO1_18_ODEN	IO1_17_ODEN	IO1_16_ODEN	IO1_15_ODEN	IO1_14_ODEN	IO1_13_ODEN	IO1_12_ODEN	IO1_11_ODEN	IO1_10_ODEN	IO1_9_ODEN	IO1_8_ODEN	IO1_7_ODEN	IO1_6_ODEN	IO1_5_ODEN	IO1_4_ODEN	IO1_3_ODEN	IO1_2_ODEN	IO1_1_ODEN	IO1_0_ODEN
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
IO1_n_ODEN	[31:0]	RW	GPIO1.n Open-Drain Enable/Disable (n = 0 ~ 31) 0 = Disable Open-Drain 1 = Enable Open-Drain	0x0000_0000

If Open-Drain is enabled on corresponding pin, it drives only “LOW” level. A pull-up resistor needs to ensure the level of the pin (HIGH) when it is not drive

16.3.1.9 IOCONF_P2MLR (PORT 2 Mode Low Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
IO2_15_FSEL		IO2_14_FSEL		IO2_13_FSEL		IO2_12_FSEL		IO2_11_FSEL		IO2_10_FSEL		IO2_9_FSEL		IO2_8_FSEL		IO2_7_FSEL		IO2_6_FSEL		IO2_5_FSEL		IO2_4_FSEL		IO2_3_FSEL		IO2_2_FSEL		IO2_1_FSEL		IO2_0_FSEL	
0	0	0	0	0	0	0	0	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

Name	Bit	Type	Description	Reset Value
IO2_n_FSEL[1:0]	[31:0]	RW	GPIO2.n Function Selection (n = 0 ~ 15) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-3* for more detail	0x0000_0000

16.3.1.10 IOCONF_P2MHR (PORT 2 Mode High Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD								IO2_27_FSEL		IO2_26_FSEL		IO2_25_FSEL		IO2_24_FSEL		IO2_23_FSEL		IO2_22_FSEL		IO2_21_FSEL		IO2_20_FSEL		IO2_19_FSEL		IO2_18_FSEL		IO2_17_FSEL		IO2_16_FSEL	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO2_n_FSEL[1:0]	[23:0]	RW	GPIO2.n Function Selection (n = 16 ~ 27) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-3* for more detail	0x0000_0000

16.3.1.11 IOCONF_P2PUR (PORT 2 Pull-up Control Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD				IO2_27_PUEN	IO2_26_PUEN	IO2_25_PUEN	IO2_24_PUEN	IO2_23_PUEN	IO2_22_PUEN	IO2_21_PUEN	IO2_20_PUEN	IO2_19_PUEN	IO2_18_PUEN	IO2_17_PUEN	IO2_16_PUEN	IO2_15_PUEN	IO2_14_PUEN	IO2_13_PUEN	IO2_12_PUEN	IO2_11_PUEN	IO2_10_PUEN	IO2_9_PUEN	IO2_8_PUEN	IO2_7_PUEN	IO2_6_PUEN	IO2_5_PUEN	IO2_4_PUEN	IO2_3_PUEN	IO2_2_PUEN	IO2_1_PUEN	IO2_0_PUEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	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

Name	Bit	Type	Description	Reset Value
IO2_n_PUEN	[27:0]	RW	GPIO2.n Pull-Up Enable/Disable (n = 0 ~ 27) 0 = Disable Pull-Up resistor 1 = Enable Pull-Up resistor	0x0000_0000

16.3.1.12 IOCONF_P2ODCR (PORT 2 Open-Drain Control Register)

- Address = Base Address + 0x02C, Reset Value = 0x0000_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
RSVD				IO2_27_ODEN	IO2_26_ODEN	IO2_25_ODEN	IO2_24_ODEN	IO2_23_ODEN	IO2_22_ODEN	IO2_21_ODEN	IO2_20_ODEN	IO2_19_ODEN	IO2_18_ODEN	IO2_17_ODEN	IO2_16_ODEN	IO2_15_ODEN	IO2_14_ODEN	IO2_13_ODEN	IO2_12_ODEN	IO2_11_ODEN	IO2_10_ODEN	IO2_9_ODEN	IO2_8_ODEN	IO2_7_ODEN	IO2_6_ODEN	IO2_5_ODEN	IO2_4_ODEN	IO2_3_ODEN	IO2_2_ODEN	IO2_1_ODEN	IO2_0_ODEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	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

Name	Bit	Type	Description	Reset Value
IO2_n_ODEN	[27:0]	RW	GPIO**2**.n Open-Drain Enable/Disable (n = 0 ~ 27) 0 = Disable Open-Drain 1 = Enable Open-Drain	0x0000_0000

If Open-Drain is enabled on corresponding pin, it drives only “LOW” level. A pull-up resistor needs to ensure the level of the pin (HIGH) when it is not drive

16.3.1.13 IOCONF_P3MLR (PORT 3 Mode Low Register)

- Address = Base Address + 0x030, Reset Value = 0x0005_5400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												IO3_9_FSEL		IO3_8_FSEL		IO3_7_FSEL		IO3_6_FSEL		IO3_5_FSEL		IO3_4_FSEL		IO3_3_FSEL		IO3_2_FSEL		IO3_1_FSEL		IO3_0_FSEL	
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
IO3_n_FSEL[1:0]	[19:0]	RW	GPIO3.n Function Selection (n = 0 ~ 9) 00 = Function 0 (GPIO) 01 = Function 1 10 = Function 2 11 = Function 3 The defined functions for each pin are different. Refer to the *Table 16-4* for more detail	0x5_5400

16.3.1.14 IOCONF_P3PUR (PORT 3 Pull-up Control Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_03A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																						IO3_9_PUEN	IO3_8_PUEN	IO3_7_PUEN	IO3_6_PUEN	IO3_5_PUEN	IO3_4_PUEN	IO3_3_PUEN	IO3_2_PUEN	IO3_1_PUEN	IO3_0_PUEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IO3_n_PUEN	[9:0]	RW	GPIO3.n Pull-Up Enable/Disable (n = 0 ~ 9) 0 = Disable Pull-Up resistor 1 = Enable Pull-Up resistor	0x3A0

16.3.1.15 IOCONF_P3ODCR (PORT 3 Open-Drain Control Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																						IO3_9_ODEN	IO3_8_ODEN	IO3_7_ODEN	IO3_6_ODEN	IO3_5_ODEN	IO3_4_ODEN	IO3_3_ODEN	IO3_2_ODEN	IO3_1_ODEN	IO3_0_ODEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
IO3_n_ODEN	[9:0]	RW	GPIO3.n Open-Drain Enable/Disable (n = 0 ~ 9) 0 = Disable Open-Drain 1 = Enable Open-Drain	0x000

If Open-Drain is enabled on corresponding pin, it drives only “LOW” level. A pull-up resistor needs to ensure the level of the pin (HIGH) when it is not drive

17 LCD Controller

17.1 Overview

The LCD driver controller has 4-com x 40-segment drivers so that a maximum of 160 LCD segments (panel) are controllable. Each segment is controlled by a corresponding bit in the LCD Display Memory Register.

17.1.1 Features

The following is the distinctive features that are described in detail in this user's manual:

- Four common output pins COM[3:0]
- Forty segment output pins SEG[39:0]
- LCD bias by internal or external resistor
- Supports bias and duty mode for each corresponding operation
- Programmable frame clock generator
- Display Memory Registers contain the data to be displayed on the LCD

17.1.2 Pin Description

Table 17-1 Pin Description

Pin Name	Function	I/O Type	Note
COM[3:0]	Appropriate voltage level for COM driver signal	O	–
SEG[39:0]	Appropriate voltage level for SEG driver signal	O	–
VLCD[3:1]	LCD bias circuit pin	I/O	–

17.2 Functional Description

17.2.1 Block Diagram

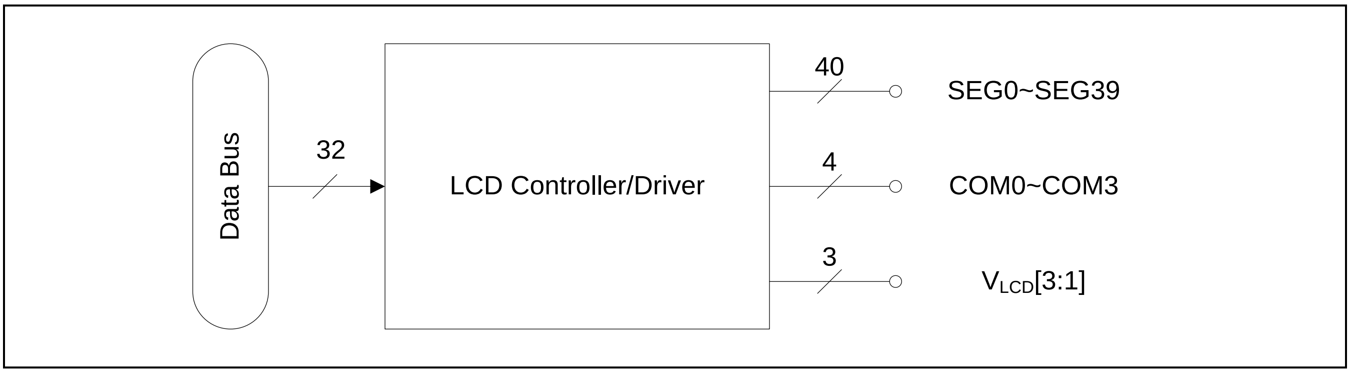


Figure 17-1 LCD Controller Top Block Diagram

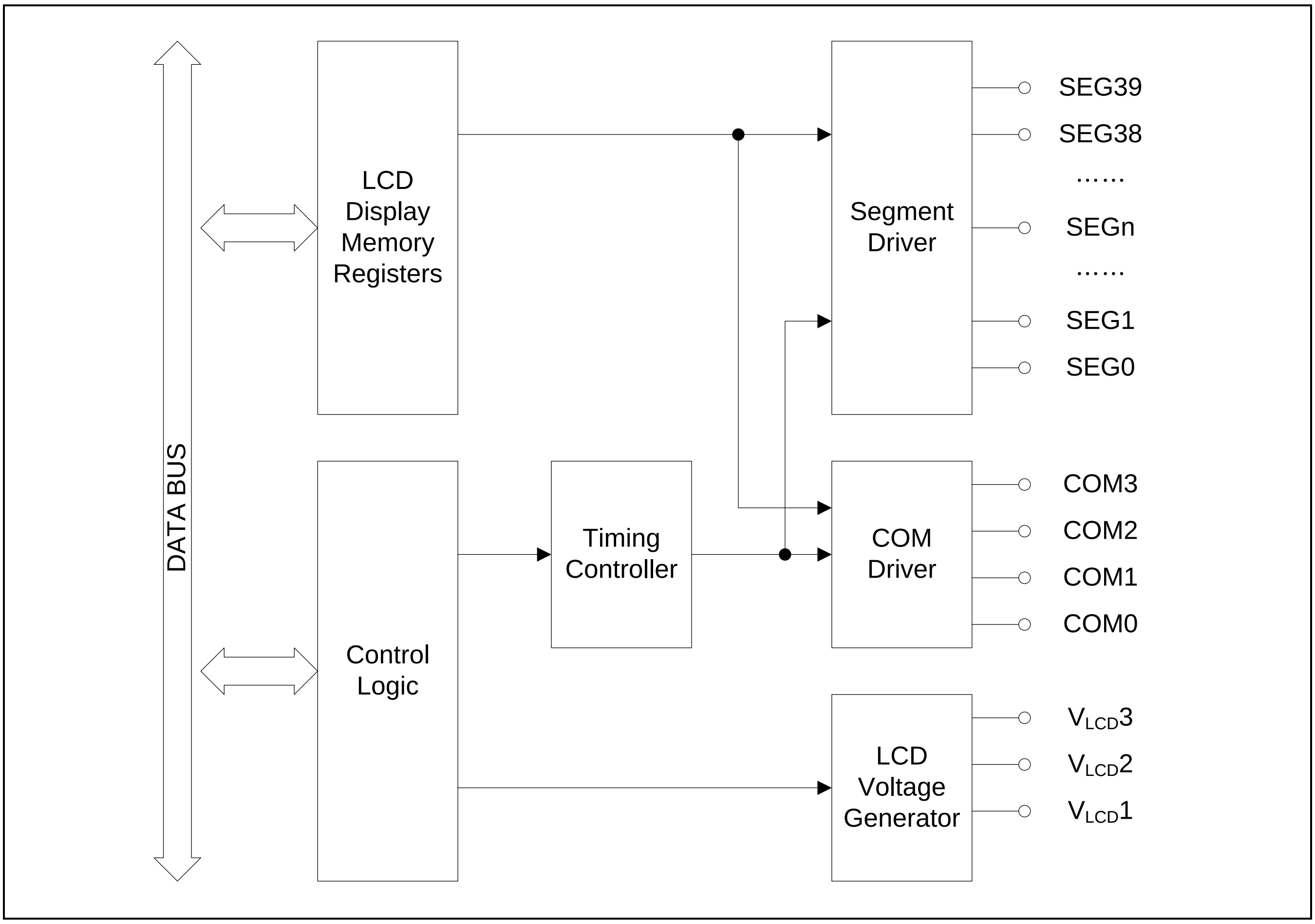


Figure 17-2 LCD Controller Block Diagram

This section provides a complete functional description of the LCD controller, detailing the operation of the design from the end user perspective in a number of subsections.

Many functions as LCD Display Memory, Clock Generator, Timing Controller, COM Driver, SEG Driver and LCD Voltage Generator will allow the user to configure the controller for a specific application.

The pins from COM0 to COM3 are used as LCD common output pins. This output signals represent the analog COM waveforms of the LCD module and are connected directly to the corresponding pins.

The pins SEG0 to SEG39 are used as LCD segment output pins. The output signals represent the analog SEG waveforms of the LCD module and are connected directly to the corresponding pins.

17.2.2 LCD Display Memory

The waveforms generated are dependent on the state (ON or OFF) of the LCD segments as defined in the LCD display memory. If each bit of display memory register is set to ‘1’, the segment output signal is converted to the selected voltage (LCD displayed). If each bit of display memory is set to ‘0’, the segment output signal is converted to the non-select voltage (LCD not displayed) and then output.

Table 17-2 4COM x 40SEG Display Memory Organization

31	30 – 5	4	COM3	COM2	COM1	COM0
–	–	–	SEG0			
–	–	–	SEG1			
–	–	–	…			
–	–	–	SEG38			
–	–	–	SEG39			

17.2.3 LCD Clock Generator and Timing Controller

The LCD controller needs the target clock signal for proper operation. This module generates clock signal from the ESCLK, EMCLK, ISCLK, IMCLK or PLLCLK. This module generates pre-selection signal according to the pre-defined LCD frequency and the bias-duty operating mode.

$F_{LCDFREQ}$ = F_{LCDCLK} /2CDIV[3:0] / (CPRE[15:0] +1)

$F_{LCDFREQ}$ means LCD Frequency

F_{LCDCLK} means LCD input clock (See Clock Manager block)

17.2.4 LCD Voltage Driving Method

Voltage Generator based on positive supply voltage applied V_{LCD}, it generates the voltage levels for the timing and control logic to produce the COM and SEG waveforms.

The LCD display is turned on only when the voltage difference between the common and segment signals is greater than V_{LCD}. The LCD display is turned off when the difference between the common and segment signal voltages is less than V_{LCD}. The turn-on voltage, + V_{LCD} or - V_{LCD}, is generated only when both signals are the selected signals of the bias.

Table 17-3 shows LCD drive voltages for static, 1/2 bias, and 1/3 bias.

Table 17-3 LCD Drive voltage Values

LCD Power Supply	1/3 Bias	1/2 Bias	Static Mode
VLCD1	V_{LCD}	V_{LCD}	V_{LCD}
VLCD2	2/3 V_{LCD}	V_{LCD}	V_{LCD}
VLCD3	1/3 V_{LCD}	1/2 V_{LCD}	V_{LCD}
V_{SS}	0V	0V	0V

The LCD panel display may be deteriorated if a DC voltage is applied that lies between the common and segment signal voltage. Therefore, always drive the LCD panel with AC voltage.

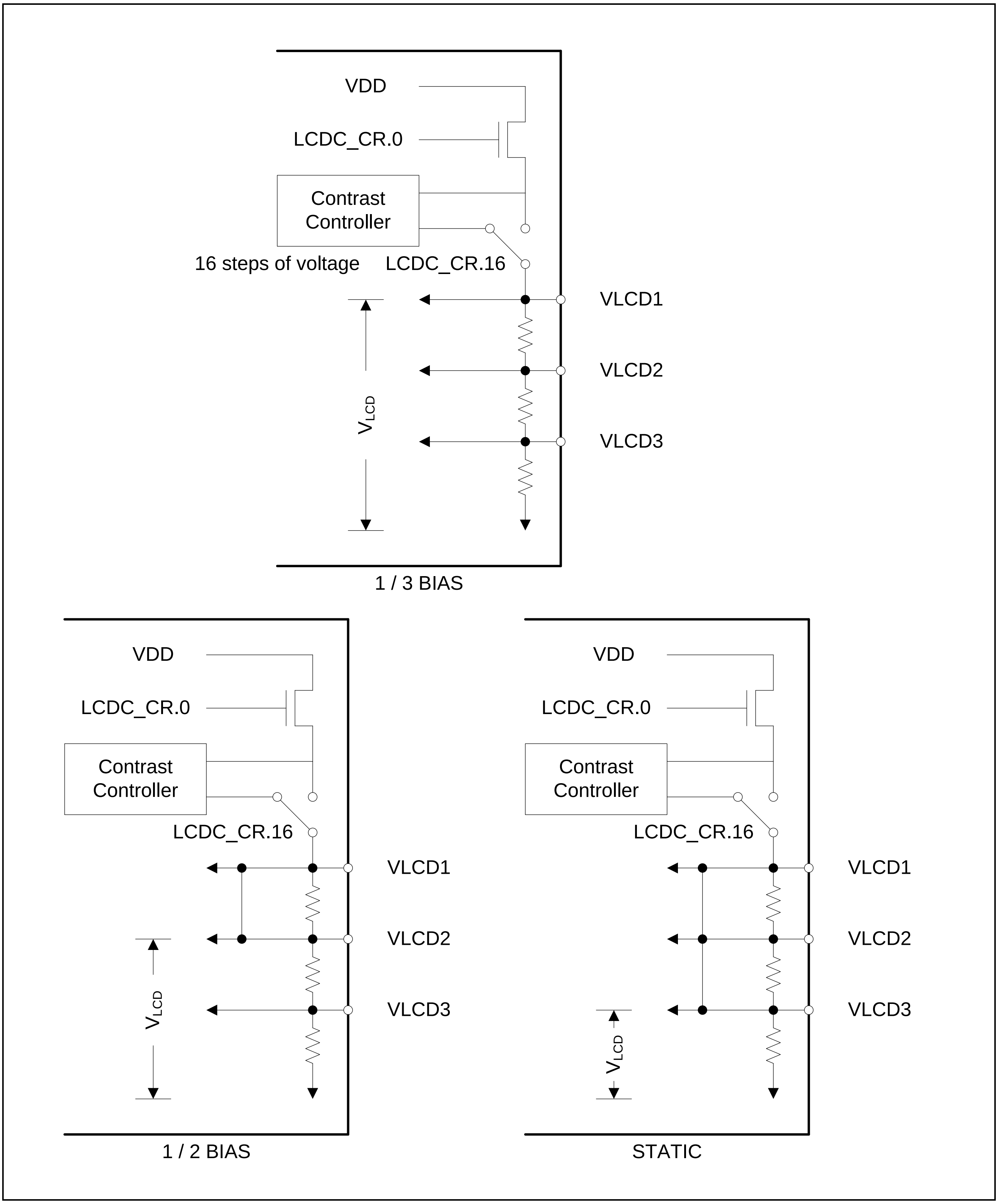


Figure 17-3 Voltage Dividing Internal Resistor Circuit Diagram

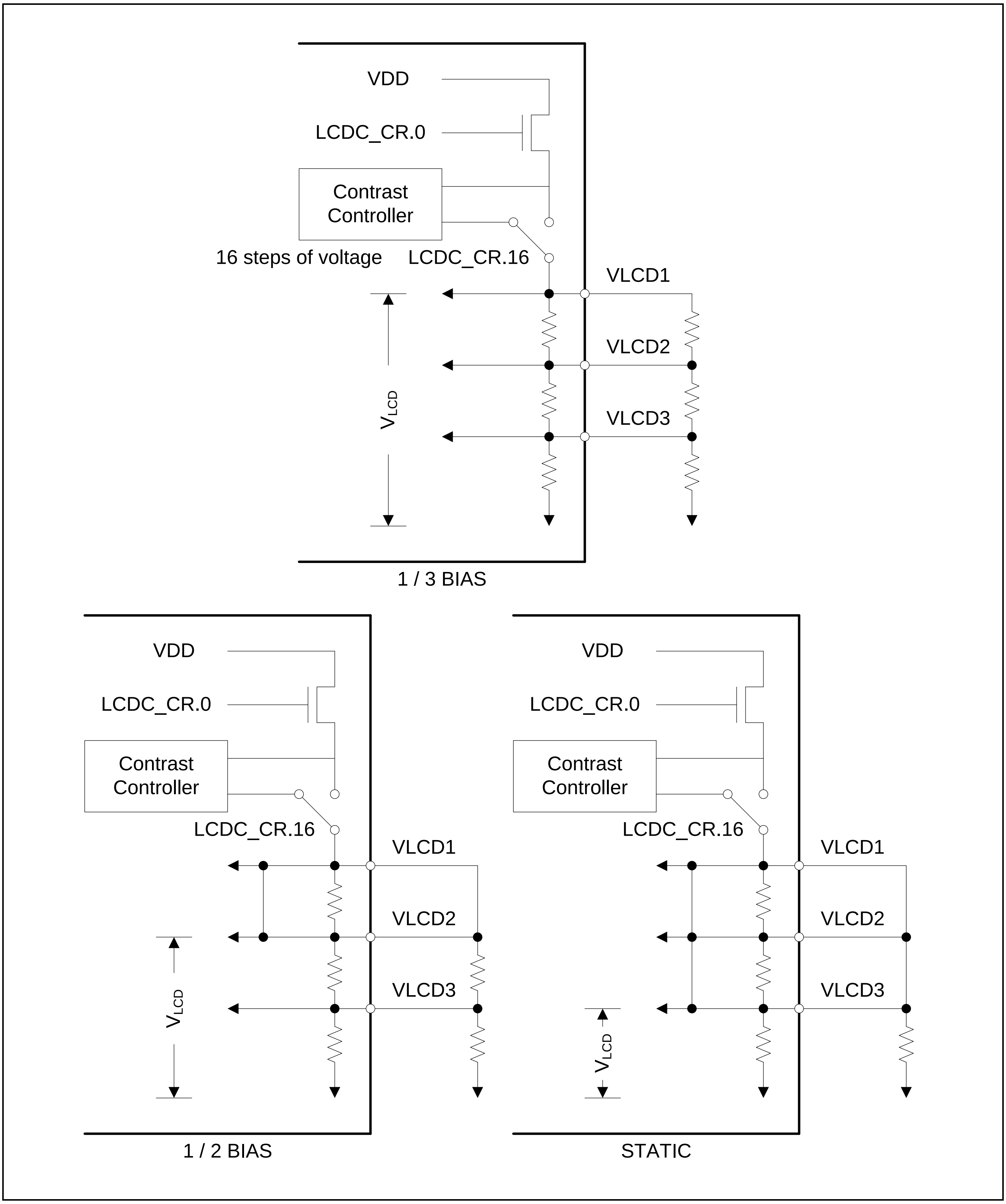


Figure 17-4 Voltage Dividing External Resistor Circuit Diagram

17.2.5 LCD Output Signal Waveform

If the LCD is supplied with DC power, the LCD element undergoes a chemical change causing a deterioration of the element. Therefore, the LCD controller/driver has a built-in AC circuit to drive the. The LCD controller supports five operation modes with different numbers of com and different biasing levels. There are six types of output waveform (5 modes of operation):

- 1/1 Duty (1 COM), 1/1 Bias
- 1/2 Duty (2 COM), 1/2 Bias
- 1/3 Duty (3 COM), 1/2 Bias
- 1/3 Duty (3 COM), 1/3 Bias
- 1/4 Duty (4 COM), 1/3 Bias

17.2.5.1 Static Mode

The AC timing diagram of COM/SEG signals under static mode is described in *Figure 17-5*.

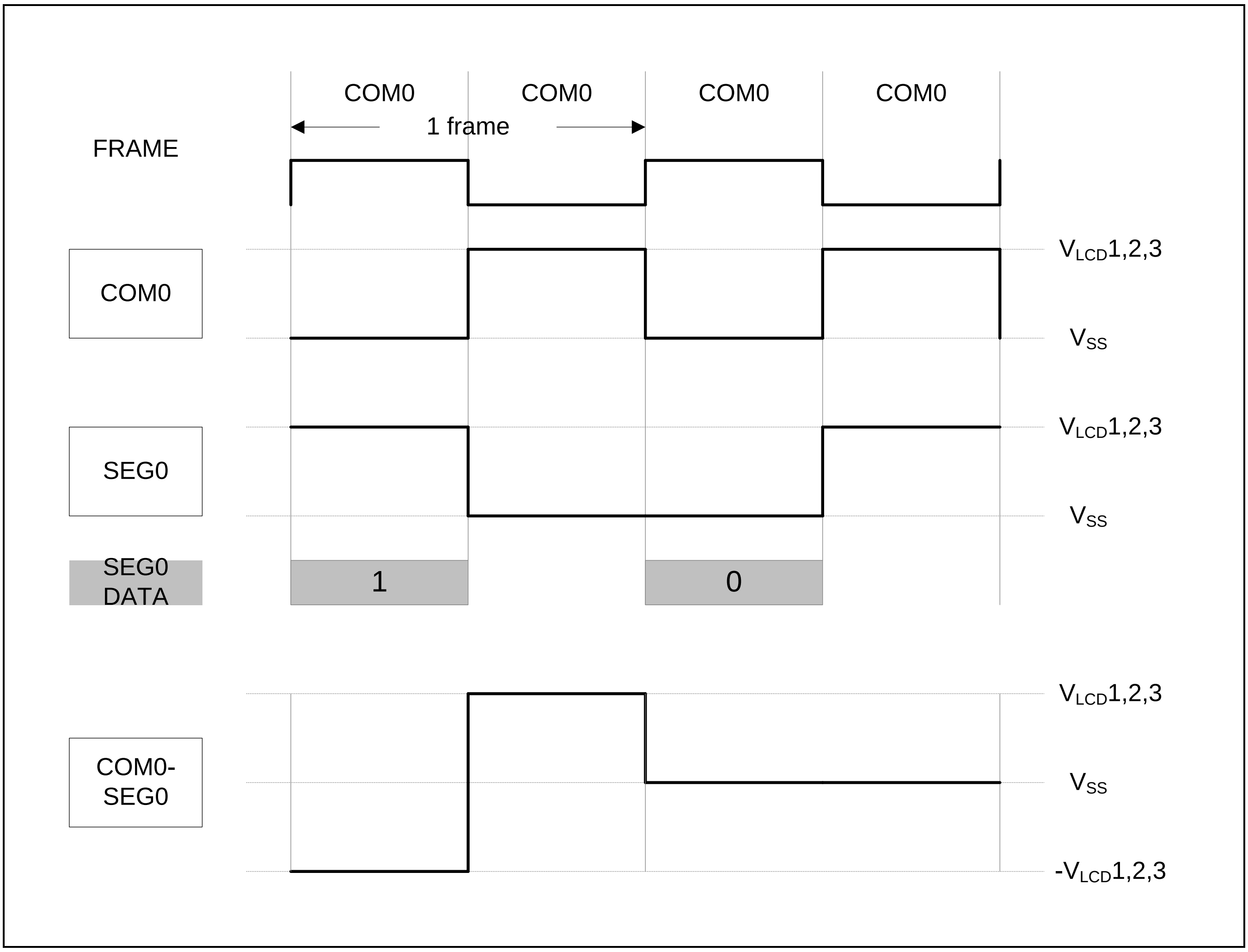


Figure 17-5 COM/SEG Signal in Static Mode

17.2.5.2 1/2 Duty, 1/2 Bias Mode

The AC Timing diagram of COM/SEG signals under 1/2 duty 1/2 bias mode is described in *Figure 17-6*.

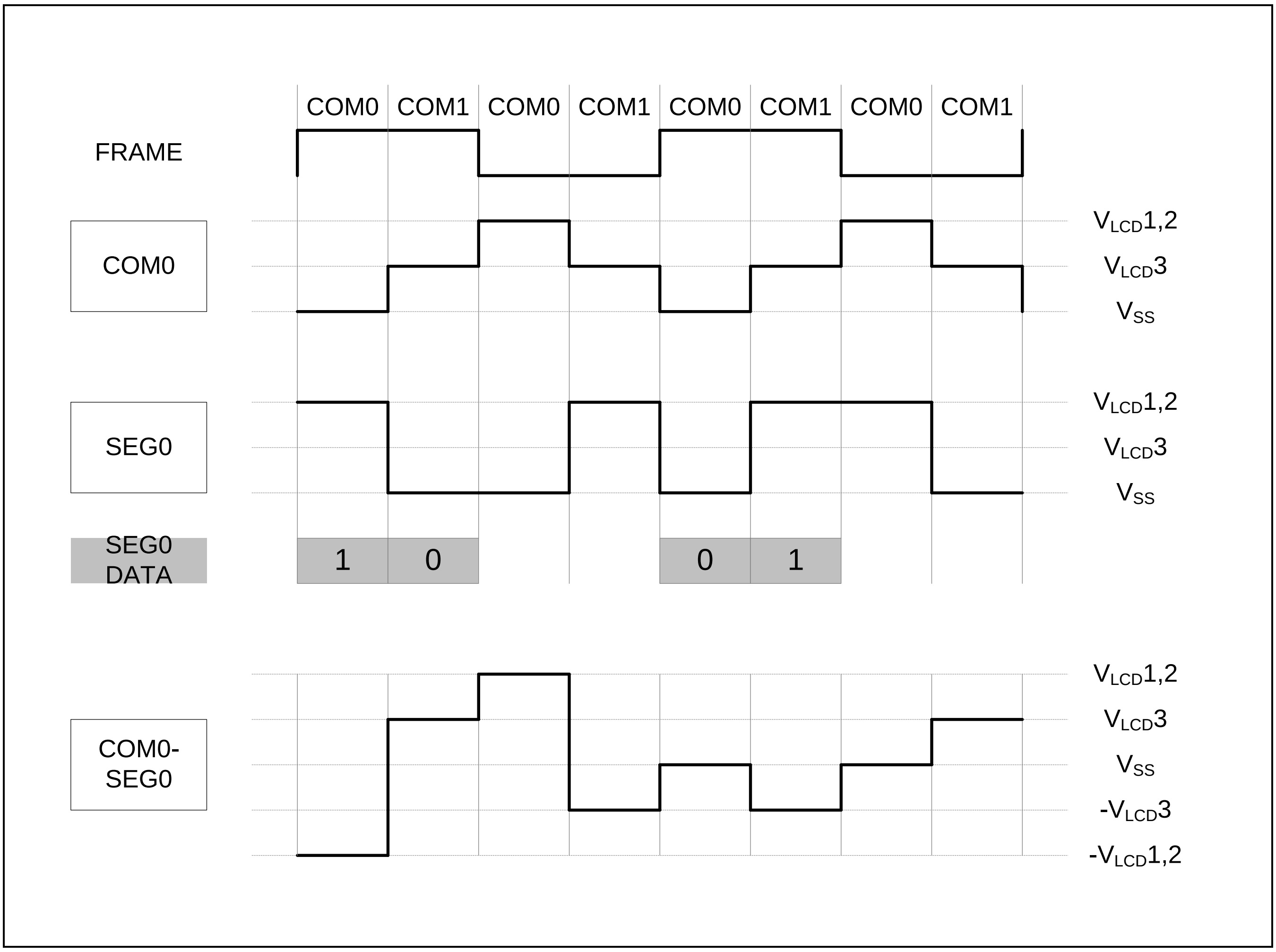


Figure 17-6 COM/SEG Signal in 1/2Duty and 1/2Bias Mode

17.2.5.3 1/3 Duty, 1/2 Bias Mode

The AC timing diagram of COM/SEG signals under 1/3 duty 1/2 bias mode is described in *Figure 17-7*.

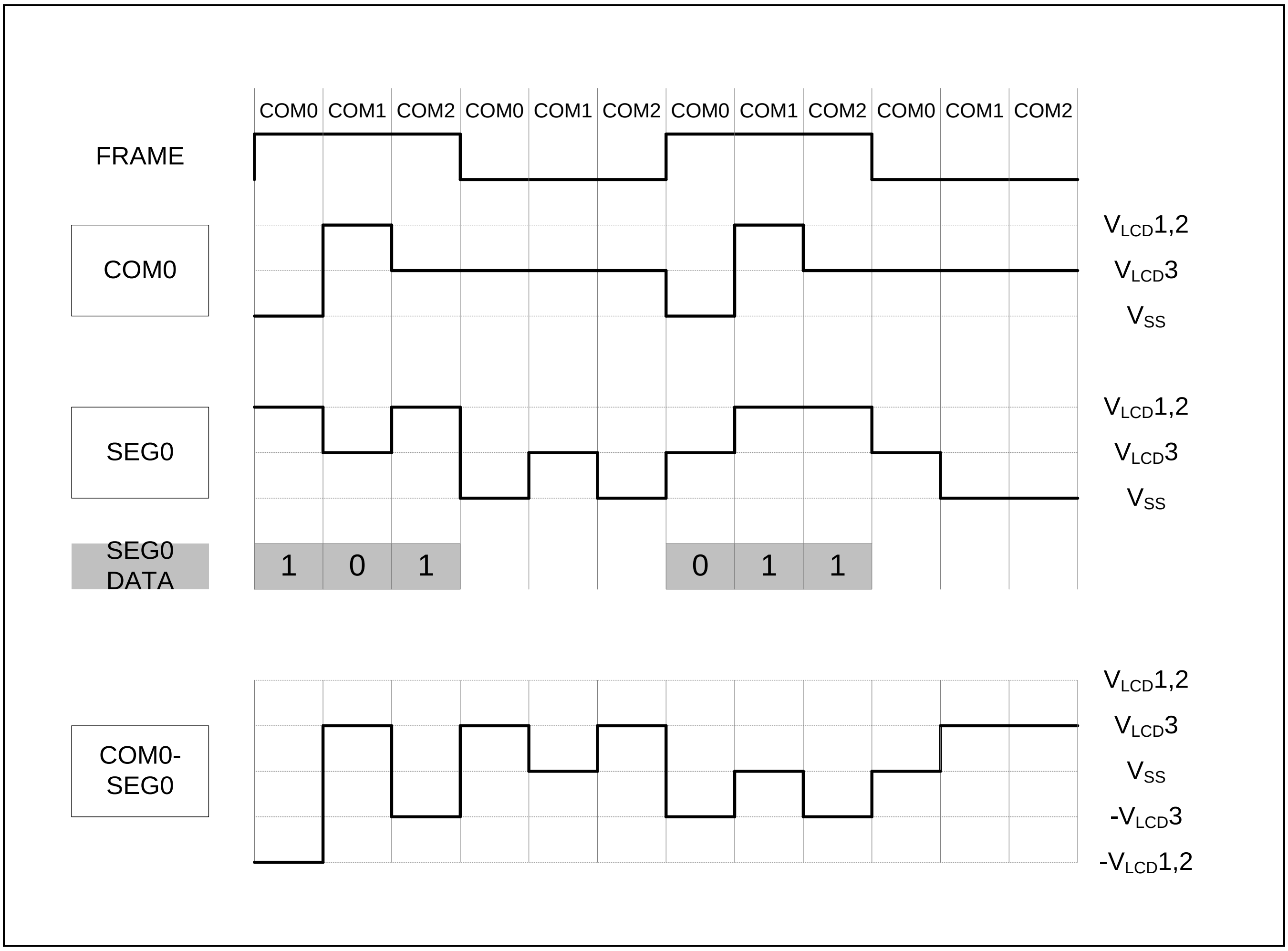


Figure 17-7 COM/SEG Signal in 1/3Duty and 1/2Bias Mode

17.2.5.4 1/3 Duty, 1/3 Bias Mode

The AC timing diagram of COM/SEG signals under 1/3 duty 1/3 bias mode is described in *Figure 17-8*.

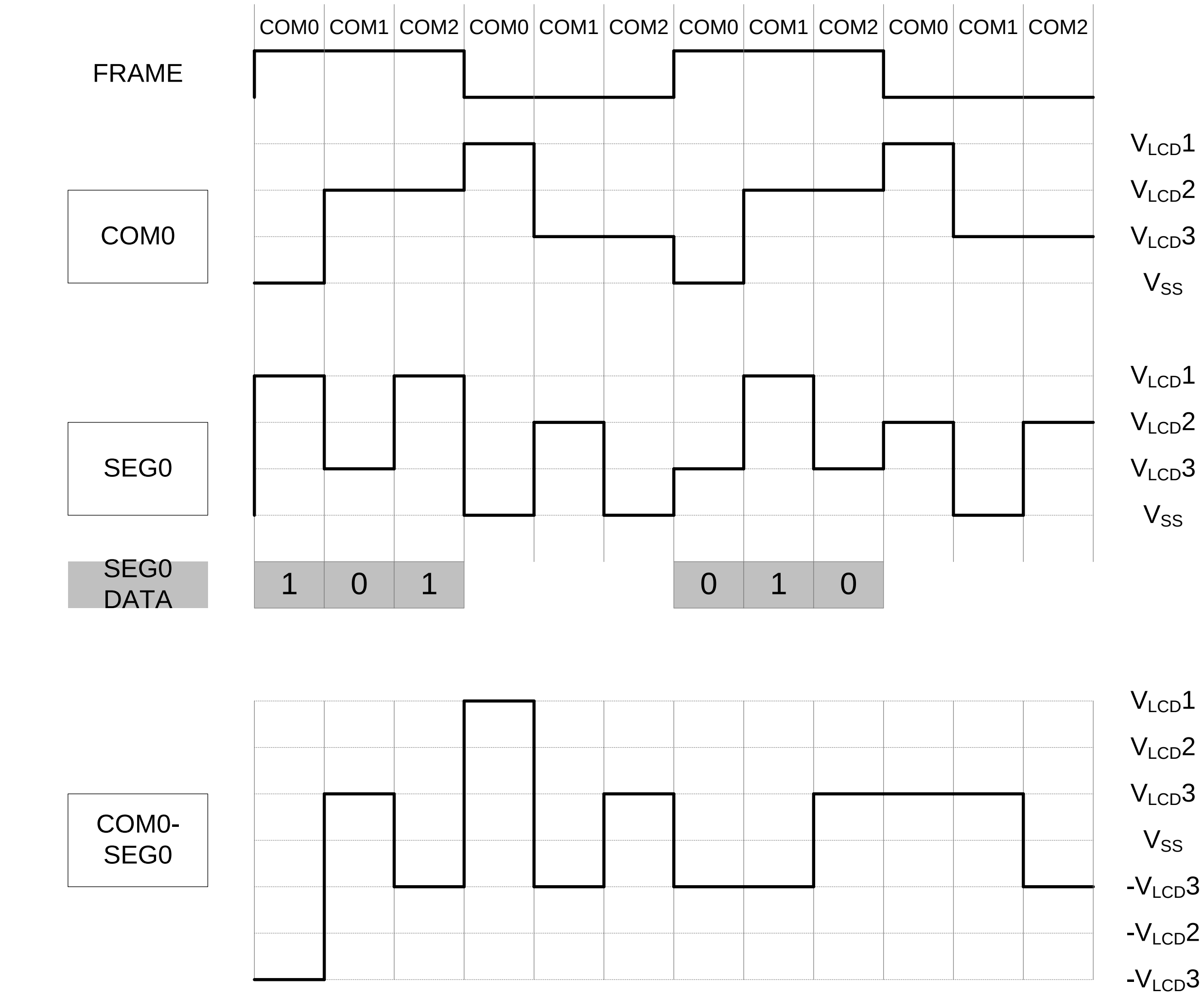


Figure 17-8 COM/SEG Signal in 1/3Duty and 1/3Bias Mode

17.2.5.5 1/4 Duty, 1/3 Bias Mode

The AC timing diagram of COM/SEG signals under 1/4 duty 1/3 bias mode is described in *Figure 17-9*.

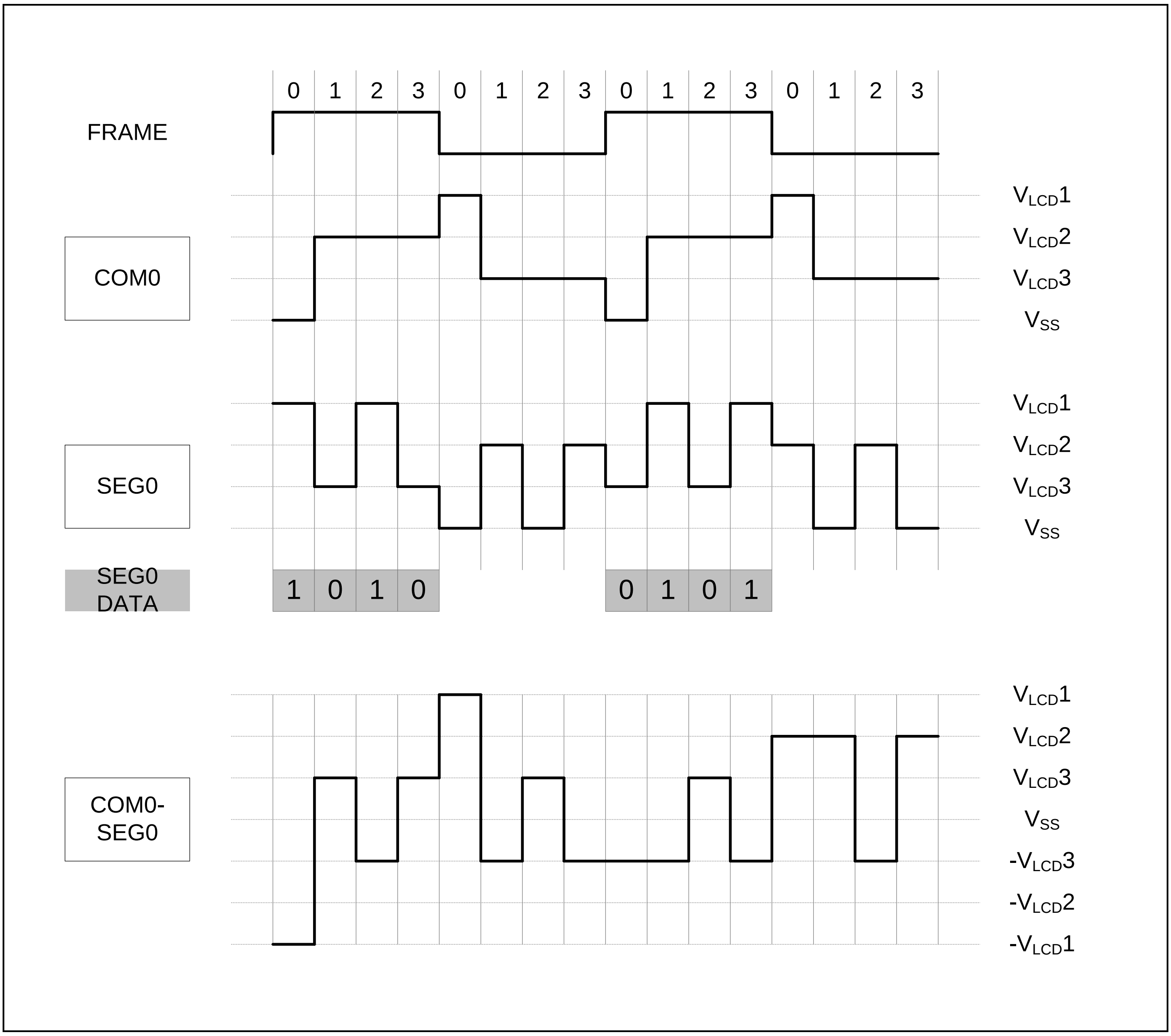


Figure 17-9 COM/SEG Signal in 1/4Duty and 1/3Bias Mode

17.3 Register Description

17.3.1 Register Map Summary

- Base Address: 0x400D_0000

Register	Offset	Description	Reset Value
LCD_IDR	0x000	ID Register	0x0001_0022
LCD_CEDR	0x004	Clock Enable/Disable Register	0x0000_0000
LCD_SRR	0x008	Software Reset Register	0x0000_0000
LCD_CR	0x00C	Control Register	0x0000_0000
LCD_CDR	0x010	Clock Divider Register	0x0000_0000
Reserved	0x014 ~ 0x3FC	Reserved	–
LCD_DMRx	0x400 + (x*4)	Display Memory Register x	0x0000_0000

NOTE: x = from 0 to 39

17.3.1.1 LCD_IDR (LCD ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_0022

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

Name	Bit	Type	Description	Reset Value
IDCODE[25:0]	[25:0]	R	Identification Code Register This field stores the ID code for the corresponding IP.	0x0001_0022

17.3.1.2 LCD_CEDR (LCD Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
CEN	[0]	RW	LCD Clock Enable Bit 0 = Disable LCD Controller Clock 1 = Enables LCD Controller Clock LCD software reset does not affect CLKEN bit status.	0

17.3.1.3 LCD_SRR (LCD Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	LCD Software Reset 0 = No effect 1 = Perform LCD Controller Software Reset operation	0

17.3.1.4 LCD_CR (LCD Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD								CONTRASTLEVEL[3:0]				RSVD		DIMINISHEN	CONTRASTEN	RSVD					DBSEL[2:0]			RSVD			BTSEL	RSVD	DISC[1:0]		LCDEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	RW	RW	RW	RW	R	R	RW	RW	R	R	R	R	R	RW	RW	RW	R	R	R	RW	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
LCDEN	[0]	RW	LCD Enable/Disable Control Bit 0 = Disable LCD Controller, Output Low; Turn LCD display off, Cut off voltage booster → LCD display off Bias resistor string and VLCD path disconnection 1 = Enable LCD Controller Bias resistor string and VLCD path connection **NOTE:** When the LCDEN bit is 0, LCD output signals (COM and SEG signals) will have logic value 0. In other words, LCD is in a display turn-off state.	0
DISC[1:0]	[2:1]	RW	LCD Display Control Field 00 = all LCD Segment Off 01 = all LCD Segment On 10 = Normal Mode 11 = Not used This bit is valid when LCDEN bit sets to '1'.	00'b
BTSEL	[3]	RW	Bias Type Selection Bit 0 = Internal Resistor Bias 1 = External Resistor Bias	0
BSEL[2:0]	[10:8]	RW	Duty and Bias Selection Field This filed decides the duty and bias for LCD display. It supports five operations so that there are five types of output waveform by a bias and duty mode.	000'b

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><th colspan="3">DBSEL [2:0]</th><th>Duty and Bias Mode</th><th>Note</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Static Mode</td><td>Use COM0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1/2 Duty, 1/2 Bias Mode</td><td>Use COM0 and COM1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1/3 Duty, 1/2 Bias Mode</td><td>Use from COM0 to COM2</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1/3 Duty, 1/3 Bias Mode</td><td>Use from COM0 to COM2</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1/4 Duty, 1/3 Bias Mode</td><td>Use from COM0 to COM3</td></tr>
<tr><td colspan="3">Others</td><td>Reserved</td><td>–</td></tr>
</table>
</td><td></td></tr>
<tr><td>CONTRASTEN</td><td>[16]</td><td>RW</td><td>Contrast Control Enable/Disable Bit
0 = Disable LCD contrast controller
1 = Enable LCD contrast controller</td><td>0</td></tr>
<tr><td>DIMINISHEN</td><td>[17]</td><td>RW</td><td>LCD Dividing Resistors Selection Bit
0 = Normal LCD dividing resistors
1 = Diminish LCD dividing resistors to strengthen LCD drive</td><td>0</td></tr>
<tr><td>CONSTRASTLEVEL[3:0]</td><td>[23:20]</td><td>RW</td><td>LCD Contrast Level Control Bits (16 steps)
0000 = 1/16 step (The dimmest level)
0001 = 2/16 step
0001 = 3/16 step
0001 = 4/16 step
0001 = 5/16 step
0001 = 6/16 step
0001 = 7/16 step
0001 = 8/16 step
0001 = 9/16 step
0001 = 10/16 step
0001 = 11/16 step
0001 = 12/16 step
0001 = 13/16 step
0001 = 14/16 step
0001 = 15/16 step
0001 = 16/16 step (The brightest level)
<b>NOTE:</b> VLCD = VDD x (n + 17)/32, where n=0-15(At normal LCD dividing resisters)</td><td>0x0</td></tr>
</table>

17.3.1.5 LCD_CDR (LCD Clock Divide Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD								CPRE[15:0]																CDC	RSVD				CDIV[3:0]		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R	R	R	R	R W	R W	R W

Name	Bit	Type	Description	Reset Value
CDIV[3:0]	[2:0]	RW	LCD Clock Dividing Value	0x0
CDC	[7]	RW	Clock Divider Control Bit 0 = $F_{LCDFREQ}$ is the same with F_{LCDCLK}. 1 = LCD frequency is generated the following; $F_{LCDFREQ} = F_{LCDCLK} / 2^{CDIV[2:0]}$ / (CPRE[15:0] +1) /2 **NOTE:** If CDC is set to 0, CDIV and CPRE are no effect.	0
CPRE[15:0]	[23:8]	RW	LCD Clock Pre-scaler Value	0x0000

17.3.1.6 LCD_DMRx (LCD Display Memory Register x)

- Address = Base Address + 0x0400 + (x*4), Reset Value = 0x0000_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
RSVD																												SEGx, COM3	SEGx, COM2	SEGx, COM1	SEGx, COM0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
SEGxCOMn	[y]	RW	Display Memory Control Bits 0 = LCD segment off 1 = LCD segment on	0

18 Operational Amplifier (OPAMP)

18.1 Overview

This microcontroller has an Operational Amplifier (OP-AMP). The op-amp can operate separately or with ADC. The gain can be configurable by control bits.

18.1.1 Features

- Configurable internal gain from x2.32 to x8.86
- Gain can be controlled by an external circuit.

18.1.2 Pin Description

Table 18-1 OP AMP Pin Description

Pin Name	Function	I/O Type	Comments
OPA_O[4:0]	Operational Amplifier Output Pin	O	–
OPA_P[4:0]	Operational Amplifier Positive Input Pin	I	–
OPA_N[4:0]	Operational Amplifier Negative Input Pin	I	–

18.2 Functional Description

18.2.1 Block Diagram

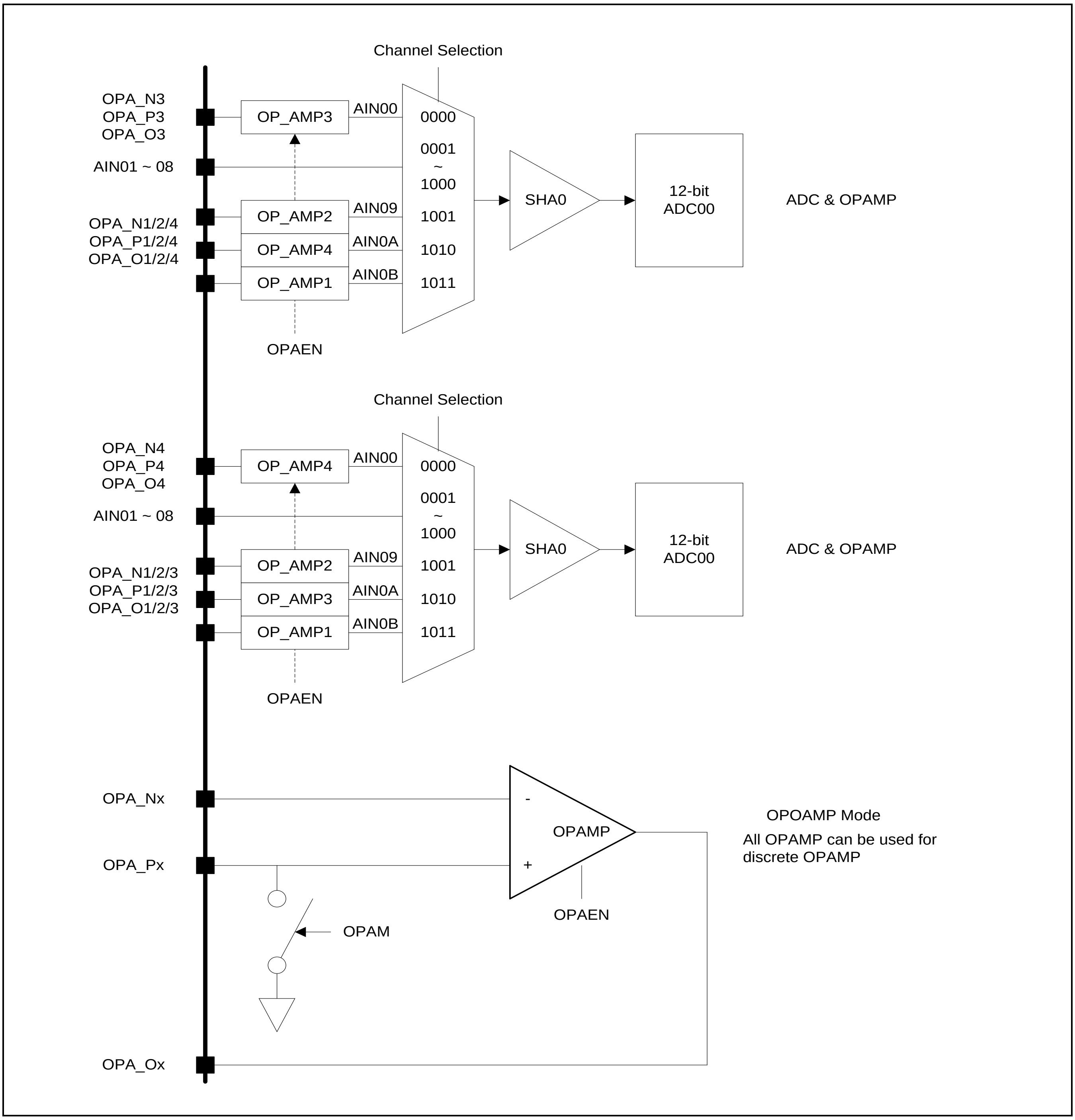


Figure 18-1 OP-AMP Block Diagram

If ADC converts AIN0x signal amplified by OP-AMP, OP-AMP should be enable and programmed gain value before ADC operation. And ADC should select AIN0 as a conversion input channel.

18.2.2 Gain Generation Circuit

User can use an external gain generation circuit with OPA_N, OPA_P, and OPA_O pin or internal gain control. In this case ‘OPAxEN’ bit in OPA_CR0 should be configurable the corresponding each mode.

18.2.3 Edge Detection circuit

User can use OP-amp as comparator. The edge detection of 5 comparators can make IMC PWM signal floating.

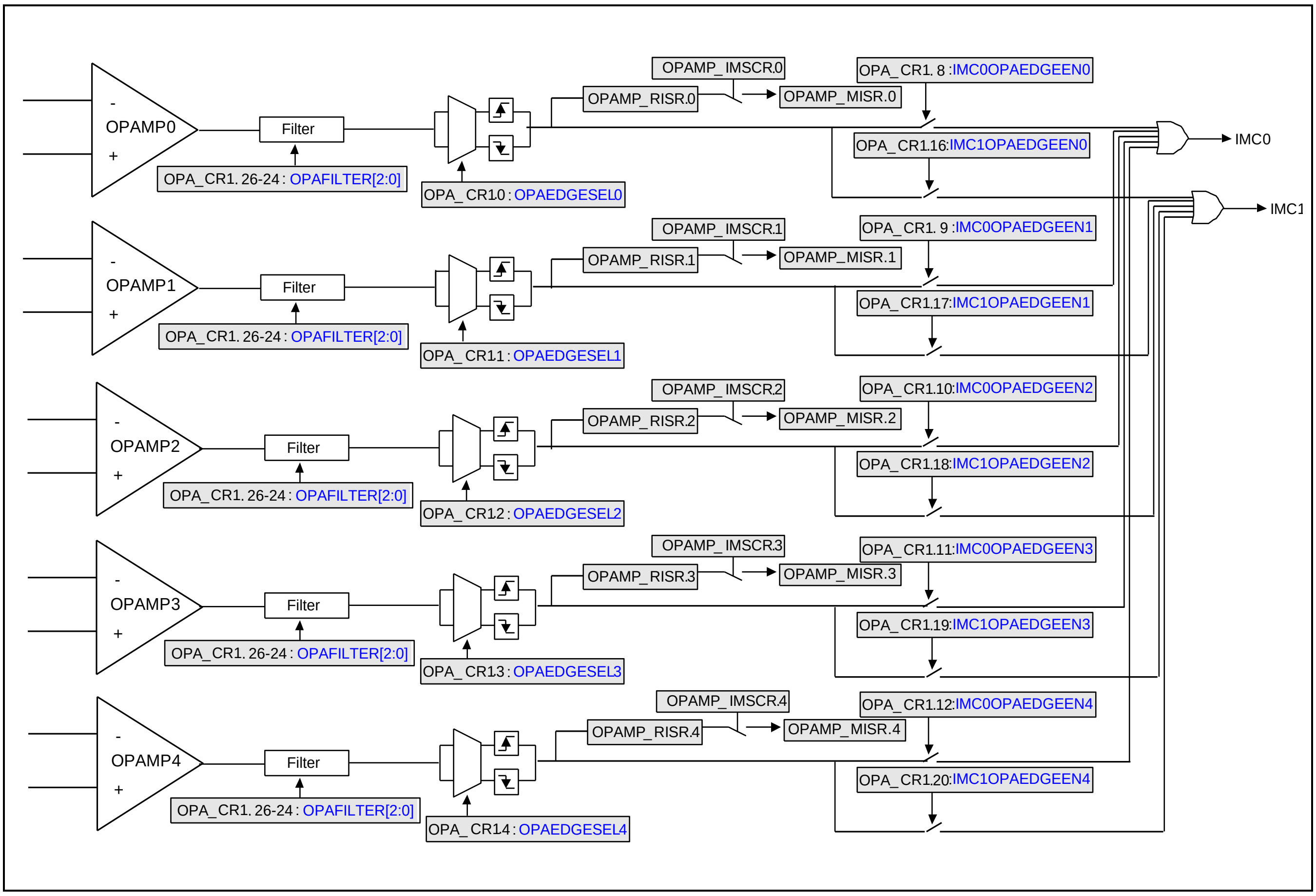


Figure 18-2 Edge Detection Diagram

18.3 Register Description

18.3.1 Register Map Summary

- Base Address: 0x4004_2000

Register	Offset	Description	Reset Value
OPA_IDR	0x000	OP-AMP ID Register	0x0001_001D
OPA_CEDR	0x004	OP-AMP Clock Enable/Disable Register	0x0000_0000
OPA_SRR	0x008	OP-AMP Software Reset Register	0x0000_0000
OPA_CR0	0x00C	OP-AMP Control Register 0	0x0000_0000
OPA_CR1	0x010	OP-AMP Control Register 1	0x0000_0000
OPA_GCR0	0x014	OP-AMP Gain Control Register 0	0x0000_0000
OPA_GCR1	0x018	OP-AMP Gain Control Register 1	0x0000_0000
OPA_IMSCR	0x01C	Interrupt Mask Set/Clear Register	0x0000_0000
OPA_RISR	0x020	Raw Interrupt Status Register	0x0000_0000
OPA_MISR	0x024	Masked Interrupt Status Register	0x0000_0000
OPA_ICR	0x028	Interrupt Clear Register	0x0000_0000

18.3.1.1 OPA_IDR (OPAMP ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_001D

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_001D

18.3.1.2 OPA_CEDR (OPAMP Clock Enable/ Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/ Disable Control Bit. 0 = Disable OP-AMP Clock 1 = Enables OP-AMP Clock OP-AMP software reset does not affect CLKEN bit status.	0

18.3.1.3 OPA_SRR (OPAMP Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No effect 1 = Perform OP-AMP Software Reset operation	0

18.3.1.4 OPA_CR0 (OPAMP Control Register 0)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																			OPAM4	OPAM3	OPAM2	OPAM1	OPAM0	RSVD			OPA4	OPA3	OPA2	OPA1	OPA0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	R	R	R	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
OPA0 OPA1 OPA2 OPA3 OPA4	[0] [1] [2] [3] [4]	RW	OP-AMP Enable Bit. 0 = Disable OP-AMP 1 = Enable OP-AMP	00000'b
OPAM0 OPAM1 OPAM2 OPAM3 OPAM4	[8] [9] [10] [11] [12]	RW	OPAMP Mode Selection Bit. 0 = Off ADC mode 1 = On Chip mode	00000'b

18.3.1.5 OPA_CR1 (OPAMP Control Register 1)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD					OPAFILTER			RSVD			IMC1OPAEDGEEN4	IMC1OPAEDGEEN2	IMC1OPAEDGEEN2	IMC1OPAEDGEEN1	IMC1OPAEDGEEN0	RSVD			IMC0OPAEDGEEN4	IMC0OPAEDGEEN2	IMC0OPAEDGEEN2	IMC0OPAEDGEEN1	IMC0OPAEDGEEN0	RSVD			OPAEDGESEL4	OPAEDGESEL3	OPAEDGESEL2	OPAEDGESEL1	OPAEDGESEL0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	W	R	R W	R W	R W	R	R	R	R W	R W	R W	R W	R W	R	R	R	R W	R W	R W	R W	R W	R	R	R	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
OPAEDGESEL0 OPAEDGESEL1 OPAEDGESEL2 OPAEDGESEL3 OPAEDGESEL4	[0] [1] [2] [3] [4]	RW	OPAMP Edge Selection Bit. 0 = falling edge detection 1 = rising edge detection	00000'b
IMC0OPAEDGEEN0 IMC0OPAEDGEEN1 IMC0OPAEDGEEN2 IMC0OPAEDGEEN3 IMC0OPAEDGEEN4	[8] [9] [10] [11] [12]	RW	OPAMP Edge Detection for IMC0 Enable Bit. 0 = Edge detection disable 1 = Edge detection enable	00000'b
IMC1OPAEDGEEN0 IMC1OPAEDGEEN1 IMC1OPAEDGEEN2 IMC1OPAEDGEEN3 IMC1OPAEDGEEN4	[16] [17] [18] [19] [20]	RW	OPAMP Edge Detection for IMC1 Enable Bit. 0 = Edge detection disable 1 = Edge detection enable	0
OPAFILTER[2:0]	[26:24]	RW	OPAMP Edge Detection Filter Selection Bit 000 = bypass 001 = PCLK/1 010 = PCLK/16 011 = PCLK/64 100 = PCLK/128 101 = PCLK/256 110 = PCLK/512 111 = PCLK/1024 **NOTE:** Only 5 times same level in a row is recognized as effective signal.	0

18.3.1.6 OPA_GCR0 (OPAMP Gain Control Register 0)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD								GCT2	RSVD			GV2				GCT1	RSVD			GV1				GCT0	RSVD			GV0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	RW	R	R	R	RW	RW	RW	RW	RW	R	R	R	RW	RW	RW	RW	RW	R	R	R	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>GV0
GV1
GV2
GV3</td><td>[3:0]
[11:8]
[19:16]
[27:24]</td><td>RW</td><td>Channel x OP-AMP Gain Value Field.
This has an effect on an internal gain control.
<table>
<tr><th colspan="4">GVx</th><th>Gain Specification of Operational Amplifier</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>× 2.32</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>× 2.47</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>× 2.62</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>× 2.76</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>× 3.06</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>× 3.35</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>× 3.65</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>× 4.09</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>× 4.53</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>× 5.04</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>× 5.92</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>× 7.09</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>× 8.86</td></tr>
<tr><td>–</td><td>1</td><td>0</td><td>–</td><td>Setting prohibited</td></tr>
</table>
</td><td>0000'b</td></tr>
<tr><td>GCT0
GCT1
GCT2
GCT3</td><td>[7]
[15]
[23
[31]</td><td>RW</td><td>OP-AMP Gain Control Type Selection Bit.
0 = External Gain Control
1 = Internal Gain Control</td><td>0</td></tr>
</table>

18.3.1.7 OPA_GCR1 (OPAMP Gain Control Register 1)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																								GCT4	RSVD			GV4			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	R	R	R	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>GV4</td><td>[3:0]</td><td>RW</td><td>Channel x OP-AMP Gain Value Field.
This has an effect on an internal gain control.
<table>
<tr><th colspan="4">GVx</th><th>Gain Specification of Operational Amplifier</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>× 2.32</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>× 2.47</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>× 2.62</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>× 2.76</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>× 3.06</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>× 3.35</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>× 3.65</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>× 4.09</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>× 4.53</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>× 5.04</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>× 5.92</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>× 7.09</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>× 8.86</td></tr>
<tr><td>–</td><td>1</td><td>0</td><td>–</td><td>Setting prohibited</td></tr>
</table>
</td><td>0000'b</td></tr>
<tr><td>GCT4</td><td>[7]</td><td>RW</td><td>OP-AMP Gain Control Type Selection Bit.
0 = External Gain Control
1 = Internal Gain Control</td><td>0</td></tr>
</table>

18.3.1.8 OPA_IMSCR (OPAMP Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																											EDGEDET4	EDGEDET3	EDGEDET2	EDGEDET1	EDGEDET0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
EDGEDET0 EDGEDET1 EDGEDET2 EDGEDET3 EDGEDET4	[0] [1] [2] [3] [4]	RW	Edge Detection Interrupt Mask. 0 = Edge Detection interrupt is masked. (Disable the interrupt) 1 = Edge Detection interrupt is not masked. (Enable the interrupt)	00000'b

18.3.1.9 OPA_RISR (OPAMP Raw Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																											EDGEDET4	EDGEDET3	EDGEDET2	EDGEDET1	EDGEDET0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EDGEDET0 EDGEDET1 EDGEDET2 EDGEDET3 EDGEDET4	[0] [1] [2] [3] [4]	R	Edge Detection Raw Interrupt Status Gives the raw interrupt state(prior to masking) of the Edge Detection interrupt	00000'b

18.3.1.10 OPA_MISR (OPAMP Masked Interrupt Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																											EDGEDET4	EDGEDET3	EDGEDET2	EDGEDET1	EDGEDET0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
EDGEDET0	[0]	R	Edge Detection Masked Interrupt Status Gives the masked interrupt state(after masking) of the Edge Detection interrupt	00000'b
EDGEDET1	[1]			
EDGEDET2	[2]			
EDGEDET3	[3]			
EDGEDET4	[4]			

18.3.1.11 OPA_ICR (OPAMP Interrupt Clear Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																											EDGEDET4	EDGEDET3	EDGEDET2	EDGEDET1	EDGEDET0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
EDGEDET0 EDGEDET1 EDGEDET2 EDGEDET3 EDGEDET4	[0] [1] [2] [3] [4]	W	Edge Detection interrupt 0 = No effect 1 = Clears the Edge Detection interrupt	00000'b

19 Program Flash Controller

19.1 Overview

The S3FM02G has an on-chip program flash ROM, internally. The memory flash size is 384Kbytes.

19.1.1 Features

- Flash memory size: 384Kbytes
- Program Size: word(32bit)
- Page Size: 1Kbyte
- Sector Size: 32Kbyte
- Boot sector erase enable / disable by entire program flash erase supports
- Protection supports: JTAG protection, Hardware protection, and Read protection

19.2 Functional Description

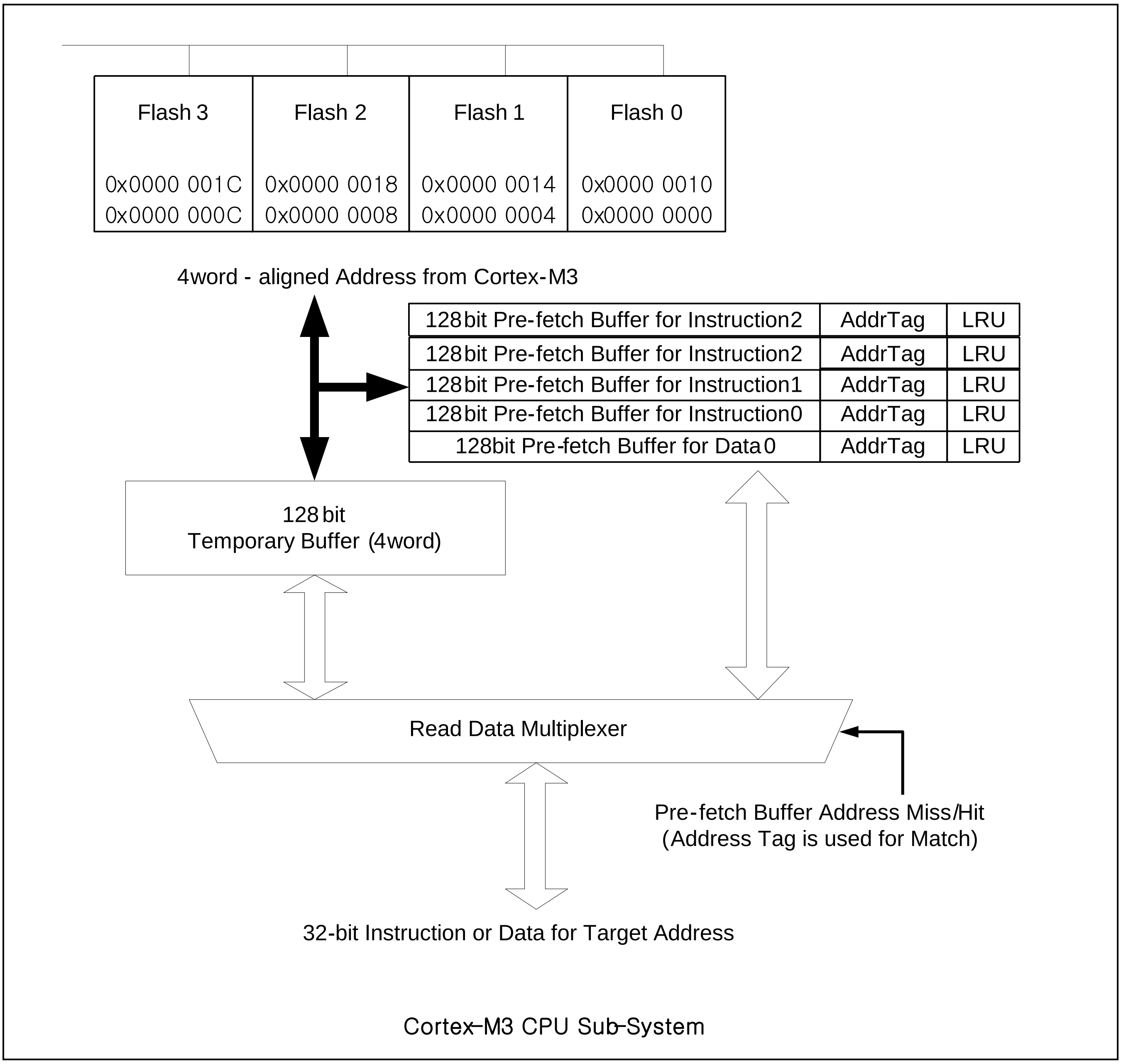
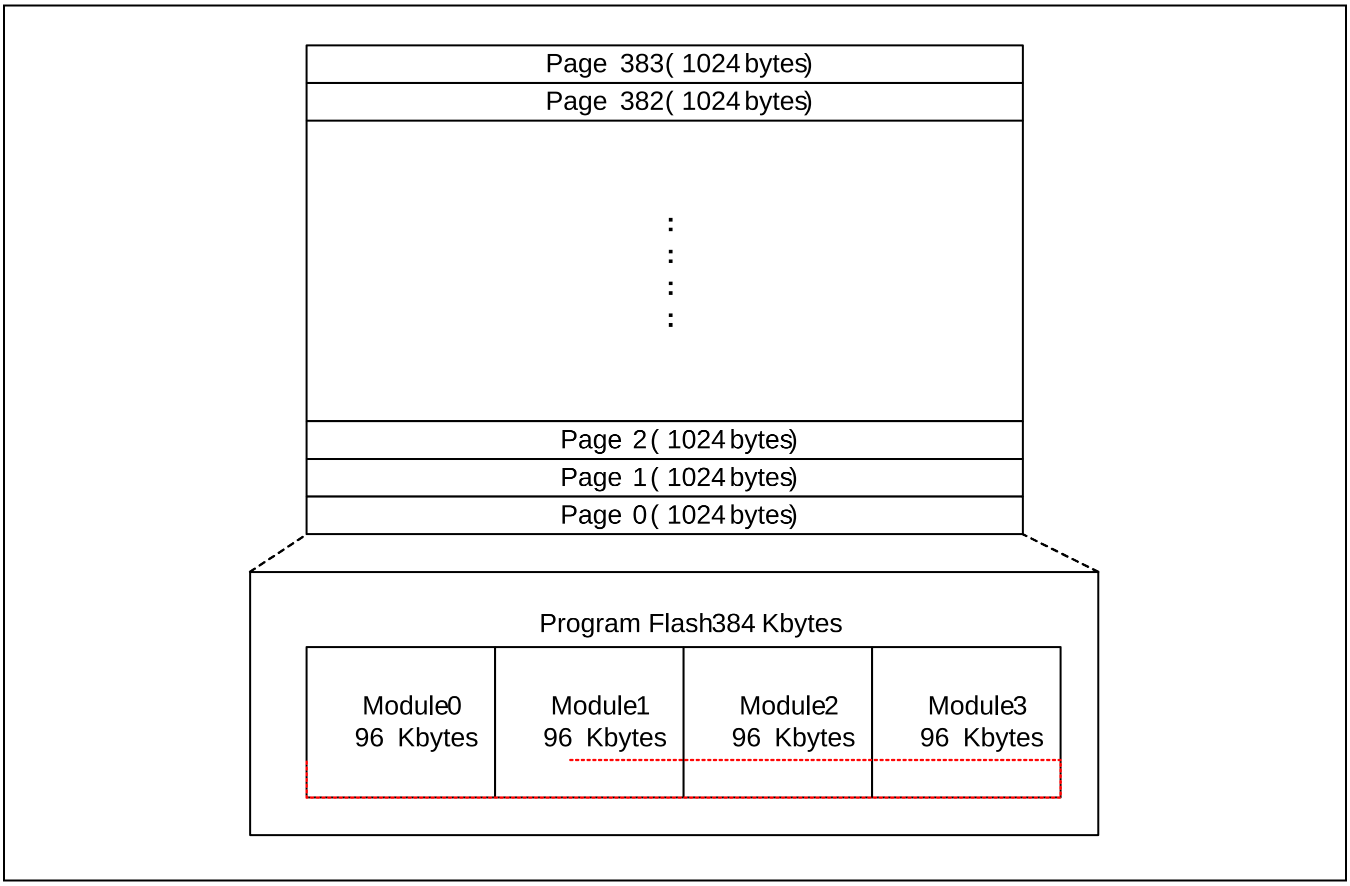

Flash 3
0x0000 001C
0x0000 000C
Flash 2
0x0000 0018
0x0000 0008
Flash 1
0x0000 0014
0x0000 0004
Flash 0
0x0000 0010
0x0000 0000
4word - aligned Address from Cortex-M3
128bit Pre-fetch Buffer for Instruction2
AddrTag
LRU
128bit Pre-fetch Buffer for Instruction2
AddrTag
LRU
128bit Pre-fetch Buffer for Instruction1
AddrTag
LRU
128bit Pre-fetch Buffer for Instruction0
AddrTag
LRU
128bit Pre-fetch Buffer for Data0
AddrTag
LRU
128 bit
Temporary Buffer (4word)
Read Data Multiplexer
Pre-fetch Buffer Address Miss/Hit
(Address Tag is used for Match)
32-bit Instruction or Data for Target Address
Cortex-M3 CPU Sub-System

Figure 19-1 Program Flash Block Diagram

19.2.1 Organization

19.2.1.1 Program Flash Configuration

The 384KBytes program flash ROM consists of 384 pages and 12 sectors. Page size is 1KB and sector size is 32KB. User can erase the flash memory a page, sector and entire program flash-unit at a time and write the data into the flash memory a word-unit(4 bytes) at a time.



19.2.1.2 Address Alignment

To set an address value in PF_AR register, abide by the following rules.

19.2.1.2.1 Program

When programming program flash, the lower 2-bits should be ‘0’, because data should be written to the flash by a word-unit (4bytes). User can select one as the ADDRESS from 0x0000 to 0x5FFFF, in 384Kbytes range. In the tool program, the low 2-bit address also should be 00b.

PF_AR.ADDR [31:0] = ADDRESS & 0xFFFFFFFC

19.2.1.2.2 Page Erase

When erasing a sector, the lower 10-bits of address should be ‘0’, because the size of a page is 1024Bytes. User can select one as the PAGE_ORDER from 0 to 383, among 384 sectors.

PF_AR.ADDR [31:0] = (PAGE_ ORDER << 10)

- Page 0: 0x0000 0000 ~ 0x0000_03FF → ADDR[31:0]= 0x00000000 = 0b<<10
- Page 1: 0x0000_0400 ~ 0x0000_07FF → ADDR[31:0]= 0x00000400 = 1<<10
- Page 2: 0x0000_0800 ~ 0x0000_0BFF→ ADDR[31:0]= 0x00000800 = 2 <<10

Page Number	Page Address
0	0x0000 0000
1	0x0000 0400
2	0x0000 0800
3	0x0000 0C00
4	0x0000 1000
5	0x0000 1400
6	0x0000 1800
7	0x0000 1C00
8	0x0000 2000
9	0x0000 2400
10	0x0000 2800
11	0x0000 2C00
12	0x0000 3000
13	0x0000 3400
…	…
380	0x0005 F000
381	0x0005 F400
382	0x0005 F800
383	0x0005 FC00

19.2.1.2.3 Sector Erase

When erasing a sector, the lower 15-bits of address should be ‘0’, because the size of a sector is 32KBytes. User can select one as the SECTOR_ORDER from 0 to 11, among 12 sectors.

PF_AR.ADDR [31:0] = (SECTOR_ ORDER << 10)

- Sector 0: 0x0000 0000 ~ 0x0000_03FF → ADDR[31:0]= 0x00000000 = 0b<<15
- Sector 1: 0x0000_0400 ~ 0x0000_07FF → ADDR[31:0]= 0x00008000 = 1<<15
- Sector 2: 0x0000_0800 ~ 0x0000_0BFF→ ADDR[31:0]= 0x00010000 = 2 <<15

Sector Number	Sector Base Address
0	0x0000 0000
1	0x0000 8000
2	0x0001 0000
3	0x0001 8000
4	0x0002 0000
5	0x0002 8000
6	0x0003 0000
7	0x0003 8000
8	0x0004 0000
9	0x0004 8000
10	0x0005 8000
11	0x0006 0000

19.2.2 Smart Option

There are two kinds of smart option. One is ‘protection’ smart option. The other is a configuration smart option.

- Flash Memory Address for Configuration Smart Option: 0x00000400
- Flash Memory Address for Protection Smart Option: 0x00000404

19.2.2.1 Protection Smart Option

The data or code programmed in flash memory need to be protected. For this situation, the program flash memory controller of S3FM02G supports three kinds of protection mechanism. These protections can be enabled by programming the protection smart option bits. The protection smart option can be enabled or disabled with configuration at address 0x00000404.

- HARDWARE (HARDLOCK) PROTECTION
  Protection for selected regions among 12regions or full region
- READ PROTECTION
  Flash read protection for serial interface
- JTAG PROTECTION
  Flash read protection on JTAG interface

Table 19-1 Protection Smart Option Address and Protection Bits

PF_AR	PF_DR	Description	Description	Reset Value
0x0000_0404	Bit[8]	0 = Enable JTAG Protection 1 = Disable JTAG Protection	–	1
	Bit[17]	0 = Enable Hardware Protection 1 = Disable Hardware Protection	(NOTE)	1
	Bit[27]	0 = Enable Read Protection 1 = Disable Read Protection	–	1
	Bit[4]	Hardware Protection Region Selection Bit These bits are each mapped to a corresponding region which is composed of 1 sector (32KB). 0 = Enable H/W protection of selected regions 1 = Disable H/W protection of selected regions	Sector 0	1
	Bit[5]		Sector 1	1
	Bit[6]		Sector 2	1
	Bit[7]		Sector 3	1
	Bit[12]		Sector 4	1
	Bit[13]		Sector 5	1
	Bit[14]		Sector 6	1
	Bit[15]		Sector 7	1
	Bit[20]		Sector 8	1
	Bit[21]		Sector 9	1
	Bit[22]		Sector 10	1
	Bit[23]		Sector 11	1
	Others	Not used	–	1

NOTE: For enabling Hardware Protection, user must define the region or regions for protection, and do smart option program. The hardware protection data in protection smart option must include Hardware Protection Bit (Bit[17]) and hardware protection region bits.

19.2.2.1.1 JTAG Protection Bit.8

This Bit is used for JTAG access enable or disable (If chip designers would like to debug through JTAG in initial chip development state, JTAG Interface Protection Bit should be disabled, but, in final design development state, if chip designers enable the JTAG interface Protection, other user can not access the flash memory data via JTAG interface)

19.2.2.1.2 Hardware (Hard-Lock) Protection Bit.17

If hard-lock protection was enabled, user cannot write or erase the data in selected regions of flash memory.

In a normal operating mode, if hard-lock protected region is not full and code that can execute smart option erase is in an un-protected space, it can be release by s/w. That means user can release only chip erase when full region is protected. In a tool, hard-lock protection can be released by the chip erase execution.

Hard-lock protection can be set in user program mode as follows. First user should write the key value (0x5A5A5A5A) into key register (PF_KR). Next user should write 0x00000404 into the address (PF_AR) and the proper data (Refer to the above Protection Bit table) into the data register (PF_DR), respectively. Finally, set operation START. Please refer to flowchart.

Table 19-2 Hardware Protection Area Configuration

PF_AR	Group	Group	Description
0x00000404	Bit[17]	0 = Enable Hardware Protection 1 = Disable Hardware Protection	–
	Bit[4]	Sector 0	H/W protection is disable / enable These bits are each mapped to a corresponding group which is composed of 1 sector (32KB). The reset value of each bit is '1' (disable). 0 = Enable H/W protection of selected group 1 = Disable H/W protection of selected group
	Bit[5]	Sector 1	
	Bit[6]	Sector 2	
	Bit[7]	Sector 3	
	Bit[12]	Sector 4	
	Bit[13]	Sector 5	
	Bit[14]	Sector 6	
	Bit[15]	Sector 7	
	Bit[20]	Sector 8	
	Bit[21]	Sector 9	
	Bit[22]	Sector 10	
	Bit[23]	Sector 11	

19.2.2.1.3 Read Protection Bit.27

Most users want that their data and code in memory would not be read by others. Read protection can give the solution for it by preventing the flash data from being read serially in the tool program mode. When this function is enabled, reading the flash data with serial interface (tool program mode) will result in all zero read-out.

User should write the values (0x5A5A5A5A) into key register (PF_KEY) to program read protection. And user should write the proper data (Refer to the above protection Bit table) into the address 0x404. The address 0x00000404 should be written the register PF_AR. The data consisting of protection bit should be written the register PF_DR. Finally, set operation START. Please refer to flow chart.

19.2.3 Programming Method

User can program (write) the data or code in flash memory with several programming methods.

- User program mode (AHB Interface)
- Programming through JTAG interface
- Programming with UART interface (Added new feature)
- Programming with Flash Writing Tool (Serial interface)

19.2.3.1 Program in User Program Mode

The user program mode for flash memory programming and page / sector / entire program memory erasing uses the internal high voltage generator, which is necessary for flash memory programming and page / sector erasing. The Flash Memory Controller has an internal high voltage pumping circuit. Therefore, high voltage to VPP pin is not needed. To program the data into the flash ROM or erase page / sector / entire program memory in this mode, several control registers should be used, which will be explained below.

In order to program to flash memory, user should write the value 0x5A5A5A5A into the PF_KEY register. As a next step, user should write the address to be written into the address register (PF_AR) and the data into the data register (PF_DR), respectively. Finally user send operation start signal with START bit in control register. To perform the next writing operation, all register should be written again as before.

In order to perform sector erase procedure is the same as program procedure except not setting the data register (PF_DR) in program flash Controller. In order to perform chip erase procedure will be enough to setting the key register (PF_KR) and control register (PF_CR).

19.2.3.2 Program through JTAG interface

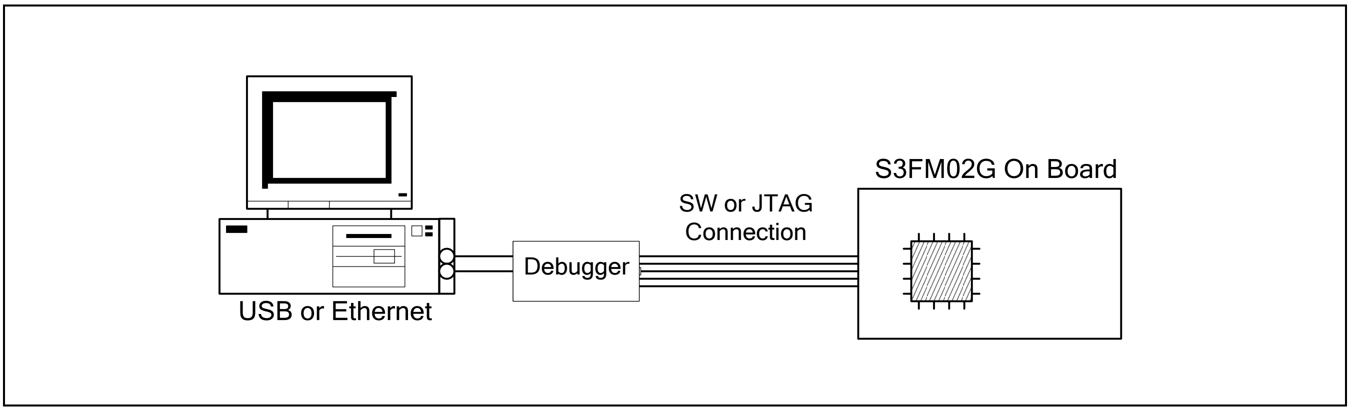


Figure 19-2 JTAG Interface Program

19.2.3.2.1 Program with UART

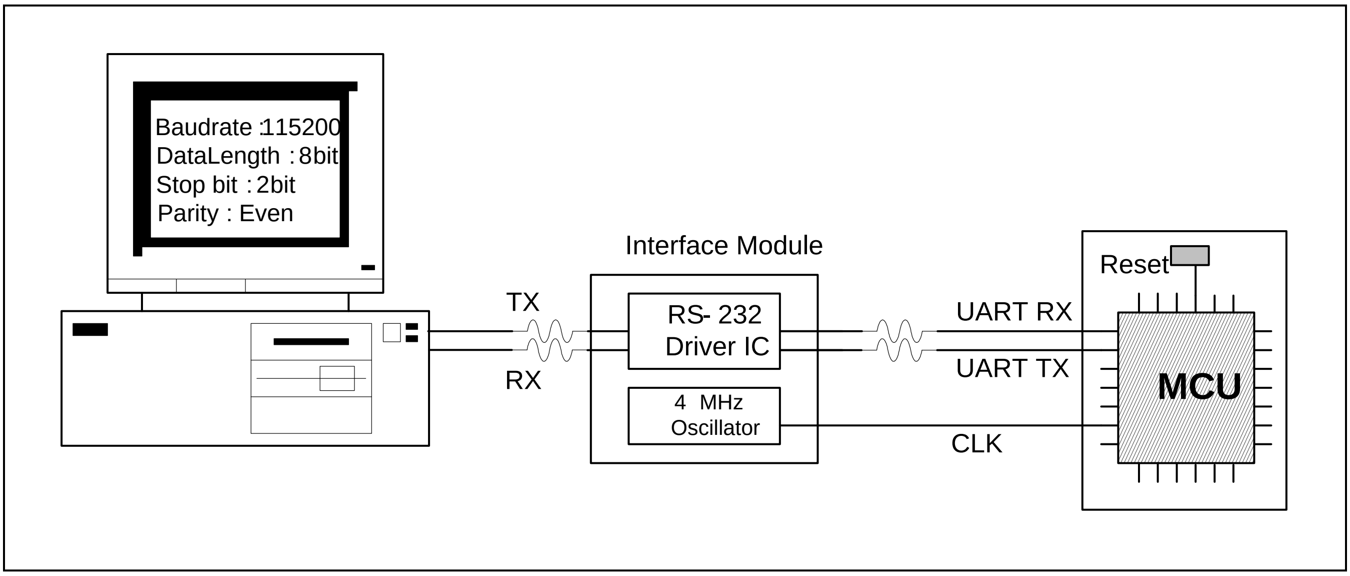


Figure 19-3 UART Interface Program

Program flash can be written using UART interface. This mode can enter when PMODE[2:0] pins are tied the logical value to 010'b. This mode will configure two pins for UART interface automatically by hardware.

To program by UART interface, UARTRX, UARTTX, CLK, Reset and flash power pins are needed. UARTRX and TX pins are for data. The interface data length is 8-bit. The data consists of serial program protocol command and data or code to be written on flash memory cell.

To use flash program by UART, user should set COM port to the following configuration;

- Baud-rate: 115200 bps
- Data Length: 8-bit
- Parity: None
- Stop bit: 2-bit

Table 19-3 Used to Read/Write/Erase the Flash ROM with UART Interface

Interface Signal	I/O	Pin Name	Description
Serial Data	I	PGP1[30]	Receive the command or data to write
	O	PGP1[31]	Transmit data written on flash memory cell
System Clock	I	PXI	Clock input for UART interface
RESET	I	PnRESET	Logic reset
MODE	I	PMODE[2:0]	Flash program mode by UART interface Mode value should be 010'b.
VDD/VSS	P	VDDx/VSSx	Logic and flash power

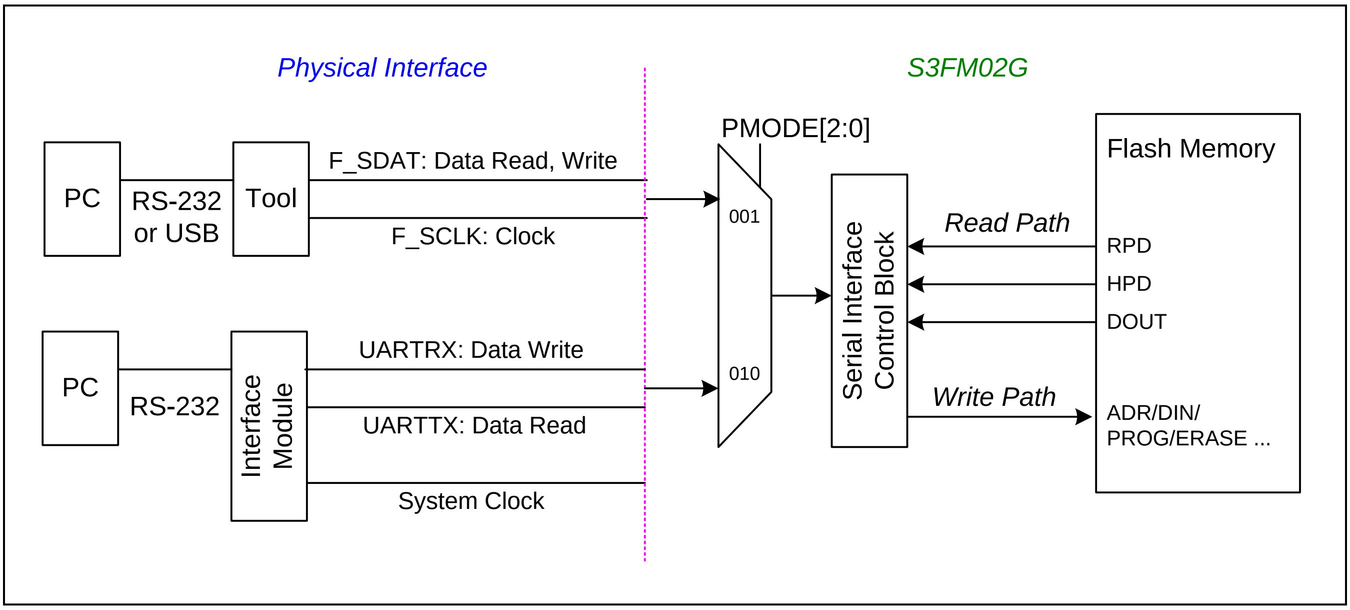


Figure 19-4 Comparison between UART Interface and Serial Interface

19.2.3.2.2 Program with Flash Writing Tool

User can program (write) using the flash ROM writer (Tool). Samsung The tool program mode can use flash memory programming, which uses an equipment tool such as SPW-uni/GW-uni/US-pro/AS-pro Flash ROM Writer.

The S3FM02G has several pins used for flash ROM writer to read/write/erase the flash memory (VDDIO[3:0], VSSIO[3:0], nRESET, VDDCORE[3:0], VSSCORE[3:0], SDAT, SCLK), which is the programming by tool program mode. These several pins are multiplexed with other functional pins.

Table 19-4 The Pins Used to Read/Write/Erase the Flash ROM in Tool Program Mode

Signal	S3FM02G Pin	I/O	Function
SDAT	PGP1[25]	I/O	Serial bi-directional DATA pin(Output when reading, Input when writing). Input & push-pull output port can be assigned
SCLK	PGP1[24]	I	Serial CLOCK input pin. (Write speed: Max 200 KHz, Read speed : Max 10 MHz)
RESET	PnRESET	I	Chip Initialization
VDD	VDDx	P	Power supply pin for Flash block
VSS	VSSx	P	GND Power supply pin for Flash block
MODE	PMODE[2:0]	–	Flash program mode using flash writing tool (serial interface) Mode value should be 001'b.

19.3 Tool Program Mode

The tool program mode is the flash memory program mode, which uses an equipment tool such as SPW2Plus Flash ROM Writer or US-PRO Flash ROM Writer. If user wants to make a dedicated Flash ROM writer for S3FM02G, please contact us for more detail document.

19.3.1 Tool Information

- SPW-uni, GW-uni
- US-pro, AS-pro
- Contact to third party (http://www.seminix.com)

19.3.2 Operation

19.3.2.1 Read Operation

User can read & access 8bit, 16bit or 32bit.

When prefetch buffer miss, read cycle is WAIT[1:0] + 1. But when prefetch buffer hit, no wait cycle is inserted.

19.3.2.2 Normal Program Operation

The normal program operation writes one word (data or code) to the target address selected by PF_AR register. That means the writing size for normal operation is 4 bytes. When executes a normal operation, the lower 2-bits should be ‘0’, because data should be written to the program flash by word-unit (32bits).

- PF_AR[31:0] = ADDRESS & 0xFFFFFFFC

Normal program for program flash must be done in the program flash, data flash or SRAM. In the program flash

In order to operation as like normal program, target address for normal operation should be initialized (0xFFFFFFFF) by sector erase, or chip erase before normal program. Next user should write the value 0x5A5A5A5A into the PF_KEY register. After that, user should select a target address by writing to PF_AR register. The operation can execute when CMD[2:0] field should be written 001’b and set START bit. User can check by END bit, if whether the operation is completed or not.

During a normal program operation, the busy bit in the PF_SR is set to high and automatically cleared at the end of the operation.

19.3.2.3 Page Erase Operation

A page has 1Kbytes size. The page erase operation erases one page including address written by PF_AR (program flash address register). For page erase, user should write the value 0x5A5A5A5A into the PF_KEY register. Next user should select a target sector to erase by writing PF_AR. The operation can execute when CMD[2:0] field should be written 010’b and set START bit. User can check by END bit, if whether the operation is completed or not.

19.3.2.4 Sector Erase Operation

A sector has 32Kbytes size. The sector erase operation erases one sector including address written by PF_AR (program flash address register). For sector erase, user should write the value 0x5A5A5A5A into the PF_KEY register. Next user should select a target sector to erase by writing PF_AR. The operation can execute when CMD[2:0] field should be written 011'b and set START bit. User can check by END bit, if whether the operation is completed or not.

19.3.2.5 Entire Program Flash Erase Operation

The chip erase operation erases entire program flash region. For entire program flash erase, user should write the value 0x5A5A5A5A into the PF_KEY register.. In case of entire program flash erase, it doesn't need to select the address and data. The operation can execute when CMD[2:0] field should be written 100'b and set START bit. User can check by END bit, if whether the operation is completed or not.

19.3.2.6 Smart Option Program Operation

- Smart Option Write
  - The writing address of smart option is the following;
    PF_AR[31:0] = 0x00000400 when program the configuration smart option
    PF_AR[31:0] = 0x00000404 when program the protection smart option
    Data for smart option shoule be written to program flash data register(PF_DR).

19.3.2.7 Smart Option Erase Operation

- Smart option Erase
  - The erasing unit of smart option flash memory is 4-byte.
    PF_AR[31:0] = 0x00000400 when erase the configuration smart option
    PF_AR[31:0] = 0x00000404 when erase the protection smart option

19.3.2.8 Flow Chart - Normal Program

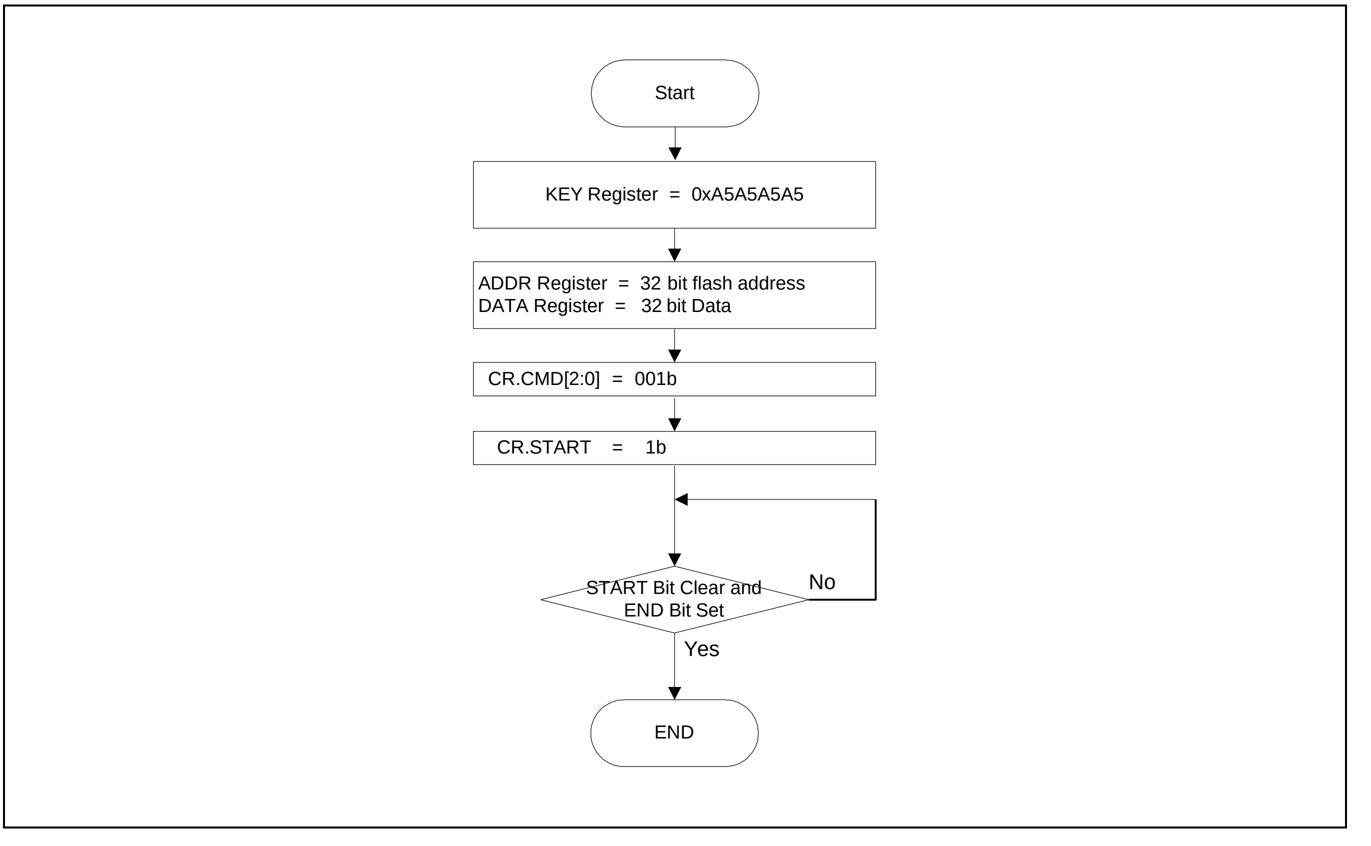

Start
KEY Register = 0xA5A5A5A5
ADDR Register = 32 bit flash address
DATA Register = 32 bit Data
CR.CMD[2:0] = 001b
CR.START = 1b
START Bit Clear and
END Bit Set
No
Yes
END

Figure 19-5 Normal Program Flow Chart

19.3.2.9 Flow Chart - Page Erase

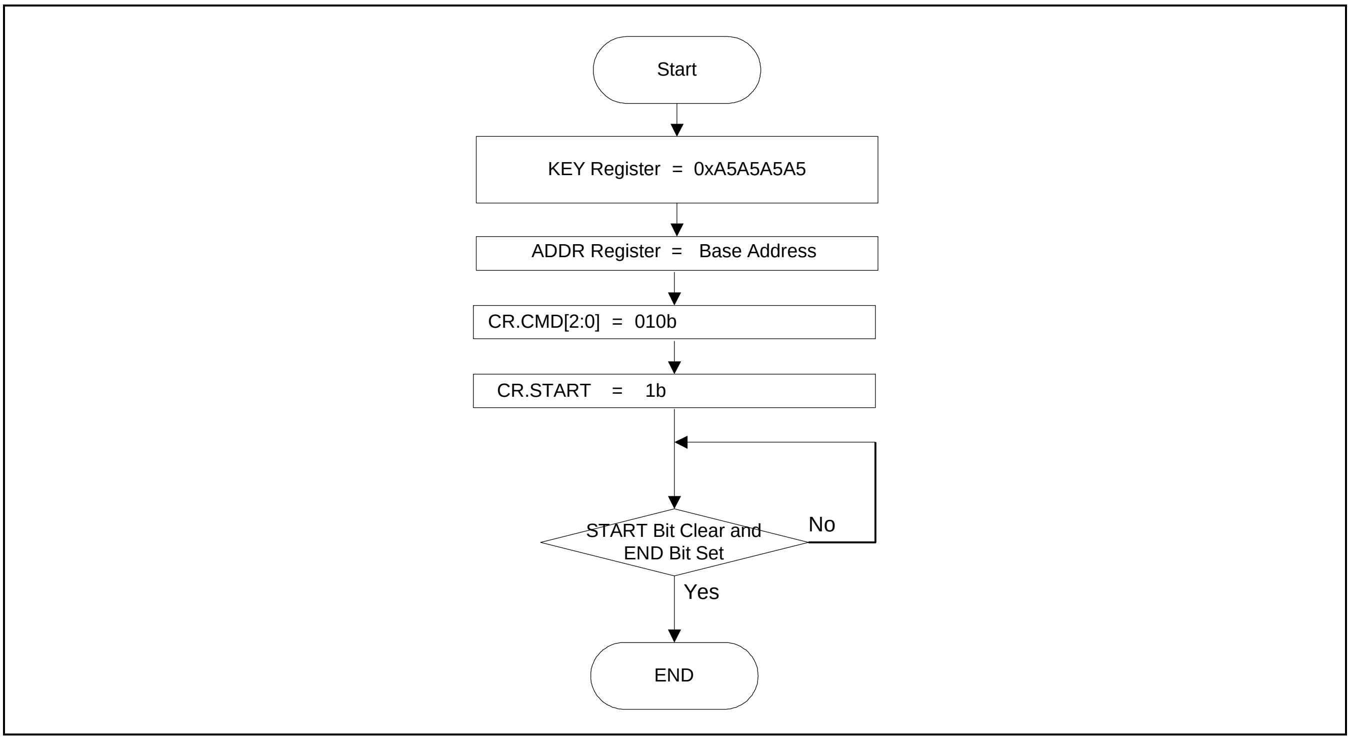

Start
KEY Register = 0xA5A5A5A5
ADDR Register = Base Address
CR.CMD[2:0] = 010b
CR.START = 1b
START Bit Clear and END Bit Set
No
Yes
END

Figure 19-6 Page Erase Flow Chart

19.3.2.10 Sector Erase

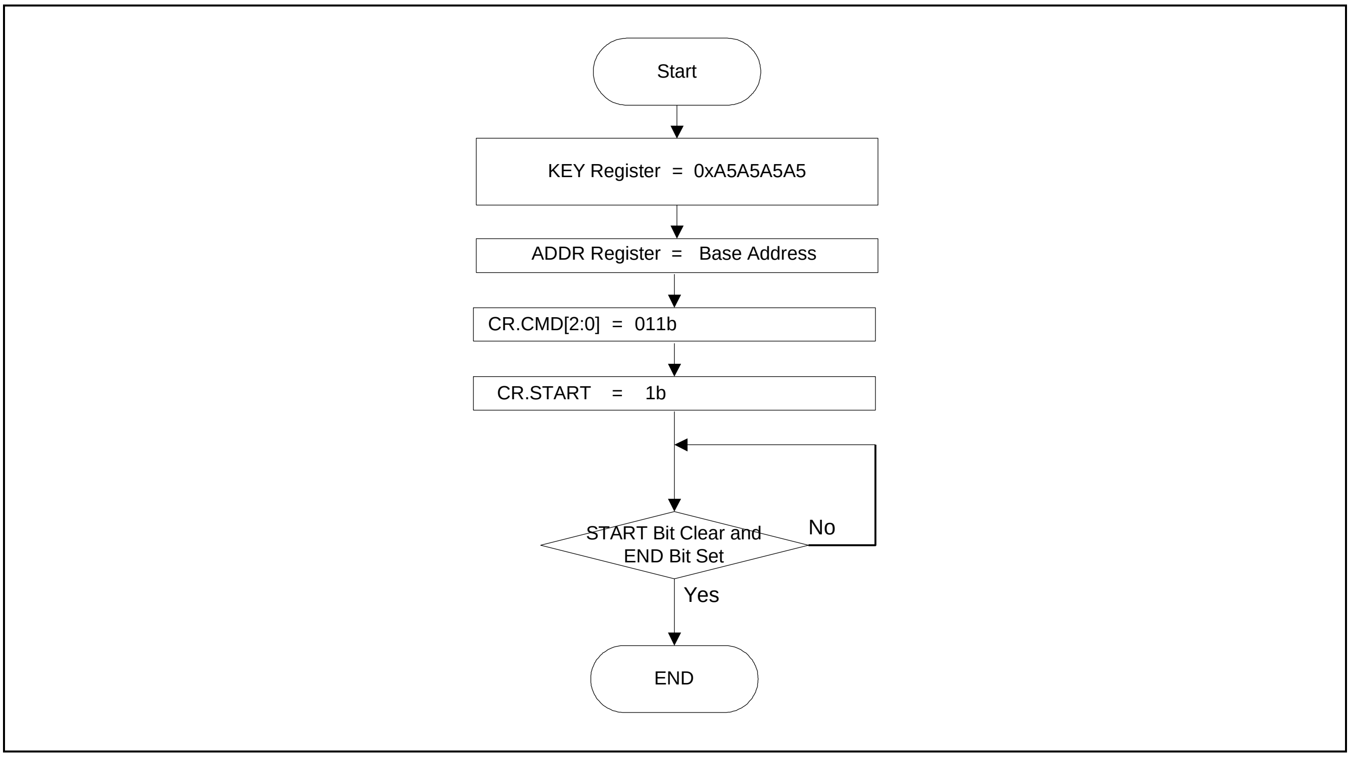

Start
KEY Register = 0xA5A5A5A5
ADDR Register = Base Address
CR.CMD[2:0] = 011b
CR.START = 1b
START Bit Clear and END Bit Set
No
Yes
END

Figure 19-7 Sector Erase Flow Chart

19.3.2.11 Flow Chart – Entire Program Flash Erase

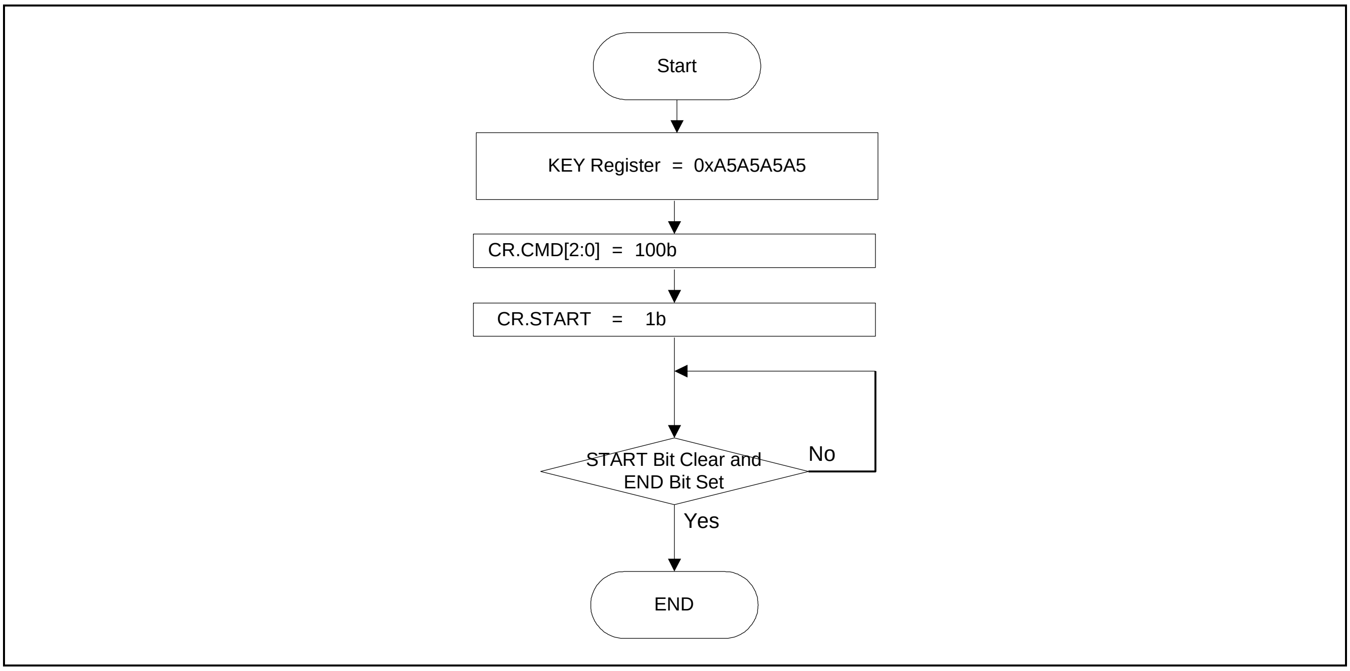


Figure 19-8 Entire Program Flash Erase Flow Chart

19.3.2.12 Flow Chart - Smart Option Program

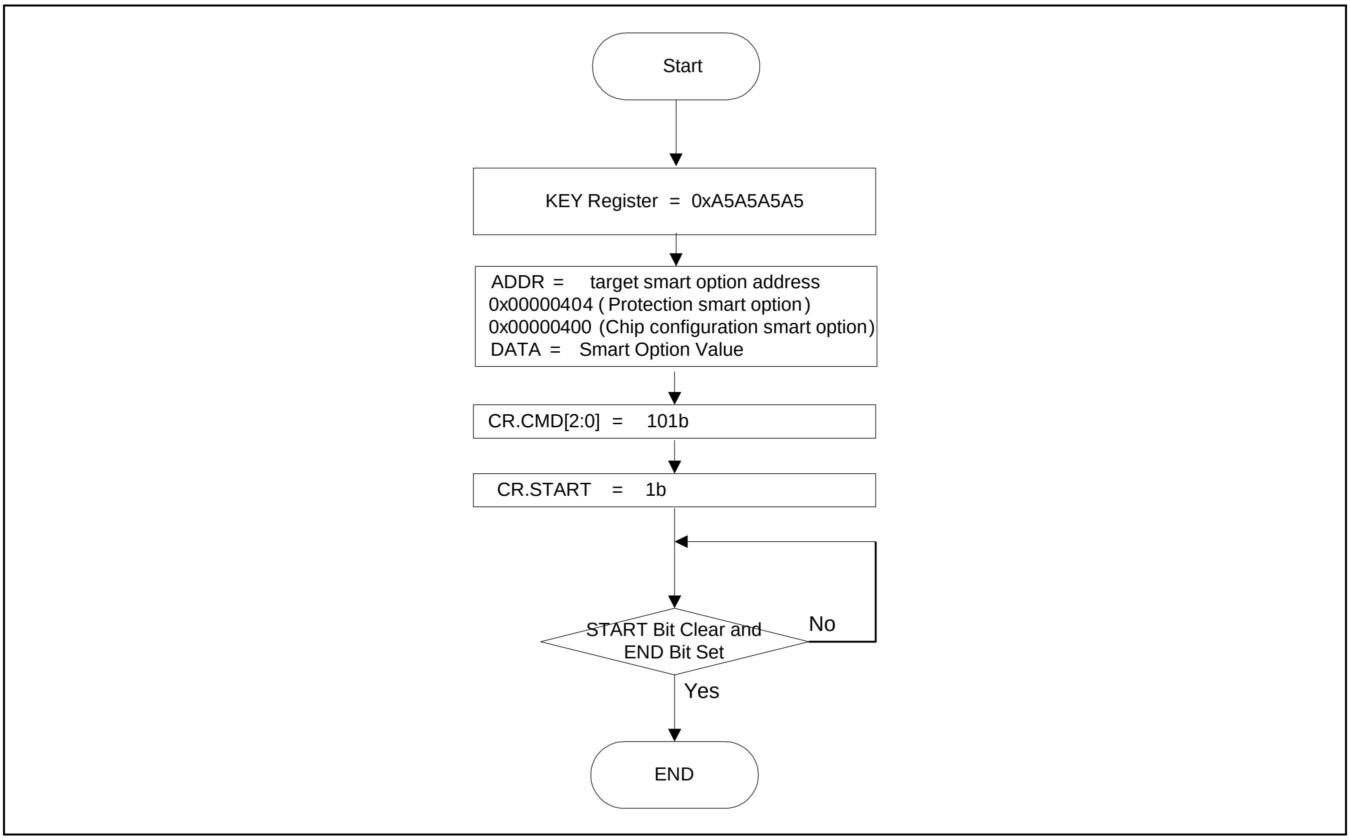

Start
KEY Register = 0xA5A5A5A5
ADDR = target smart option address
0x00000404 (Protection smart option)
0x00000400 (Chip configuration smart option)
DATA = Smart Option Value
CR.CMD[2:0] = 101b
CR.START = 1b
START Bit Clear and
END Bit Set
No
Yes
END

Figure 19-9 Smart Option Program Flow Chart

19.3.2.13 Flow Chart - Smart Option Erase

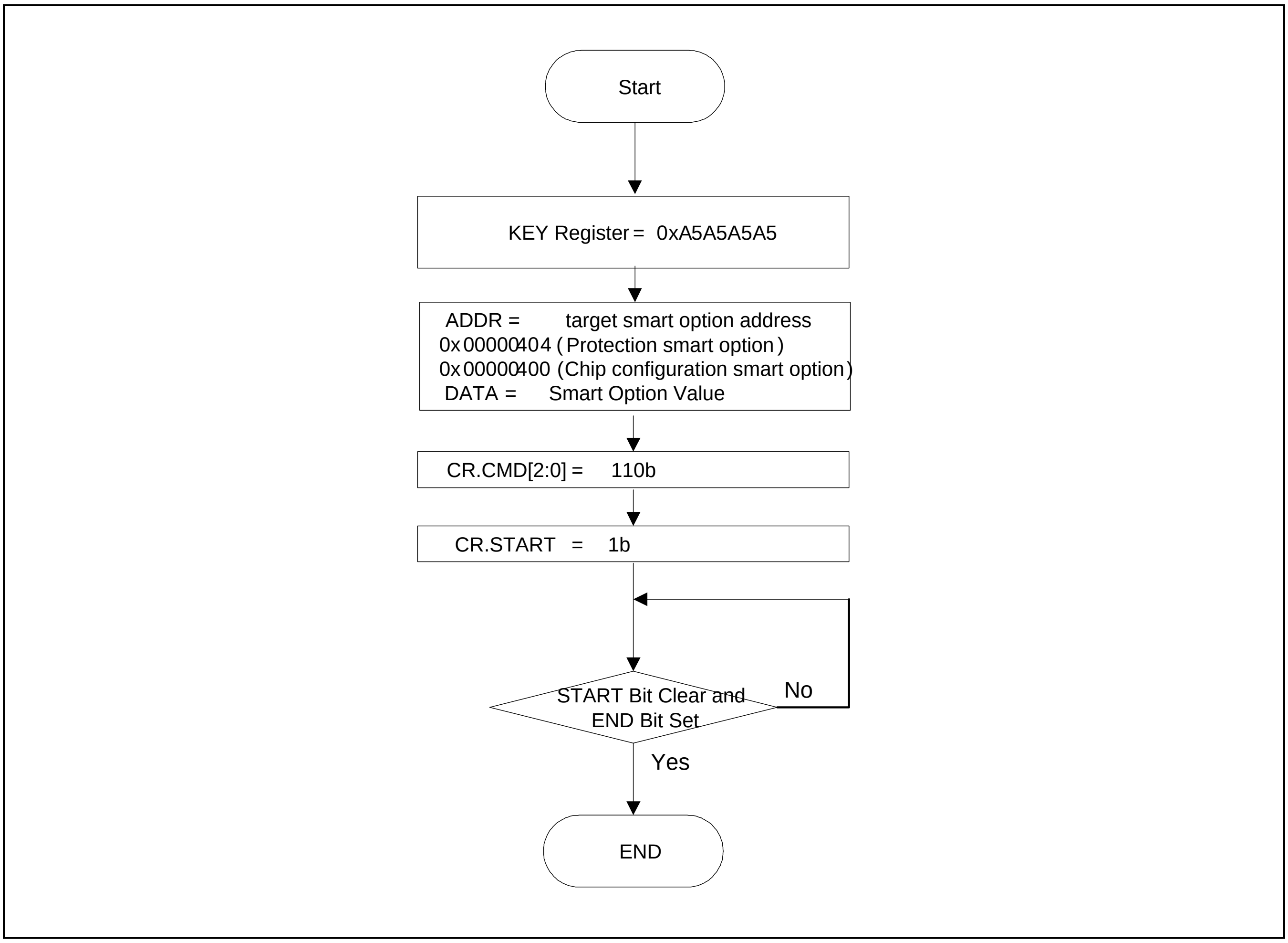

Start
KEY Register = 0xA5A5A5A5
ADDR = target smart option address
0x00000404 (Protection smart option)
0x00000400 (Chip configuration smart option)
DATA = Smart Option Value
CR.CMD[2:0] = 110b
CR.START = 1b
START Bit Clear and END Bit Set
No
Yes
END

Figure 19-10 Smart Option Erase Flow Chart

19.3.2.14 Error Case 0

When Code0 is not finished about writing and erasing and other program or erase operation is executed, error case 0 is occurred.

19.3.2.15 Error Case 1

When Code0 executes undefined command, error case 1 is occurred.

19.3.2.16 Error Case 2

When Code0 executes programming or erasing to protected area, error case 2 is occurred.

19.4 Register Description

19.4.1 Register Map Summary

- Base Address: 0x4001_0000

Register	Offset	Description	Reset Value
PF_IDR	0x0000	ID Register	0x0012_0019
PF_CEDR	0x0004	Clock Enable/Disable Register	0x0000_0000
PF_SRR	0x0008	Software Reset Register	0x0000_0000
PF_CR	0x000C	Control Register	0x0000_0000
PF_MR	0x0010	Mode Register	0x0000_0000
PF_IMSCR	0x0014	Interrupt Mask Set/Clear Register	0x0000_0000
PF_RISR	0x0018	Raw Interrupt Status Register	0x0000_0000
PF_MISR	0x001C	Masked Interrupt Status Register	0x0000_0000
PF_ICR	0x0020	Interrupt Clear Register	0x0000_0000
PF_SR	0x0024	Status Register	0x0000_0000
PF_AR	0x0028	Address Register	0x0000_0000
PF_DR	0x002C	Data Register	0x0000_0000
PF_KR	0x0030	Key Register	0x0000_0000
SO_PSR	0x0034	Smart Option Protection Status Register	0XFFFF_FFFF(1)
SO_CSR	0x0038	Smart Option Configuration Status Register	0xFFFF_FFFF(2)
PF_IOTR	0x003C	Internal OSC Trimming Register	0x00XX_XXXX

NOTE: (1) & (2) The reset value by fabrication is ‘0xFFFFFFFF’. But user can change using smart option program.

Caution: PF_CR, PF_AR, PF_DR, and PF_KR are auto-cleared as soon as finishing operation by command. Other writable registers except PF_IOTR can write when flash is not in running (Operation by command).

19.4.1.1 PF_IDR (P-Flash ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0012_0019

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	Identification Code Register This field stores the ID code for the corresponding IP.	0x0012_0019

19.4.1.2 PF_CEDR (P-Flash Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable Bit. 0 = Disable Flash Clock 1 = Enables Flash Clock Flash software reset does not affect CLKEN bit status. Flash clock controls to access SFR (special function register) for flash controller.	0

19.4.1.3 PF_SRR (P-Flash Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No effect 1 = Perform Flash Controller Software Reset operation. All registers except smart option, clock enable/disable and internal osc trimming registers are initialized.	0

19.4.1.4 PF_CR (P-Flash Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																									CMD[2:0]			RSVD			START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R	R	R	R W

Name	Bit	Type	Description	Reset Value
START	[0]	RW	Operation Start Bit. After start and finish, selected operation command and START bit will be clear automatically. 0 = No effect 1 = Start While executing command (normal program, page/sector/entire erase, smart option program, or erase), this bit can't be written to keep the sequence for an operation. That means it doesn't affect. But that becomes one of error cases. If it is written with a valid command, that will be an ERR0. If CMD[2:0] are 111'b and user writes 'START' bit, that will be an ERR1. User can catch that error occurred with RISR (status) and MISR (interrupt) register.	0
CMD[2:0]	[6:4]	RW	Flash Program/Erase Command Field. 000 = No Effect 001 = Select 'Normal Program' Command 010 = Select 'Page Erase' Command 011 = Select 'Sector Erase' Command 100 = Select 'Entire Flash Erase' Command 101 = Select 'Smart Option Program' Command 110 = Select 'Smart Option Erase' Command 111 = Prohibited Smart option erase operation makes both protection and configuration smart option clear. So user should read the other option value and rewrite to change only one smart option region.	000'b

NOTE: This register will be auto-clear when finishing operation by command.

1. This register isn’t written the following conditions. Although user writes any value in this register, leit doesn’t affect. User should check ‘END’ status (interrupt) to next command operation.
2. While operating by any command
3. When any bit in ERRx is ‘1’ (Error occurred) in RISR register
4. When the key value written before START is not valid
5. It supports the following program type - Entire Flash Erase, Sector Erase, Normal Program and Option Program.

Command	Description
NP	Normal Program
PE	Page Erase
SE	Sector Erase
CE	Entire Flash Erase
SOP	Smart Option Program
SOE	Smart Option Erase

6. The PF_CR can determine the program / erase operation. In user program mode, program flash controller can support normal program, smart option program, smart option erase, sector erase and entire flash erase. Among 6 operations, only one operation must be selected.

19.4.1.5 PF_MR (P-Flash Mode Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																							PBUFEN	RSVD		WAIT[1:0]		RSVD			BACEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R	R	R W	R W	R	R	R	R W

Name	Bit	Type	Description	Reset Value
BACEN	[0]	RW	Boot Area Erase Enable/Disable by Entire Program Memory Erase 0 = Disable 1 = Enable	0
WAIT[1:0]	[5:4]	RW	Program Flash Access Wait Cycle 00 = 1 Wait Cycle during the Programming Flash Read Access 01 = 2 Wait Cycle during the Programming Flash Read Access 10 = 3 Wait Cycle during the Programming Flash Read Access 11 = 4 Wait Cycle during the Programming Flash Read Access	00'b
PBUFEN	[8]	RW	Pre-Fetch Buffer Enable 0 = Disable Pre-fetch Buffer 1 = Enable Pre-fetch Buffer **NOTE:** PBUFEN must be enabled when the period of HCLK is greater than 50ns.	0

NOTE: WAIT[1:0] must be set the value which meets this equation: (WAIT[1:0] + 1) * 20MHz(50ns) ≥ the period HCLK.
WAIT[1:0] is no effect when PBUFEN is 0.
for example: HCLK = 75MHz, (WAIT[1:0] + 1) * 20MHz ≥ 75MHz, WAIT[1:0] = 3.

19.4.1.6 PF_IMSCR (P-Flash Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
END	[0]	RW	END Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	0
ERRn	[10:8]	RW	ERR Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	000'b

19.4.1.7 PF_RISR (P-Flash Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
END	[0]	RW	END Interrupt Mask Status 0 = Disable 1 = Enable	0
ERRn	[10:8]	RW	ERR Interrupt Mask Status 0 = Disable 1 = Enable	000'b

19.4.1.8 PF_MISR (P-Flash Raw Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
END	[0]	RW	END Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	0
ERRn	[10:8]	RW	ERR Interrupt Enable/Disable Bit 0 = Disable 1 = Enable	000'b

19.4.1.9 PF_ICR (P-Flash Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																					ERR2	ERR1	ERR0	RSVD							END
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
END	[0]	W	END Interrupt Clear Bit 0 = No effect 1 = Clear Interrupt	0
ERRn	[10:8]	W	ERR Interrupt Clear Bit 0 = No effect 1 = Clear Interrupt	000'b

19.4.1.10 PF_SR (P-Flash Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																															BUSY
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
BUSY	[0]	R	Busy Status Flag 0 = Not in operating (Programming or Erasing) 1 = Reported flag when program flash is programming or erasing.	0

19.4.1.11 PF_AR (P-Flash Address Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
ADDR[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ADDR[31:0]	[31:0]	RW	Program Flash Memory Address to program(Normal/Smart Option) or erase(Chip/Sector)	0x0000_0000

NOTE:

1. PF_AR [31:0] ← Address to be selected by user in flash memory range
2. PF_AR [31:0] ← 0x00000400 on programming configuration smart option
3. PF_AR [31:0] ← 0x00000E3C on programming protection smart option

19.4.1.12 PF_DR (P-Flash Data Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
DATA[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA[31:0]	[31:0]	RW	Data to write on the target address (PF_AR) of the program flash memory.	0x0000_0000

NOTE:

1. PFDATA [31:0] ← Specific word data (4bytes) selected by user to be written into the flash memory
2. PFDATA [31:0] ← Trimming data in case of configuration smart option
3. PFDATA [31:0] ← Protection option data in case of protection smart option

19.4.1.13 PF_KR (P-Flash Key Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
KEY[31:0]																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
KEY[31:0]	[31:0]	RW	Key register value for program, erase operation To program any data into the flash memory or erase by the user program mode, a specific key register with 0x5A5A5A5A is required to prevent flash data from being destroyed under undesired situations. 'START' in the control register can't be written on pre-condition by a non-valid key value. If the sequence for a flash operation has the invalid key value, not 0x5a5a5a5a, it has no-effect.	0x0000_0000

NOTE: The PF_KR register will be cleared automatically just after the completion of erase or program.

19.4.1.14 SO_PSR (Smart Option Protection Status Register)

- Address = Base Address + 0x0034, Reset Value = 0xFFFF_FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				nRDP	RSVD			HWPA11	HWPA10	HWPA9	HWPA8	RSVD		nHWP	RSVD	HWPA7	HWPA6	HWPA5	HWPA4	RSVD			nJTAGP	HWPA3	HWPA2	HWPA1	HWPA0	RSVD			
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
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
nJTAGP	[8]	R	JTAG Protection Status Flag 0 = JTAG Protection is enabled. 1 = JTAG Protection is disabled.	1
nHWP	[17]	R	Hardware Protection Status Flag 0 = Hardware Protection is enabled. 1 = Hardware Protection is disabled.	1
nRDP	[27]	R	Read Protection Status Flag 0 = Serial Read Protection is enabled. 1 = Serial Read Protection is disabled.	1
HWPAn	[7:4] [15:12] [23:20]	R	Hardware Protection Area n Area is a unit to be consist of 0 = Area to be protected by Hardware Protection Smart Option. 1 = Area not to be protected because hardware protection for corresponding area is disabled.	0xFFF

User can read what value smart option about a protection is programmed with. If program smart option and chip reset don’t occur, the value to be read via this register is different the current smart option. Because the programmed smart option (new option) will be effected after chip reset.

The reset value by fabrication is ‘0xFFFFFFFF’. But user can change using smart option program

19.4.1.15 SO_CSR (Smart Option Configuration Status Register)

- Address = Base Address + 0x0038, Reset Value = 0xFFFF_FFFF

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

Name	Bit	Type	Description	Reset Value
POCCS	[1:0]	R	Power-On System Clock Source Selection Field When power-on, the selected clock source by smart option is used for operation (CPU Clock). 00 = External Sub-Clock, 32.768KHz (ESCLK: External Sub-Clock) 01 = External Main Clock, 4~8MHz (EMCLK: External Main Clock) 10 = Internal Sub-Clock, 32.768KHz (ISCLK: Internal Sub-Clock) 11 = Internal Main Clock (IMCLK: Internal Main Clock)	11'b
IMSEL	[7:6]	R	Internal Main OSC Frequency Selection Field 00 = Not used 01 = 8MHz 10 = 16MHz 11 = 20MHz	11'b
BTDIV	[15:12]	R	Basic timer divider selection bit in the reset time 0000 = Not used 0001 = Not used 0010 = Not used 0011 = 1 0100 = 2 0101 = 4 0110 = 8 0111 = 16 1000 = 32 1001 = 64 1010 = 128 1011 = 256 1100 = 512 1101 = 1024 1110 = 2048 1111 = 4096	0xF

Control bits have the relationship with clock manager for chip operation. The smart option value programmed by user can change the reset value of a clock manager. If program smart option and chip reset don’t occur, the value to be read via this register is different the current smart option. Because the programmed smart option (new option) will be effected after chip reset.

The reset value by fabrication is ‘0xFFFFFFFF’. But user can change using smart option program. If you programs with target value to change a configuration option, other bits except to control bits (field) should be ‘1’.

19.4.1.16 PF_IOTR (Internal OSC Trimming Register)

- Address = Base Address + 0x003C, Reset Value = 0x00XX_XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOTKEY[7:0]								RSVD		OSC[25:0]						RSVD	OSC1[6:1]							RSVD	OSC0[6:0]						
0	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	0	X	X	X	X	X	X	X	0	X	X	X	X	X	X	X
W	W	W	W	W	W	W	W	R	R	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W	R	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
OSC0[6:0]	[6:0]	RW	Internal Oscillator 0 (20MHz) Trim Value This field can use for user trimming value on changed condition (Voltage, Temperature, and so on).	Undefined
OSC1[6:0]	[14:8]	RW	Internal Oscillator 1(8/16MHz) Trim Value This field can use for user trimming value on changed condition (Voltage, Temperature, and so on).	Undefined
OSC2[5:0]	[21:16]	RW	Internal Oscillator 2(32.768KHz) Trim Value This field can use for user trimming value on changed condition (Voltage, Temperature, and so on).	Undefined
IOTKEY[7:0]	[31:24]	W	Key for write access into the PF_IOTR register Any write in PF_IOTR register bits is only effective if the IOTKEY is equal to 0x53.	0x00

Reset value is dependent on trimming value by fabrication.

20 Pulse Width Modulation (PWM)

20.1 Overview

This document defines the pulse width modulation (PWM) for the general. This module generates a pulse with the programmable width and frequency. PWM can be the 22-bit resolution including 6-bit extension bit.

20.1.1 Features

- A control bit to take into account the new configuration:
  - Programmable duty cycle and frequency
  - Programmable active level
- Enable/disable the PWM output signal:
  - Programmable idle level
- Enable/disable the interrupt sources:
  - Pulse start
  - Period end
  - Pulse match
- PWM extension function by an external cycle circuit

20.1.2 Pin Description

Table 20-1 PWM Pin Description

Pin Name	Function	I/O Type	Comments
PWM[7:0]	Pulse width modulation output	O	–

20.2 Functional Description

20.2.1 Block Diagram

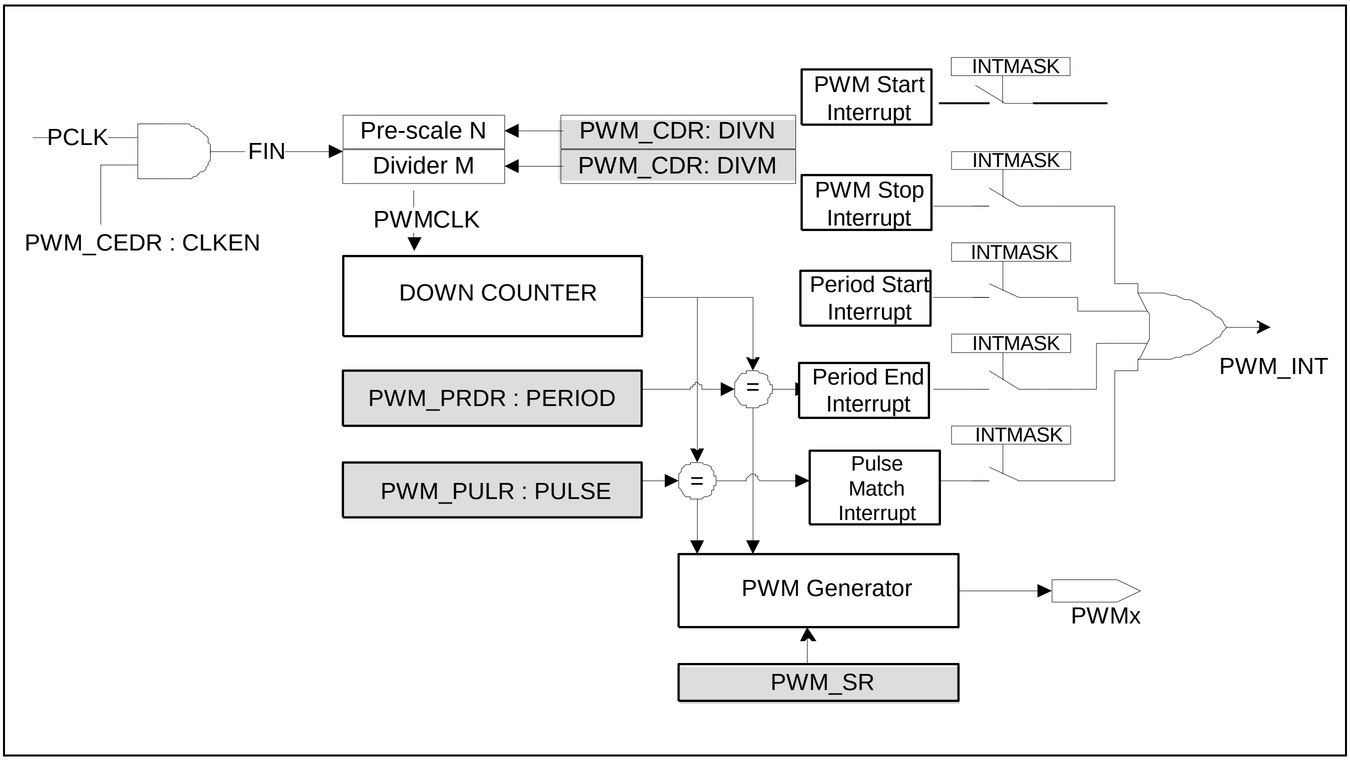


Figure 20-1 Pulse Width Modulation (PWM) Block Diagram

20.2.2 General Description

The PWM generates the periodic waveform. The waveform consist s of ACTIVE and nACTIVE width.

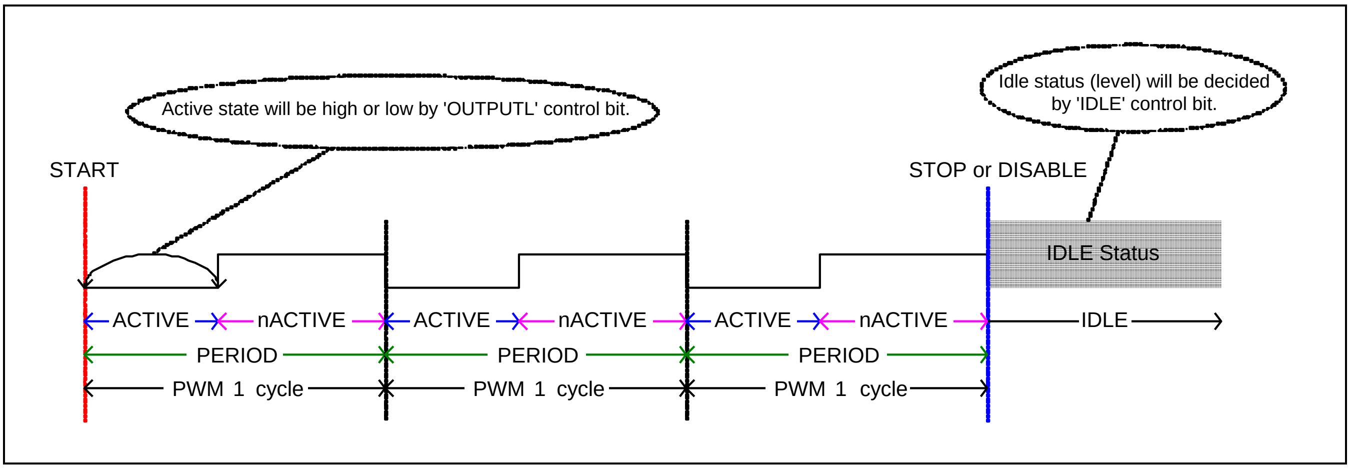


Figure 20-2 PWM Cycle Description

- PERIOD: Period Width
  User can define the periodic width (time) with PERIOD[15:0] field in the PWM_PRDR register.
- ACTIVE: Active Width
  The active width (time) can be calculated by (Period–PULSE[15:0]) value.
- nACTIVE: nActive Width
  This width (time) can be decided by PULSE[15:0] field value in the PWM_PULR register.
- IDLE: Idle Level
  The idle level can be decided by IDLEL bit in the PWM_CSR/PWM_CCR register.
- OUTPUTL: Output Level
  This is the level for an active width. It means a start level at each period and is defined by OUTPUTL bit in the PWM_CSR/PWM_CCR register.

The ‘CLKEN’ bit in the PWM_CEDR (Clock Enable/Disable Register) register enables/disables the PWM clock. The module is reset by MCU chip resets or by the IP software reset via the SWRST bit in the PWM_SRR (Software Reset Register) register.

When user stops or disables PWM, PWM will stop after generation until already started period (cycle).

Caution: The UPDATE bit in the PWM_CSR/PWM_CCR register allows to take into account the new parameters configuration immediately if the PWM channel is disabled or at the end of “PERIOD” cycle if the PWM channel is enabled.

20.2.3 Clock and Operation Frequency

PWM clock source is PCLK (Peripheral Clock). PWM operation clock frequency is decided by divider value M and N in PWM_CDR (Clock Divider Register).

PWM Counter Clock is defined as follow

- PWM Counter Clock Frequency, PWMCLK = PCLK / 2^N / (M+1)

$$0 \le N < 16, \qquad 0 \le M < 2^{12}$$

20.2.4 Period

The "PERIOD" cycle represents the periodic PWM output. The PERIOD field in the PWM_PRDR register controls the number of down-counter cycles to fix the period.

- PERIOD[15:0] for the 16-bit PWM period width

If the PERIOD field is configured at 0, the PWM output is driven at logical level configured by the IDLEL (Idle Level) bit. The PWM channel is automatically disabled after setting the UPDATE bit.

20.2.5 Active Level

The level of the active width is controlled by the OUTSL bit to fix the PWM output active. The OUTSL bit affects the PWM output start level.

If OUTSL bit is set to '0', the ACTIVE width level is low and the nACTIVE width level is high. On the other hands, if OUTSL bit is set to '1', the ACTIVE width level is high and the nACTIVE width becomes low.

20.2.6 nActive Witdh

The PWM output starts with an ACTIVE width when the PWM channel is enabled. At the end of ACTIVE width, the nACTIVE width starts. The PWM output finishes with a nACTIVE state when the PWM channel is disabled.

The nACTIVE width represents the PWM output nACTIVE state. The PULSE field in the PWM_PULR register controls the number of down-counter cycles to fix nACTIVE width.

- PULSE[15:0] for the 16-bit PWM nACTIVE width

PMATCH bit in the PWM_RISR register indicates the start of nACTIVE width and PEND bit indicates the "PERIOD" cycle end. Both status bits are cleared via the appropriate bit of PWM_CSR register. The software has the possibility to enable/disable the PSTA and/or PEND interrupts.

If the PULSE field is configured at 0, the "PULSE" cycle does not exist. The PWM output is driven at a logical level configured by the OUTPUTL bit as long as the PWM channel is enabled. For this particular case, the new configuration set by the UPDATE bit is taken into account at the end of "PERIOD" cycle.

20.2.7 Idle Level

The idle level is controlled by the IDLEL bit to fix the PWM output when the PWM stops via ‘START’ bit in the PWM_CCR (Control Clear Register) register or clock is disabled by ‘CLKEN’ bit in the PWM_CEDR (Clock Enable/Disable Register). The PWM is effectively disabled (idle level) at the end of “PERIOD” cycle in progress.

20.2.8 Parameter Relationship

Table 20-2 PERIOD and PULSE Field Relationship

Configuration		Normal Mode	Interval Mode
PERIOD Value	**PULSE Value**	**PWMx Output**	**PWMx Output**
PERIOD[15:0] > PULSE[15:0]		Normal PWM function	Interval PWM function
[0x1:0xFFFF]	0	OUTPUTL bit value	Interval PWM function

NOTE: Above a table, PERIOD[15:0] and PULSE[15:0] have the value over ‘0’. That means the value dose not include 0. The case having ‘0’ value is defined as other cases. In case of an interval mode, if PERIOD is not ‘0’ regardless of PULSE, it’s possible to operate.
The following configurations are prohibited.
PERIOD[15:0] = 0, PERIOD[15:0] = PULSE[15:0], or PERIOD[15:0] < PULSE[15:0]

20.2.9 Extension Bit

The value in the extension counter is compared with the extension settings in the extension bits. This extension counter value, together with extension logic and the PWM module's extension bits, is then used to "stretch" the duty cycle of the PWM output. The "stretch" value is one extra clock period at specific intervals, or cycles.

If, for example, in 8-bit base and 6-bit extension, the value of PWMEX.0 is ‘1’, the 32nd cycle will be one pulse longer than the other 63 cycles. If the base duty cycle is 50%, the duty of the 32nd cycle will therefore be "stretched" to approximately 51% duty.

PWMEX Bit	“Stretched” Cycle Number
PWMEX0	32
PWMEX1	16, 48
PWMEX2	8, 24, 40, 56
PWMEX3	4, 12, 20, 28, 36, 44, 52, 60
PWMEX4	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
PWMEX5	1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63

For example, if PWMEX[4:0] bits are ‘1’, all odd-numbered cycles will be one pulse longer. If all extension bits, PWMEX[5:0], set to ‘1’, all cycles will be stretched by one pulse except the 64th cycle. In this way, you can obtain high output resolution at high frequencies.

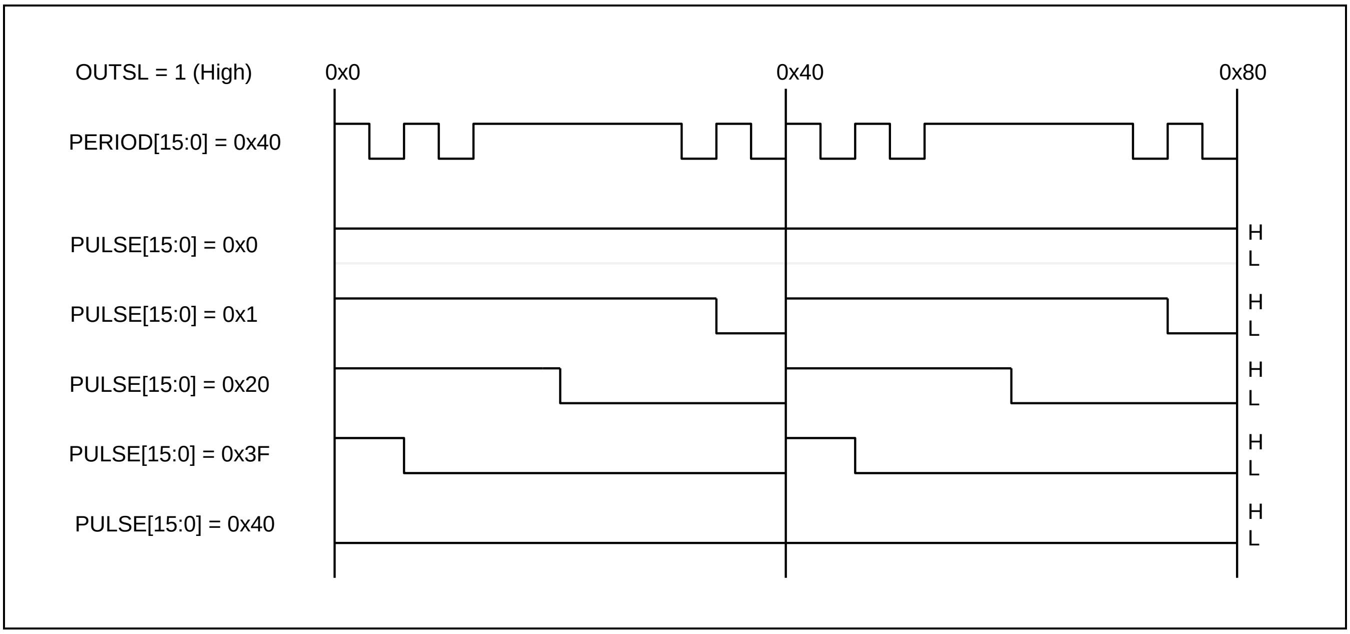


Figure 20-3 PWM Basic Waveform (OUTSL=1, PWM Period=0x40, Pulse=0x0, 0x1, 0x20, 0x3F, 0x40)

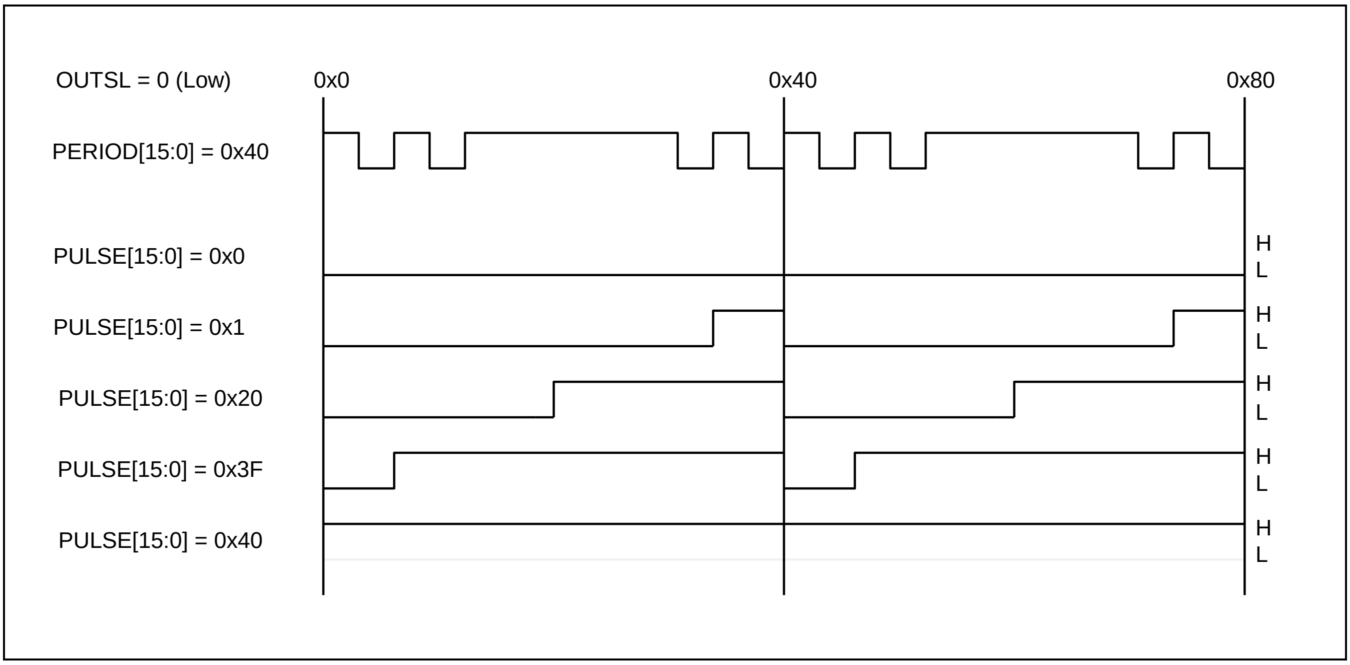


Figure 20-4 PWM Basic Waveform (OUTSL=0, PWM Period=0x40, Pulse=0x0, 0x1, 0x20, 0x3F, 0x40)

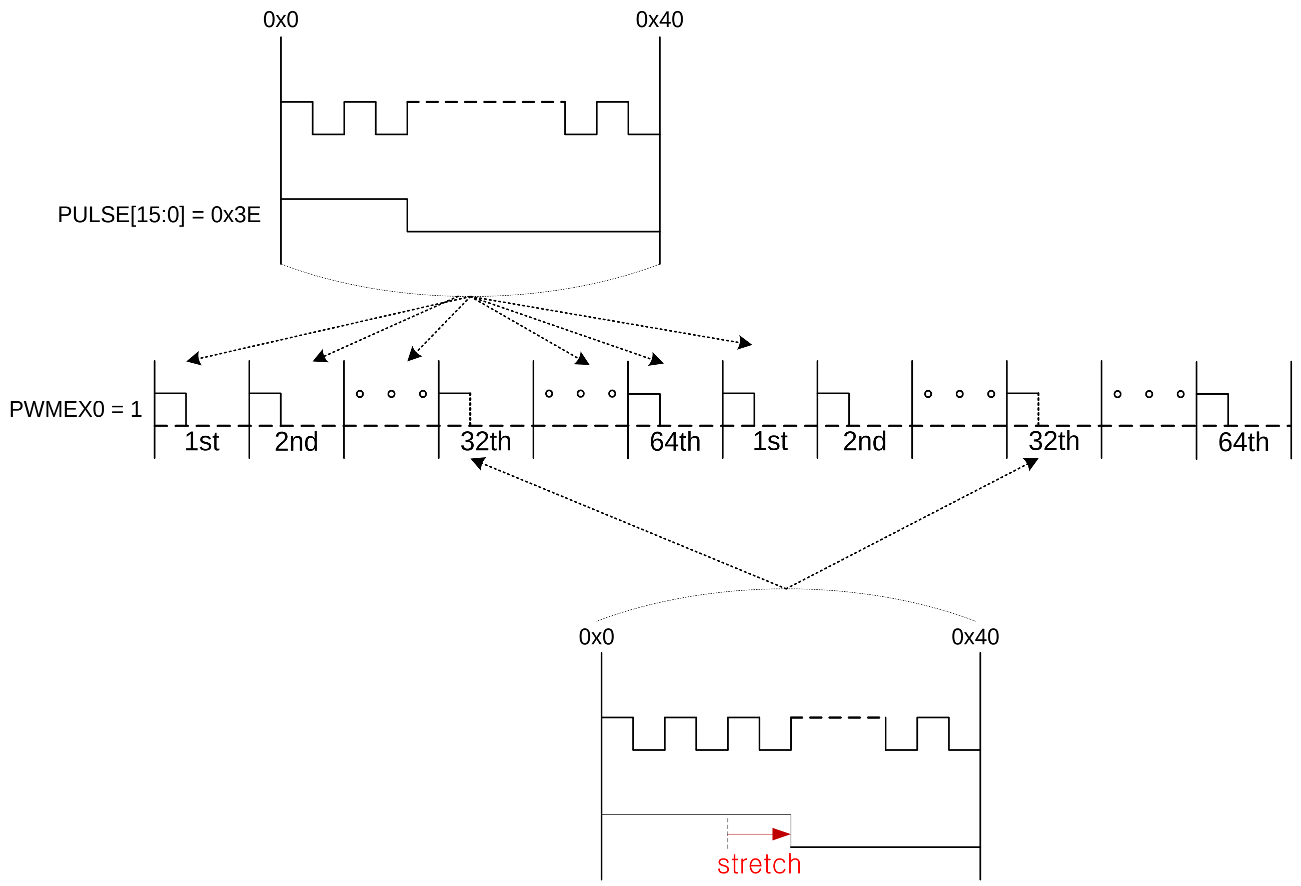


Figure 20-5 Extended PWM Waveform (PWM Period = 0x40, Pulse=0x3E, PWMEX0)

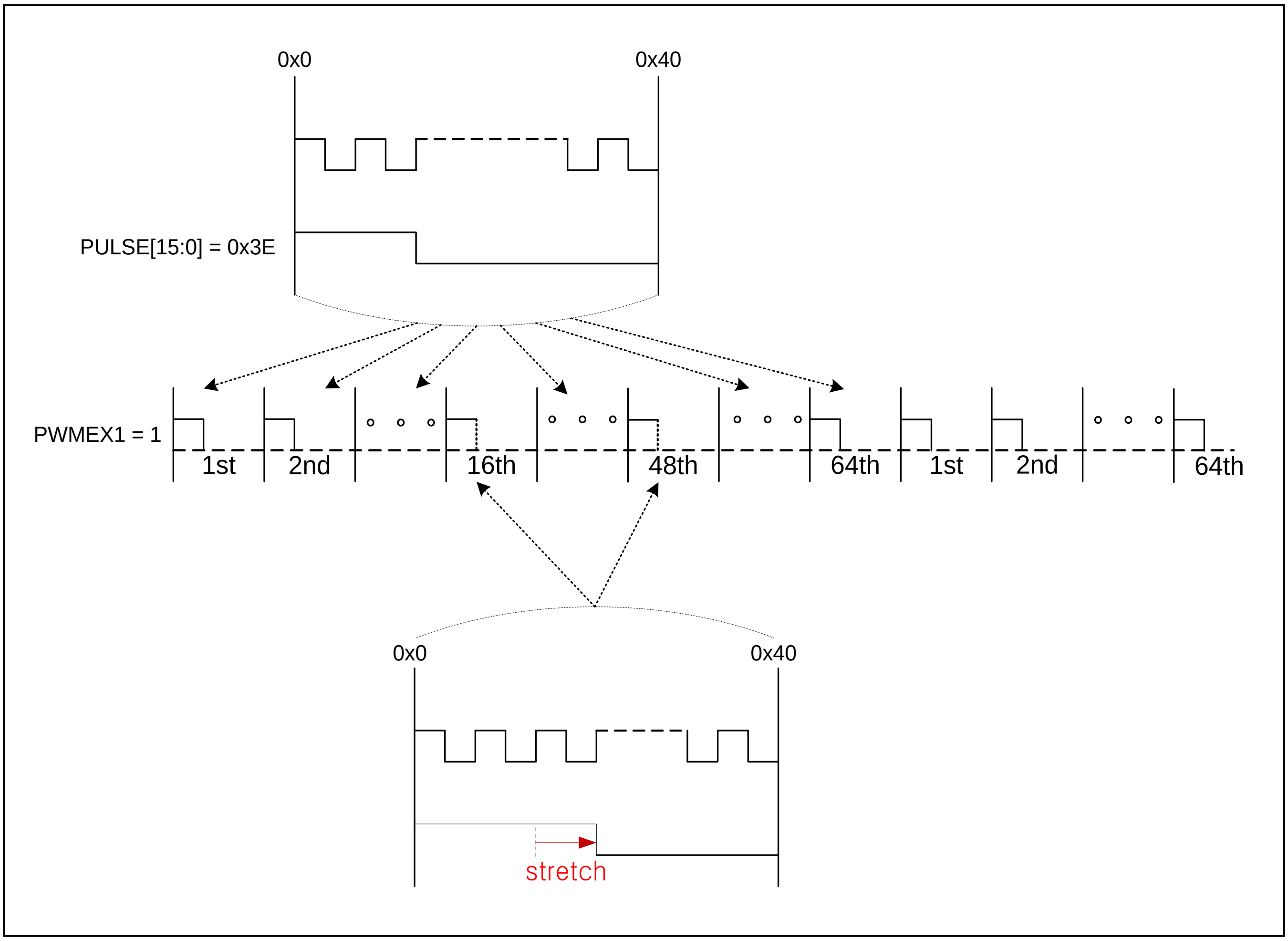


Figure 20-6 Extended PWM Waveform (PWM Period = 0x40, Pulse=0x3E, PWMEX1)

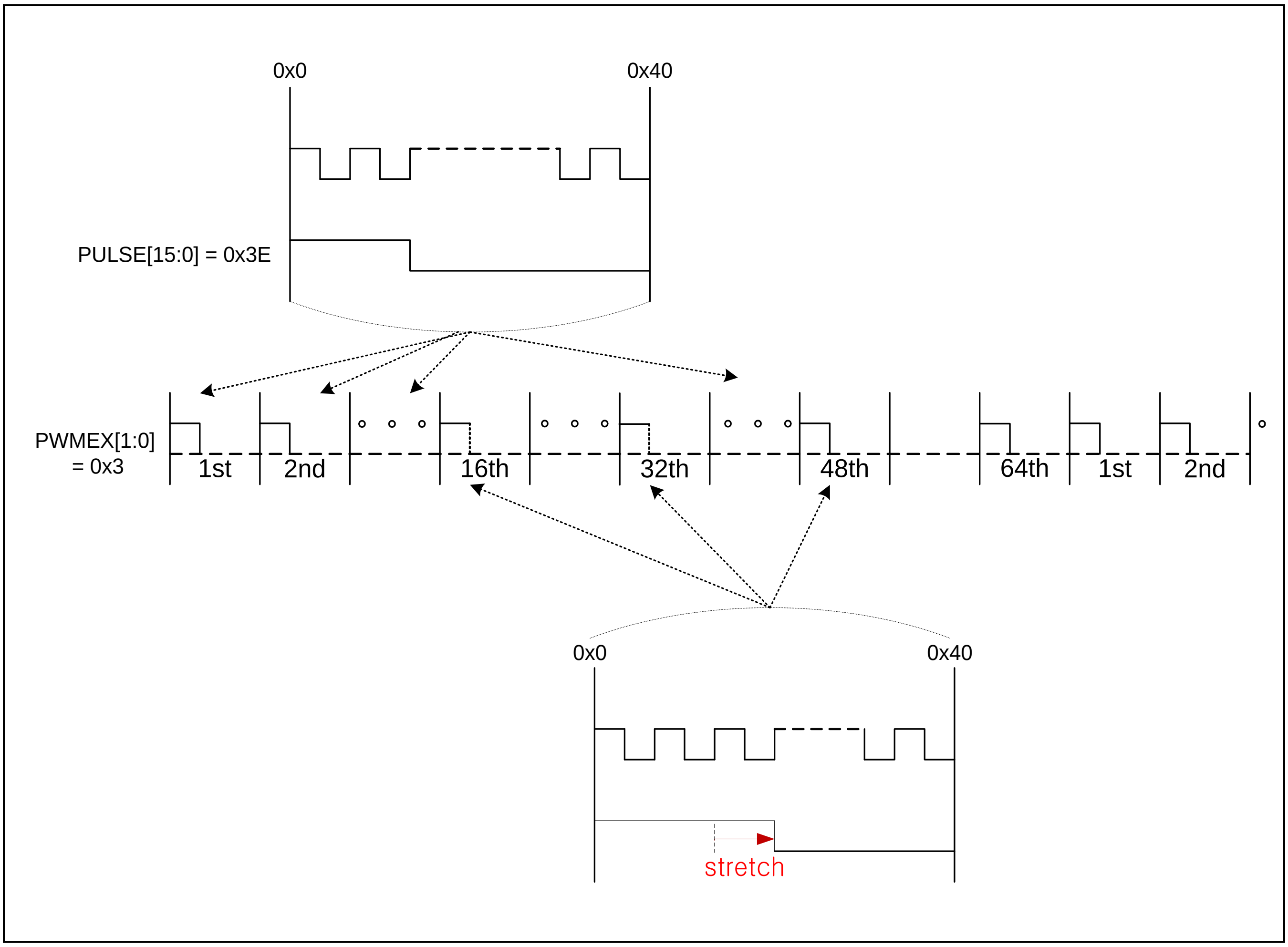


Figure 20-7 Extended PWM Waveform (PWM Period = 0x40, Pulse=0x3E, PWMEX1 and PWMEX

20.3 Register Description

20.3.1 Register Map Summary

- Base Address: 0x4007_0000
- Base Address: 0x4007_1000
- Base Address: 0x4007_2000
- Base Address: 0x4007_3000
- Base Address: 0x4007_4000
- Base Address: 0x4007_5000
- Base Address: 0x4007_6000
- Base Address: 0x4007_7000

Register	Offset	Description	Reset Value
PWM_IDR	0x0000	ID Register	0x0001_0009
PWM_CEDR	0x0004	Clock Enable/Disable Register	0x0000_0000
PWM_SRR	0x0008	Software Reset Register	0x0000_0000
PWM_CSR	0x000C	Control Set Register	0x0000_0000
PWM_CCR	0x0010	Control Clear Register	0x0000_0000
PWM_SR	0x0014	Status Register	0x0000_0000
PWM_IMSCR	0x0018	Interrupt Mask Set/Clear Register	0x0000_0000
PWM_RISR	0x001C	Raw Interrupt Status Register	0x0000_0000
PWM_MISR	0x0020	Masked Interrupt Status Register	0x0000_0000
PWM_ICR	0x0024	Interrupt Clear Register	0x0000_0000
PWM_CDR	0x0028	Clock Divider Register	0x0000_0000
PWM_PRDR	0x002C	Period Register	0x0000_0000
PWM_PULR	0x0030	Pulse Register	0x0000_0000
PWM_CCDR	0x0034	Current Clock Divider Register	0x0000_0000
PWM_CPRDR	0x0038	Current Period Register	0x0000_0000
PWM_CPULR	0x003C	Current Pulse Register	0x0000_0000

20.3.1.1 PWM_IDR (PWM ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_0009

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_0009

20.3.1.2 PWM_CEDR (PWM Clock Enable/Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable Bit. 0 = Disable PWM Clock 1 = Enables PWM Clock PWM software reset does not affect CLKEN bit status.	0
DBGEN	[31]	RW	Debug mode enable. 0 = Disable the debug mode The DEBUG ACKNOWLEDGE generated by the ICE interface has no influence on the PWM function. 1 = Enable the debug mode The DEBUG ACKNOWLEDGE freezes the PWM function when the debugger interface is activated. However full read/write access to internal register is kept for the debug purpose.	0

20.3.1.3 PWM_SRR (PWM Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No effect 1 = Perform PWM Software Reset operation. If CLKEN bit in PWM_CEDR is "1", this bit is auto cleared. Otherwise user should clear this bit.	0

20.3.1.4 PWM_CSR (PWM Control Set Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD												PWMIM	KEEP	OUTSL	IDLESL	RSVD						UPDATE	START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	R	R	R	R	R	R	W	W

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>START</td><td>[0]</td><td>W</td><td>Start PWM.
0 = No effect
1 = Start PWM Operation</td><td>0</td></tr>
<tr><td>UPDATE</td><td>[1]</td><td>W</td><td>Update PWM Parameter.
0 = No effect
1 = Update PWM Parameter (Period, Pulse, Clock Divider)</td><td>0</td></tr>
<tr><td>IDLESL</td><td>[8]</td><td>W</td><td>Idle State Level.
0 = No effect
1 = Idle State PWM Output Level is High (Logic-1)</td><td>0</td></tr>
<tr><td>OUTSL</td><td>[9]</td><td>W</td><td>PWM Output Start Level.
0 = No effect
1 = PWM Output Level Starts from High (Logic-1) for the given PERIOD.</td><td>0</td></tr>
<tr><td>KEEP</td><td>[10]</td><td>W</td><td>Keep Last Period State.
0 = No effect
1 = PWM Output level is kept when PWM stops.
In this case, IDLELEVEL is not effective.</td><td>0</td></tr>
<tr><td>PWMIM</td><td>[11]</td><td>W</td><td>PWM Interval Mode.
0 = No effect
1 = PWM Mode. PWM phase will be toggled when period matches</td><td>0</td></tr>
<tr><td>PWMEX0
PWMEX1
PWMEX2
PWMEX3
PWMEX4
PWMEX5</td><td>[24]
[25]
[26]
[27]
[28]
[29]</td><td>W</td><td>PWM Extension Control Bit.
0 = No effect
1 = Include the corresponding stretched cycle number
<table>
<tr><th>PWMEXy</th><th>“Stretched” Cycle Number</th></tr>
<tr><td>PWMEX0</td><td>32</td></tr>
<tr><td>PWMEX1</td><td>16, 48</td></tr>
<tr><td>PWMEX2</td><td>8, 24, 40, 56</td></tr>
<tr><td>PWMEX3</td><td>4, 12, 20, 28, 36 , 44, 52, 60</td></tr>
</table></td><td>0</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td><table><tr><td>PWMEX4</td><td>2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62</td></tr><tr><td>PWMEX5</td><td>1, 3, 5, 7, 9,11,13,15,17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63</td></tr></table>
If PWMEX0 and PWMEX3 set to ‘1’, PWM inserts extra 1-cycle at (32) and (4, 12, 20, 28, 36, 44, 52, 60).</td><td></td></tr>
</table>

20.3.1.5 PWM_CCR (PWM Control Clear Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD												PWMIM	KEEP	OUTSL	IDLESL	RSVD							START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	R	R	R	R	R	R	R	W

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>START</td><td>[0]</td><td>W</td><td>Start PWM.
0 = No effect
1 = Stop PWM Operation</td><td>0</td></tr>
<tr><td>IDLESL</td><td>[8]</td><td>W</td><td>Idle State Level.
0 = No effect
1 = Idle State PWM Output Level is Low (Logic-0)</td><td>0</td></tr>
<tr><td>OUTSL</td><td>[9]</td><td>W</td><td>PWM Start Level.
0 = No effect
1 = PWM Output Level Starts from Low (Logic-0) for the given PERIOD.</td><td>0</td></tr>
<tr><td>KEEP</td><td>[10]</td><td>W</td><td>Keep Last Period State.
0 = No effect
1 = PWM Output level is kept as last period output level when PWM stops. In this case, IDLELEVEL is ignored.</td><td>0</td></tr>
<tr><td>PWMIM</td><td>[11]</td><td>W</td><td>PWM Interval Mode.
0 = No effect
1 = Interval Mode. PWM phase will be toggled every period.</td><td>0</td></tr>
<tr><td>PWMEX0
PWMEX1
PWMEX2
PWMEX3
PWMEX4
PWMEX5</td><td>[24]
[25]
[26]
[27]
[28]
[29]</td><td>W</td><td>PWM Extension Control Bit.
0 = No effect
1 = Delete the corresponding stretched cycle number
<table>
<tr><th>PWMEXy</th><th>“Stretched” Cycle Number</th></tr>
<tr><td>PWMEX0</td><td>32</td></tr>
<tr><td>PWMEX1</td><td>16, 48</td></tr>
<tr><td>PWMEX2</td><td>8, 24, 40, 56</td></tr>
<tr><td>PWMEX3</td><td>4, 12, 20, 28, 36 , 44, 52, 60</td></tr>
<tr><td>PWMEX4</td><td>2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62</td></tr>
<tr><td>PWMEX5</td><td>1, 3, 5, 7, 9,11,13,15,17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53,</td></tr>
</table>
</td><td>0</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td></td><td><table><tr><td></td><td>55, 57, 59, 61, 63</td></tr></table>
NOTE: PWMEXy can be controlled by a combination with 6 control bits.
• extension case:
(PWMEX0=1), (PWMEX1=1), (PWMEX2=1), (PWMEX3=1), (PWMEX4=1), (PWMEX5=1)
• extension case:
(PWMEX0 =1 and PWMEX5 = 1),
(PWMEX2 =1 and PWMEX3 =1) etc.
• extension case:
(PWMEX1 = 1, PWMEX3 = 1, PWMEX4 = 1),
(PWMEX0 = 1, PWMEX2 = 1, PWMEX5 = 1) etc.</td><td></td></tr>
</table>

20.3.1.6 PWM_SR (PWM Status Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD												PWMIM	KEEP	OUTSL	IDLESL	RSVD							START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
START	[0]	R	PWM start/stop status. 0 = PWM Stopped 1 = PWM Started	0
IDLESL	[8]	R	Idle State PWM Output Level Status. 0 = Idle State PWM Output Level is Low (Logic-0) 1 = Idle State PWM Output Level is High (Logic-1)	0
OUTSL	[9]	R	PWM Output Start Level Status. 0 = PWM Output Level Starts from Low (Logic-0) for the given PERIOD. 1 = PWM Output Level Starts from High (Logic-1) for the given PERIOD.	0
KEEP	[10]	R	Keep Last Period Status. 0 = PWM Output level is determined by IDLELEVEL when PWM stops. 1 = PWM Output level is kept as last period output level when PWM stops. In this case, IDLELEVEL is ignored.	0
PWMIM	[11]	R	PWM Output Mode Status. 0 = PWM Mode 1 = Interval Mode	0
PWMEX0 PWMEX1 PWMEX2 PWMEX3 PWMEX4 PWMEX5	[24] [25] [26] [27] [28] [29]	R	6-bit PWM Extension. Bit-control. 0 = PWM Extension is not effective to the corresponding bit-field. 1 = PWM Extension is effective to the corresponding bit-field.	0

20.3.1.7 PWM_IMSCR (PWM Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																											PMATCH	PEND	PSTART	PWMSTOP	PWMSTART
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PWMSTART	[0]	RW	PWM Start Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PWMSTOP	[1]	RW	PWM Stop Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PSTART	[2]	RW	Period Start Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PEND	[3]	RW	Period End Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
PMATCH	[4]	RW	Pulse Match Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

20.3.1.8 PWM_RISR (PWM Raw Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																											PMATCH	PEND	PSTART	PWMSTOP	PWMSTART
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PWMSTART	[0]	R	PWM Start Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PWMSTART interrupt.	0
PWMSTOP	[1]	R	PWM Stop Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PWMSTOP interrupt.	0
PSTART	[2]	R	Period Start Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PSTART interrupt.	0
PEND	[3]	R	Period End Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PEND interrupt.	0
PMATCH	[4]	R	Pulse Match Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the PMATCH interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

20.3.1.9 PWM_MISR (PWM Masked Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																											PMATCH	PEND	PSTART	PWMSTOP	PWMSTART
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PWMSTART	[0]	R	PWM Start Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PWMSTART interrupt.	0
PWMSTOP	[1]	R	PWM Stop Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PWMSTOP interrupt.	0
PSTART	[2]	R	Period Start Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PSTART interrupt.	0
PEND	[3]	R	Period End Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PEND interrupt.	0
PMATCH	[4]	R	Pulse Match Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PMATCH interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

20.3.1.10 PWM_ ICR (PWM Interrupt Clear Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																											PMATCH	PEND	PSTART	PWMSTOP	PWMSTART
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
PWMSTART	[0]	W	PWM Started Interrupt clear. 0 = No effect 1 = Clear PWM Started interrupt	0
PWMSTOP	[1]	W	PWM Stopped Interrupt clear. 0 = No effect 1 = Clear PWM Stopped interrupt	0
PSTART	[2]	W	PWM Period Started Interrupt clear. 0 = No effect 1 = Clear PWM Period Started interrupt	0
PEND	[3]	W	PWM Period Ended Interrupt clear. 0 = No effect 1 = Clear PWM Period Ended interrupt	0
PMATCH	[4]	W	PWM Counter value is matched with Pulse value Status. 0 = No effect 1 = Clear Pulse match interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

20.3.1.11 PWM_CDR (PWM Clock Divider Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																	DIVM											DIVN			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DIVN	[3:0]	RW	Clock Divider for CLK / 2^N (0 < = N < 16)	0x0
DIVM	[14:4]	RW	Clock Divider for CLK / (M + 1) (0 < = M < 2^{12}) max	0x000

20.3.1.12 PWM_PRDR (PWM Period Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																PERIOD															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PERIOD	[15:0]	RW	PWM Period Value. To generate PWM waveform, PERIOD[15:0] should not be ‘0’ and greater than PULSE[15:0].	0x0000

20.3.1.13 PWM_PULR (PWM Pulse Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																PULSE															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PULSE	[15:0]	RW	PWM Pulse Value. To generate PWM waveform, PULSE[15:0] should not be ‘0’.	0x0000

20.3.1.14 PWM_CCDR (PWM Current Clock Divider Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																	DIVM											DIVN			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DIVN	[3:0]	R	Clock Divider for CLK / 2^N (0 < = N < 16)	0x0
DIVM	[14:4]	R	Clock Divider for CLK / (M + 1) (0 < = M < 2^{12}) max	0x000

20.3.1.15 PWM_CPRDR (PWM Current Period Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																PERIOD															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PERIOD	[15:0]	R	PWM Current Period Value. This field shows the operating pwm’s current period value.	0x0000

20.3.1.16 PWM_CPULR (PWM Current Pulse Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																PULSE															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PULSE	[15:0]	R	PWM Pulse Value. This field shows the operating pwm’s current pulse value.	0x0000

21 Serial Peripheral Interface (SPI)

21.1 Overview

The PrimeCell Synchronous Serial Port (SSP, PL022) is used for serial peripheral interface.

About the ARM PrimeCell SSP (PL022)

The PrimeCell Synchronous Serial Port (SSP) is an Advanced Microcontroller Bus Architecture (AMBA) slave block that connects to the Advanced Peripheral Bus (APB). The PrimeCell SSP is an AMBA compliant System-on-Chip (SOC) peripheral that is developed, tested, and licensed by ARM.

21.1.1 Features

The PrimeCell SSP is a master or slave interface that enables synchronous serial communication with slave or master peripherals having one of the following:

- A Motorola SPI-compatible interface.

In both master and slave configurations, the PrimeCell SSP performs:

- Parallel-to-serial conversion on data written to an internal 16-bit wide, 8-location deep transmit FIFO.
- Serial-to-parallel conversion on received data, buffering it in a similar 16-bit wide, 8-location deep receive FIFO.

Interrupts are generated to:

- Request servicing of the transmit and receive FIFO.
- Inform the system that a receive FIFO over-run has occurred.
- Inform the system that data is present in the receive FIFO after an idle period has expired.

21.1.1.1 Features of the PrimeCell SSP

The PrimeCell SSP has the following features:

- Compliance to the AMBA Specification (Rev 2.0) for easy integration into SOC implementation.
- Master or slave operation.
- Programmable clock bit rate and prescale.
- Separate transmit and receive first-in, first-out memory buffers, 16 bits wide, 8 locations deep.
- Programmable data frame size from 4 to 16 bits.
- Independent masking of transmit FIFO, receive FIFO, and receive overrun interrupts.
- Internal loopback test mode available.

21.1.1.2 Programmable Parameters

The following parameters are programmable:

- Master or slave mode.
- Enabling of operation.
- Frame format.
- Communication baud rate.
- Clock phase and polarity.
- Data widths from 4 to 16 bits wide.
- Interrupt masking.

21.1.1.3 SPI Features

The features of the Motorola SPI-compatible interface are:

- Full duplex, four-wire synchronous transfers.
- Programmable clock polarity and phase.

21.1.2 Pin Description

Table 21-1 SSP Pin Description

Pin Name	Function	I/O Type	Comments
SSPCLK[1:0]	SPI Serial Clock	I/O	–
SSPMOSI[1:0]	Master Out Slave In	I/O	–
SSPMISO[1:0]	Master In Slave Out	I/O	–
SSPFSS[1:0]	Frame, Slave Select (Master) Frame Input (Slave)	I/O	–

21.2 Functional Description

21.2.1 Block Diagram

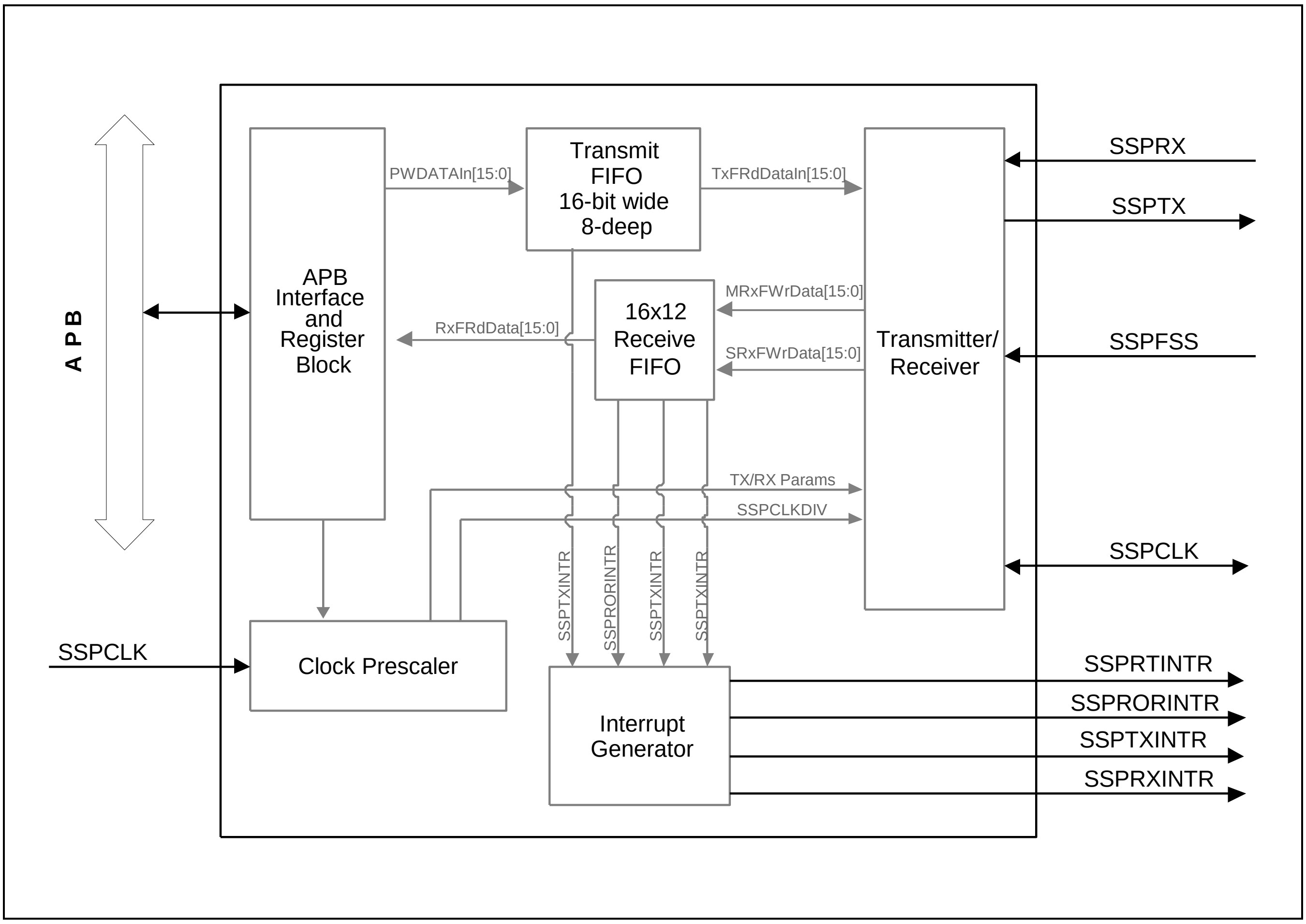


Figure 21-1 SSP Block Diagram

21.2.2 Operation

21.2.2.1 ARM PrimeCell SSP (PL022) Overview

The PrimeCell SSP is a master or slave interface for synchronous serial communication with peripheral devices that have Motorola SPI.

The PrimeCell SSP performs serial-to-parallel conversion on data received from a peripheral device. The CPU accesses data, control, and status information through the AMBA APB interface. The transmit and receive paths are buffered with internal FIFO memories allowing up to eight 16-bit values to be stored independently in both transmit and receive modes. Serial data is transmitted on SSPTXD and received on SSPRXD. The PrimeCell SSP includes a programmable bit rate clock divider and prescaler to generate the serial output clock SSPCLK from the input clock FSSPCLK. Bit rates are supported to 2MHz and higher, subject to choice of frequency for SSPCLK and the maximum bit rate is determined by peripheral devices.

The PrimeCell SSP operating mode, frame format, and size are programmed through the control registers SSP_CR0 and SSP_CR1.

Four individually mask able interrupt outputs are generated:

- SSPTXINTR requests servicing of the transmit buffer.
- SSPRXINTR requests servicing of the receive buffer.
- SSPRORINTR indicates an overrun condition in the receive FIFO.
- SSPRTINTR indicates that a timeout period expired while data was present in the receive FIFO.

Depending on the operating mode selected, the SSPFSS output operates as an active LOW slave select for SPI.

21.2.2.2 PrimeCell SSP Functional Description

AMBA APB Interface

The AMBA APB interface generates read and write decodes for accesses to status and control registers, and transmit and receive FIFO memories. The AMBA APB is a local secondary bus that provides a low-power extension to the higher bandwidth AMBA Advanced High-performance Bus (AHB) within the AMBA system hierarchy. The AMBA APB groups narrow-bus peripherals to avoid loading the system bus and provides an interface using memory-mapped registers, which are accessed under programmed control.

Register Block

The register block stores data written or to be read across the AMBA APB interface.

Clock Prescaler

When configured as a master, an internal prescaler, comprising two free-running reloadable serially linked counters, is used to provide the serial output clock SSPCLK.

You can program the clock prescaler, through the SSPCPSR register, to divide FSSPCLK by a factor of 2 to 254 in steps of two. By not utilizing the least significant bit of the SSPCPSR register, division by an odd number is not possible and this ensures a symmetrical (Equal mark space ratio) clock is generated.

The output of the prescaler is further divided by a factor of 1 to 256, through the programming of the SSPCR0 control register, to give the final master output clock SSPCLK pin.

Transmit FIFO

The common transmit FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. CPU data written across the AMBA APB interface are stored in the buffer until read out by the transmit logic.

When configured as a master or a slave parallel data is written into the transmit FIFO prior to serial conversion and transmission to the attached slave or master respectively, through the SSPTXD pin.

Receive FIFO

The common receive FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. Received data from the serial interface are stored in the buffer until read out by the CPU across the AMBA APB interface.

When configured as a master or slave, serial data received through the SSPRXD pin is registered prior to parallel loading into the attached slave or master receive FIFO respectively.

Transmit and Receive logic

When configured as a master, the clock to the attached slaves is derived from a divided down version of FSSPCLK through the prescaler operations described previously. The master transmit logic successively reads a value from its transmit FIFO and performs parallel to serial conversion on it. Then the serial data stream and frame control signal, synchronized to SSPCLK pin, are output through the SSPTXD pin to the attached slaves. The master receive logic performs serial to parallel conversion on the incoming synchronous SSPRXD data stream, extracting and storing values into its receive FIFO, for subsequent reading through the APB interface.

When configured as a slave, the SSPCLK pin clock is provided by an attached master and used to time its transmission and reception sequences. The slave transmit logic, under control of the master clock, successively reads a value from its transmit FIFO, performs parallel to serial conversion, then output the serial data stream and frame control signal through the slave SSPTXD pin. The slave receive logic performs serial to parallel conversion on the incoming SSPRXD data stream, extracting and storing values into its receive FIFO, for subsequent reading through the APB interface.

Interrupt generation logic

Four individual maskable, active HIGH interrupts are generated by the PrimeCell SSP

The individual interrupt requests could also be used with a system interrupt controller that provides masking for the outputs of each peripheral. In this way, a global interrupt controller service routine would be able to read the entire set of sources from one wide register in the system interrupt controller. This is attractive where the time to read from the peripheral registers is significant compared to the CPU clock speed in a real-time system.

The peripheral supports both the above methods.

The transmit and receive dynamic data-flow interrupts, SSPTXINTR and SSPRXINTR, are separated from the status interrupts so that data can be read or written in response to the FIFO trigger levels.

21.2.2.3 PrimeCell SSP Operation

Interface Reset

The PrimeCell SSP is reset by the global reset signal PRESETn and a block-specific reset signal nSSPRST. An external reset controller must use PRESETn to assert nSSPRST asynchronously and negate it synchronously to SSPCLK. PRESETn must be asserted LOW for a period long enough to reset the slowest block in the on-chip system, and then taken HIGH again. The PrimeCell SSP requires PRESETn to be asserted LOW for at least one period of PCLK.

Configuring the SSP

Following reset, the PrimeCell SSP logic is disabled and must be configured when in this state. Control registers SSP_CR0 and SSP_CR1 need to be programmed to configure the peripheral as a master or slave operating under Motorola SPI.

The bit rate, derived from the external SSPCLK, requires the programming of the clock prescale register SSPCPSR.

Enable PrimeCell SSP Operation

You can either prime the transmit FIFO, by writing up to eight 16-bit values when the PrimeCell SSP is disabled, or allow the transmit FIFO service request to interrupt the CPU. Once enabled, transmission or reception of data begins on the transmit (SSPTXD) and receive (SSPRXD) pins.

Clock Ratios

There is a constraint on the ratio of the frequencies of PCLK to F_{SSPCLK}. The frequency of F_{SSPCLK} must be less than or equal to that of PCLK. This ensures that control signals from the SSPCLK domain to the PCLK domain are certain to get synchronized before one frame duration:

- $F_{SSPCLK} \leftarrow F_{PCLK}$.

In the slave mode of operation, the SSPCLK pin signal from the external master is double synchronized and then delayed to detect an edge. It takes three F_{SSPCLK}S to detect an edge on SSPCLK pin. SSPTXD has less setup time to the falling edge of SSPCLK pin on which the master is sampling the line. The setup and hold times on SSPRXD with reference to SSPCLK pin must be more conservative to ensure that it is at the right value when the actual sampling occurs within the SSPMS. To ensure correct device operation, F_{SSPCLK} must be at least 12 times faster than the maximum expected frequency of SSPCLK pin.

The frequency selected for SSPCLK must accommodate the desired range of bit clock rates. The ratio of minimum F_{SSPCLK} frequency to SSPCLK pin maximum frequency in the case of the slave mode is 12 and for the master mode it is two.

To generate a maximum bit rate of 1.8432Mbps in the Master mode, the frequency of F_{SSPCLK} must be at least 3.6864MHz. With an SSPCLK frequency of 3.6864MHz, the SSPCPSR register has to be programmed with a value of two and the SCR[7:0] field in the SSPCR0 register needs to be programmed as zero.

To work with a maximum bit rate of 1.8432Mbps in the slave mode, the frequency of FSSPCLK must be at least 22.12MHz. With an F_{SSPCLK} frequency of 22.12MHz, the SSPCPSR register can be programmed with a value of 12 and the SCR[7:0] field in the SSPCR0 register can be programmed as zero. Similarly the ratio of SSPCLK maximum frequency to SSPCLK pin minimum frequency is 254 × 256. The minimum frequency of SSPCLK is governed by the following equations, both of which have to be satisfied:

- F_{SSPCLK} (min) $\rightarrow 2 \times F_{SSPCLKOUT}$ (max) [for master mode]
- F_{SSPCLK} (min) $\rightarrow 12 \times F_{SSPCLKIN}$ (max) [for slave mode]

The maximum frequency of SSPCLK is governed by the following equations, both of which have to be satisfied:

- F_{SSPCLK} (max) $\leftarrow 254 \times 256 \times F_{SSPCLKOUT}$ (min) [for master mode]
- F_{SSPCLK} (max) $\leftarrow 254 \times 256 \times F_{SSPCLKIN}$ (min) [for slave mode]

Programming the SSPCR0 Control Register

The SSPCR0 register is used to:

- Program the serial clock rate
- Select one of the three protocols
- Select the data word size (where applicable).

The Serial Clock Rate (SCR) value, in conjunction with the SSPCPSR clock prescale divisor value (CPSDVSR), is used to derive the PrimeCell SSP transmit and receive bit rate from the external SSPCLK. The frame format is programmed through the FRF bits and the data word size through the DSS bits. Bit phase and polarity, applicable to Motorola SPI format only, are programmed through the SPH and SPO bits.

Programming the SSPCR1 Control Register

The SSPCR1 register is used to:

- Select master or slave mode
- Enable the PrimeCell SSP peripheral.

To configure the PrimeCell SSP as a master, clear the SSPCR1 register master or slave selection bit (MS) to 0, which is the default value on reset. Setting the SSPCR1 register MS bit to 1 configures the PrimeCell SSP as a slave. When configured as a slave, enabling or disabling of the PrimeCell SSP SSPTXD signal is provided through the SSPCR1 slave mode SSPTXD output disable bit (SOD). This can be used in some multi-slave environments where masters might parallel broadcast.

To enable the operation of the PrimeCell SSP set the Synchronous Serial Port Enable (SSE) bit to 1.

Bit rate generation

The serial bit rate is derived by dividing down the input clock FSSPCLK. The clock is first divided by an even prescale value CPSDVSR from 2 to 254, which is programmed in SSPCPSR. The clock is further divided by a value from 1 to 256, which is 1 + SCR, where SCR is the value programmed in SSPCR0.

The frequency of the output signal bit clock SSPCLK pin is defined below:

- $F_{SSPCLK} = F_{FSSPCLK} / (CPSDVR \times (1 + SCR))$

For example, if SSPCLK is 3.6864MHz, and CPSDVSR = 2, then SSPCLK has a frequency range from 7.2kHz to 1.8432MHz.

Frame Format

Each data frame is between 4 and 16 bits long depending on the size of data programmed, and is transmitted starting with the MSB.

For all three formats, the serial clock (SSPCLK pin) is held inactive while the PrimeCell SSP is idle, and transitions at the programmed frequency only during active transmission or reception of data. The idle state of SSPCLK pin is utilized to provide a receive timeout indication that occurs when the receive FIFO still contains data after a timeout period.

For Motorola SPI, the serial frame (SSPFSS) pin is active LOW, and is asserted (pulled down) during the entire transmission of the frame.

Motorola SPI Frame Format

The Motorola SPI interface is a four-wire interface where the SSPFSS signal behaves as a slave select. The main feature of the Motorola SPI format is that the inactive state and phase of the SSPCLK pin signal are programmable through the SPO and SPH bits within the SSPSCR0 control register.

- SPO, clock polarity

When the SPO clock polarity control bit is LOW, it produces a steady state low value on the SSPCLK pin. If the SPO clock polarity control bit is HIGH, a steady state high value is placed on the SSPCLK pin when data is not being transferred.

- SPH, clock phase

The SPH control bit selects the clock edge that captures data and allows it to change state. It has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge.

When the SPH phase control bit is LOW, data is captured on the first clock edge transition. If the SPH clock phase control bit is HIGH, data is captured on the second clock edge transition.

Motorola SPI Format with SPO = 0, SPH = 0

Single and continuous transmission signal sequences for Motorola SPI format with SPO = 0, SPH = 0 are shown in below figure.

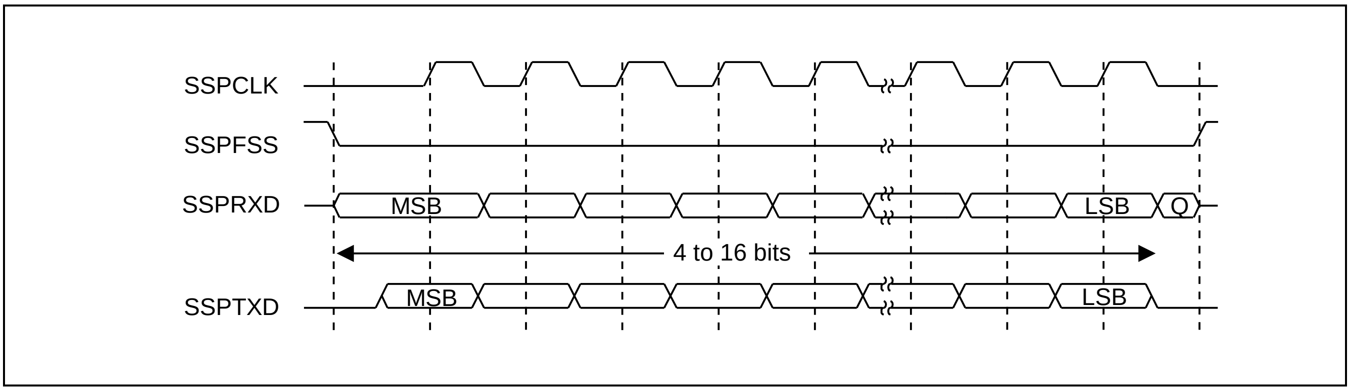


Figure 21-2 Motorola SPI Frame Format (Single Transfer) with SPO = 0 and SPH = 0

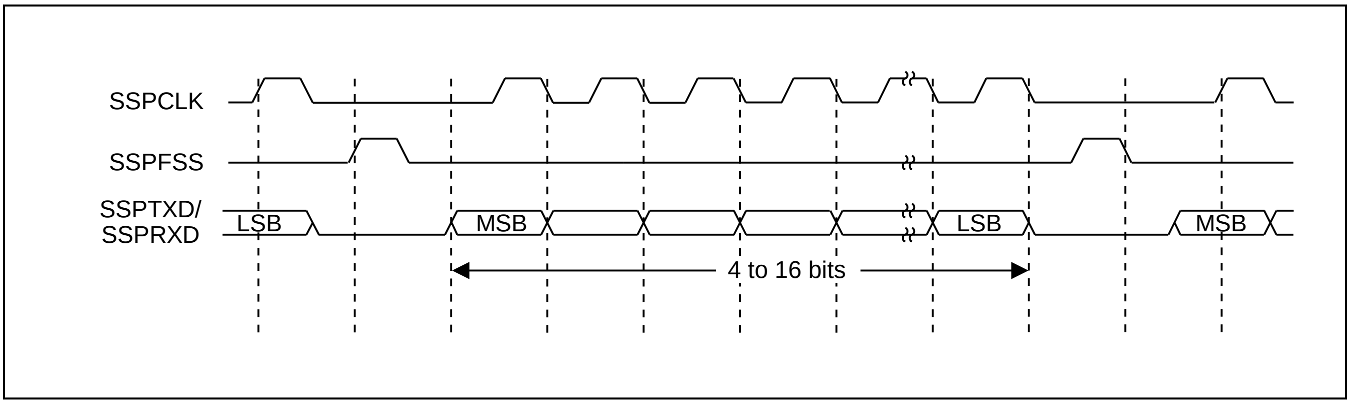


Figure 21-3 Motorola SPI Frame Format (Continuous Transfer) with SPO = 0 and SPH = 0

In this configuration, during idle periods:

- The SSPCLK pin signal is forced LOW
- SSPFSS is forced HIGH
- The transmit data line SSPTXD is arbitrarily forced LOW
- When the PrimeCell SSP is configured as a master, the SSPCLK pin is enabled
- When the PrimeCell SSP is configured as a slave, SSPCLK pin is disabled

If the PrimeCell SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSPFSS master signal being driven LOW. This causes slave data to be enabled onto the SSPRXD input line of the master.

One half SSPCLK pin period later, valid master data is transferred to the SSPTXD pin. Now that both the master and slave data have been set, the SSPCLK master clock pin goes HIGH after one further half SSPCLK pin period.

The data is now captured on the rising and propagated on the falling edges of the SSPCLK pin signal.

In the case of a single word transmission, after all bits of the data word have been transferred, the SSPFSS line is returned to its idle HIGH state one SSPCLK pin period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSPFSS signal must be pulsed HIGH between each data word transfer. This is because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is logic zero. Therefore the master device must raise the SSPFSS pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSPFSS pin is returned to its idle state one SSPCLK period after the last bit has been captured.

Motorola SPI Format with SPO = 0, SPH = 1

The transfer signal sequence for Motorola SPI format with SPO = 0, SPH = 1 is shown in below figure, which covers both single and continuous transfers.

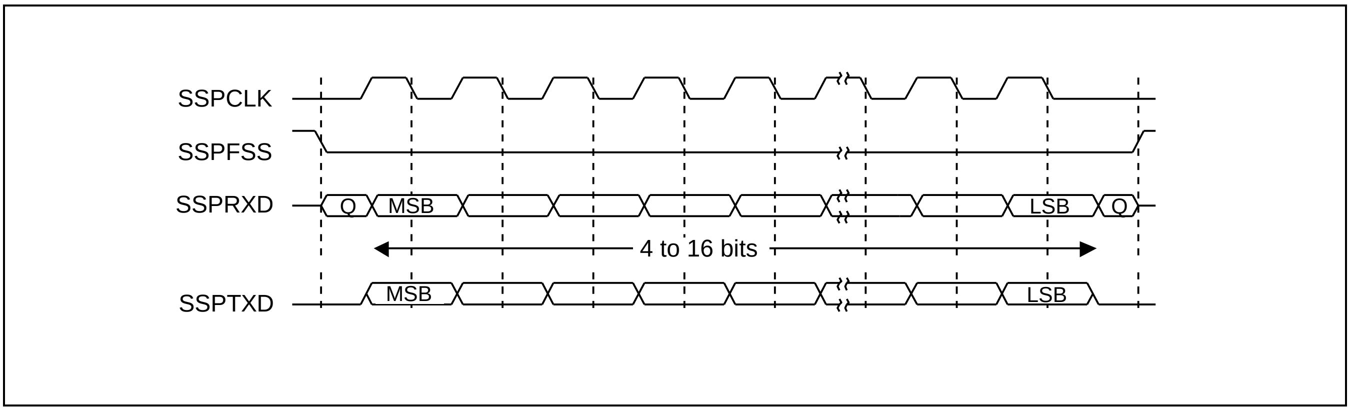


Figure 21-4 Motorola SPI Frame Format with SPO = 0 and SPH = 1

In this configuration, during idle periods:

- The SSPCLK signal is forced LOW
- SSPFSS is forced HIGH
- The transmit data line SSPTXD is arbitrarily forced LOW
- When the PrimeCell SSP is configured as a master, the SSPCLK is enabled
- When the PrimeCell SSP is configured as a slave the SSPCLK is disabled

If the PrimeCell SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSPFSS master signal being driven LOW. The master SSPTXD output pad is enabled. After a further one half SSPCLK period, both master and slave valid data is enabled onto their respective transmission lines. At the same time, the SSPCLK is enabled with a rising edge transition.

Data is then captured on the falling edges and propagated on the rising edges of the SSPCLK signal.

In the case of a single word transfer, after all bits have been transferred, the SSPFSS line is returned to its idle HIGH state one SSPCLK period after the last bit has been captured.

For continuous back-to-back transfers, the SSPFSS pin is held LOW between successive data words and termination is the same as that of the single word transfer.

Motorola SPI Format with SPO = 1, SPH = 0

Single and continuous transmission signal sequences for Motorola SPI format with SPO = 1, SPH = 0 are shown in below figure.

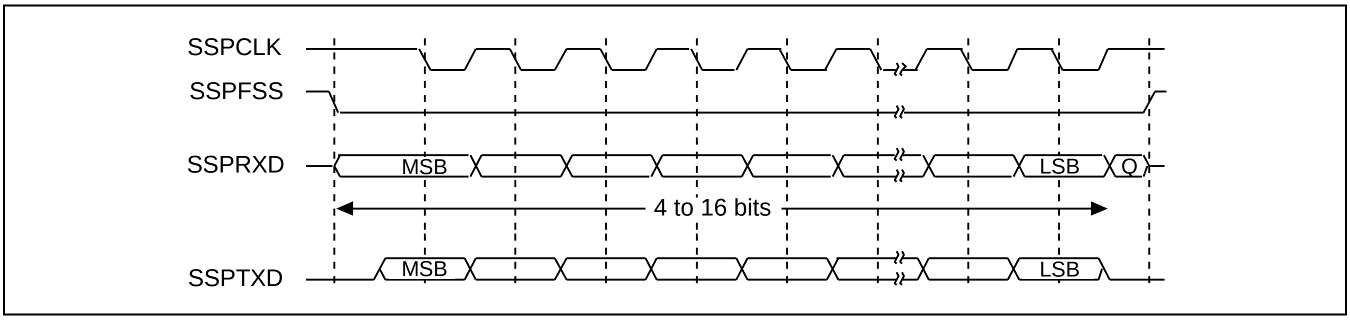


Figure 21-5 Motorola SPI Frame Format (Single Transfer) with SPO = 1 and SPH = 0

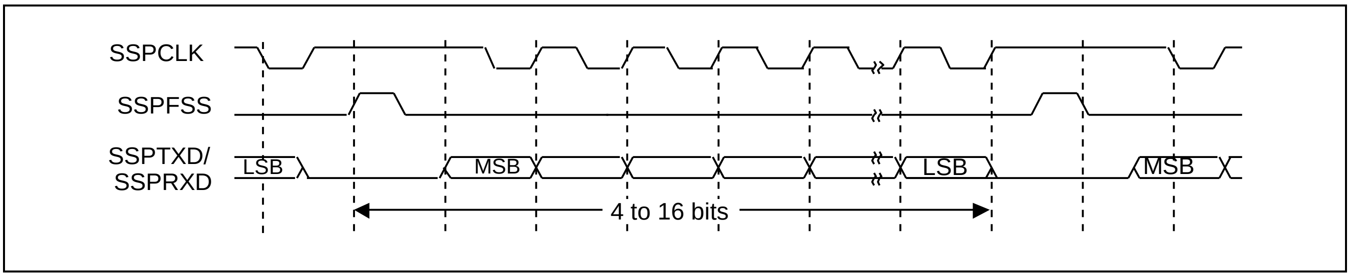


Figure 21-6 Motorola SPI Frame Format (Continuous Transfer) with SPO = 1 and SPH = 0

In this configuration, during idle periods

- The SSPCLK signal is forced HIGH
- SSPFSS is forced HIGH
- The transmit data line SSPTXD is arbitrarily forced LOW
- When the PrimeCell SSP is configured as a master, the SSPCLK is enabled
- When the PrimeCell SSP is configured as a slave, the SSPCLK is disabled

If the PrimeCell SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSPFSS master signal being driven LOW, which causes slave data to be immediately transferred onto the SSPRXD line of the master. The master SSPTXD output pad is enabled.

One half period later, valid master data is transferred to the SSPTXD line. Now that both the master and slave data have been set, the SSPCLK master clock pin becomes LOW after one further half SSPCLK period. This means that data is captured on the falling edges and be propagated on the rising edges of the SSPCLK signal.

In the case of a single word transmission, after all bits of the data word are transferred, the SSPFSS line is returned to its idle HIGH state one SSPCLK period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSPFSS signal must be pulsed HIGH between each data word transfer. This is because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is logic zero. Therefore the master device must raise the SSPFSS pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSPFSS pin is returned to its idle state one SSPCLK period after the last bit has been captured.

Motorola SPI Format with SPO = 1, SPH = 1

The transfer signal sequence for Motorola SPI format with SPO = 1, SPH = 1 is shown in below figure, which covers both single and continuous transfers.

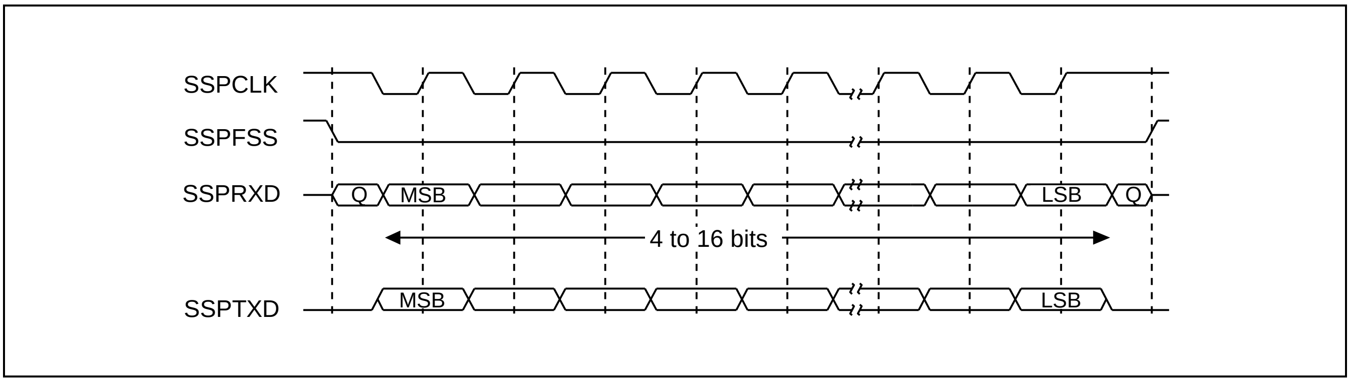


Figure 21-7 Motorola SPI Frame Format with SPO = 1 and SPH = 1

In this configuration, during idle periods:

- The SSPCLK signal is forced HIGH
- SSPFSS is forced HIGH
- The transmit data line SSPTXD is arbitrarily forced LOW
- When the PrimeCell SSP is configured as a master, the SSPCLK is enabled
- When the PrimeCell SSP is configured as a slave, the SSPCLK is disabled

If the PrimeCell SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSPFSS master signal being driven LOW. The master SSPTXD output pad is enabled.

After a further one half SSPCLK period, both master and slave data are enabled onto their respective transmission lines. At the same time, the SSPCLK is enabled with a falling edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SSPCLK signal.

After all bits have been transferred, in the case of a single word transmission, the SSPFSS line is returned to its idle HIGH state one SSPCLK period after the last bit has been captured.

For continuous back-to-back transmissions, the SSPFSS pins remains in its active LOW state, until the final bit of the last word has been captured, and then returns to its idle state as described above.

For continuous back-to-back transfers, the SSPFSS pin is held LOW between successive data words and termination is the same as that of the single word transfer.

Examples of Master and Slave Configurations

Below figures show how the PrimeCell SSP (PL022) peripheral can be connected to other synchronous serial peripherals, when it is configured as a master or slave.

NOTE: The SSP (PL022) does not support dynamic switching between master and slave in a system. Each instance is configured and connected either as a master or slave.

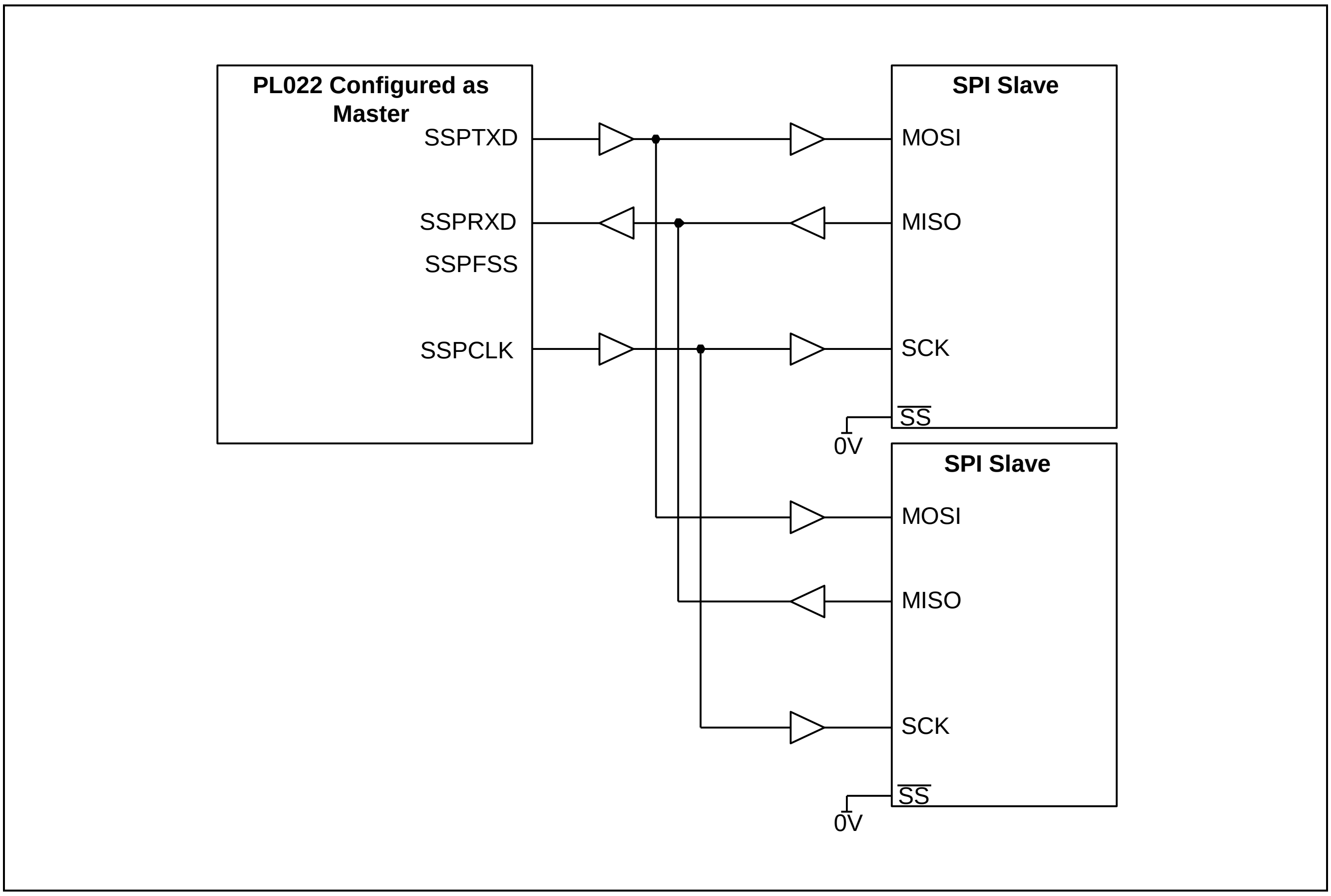


Figure 21-8 PrimeCell SSP Master Coupled to Two Slaves

Above figure shows how an PrimeCell SSP (PL022), configured as master, interfaces to two Motorola SPI slaves. Each SPI Slave Select (SS) signal is permanently tied LOW and configures them as slaves. Similar to the above operation, the master can broadcast to the two slaves through the master PrimeCell SSP SSPTXD line. In response, only one slave drives its SPI MISO port onto the SSPRXD line of the master

.

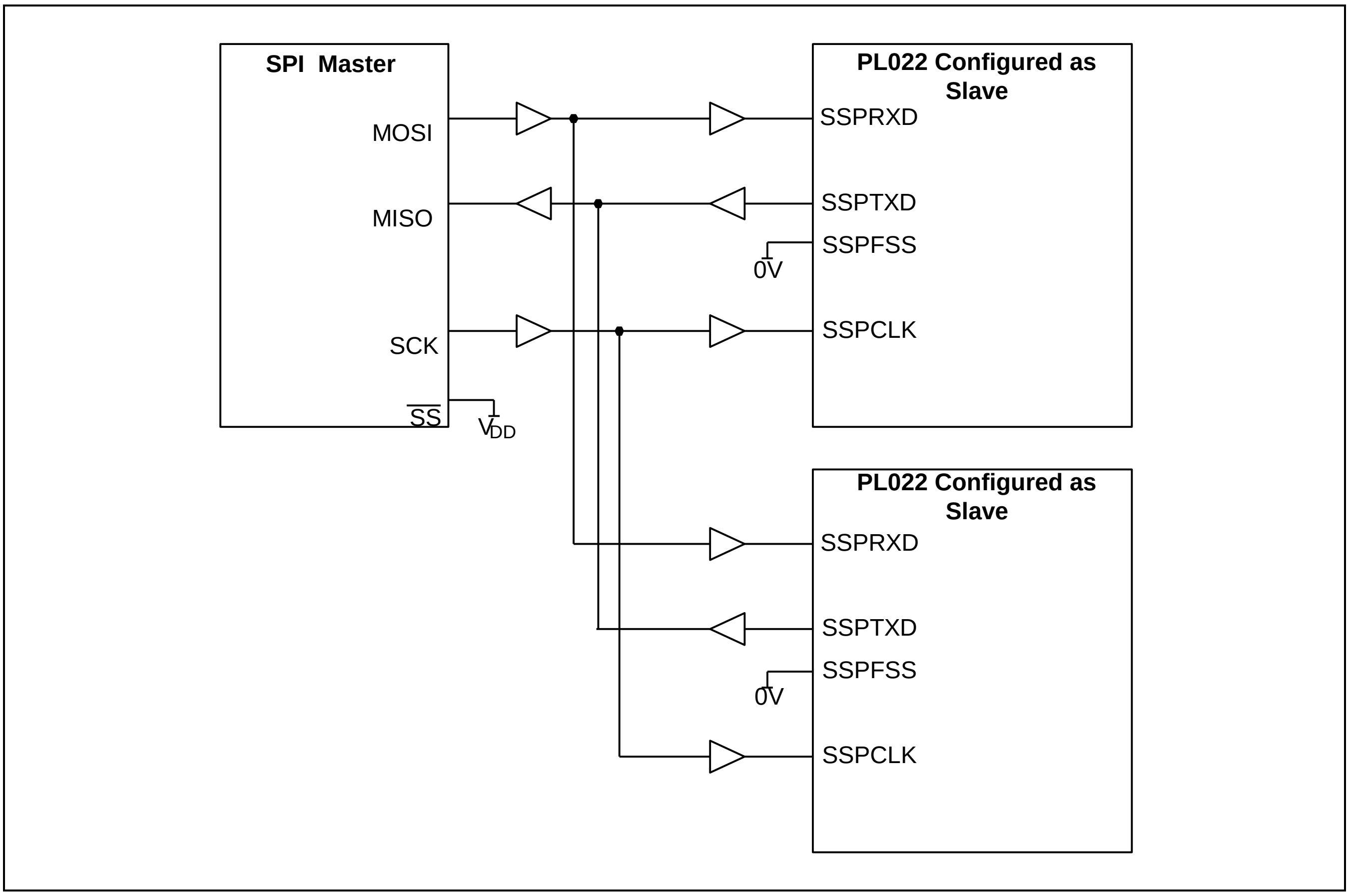


Figure 21-9 SPI Master Coupled to two PrimeCell SSP Slaves

Above figure shows a Motorola SPI configured as a master and interfaced to two instances of PrimeCell SSP (PL022) configured as slaves. In this case the slave Select Signal (SS) is permanently tied HIGH and configures it as a master. The master can broadcast to the two slaves through the master SPI MOSI line and in response, only one slave drives its nSSPOE signal LOW. This enables its SSPTXD data onto the MISO line of the master.

21.2.2.4 Interrupt

There are five interrupts generated by the PrimeCell SSP. Four of these are individual, maskable, active HIGH interrupts:

- SSPRXINTR: PrimeCell SSP receive FIFO service interrupt request.
- SSPTXINTR: PrimeCell SSP transmit FIFO service interrupt request.
- SSPRORINTR: PrimeCell SSP receive overrun interrupt request.
- SSPRTINTR: PrimeCell SSP time out interrupt request.

You can mask each of the four individual maskable interrupts by setting the appropriate bits in the SSPIMSC register. Setting the appropriate mask bit HIGH enables the interrupt.

Provision of the individual outputs as well as a combined interrupt output, allows use of either a global interrupt service routine, or modular device drivers to handle interrupts.

The transmit and receive dynamic dataflow interrupts SSPTXINTR and SSPRXINTR have been separated from the status interrupts, so that data can be read or written in response to just the FIFO trigger levels.

The status of the individual interrupt sources can be read from SSPRIS and SSPMIS registers.

- SSPRXINTR
  - The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.

- SSPTXINTR
  - The transmit interrupt is asserted when there are four or less valid entries tin the transmit FIFO. The transmit interrupt SSPTXINTR is not qualified with the PrimeCell SSP enable signal, which allows operation in one of two ways. Data can be written to the transmit FIFO prior to enabling the PrimeCell SSP and interrupts.
    Alternatively, the PrimeCell SSP and interrupts can be enabled so that data can be written to the transmit FIFO by an interrupt service routine.

- SSPRORINTR
  - The receive overrun interrupt SSPORINTR is asserted when the FIFO is already full and an additional data frame is received, causing an overrun of the FIFO. Data is over-written in the receive shift register, but not the FIFO.

- SSPRTINTR
  - The receive timeout interrupt is asserted when the receive FIFO is not empty and the PrimeCell SSP has remained idle for a fixed 32 bit period. This mechanism ensures that the user is aware that data is still present in the receive FIFO and requires servicing. This interrupt is deasserted if the receive FIFO becomes empty by subsequent reads, or if new data is received on SSPRXD. It can also be cleared by writing to the RTIC bit in the SSPICR register.

21.3 Register Description

21.3.1 Register Map Summary

- Base Address: 0x4009_0000
- Base Address: 0x4009_1000

Register	Offset	Description	Reset Value
SSP_CR0	0x0000	SSP Control Register 0	0x0000_0000
SSP_CR1	0x0004	SSP Control Register 1	0x0000_0010
SSP_DR	0x0008	SSP Receive FIFO Data Register (READ) SSP Transmit FIFO Data Register (WRITE)	0x0000_0000
SSP_SR	0x000C	SSP Status Register	0x0000_0003
SSP_CPSR	0x0010	SSP Clock Rescale Register	0x0000_0000
SSP_IMSCR	0x0014	SSP Interrupt Mask Set/Clear Register	0x0000_0000
SSP_RISR	0x0018	SSP Raw Interrupt Status Register	0x0000_0008
SSP_MISR	0x001C	SSP Masked Interrupt Status Register	0x0000_0000
SSP_ICR	0x0020	SSP Interrupt Clear Register	0x0000_0000
SSP_DMACR	0x0024	SSP DMA Control Register	0x0000_0000

NOTE: ID register’s read-only values tell the prime-cell ID information. (SSPPeriphID0/1/2/3, SSPPCellID0/1/2/3)

21.3.1.1 SSP_CR0 (SSP Control Register 0)

- Address = Base Address + 0x0000, Reset Value = 0x0000_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
RSVD																SCR								SPH	SPO	FRF		DSS			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
DSS	[3:0]	RW	Data Size Selection Field. 0000–0010 = Reserved 0011 = 4-bit data 0100 = 5-bit data 0101 = 6-bit data 0110 = 7-bit data 0111 = 8-bit data 1000 = 9-bit data 1001 = 10-bit data 1010 = 11-bit data 1011 = 12-bit data 1100 = 13-bit data 1101 = 14-bit data 1110 = 15-bit data 1111 = 16-bit data	0000'b
FRF	[5:4]	RW	Frame Format Selection Field. Must be set to 00 for Motorola SPI frame format.	00'b
SPO	[6]	RW	SSPCLK Polarity Bit. 0 = Data is captured on the first clock edge transition. 1 = Data is captured on the second clock edge transition.	0
SPH	[7]	RW	SSPCLKOUT Phase. 0 = Data is captured on the first clock edge transition. 1 = Data is captured on the second clock edge transition.	0
SCR	[15:8]	RW	Serial Clock Rate Field. The value SCR is used to generate the transmit and receive bit rate. The Bit Rate = FPCLK/ (CPSDVR × (1 + SCR)) Where CPSDVSR is an even value from 2 to 254, programmed through the SSPCPSR register and SCR is a value from 0 to 255.	0x00

21.3.1.2 SSP_CR1 (SSP Control Register 1)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																									RXIFLSEL			SOD	MS	SSE	LBM
0	0	0	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	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
LBM	[0]	RW	Loop-Back Mode Bit. 0 = Normal serial port operation enabled. 1 = Output of transmit serial shifter is connected to input of receive serial shifter internally.	0
SSE	[1]	RW	Synchronous Serial Port Enable Bit. 0 = SSP operation disabled. 1 = SSP operation enabled.	0
MS	[2]	RW	Master or Slave Mode Selection Bit. 0 = Device configured as master. 1 = Device configured as slave.	0
SOD	[3]	RW	Slave-mode Output Disable Bit. This bit is relevant only in the slave mode (MS = 1). In multiple-slave systems, it is possible for a PrimeCell SSP master to broadcast a message to all slaves in the system while ensuring that only one slave drives data onto its serial output line. In such systems the RXD lines from multiple slaves could be tied together. To operate in such systems, the SOD bit can be set if the PrimeCell SSP slave is not supposed to drive the SSPTXD line. 0 = SSP can drive the SSPTXD output in slave mode. 1 = SSP must not drive the SSPTXD output in slave mode.	0
RXIFLSEL	[6:4]	RW	Receive Interrupt FIFO Level Selection Field. 001 = Trigger points: Receive FIFO becomes > = 1/8 010 = Trigger points: Receive FIFO becomes > = 1/4 100 = Trigger points: Receive FIFO becomes > = 1/2 Others = Reserved	0001'b

21.3.1.3 SSP_DR (SSP Data Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																DATA															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA	[15:0]	RW	Transmit/ Receive FIFO. Read: Receive FIFO. Write: Transmit FIFO. You must right-justify data when the PrimeCell SSP is programmed for a data size that is less than 16 bits. Unused bits at the top are ignored by transmit logic. The receive logic automatically right-justifies.	0x0000

When SSPDR is read, the entry in the receive FIFO (pointed to by the current FIFO read pointer) is accessed. As data values are removed by the PrimeCell SSP receive logic from the incoming data frame, they are placed into the entry in the receive FIFO (pointed to by the current FIFO write pointer).

When SSPDR is written to, the entry in the transmit FIFO (pointed to by the write pointer), is written to. Data values are removed from the transmit FIFO one value at a time by the transmit logic. It is loaded into the transmit serial shifter, then serially shifted out onto the SSPTXD pin at the programmed bit rate.

When a data size of less than 16 bits is selected, the user must right-justify data written to the transmit FIFO. The transmit logic ignores the unused bits. Received data less than 16 bits is automatically right-justified in their receive buffer.

21.3.1.4 SSP_SR (SSP Status Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											BSY	RFF	RNE	TNF	TFE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
TFE	[0]	R	Transmit FIFO Empty Status Bit. 0 = Transmit FIFO is not empty. 1 = Transmit FIFO is empty.	1
TNF	[1]	R	Transmit FIFO Full Status Bit. 0 = Transmit FIFO is full. 1 = Transmit FIFO is not full.	1
RNE	[2]	R	Receive Empty Status Bit. 0 = Receive FIFO is empty. 1 = Receive FIFO is not empty.	0
RFF	[3]	R	Receive FIFO Full Status Bit. 0 = Receive FIFO is not full. 1 = Receive is full.	0
BSY	[4]	R	Prime-Cell SSP Busy Flag Bit. 0 = SSP is idle. 1 = SSP is currently transmitting and/or receiving a frame or the transmit FIFO is not empty.	0

21.3.1.5 SSP_CPSR (SSP Clock Pre-scale Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																								CPSDVSR							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
CPSDVSR	[7:0]	RW	Clock Pre-scale Divisor Field. Must be an even number from 2 to 254, depending on the frequency of FSSPCLK. The least significant bit always returns zero on reads.	0x00

SSPCPSR is the clock pre-scale register and specifies the division factor by which the input F_{PCLK} must be internally divided before further use.

The value programmed into this register must be an even number between 2 to 254. The least significant bit of the programmed number is hard-coded to zero. If an odd number is written to this register, data read back from this register has the least significant bit as zero.

21.3.1.6 SSP_IMSCR (SSP Interrupt Mask Set/ Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																												TXIM	RTIM	RTIM	RORIM
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
RORIM	[0]	RW	Receive Overrun Interrupt Mask. 0 = RxFIFO written to while full condition interrupt is masked. (Disable the interrupt) 1 = RxFIFO written to while full condition interrupt is not masked. (Enable the interrupt)	0
RTIM	[1]	RW	Receive Timeout Interrupt Mask. 0 = RxFIFO not empty and no read prior to timeout period interrupt is masked. (Disable the interrupt) 1 = RxFIFO not empty and no read prior to timeout period interrupt is not masked. (Enable the interrupt)	0
RXIM	[2]	RW	Receive FIFO Interrupt Mask. 0 = Selected Rx FIFO trigger level (1/2, 1/4, or 1/8) condition interrupt is masked. (Disable the interrupt) 1 = Selected Rx FIFO trigger level (1/2, 1/4, or 1/8) condition interrupt is not masked.(Enable the interrupt)	0
TXIM	[3]	RW	Transmit FIFO Interrupt Mask. 0 = Tx FIFO half full or less condition interrupt is masked. (Disable the interrupt) 1 = Tx FIFO half full or less condition interrupt is not masked. (Enable the interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

21.3.1.7 SSP_RISR (SSP Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																												TXRIS	RXRIS	RTRIS	RORRIS
0	0	0	0	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	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
RORRIS	[0]	R	Receive Overrun Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SSPRORINTR interrupt.	0
RTRIS	[1]	R	Receive Timeout Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SSPRTINTR interrupt.	0
RXRIS	[2]	R	Receive FIFO Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SSPRXINTR interrupt.	0
TXRIS	[3]	R	Transmit FIFO Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the SSPTXINTR interrupt.	1

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

21.3.1.8 SSP_MISR (Interrupt Priority Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																												TXMIS	RXMIS	RTMIS	RORMIS
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ROMIS	[0]	R	Receive Overrun Masked Interrupt State. Gives the receive over run masked interrupt status (After masking) of the SSPRORINTR interrupt.	0
RTMIS	[1]	R	Receive Timeout Masked Interrupt State. Gives the receive timeout masked interrupt state (After masking) of the SSPRTINTR interrupt.	0
RXMIS	[2]	R	Receive FIFO Masked Interrupt State. Gives the receive FIFO masked interrupt state (After masking) of the SSPRXINTR interrupt.	0
TXMIS	[3]	R	Transmit FIFO Masked Interrupt State. Gives the transmit FIFO masked interrupt state (After masking) of the SSPTXINTR interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

21.3.1.9 SSP_ICR (SSP Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																														RTIC	RORIC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W

Name	Bit	Type	Description	Reset Value
RORIC	[0]	W	Receive Overrun Interrupt Clear. 0 = No effect. 1 = Clears the SSPRORINTR interrupt.	0
RTIC	[1]	W	Receive Timeout Interrupt Clear. 0 = No effect. 1 = Clears the SSPRTINTR interrupt.	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

21.3.1.10 SSP_DMACR (SSP DMA Control Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																														TXDMAE	RXDMAE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
RXDMAE	[0]	RW	DMA for the receive FIFO Enable/ Disable Control Bit. 0 = Disable. 1 = Enable.	0
TXDMAE	[1]	RW	DMA for the transmit FIFO Enable/ Disable Control Bit. 0 = Disable. 1 = Enable.	0

22 Stamp Timer (STT)

22.1 Overview

The Stamp Timer provides a counter and an alarm function. It is also used to stamp the CAN messages.

22.1.1 Features

The stamp timer has a programmable 32-bit counter.

- The STT_CNT indicates the number of STT clock pulses elapsed since the last time it was reset to zero.
- An interrupt is generated at the end of the period at which the value in the counter register equals the value in the STT_ALR.
- After each transmission or reception of a CAN frame, the STT_CNT value will be automatically written in the corresponding CAN channel CAN_STPx (CAN Interface Stamp Register x).

22.2 Functional Description

22.2.1 Block Diagram

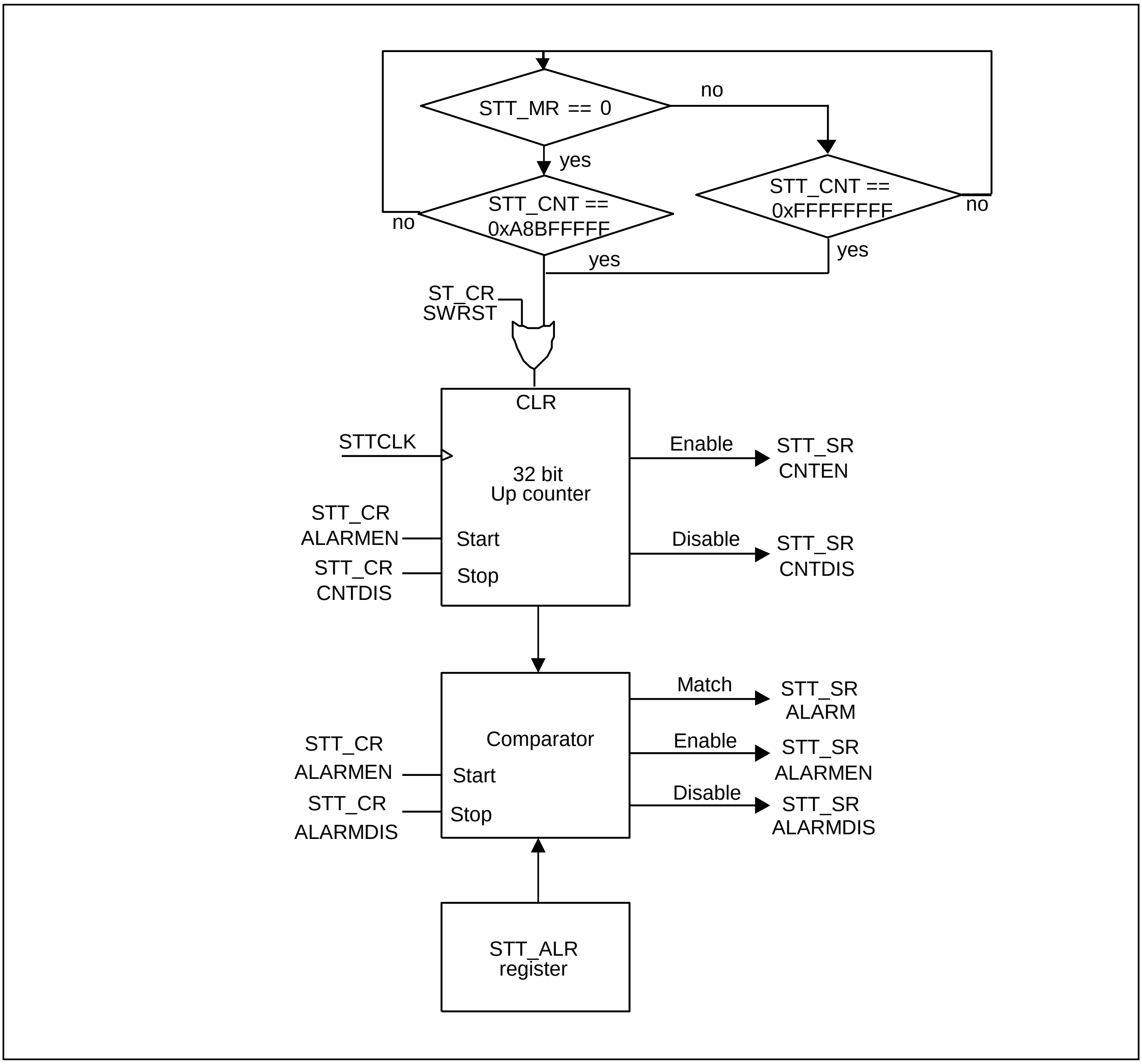


Figure 22-1 Stamp Timer (STT) Block Diagram

22.2.1.1 Counter Register

The counter is a 32-bit counter that indicates the number of STT clock pulses elapsed since the last time it was reset to zero.

The counter is incremented every period of STT clock.

The counter is reset to 0x0000_0000 when the counter reaches 0xA8C0_0000 or 0XFFFF_FFFF (it is configurable on the Mode Register).

A write access can only be performed when the counter is disabled because of an asynchronous interface (see "Asynchronous interface" below).

22.2.1.2 Alarm Register

The alarm register has the same resolution as the counter.

An interrupt is generated at the end of the period at which the value in the counter register equals the value in the alarm register.

A write access can only be performed when the alarm counter is disabled because of an asynchronous interface. An invalid data (i.e. value greater or equal to 0xA8C0_0000) will not be written into the alarm register if CNTRST = 0.

22.2.1.3 Asynchronous Interface

As the counter is an asynchronous counter, some precautions must be taken with it.

When enabling or disabling alarm or counter, software must wait for enabled or disabled interrupt to be sure that the alarm or the counter is really enabled or disabled.

22.2.1.4 CAN Time Stamp

The 32 bits register forming the counter is provided to the CAN module. After each transmission or reception of a CAN frame, the value of the current counter will be automatically written in the corresponding CAN channel CAN_STPx register.

22.2.2 Programming Examples

Example of use of the timer:

Use of the timer to generate an interrupt after 1 second according that the low frequency clock is equal to 1MHz. (1 second = 0xF4240/1MHz)

Configuration:

- Do a software reset of the timer to be in a known state by writing bit SWRST in STT_SRR and wait about four LFCLK period for the circuitry to be stabilized
- Configuration of STT_ ALARM: When the counter will be equal to this value, an interrupt can be generated. For 1 second at 1MHz, Alarm value should be 0xF423F.
- Configuration of STT_CNT: Starting value from which counter will start to increase. Should be left to 0.
- Configuration of STT_IMSCR: bit ALARM enables interrupt at the peripheral level when counter is equal to the programmed value in STT_ALARM. Other interrupts can be activated to indicate when the counter or alarm functionality are really enabled or disabled as stamp timer is clocked on the LFCLK.
- Configuration of STT_CR: Starts the counter and enables the alarm (bit ALARMEN and CNTEN). Counter will start to increment when bit CNTEN will be set in STT_SR, same for alarm functionality bit ALARMEN, an interrupt can be programmed with those this events.

Interruption Handling:

- IRQ Entry and call C function.
- Read STT_SR and verify the source of the interrupt.
- Clear the corresponding interrupt at peripheral level by writing in the STT_CSR.
- Interrupt treatment, counter will restart automatically counting from 0 when it reaches 0xF423F (STT_MR). If users want to set a lower value in STT_ALARM, and if they want to restart counter, they must disable counter and set it to 0 and enable it again.
- IRQ Exit.

Caution: If internal oscillator is used for clock source of STT, STT cannot be used for application which needs accurate timing measurement because internal oscillator (1MHz) has not good accuracy.

22.3 Register Description

22.3.1 Register Map Summary

- Base Address: 0x4007_8000

Register	Offset	Description	Reset Value
STT_IDR	0x0000	ID Register	0x0001_1234
STT_CEDR	0x0004	Clock Enable/ Disable Register	0x0000_0000
STT_SRR	0x0008	Software Reset Register	0x0000_0000
STT_CR	0x000C	Control Register	0x0000_0000
STT_MR	0x0010	Mode Register	0x0000_0000
STT_IMSCR	0x0014	Interrupt Mask Set/ Clear Register	0x0000_0000
STT_RISR	0x0018	Raw Interrupt Status Register	0x0000_0000
STT_MISR	0x001C	Masked Interrupt Status Register	0x0000_0000
STT_ICR	0x0020	Interrupt Clear Register	0x0000_0000
STT_SR	0x0024	Status Register	0x0000_0000
STT_CNTR	0x0028	Counter Register	0x0000_0000
STT_ALR	0x002C	Alarm Register	0x0000_0000

22.3.1.1 STT_IDR (STT ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_1234

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_1234

22.3.1.2 STT_CEDR (STT Clock Enable/ Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable Control Bit. 0 = STT Clock is disabled 1 = STT Clock is enabled	0
DBGEN	[31]	RW	Debug Mode Enable/Disable Control Bit. 0 = Disable Debug Mode, STT is not halted during processor debug mode. 1 = Enable Debug Mode, STT is halted during processor debug mode.	0

22.3.1.3 STT_SRR (STT Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No Effect 1 = STT Software Reset	0

22.3.1.4 STT_CR (STT Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																											ALARMDIS	ALARMEN	CNTDIS	CNTEN	RSVD
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	R

Name	Bit	Type	Description	Reset Value
CNTEN	[1]	W	STT Counter Enable. 0 = No effect 1 = Enables the STT seconds counter	0
CNTDIS	[2]	W	STT Counter Disable. 0 = No effect 1 = Disables the STT seconds counter. In case both CNTEN and CNTDIS are equal to one when the control register is written, the STT counter will be disabled.	0
ALARMEN	[3]	W	STT Alarm Enable. 0 = No effect 1 = Enables the STT alarm	0
ALARMDIS	[4]	W	STT Alarm Disable. 0 = No effect 1 = Disables the STT alarm In case both ALARMEN and ALARMDIS are equal to one when the control register is written the STT alarm will be disabled.	0

22.3.1.5 STT_MR (STT Mode Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD																															CNTRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CNTRST	[0]	RW	Counter Reset. 0 = The counter is reset to 0x000_00000 at the end of the period when it reaches 0xA8BF_FFFF. 1 = The counter is reset to 0x0000_0000 at the end of the period when it reaches 0xFFFF_FFFF.	0

Caution: CNTRST bit can only be written Stamp Timer is not enabled.

22.3.1.6 STT_IMSCR (STT Interrupt Mask Set/ Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																											ALARMDIS	ALARMEN	CNTDIS	CNTEN	ALARM
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
ALARM	[0]	RW	Alarm Interrupt Mask Set/ Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
CNTEN	[1]	RW	Counter Interrupt Mask Set/ Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
CNTDIS	[2]	RW	Counter Interrupt Mask Set/ Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
ALARMEN	[3]	RW	Alarm Interrupt Mask Set/ Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
ALARMDIS	[4]	RW	Alarm Interrupt Mask Set/ Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

22.3.1.7 STT_ RISR (STT Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD																											ALARMDIS	ALARMEN	CNTDIS	CNTEN	ALARM
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ALARM	[0]	R	Gives the raw interrupt state (Prior to masking) of the ALARM interrupt.	0
CNTEN	[1]	R	Gives the raw interrupt state (Prior to masking) of the CNTEN interrupt.	0
CNTDIS	[2]	R	Gives the raw interrupt state (Prior to masking) of the CNTDIS interrupt.	0
ALARMEN	[3]	R	Gives the raw interrupt state (Prior to masking) of the ALARMEN interrupt.	0
ALARMDIS	[4]	R	Gives the raw interrupt state (Prior to masking) of the ALARMDIS interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

22.3.1.8 STT_MISR (STT Masked Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																											ALARMDIS	ALARMEN	CNTDIS	CNTEN	ALARM
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
ALARM	[0]	R	Gives the masked interrupt status (After masking) of the ALARM interrupt.	0
CNTEN	[1]	R	Gives the masked interrupt status (After masking) of the CNTEN interrupt.	0
CNTDIS	[2]	R	Gives the masked interrupt status (After masking) of the CNTDIS interrupt.	0
ALARMEN	[3]	R	Gives the masked interrupt status (After masking) of the ALARMEN interrupt.	0
ALARMDIS	[4]	R	Gives the masked interrupt status (After masking) of the ALARMDIS interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

22.3.1.9 STT_ ICR (STT Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																											ALARMDIS	ALARMEN	CNTDIS	CNTEN	ALARM
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
ALARM	[0]	W	0 = No Effect 1 = Clears the ALARM interrupt	0
CNTEN	[1]	W	0 = No Effect 1 = Clears the CNTEN interrupt	0
CNTDIS	[2]	W	0 = No Effect 1 = Clears the CNTDIS interrupt	0
ALARMEN	[3]	W	0 = No Effect 1 = Clears the ALARMEN interrupt	0
ALARMDIS	[4]	W	0 = No Effect 1 = Clears the ALARMDIS interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

22.3.1.10 STT_SR (STT Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																						ALARMENS	CNTENS	RSVD	WSEC	RSVD					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
WSEC	[6]	R	Write Counter. 0 = No effect 1 = A write is occurring on the second counter register	0
CNTENS	[8]	R	Counter Enable Status. 0 = Counter is disabled 1 = Counter is enabled	0
ALARMENS	[9]	R	Alarm Enable Status. 0 = Alarm is disabled 1 = Alarm is enabled	0

22.3.1.11 STT_CNTR (STT Count Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
COUNT																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
COUNT	[31:0]	RW	Counter register. Number of LFCLK clock cycles periods elapsed since last reset to zero.	0x0000_0000

Caution: This register can only be written when CNTENS = 0.
An invalid data (i.e. value greater or equal to 0xA8C0_0000) will not be written into the counter register if CNTRST = 0.

22.3.1.12 STT_ALR (STT Alarm Register)

- Address = Base Address + 0x002C Reset Value = 0x0000_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
ALARMREG																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
ALARMREG	[31:0]	RW	Alarm Register. An interrupt can be generated when the counter register reaches this value.	0x0000_0000

Caution: This register can only be written when CNTENS = 0.
An invalid data (i.e. value greater or equal to 0xA8C0_0000) will not be written into the counter register if CNTRST = 0.

23 Timer/Counter

23.1 Overview

This sector describes 16-bit timer/counter. The 16-bit timer can operate in interval mode, capture mode, match & overflow, or PWM mode. The clock source for timer can be an internal or an external clock. User can enable or disable the timer by setting control bits in the corresponding timer mode register.

23.1.1 Features

- Programmable clock source for timer, including an external clock
- Input capture capability with programmable trigger edge on input pin
- Support several operation modes
    - Interval mode, Capture mode, Match & Overflow mode, PWM mode

23.1.2 Pin Description

Table 23-1 Pin Description

Pin Name	Function	I/O Type	Comments
TCLK[7:0]	External Clock Input Pin	I	–
TCAP[7:0]	Capture Pin	I	–
TPWM[7:0]	PWM Output Pin	O	–

23.2 Functional Description

23.2.1 Block Diagram

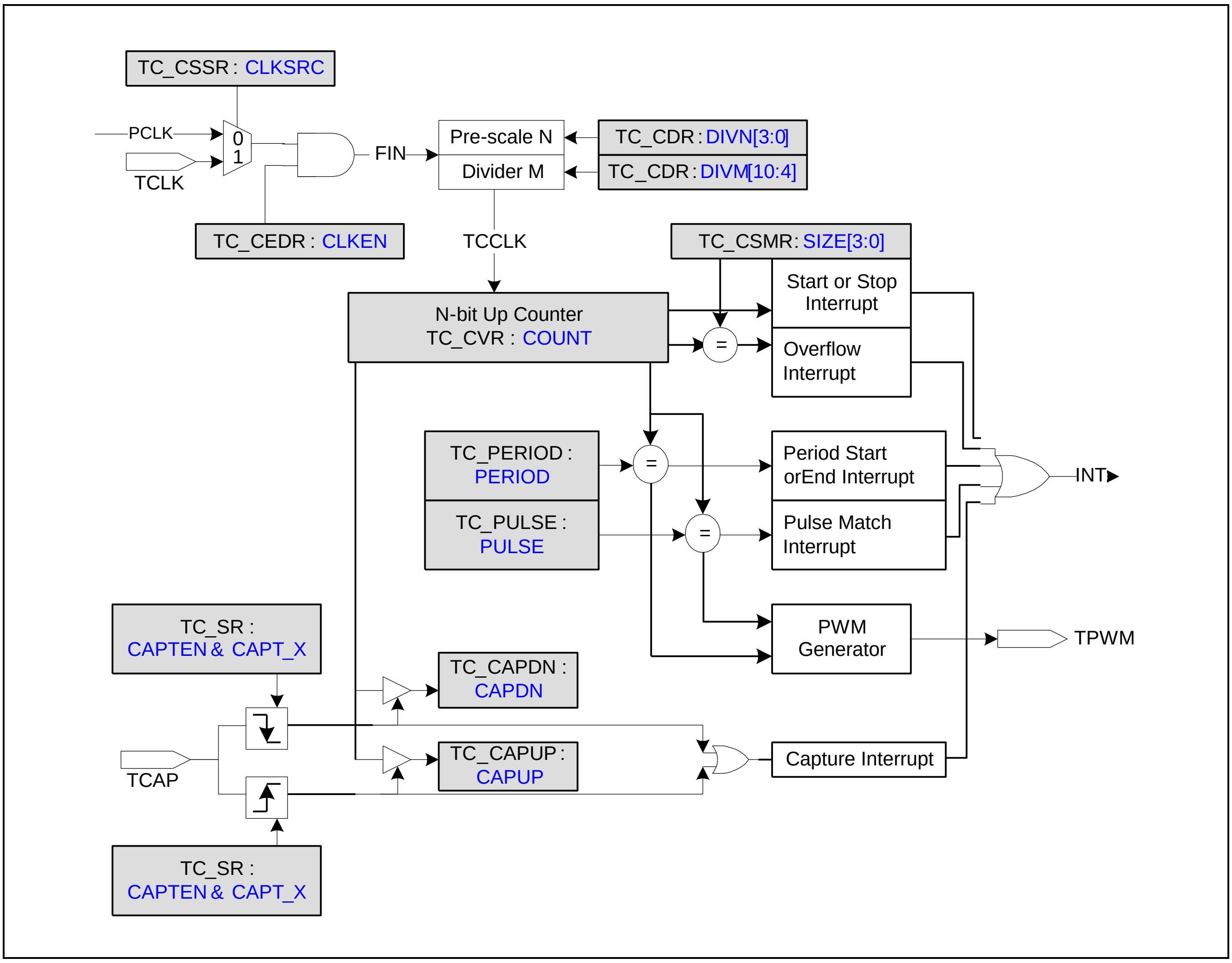


Figure 23-1 TC Block Diagram

NOTE: TCLK is a pin defined to assert the timer clock. That will be supported from an external clock source, instead of PCLK supported from an internal clock source. It must be equal or less than PCLK.

23.2.2 Timer Count Size

There are two registers for the timer count size, TC_CSMR (Counter Size Mask Register) and TC_CCSMR (Current Counter Size Mask Register). They include the SIZE[3:0] filed. TC_CSMR is copied to the TC_CCSMR when a timer starts or UPDATE in TC_CSR (Control Set Register) sets to ‘1’. It shows the information for the counter bit size of a current operating timer. During operating, it can prepare the new counter size with TC_CSMR. If the value of SIZE[3:0] is n, TC will become a (n+1) bit timer. In other words, this timer can count from 1 to $2^{(n+1)}$–1. It can be up to a 16-bit timer.

23.2.3 Timer Clock

23.2.3.1 Clock Source

TC can use one of 2 different clocks (Internal or external clock). An internal clock is PCLK and external clock is asserted via TCLK pin. To use the external clock supplied by TCLK, pin configuration should be done before enable timer. In case of an external clock, it must be equal or less than PCLK.

23.2.3.2 Count Clock

The count clock based on FIN is determined by DIVM[10:0] and DIVN[3:0] in TC_CDR (Clock Divider Register) finally. The frequency of the counting clock is TCCLK determined by FIN and the internal clock divider (DIVM and DIVN).

- TCCLK = FIN/2^{DIVN}/(DIVM[10:0] +1)

Counter Resolution = 1/TCCLK

Also it has read-only TC_CCDR (Current Clock Divider Register). That includes the value of current clock divider during operating. That is copied from TC_CDR when a timer starts or UPDATE bit sets to '1'. User can control the resolution of timer by changing the clock divider.

23.2.4 Operation Modes

23.2.4.1 Match & Overflow Mode Operation

In this mode, a match signal can be generated when the counter value is identical to the value written to the timer data register TC_PERIOD. However, the match signal does not clear the counter even if it can generate a match interrupt as same as the interval mode. Because it does not clear the counter value, the timer can run up to the overflow of counter value and generate an overflow interrupt, also. After the overflow of counter value, the counter value will be counted from 0x0001, again.

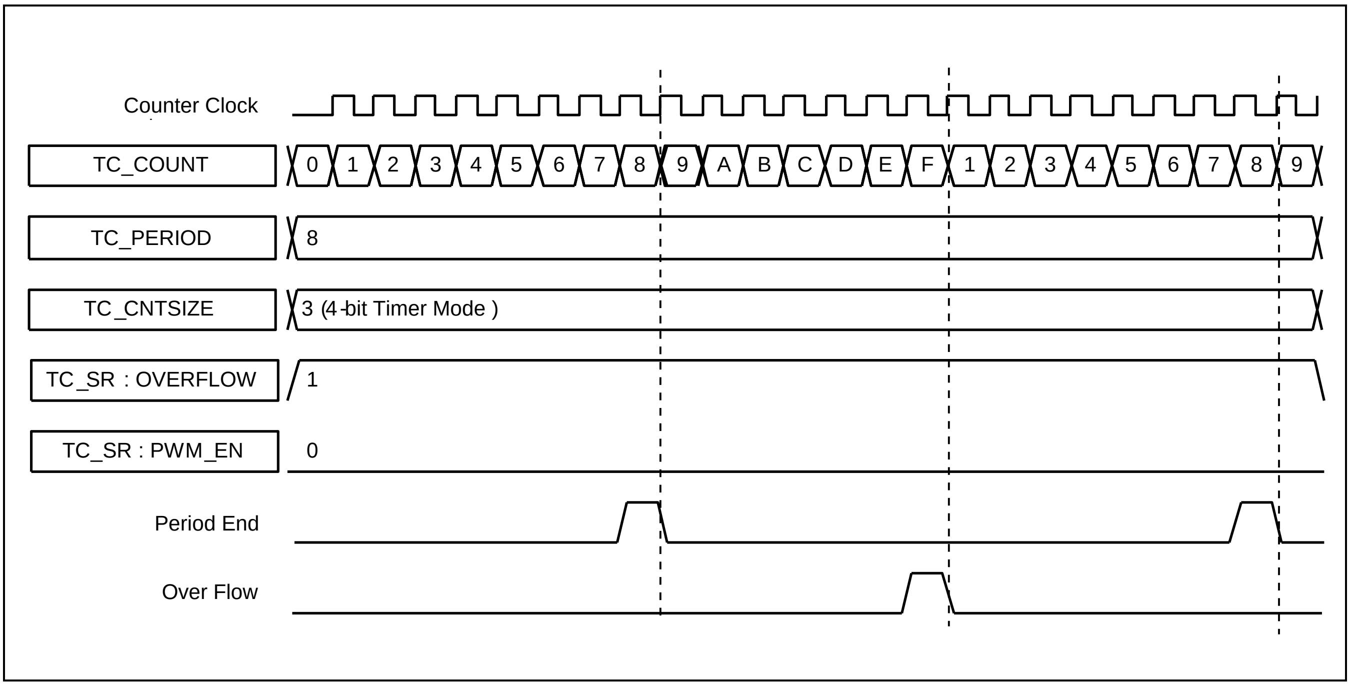


Figure 23-2 Simplified Timer Function Diagram: Match & Overflow Timer Mode

23.2.4.2 Capture Mode Operation

In capture mode, the timer can perform the capturing operation, which is that the counter value is transferred into the capture register (TC_CAPUP and TC_CAPDN) in synchronization with an external trigger. The external triggering signal for capturing operation is a pre-defined valid edge on the capture input pin. When this valid signal happens, the counter value in process should be copied into the capture register (TC_CAPUP and TC_CAPDN). By using the capturing function, you can measure the time difference between external events. If a valid trigger signal on the pin does not happen before the overflow, an overflow interrupt will be generated and the counter value will be counted from 0x0001, again.

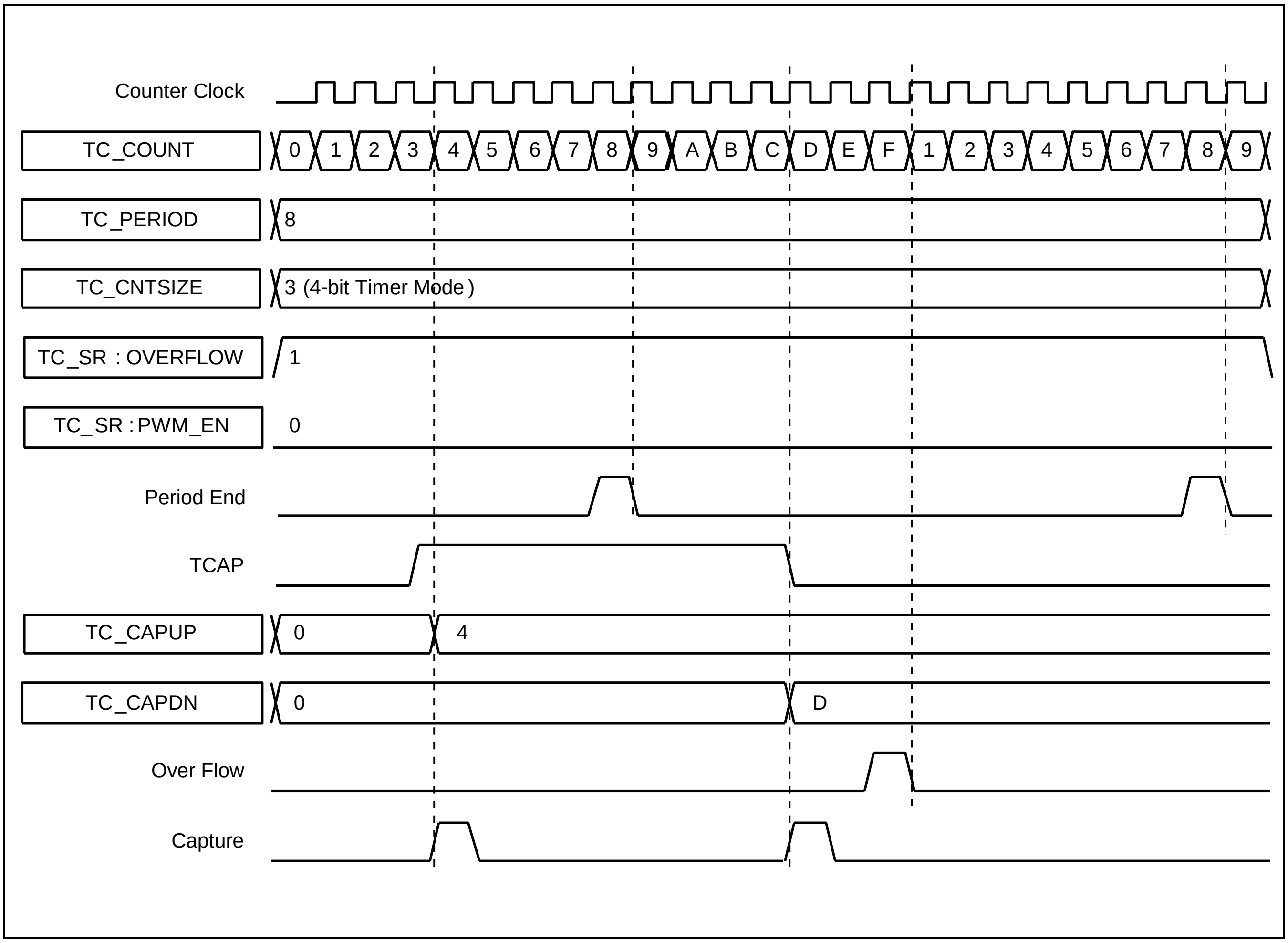


Figure 23-3 Simplified Timer Function Diagram: Capture Mode

23.2.4.3 Interval Mode Operation

In an interval mode, TPWM output level toggles whenever the period end condition detected. A period end signal should be generated when the counter value is identical to the value written to the timer data register, TC_PERIOD. Then the up counter restarts from 0x0001.

If, for example, you write the value '0x08' to TC_PERIOD, the counter will increment until it reaches '0x08'. At this point, the timer period end interrupt is generated. The period is equal to (TCCLK × TC_PERIOD).

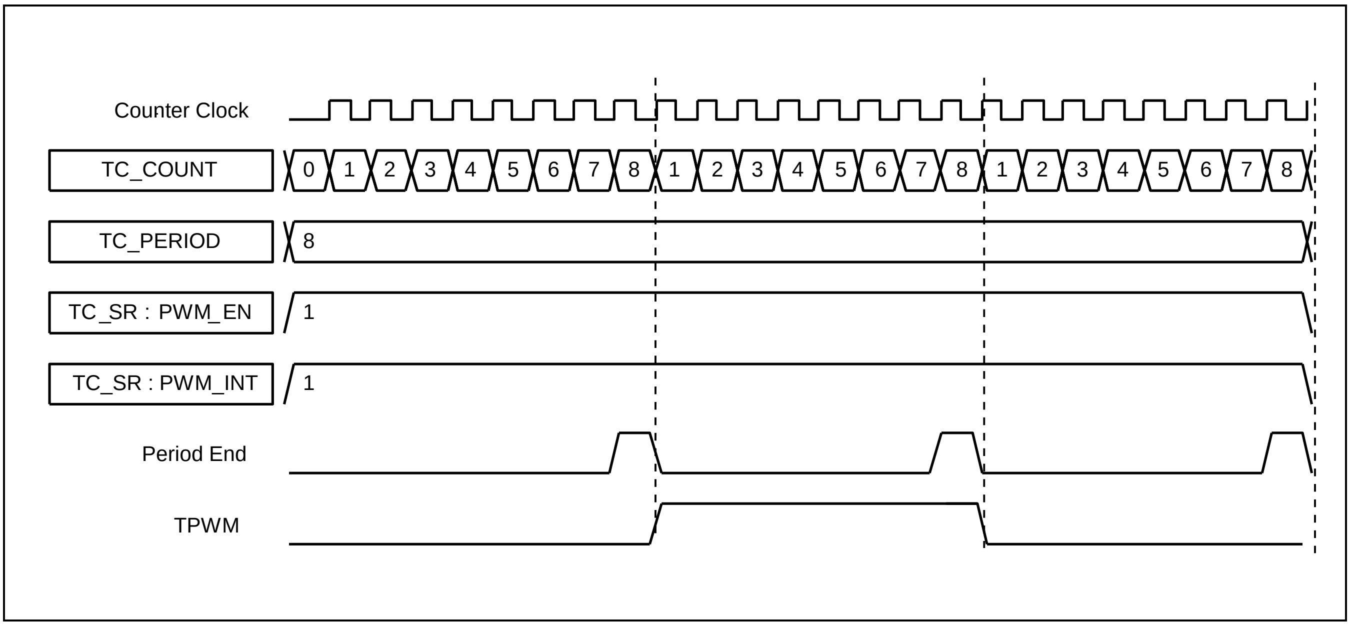


Figure 23-4 Simplified Timer Function Diagram: Interval Timer Mode

23.2.4.4 PWM Mode Operation

The timer can be used for generating the PWM (Pulse Width Modulation) signal. In this mode, user must define PWM period and pulse by writing TC_PERIOD and TC_PULSE register. During operating, TC_CPERIOD and TC_CPULSE show the current configuration value. That is update by START or UPDATE control bit.

TC_PULSE must be smaller than or equal to TC_PERIOD. When the counter value became equal to TC_PULSE, the pulse match signal generated and TPWM output is toggled. After then, the counter value became equal to TC_PERIOD, the period end signal generated and TPWM output is toggled and counter restart to count from 0x0001 if TC_SR.REPEAT is 1. If TC_SR.REPEAT is 0, the counter stops when the counter value became equal to TC_PERIOD.

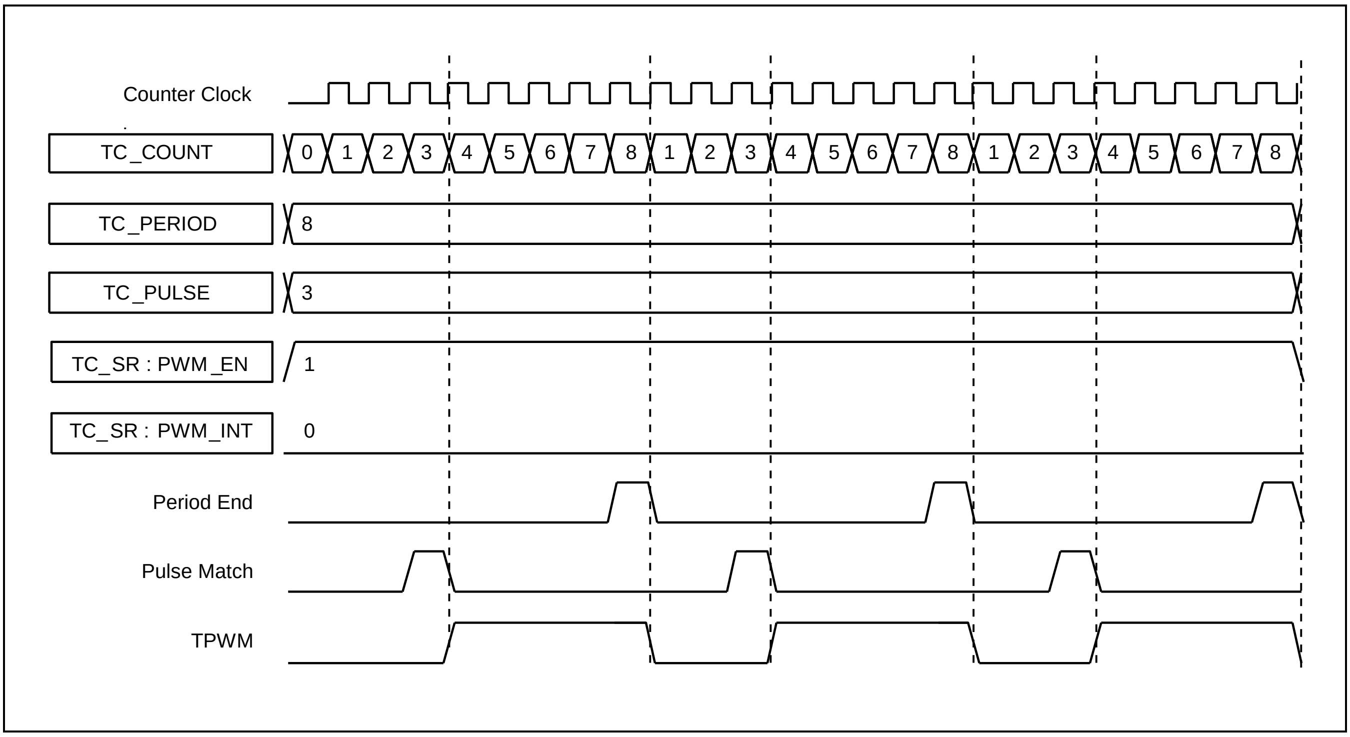


Figure 23-5 Simplified Timer Function Diagram: PWM Mode

23.2.5 Interrupt

There are 7 types of an interrupt, STARTI, STOPI, PSTARTI, PENDI, MATCHI, OVFI and CAPTI.

23.2.5.1 Start Interrupt

A start signal can be generated when the timer starts.

23.2.5.2 Stop Interrupt

A stop signal can be generated when the timer stops.

23.2.5.3 Period Start Interrupt

A period start signal can be generated when the timer is configured as PWM mode and PWM starts.

23.2.5.4 Period End Interrupt

A period end signal can be generated when the timer is configured as PWM mode and PWM ends after start.

23.2.5.5 Match Interrupt

A match signal can be generated when the counter value is identical to the value read from the current period register, TC_CPERIOD.

23.2.5.6 Overflow Interrupt

The timer can run up to the overflow of counter value and generate an overflow interrupt, also. After the overflow of counter value, the counter value will be counted from 0, again.

23.2.5.7 Capture Interrupt

A capture signal can be generated when the counter value is identical to the value written to the timer data register, FRT_DBR.

23.2.5.8 Interruption Handling

- Interrupt Service Routine (ISR) Entry and call C function.
- Read TC_IMSR and verify the source of the interrupt.
- Clear the corresponding interrupt at peripheral level by writing in the TC_ICR
- Interrupt treatment
- ISR Exit.

23.3 Register Description

23.3.1 Register Map Summary

- Base Address: 0x4006_0000
- Base Address: 0x4006_1000
- Base Address: 0x4006_2000
- Base Address: 0x4006_3000
- Base Address: 0x4006_4000
- Base Address: 0x4006_5000

Register	Offset	Description	Reset Value
TC_IDR	0x0000	ID Register	0x0011_000A
TC_CSSR	0x0004	Clock Source Selection Register	0x0000_0000
TC_CEDR	0x0008	Clock Enable/Disable Register	0x0000_0000
TC_SRR	0x000C	Software Reset Register	0x0000_0000
TC_CSR	0x0010	Control Set Register	0x0000_0000
TC_CCR	0x0014	Control Clear Register	0x0000_0000
TC_SR	0x0018	Status Register	0x0000_0000
TC_IMSCR	0x001C	Interrupt Mask Set/Clear Register	0x0000_0000
TC_RISR	0x0020	Raw Interrupt Status Register	0x0000_0000
TC_MISR	0x0024	Masked Interrupt Status Register	0x0000_0000
TC_ICR	0x0028	Interrupt Clear Register	0x0000_0000
TC_CDR	0x002C	Clock Divider Register	0x0000_0000
TC_CSMR	0x0030	Counter Size Mask Register	0x0000_000F
TC_PRDR	0x0034	Period Register	0x0000_0000
TC_PULR	0x0038	Pulse Register	0x0000_0000
TC_CCDR	0x003C	Current Clock Divider Register	0x0000_0000
TC_CCSMR	0x0040	Current Counter Size Mask Register	0x0000_000F
TC_CPRDR	0x0044	Current Period Register	0x0000_0000
TC_CPULR	0x0048	Current Pulse Register	0x0000_0000
TC_CUCR	0x004C	Capture Up Count Register	0x0000_0000
TC_CDCR	0x0050	Capture Down Count Register	0x0000_0000
TC_CVR	0x0054	Counter Value Register	0x0000_0000

23.3.1.1 TC_IDR (Timer/Counter ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0011_000A

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0011_000A

23.3.1.2 TC_CSSR (Timer/Counter Clock Source Selection Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
RSVD																															CLKSRC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKSRC	[0]	RW	Clock Source Selection Field. 0 = Counter Clock Source is PCLK 1 = Counter Clock Source is External Clock which is provided by system.	0

Caution: The frequency of external clock should be slower than PCLK.
When the Timer/Counter clock is enabled, User cannot change CLKSRC. Thus before change CLKSRC, user must disable the clock enable bit of clock enable register.

23.3.1.3 TC_CEDR (Timer/Counter Clock Enable/Disable Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable Control Bit. 0 = Counter Clock is disabled 1 = Counter Clock is enabled	0
DBGEN	[31]	RW	Debug Mode Enable/Disable Control Bit. 0 = Disable Debug Mode, In debug mode, TC is not halted (No influence on the TC function) 1 = Enable Debug Mode, In debug mode, TC is halted (Freeze the TC function)	0

23.3.1.4 TC_SRR (Timer/Counter Software Reset Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No Effect 1 = Timer/Counter Software Reset	0

23.3.1.5 TC_CSR (Timer/Counter Control Set Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD					CAPT_R	CAPT_F	CAPTEN	ADCTRG	OVFM	REPEAT	PWMEN	PWMIM	KEEP	OUTSL	IDLESL	RSVD				STOPCLEAR	STOPHOLD	UPDATE	START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	W	W	W	W	W	W	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	W	W	W	W

Name	Bit	Type	Description	Reset Value
START	[0]	W	0 = No Effect 1 = Counter Start	0
UPDATE	[1]	W	Update PWM parameter. 0 = No Effect 1 = PERIOD, PULSE, CLOCKDIV, CNTSIZE register updated at the same time.	0
STOPHOLD	[2]	W	Stop Count Hold. 0 = No Effect 1 = Enable Stop Hold. When the user stops counter, the counter stops and holds current counter value.	0
STOPCLEAR	[3]	W	Stop Count Clear. 0 = No Effect 1 = Enable Stop Clear. When the user stops counter, the counter stops and counter initialized.	0
IDLESL	[8]	W	IDLE State Level. 0 = No Effect 1 = Idle level of PWM output will be high	0
OUTSL	[9]	W	PWM Output Start Level. 0 = No Effect 1 = PWM output level will be high when starting counting	0
KEEP	[10]	W	Keep Stop Level. 0 = No Effect 1 = Keep State Mode, Override I_LEVEL and keep PWM output level as the last output level even if the counter stops.	0
PWMIM	[11]	W	0 = No Effect 1 = Interval Mode, PWM output will be toggled every period	0
PWMEN	[12]	W	PWM Enable. 0 = No Effect	0

Name	Bit	Type	Description	Reset Value
			1 = PWM Output Enabled	
REPEAT	[13]	W	Auto Repeat Mode. 0 = No Effect 1 = Counter period counting is automatically repeated	0
OVFM	[14]	W	Overflow Mode. 0 = No Effect 1 = Overflow Mode, When the Counter value equals Period, Counter continues to count and finally overflow interrupt will be generated.	0
ADTRIG	[15]	W	ADC Trigger. 0 = No Effect 1 = Enable ADC Trigger Signal-Out	0
CAPTEN	[16]	W	Capture Enable. 0 =No Effect 1 =Capture Mode enabled	0
CAPT_F	[17]	W	Capture by Falling Edge Trigger. 0 = No Effect 1 = Falling edge of external input signal is detected and the Counter value is stored to capture up register.	0
CAPT_R	[18]	W	Capture by Rising Edge Trigger. 0 = No Effect 1 = Rising edge of external input signal is detected and the Counter value is stored to capture up register.	0
PWMEXy	[29:24]	W	PWM output extension.	0

23.3.1.6 TC_CCR (Timer/Counter Control Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD					CAPT_R	CAPT_F	CAPTEN	ADCTRG	OVFM	REPEAT	PWMEN	PWMIM	KEEP	OUTSL	IDLESL	RSVD				STOPCLEAR	STOPHOLD	UPDATE	START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	W	W	W	W	W	W	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W	W	R	R	R	R	W	W	W	W

Name	Bit	Type	Description	Reset Value
START	[0]	W	0 = No Effect 1 = Counter Stop	0
UPDATE	[1]	W	Update PWM parameter. 0 = No Effect 1 = PERIOD, PULSE, CLOCKDIV, CNTSIZE register updated at the same time.	0
STOPHOLD	[2]	W	0 = No Effect 1 = Disable Stop Hold Mode.	0
STOPCLEAR	[3]	W	0 = No Effect 1 = Disable Stop Clear Mode.	0
IDLESL	[8]	W	IDLE State Level Selection Bit. 0 = No Effect 1 = Idle level of PWM output will be Low	0
OUTSL	[9]	W	PWM Start Level Selection Bit. 0 = No Effect 1 = PWM output level will be low when starting counting	0
KEEP	[10]	W	0 = No Effect 1 = Disable Keep State Mode	0
PWMIM	[11]	W	0 = No Effect 1 = PWM Mode	0
PWMEN	[12]	W	0 = No Effect 1 = PWM Output disabled	0
REPEAT	[13]	W	0 = No Effect 1 = Disable Repeat Mode	0
OVFM	[14]	W	0 = No Effect 1 = Disable Overflow Mode	0
ADTRIG	[15]	W	ADC Trigger. 0 = No Effect 1 = Disable ADC Trigger Signal-Out	0

Name	Bit	Type	Description	Reset Value
CAPTEN	[16]	W	0 = No Effect 1 = Capture Mode disabled	0
CAPT_F	[17]	W	0 = No Effect 1 = Disable falling edge capture.	0
CAPT_R	[18]	W	0 = No Effect 1 = Disable rising edge capture	0
PWMEXy	[29:24]	W	PWM output extension.	0

23.3.1.7 TC_SR (Timer/Counter Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD		PWMEX5	PWMEX4	PWMEX3	PWMEX2	PWMEX1	PWMEX0	RSVD					CAPT_R	CAPT_F	CAPTEN	ADCTRG	OVFM	REPEAT	PWMEN	PWMIM	KEEP	OUTSL	IDLESL	RSVD				STOPCLEAR	STOPHOLD	RSVD	START
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
START	[0]	R	0 = Counter is stopped 1 = Counter is started	0
STOPHOLD	[2]	R	0 = Stop Hold mode is disabled 1 = Stop Hold mode is enabled	0
STOPCLEAR	[3]	R	0 = Stop Clear mode is disabled 1 = Stop Clear mode is enabled	0
IDLESL	[8]	R	0 = Idle level of PWM is low 1 = Idle level of PWM is high	0
PWMSL	[9]	R	0 = Output level of PWM is started from low 1 = Output level of PWM is started from high	0
KEEP	[10]	R	0 = Keep state mode disabled 1 = Keep state mode enabled	0
PWMIM	[11]	R	0 = PWM output is PWM mode 1 = PWM output is Interval mode	0
PWMEN	[12]	R	0 = PWM output is disabled 1 = PWM output is enabled	0
REPEAT	[13]	R	0 = Automatic repeat is disabled 1 = Automatic repeat is enabled	0
OVFM	[14]	R	0 = Overflow mode is disabled 1 = Overflow mode is enabled	0
ADTRIG	[15]	R	ADC Trigger. 0 = Disable ADC Trigger Signal-Out 1 = Enable ADC Trigger Signal-Out If ADC Trigger selection field value is 'TC' and this bit sets to '1', ADC conversion occurs by timer match signal (event).	0
CAPTEN	[16]	R	0 = Capture mode is disabled 1 = Capture mode is enabled	0
CAPT_F	[17]	R	0 = External falling edge capture is disabled 1 = External falling edge capture is enabled	0

Name	Bit	Type	Description	Reset Value
CAPT_R	[18]	R	0 = External rising edge capture is disabled 1 = External rising edge capture is enabled	0
PWMEXy	[29:24]	R	PWM output extension status.	0

23.3.1.8 TC_IMSCR (Timer/Counter Interrupt Mask Set/Clear Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																									CAPTI	OVFI	MATI	PENDI	PSTARTI	STOPI	STARTI
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
STARTI	[0]	RW	Start Interrupt Mask Set/Clear Bit. This interrupt will occur when user starts the enabled PWM. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
STOPI	[1]	RW	Stop Interrupt Mask Set/Clear Bit. This interrupt will occur when user stops the operating PWM. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
PSTARTI	[2]	RW	Period Start Interrupt Mask Set/Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
PENDI	[3]	RW	Period End Interrupt Mask Set/Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
MATI	[4]	RW	Pulse Start Interrupt Mask Set/Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
OVFI	[5]	RW	Timer/Counter Overflow Interrupt Mask Set/Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0
CAPTI	[6]	RW	Capture Trigger Interrupt Mask Set/Clear Bit. 0 = Each interrupt is masked. (Disable an interrupt) 1 = Interrupt is not masked. (Enable an interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

23.3.1.9 TC_RISR (Timer/Counter Raw Interrupt Status Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																									CAPTI	OVFI	MATI	PENDI	PSTARTI	STOPI	STARTI
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
STARTI	[0]	R	Gives the raw interrupt state (Prior to masking) of the START interrupt.	0
STOPI	[1]	R	Gives the raw interrupt state (Prior to masking) of the STOP interrupt.	0
PSTARTI	[2]	R	Gives the raw interrupt state (Prior to masking) of the PSTA interrupt.	0
PENDI	[3]	R	Gives the raw interrupt state (Prior to masking) of the PEND interrupt.	0
MATI	[4]	R	Gives the raw interrupt state (Prior to masking) of the PULSESTA interrupt.	0
OVFI	[5]	R	Gives the raw interrupt state (Prior to masking) of the OVF interrupt.	0
CAPTI	[6]	R	Gives the raw interrupt state (Prior to masking) of the CAPT interrupt.	0

TC_RISR does not affected by TC_IMSCR

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

23.3.1.10 TC_MISR (Timer/Counter Masked Interrupt Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_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
RSVD																									CAPTI	OVFI	MATI	PENDI	PSTARTI	STOPI	STARTI
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
STARTI	[0]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
STOPI	[1]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
PSTARTI	[2]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
PENDI	[3]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
MATI	[4]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
OVFI	[5]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0
CAPTI	[6]	R	Gives the masked interrupt status (After masking) of the START interrupt.	0

TC_MISR is affected by TC_IMSCR
TC_MISR = TC_IMSCR & TC_RISR

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

0 = Each interrupt doesn't occur.
1 = Each interrupt occurs.

23.3.1.11 TC_ICR (Timer/Counter Interrupt Clear Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																									CAPTI	OVFI	MATI	PENDI	PSTARTI	STOPI	STARTI
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
STARTI	[0]	W	0 = No Effect 1 = Clears the START interrupt	0
STOPI	[1]	W	0 = No Effect 1 = Clears the STOP interrupt	0
PSTARTI	[2]	W	0 = No Effect 1 = Clears the PSTART interrupt	0
PENDI	[3]	W	0 = No Effect 1 = Clears the PEND interrupt	0
MATI	[4]	W	0 = No Effect 1 = Clears the MATCH interrupt	0
OVFI	[5]	W	0 = No Effect 1 = Clears the OVF interrupt	0
CAPTI	[6]	W	0 = No Effect 1 = Clears the CAPT interrupt	0

On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

23.3.1.12 TC_CDR (Timer/Counter Clock Divider Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																	DIVM											DIVN			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DIVN	[3:0]	RW	DIVN[3:0] is 4-bit.	0
DIVM	[14:4]	RW	Writing TC_CDR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0

Counter Clock is defined as following equation.

Timer Counter Clock = (Clock Source) / (DIVM + 1) / 2^DIVN

NOTE: If a timer obtains the clock from an external source via TCLK pin, operating clock should be equal or less than PCLK.

23.3.1.13 TC_CSMR (Timer/Counter Counter Size Mask Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_000F

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

Name	Bit	Type	Description	Reset Value
SIZE	[3:0]	RW	Timer/Counter Size Mask Field. For example) If SIZE is 0x07 then Timer/Counter acts as 8-bit Timer/Counter. If SIZE is 0x09 then Timer/Counter acts as 10-bit Timer/Counter. If SIZE is 0x0f then Timer/Counter acts as 16-bit Timer/Counter. Writing TC_CSMR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0xF

23.3.1.14 TC_PRDR (Timer/Counter Period Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																PERIOD															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
PERIOD	[15:0]	RW	PWM Period Width. If Timer/Counter is 16-bit, PERIOD[15:0] is valid and PERIOD[31:16] will be read as 0x0000. Writing TC_PRDR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0x0000

23.3.1.15 TC_PULR (Timer/Counter Pulse Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																PULSE															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Name	Bit	Type	Description	Reset Value
PULSE	[15:0]	RW	PWM Pulse Width. If Timer/Counter is 16-bit, PULSE[15:0] is valid and PULSE[31:16] will be read as 0x0000. Writing TC_PULR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0x0000

23.3.1.16 TC_CCDR (Timer/Counter Current Clock Divider Register)

- Address = Base Address + 0x003C, Reset Value = 0x0000_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
RSVD																	DIVM											DIVN			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DIVN	[3:0]	R	DIVN[3:0] is 4-bit.	0x0
DIVM	[14:4]	R	Writing TC_CDR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0x000

Counter Clock is defined as following equation.

Counter Clock = (Clock Source) / (DIVM + 1) / 2^DIVN

23.3.1.17 TC_CCSMR (Timer/Counter Current Counter Size Mask Register)

- Address = Base Address + 0x0040, Reset Value = 0x0000_000F

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

Name	Bit	Type	Description	Reset Value
SIZE	[3:0]	R	Timer/Counter Size Mask Field. For example) If SIZE is 0x07, then Timer/Counter acts as 8-bit Timer/Counter. If SIZE is 0x09 then Timer/Counter acts as 10-bit Timer/Counter. If SIZE is 0x0f then Timer/Counter acts as 16-bit Timer/Counter. Writing TC_CSMR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0xF

23.3.1.18 TC_CPRDR (Timer/Counter Current Period Register)

- Address = Base Address + 0x0044, Reset Value = 0x0000_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
RSVD																PERIOD															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PERIOD	[15:0]	R	PWM Period Width. If Timer/Counter is 16-bit, PERIOD[15:0] is valid and PERIOD[31:16] will be read as 0x0000. Writing TC_PRDR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0x0000

23.3.1.19 TC_CPULR (Timer/Counter Current Pulse Register)

- Address = Base Address + 0x0048, Reset Value = 0x0000_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
RSVD																PULSE															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PULSE	[15:0]	R	PWM Pulse Width. If Timer/Counter is 16-bit, PULSE[15:0] is valid and PULSE[31:16] will be read as 0x0000. Writing TC_PULR register is completed when UPDATE = 1 or START = 1 condition of TC_CSR register.	0x0000

23.3.1.20 TC_CUCR (Timer/Counter Capture up Counter Value Register)

- Address = Base Address + 0x004C, Reset Value = 0x0000_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
RSVD																COUNT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
COUNT	[15:0]	R	Capture Up Count Value. If Timer/Counter is 16-bit, COUNT[15:0] is valid and COUNT[31:16] will be read as 0x0000. The TC_CVR register value is automatically copied to TC_CUCR register when rising edge of external input is detected.	0x0000

23.3.1.21 TC_CDCR (Timer/Counter Capture down Counter Value Register)

- Address = Base Address + 0x0050, Reset Value = 0x0000_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
RSVD																COUNT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
COUNT	[15:0]	R	Capture Down Counter Value. If Timer/Counter is 16-bit, COUNT[15:0] is valid and COUNT[31:16] will be read as 0x0000. The TC_CVR register value is automatically copied to TC_CDCR register when falling edge of external input is detected.	0x0000

23.3.1.22 TC_CVR (Timer/Counter Counter Value Register)

- Address = Base Address + 0x0054, Reset Value = 0x0000_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
RSVD																COUNT															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
COUNT	[15:0]	R	Current Count Value. If Timer/Counter is 16-bit, COUNT[15:0] is valid and COUNT[31:16] will be read as 0x0000.	0x0000

24 Universal Sync/Async Receiver/Transmitter

24.1 Overview

The Universal Asynchronous/Synchronous Receiver/Transmitter (USART) controller is used for communications between microcontrollers. The USART takes bytes of data and transmits the individual bits sequentially (Low Significant Bit first). At the destination, a second USART re-assembles the bits into complete bytes.

Serial transmission is commonly used with modems and for non-networked communication between computers, terminals and other devices.

There are two primary forms of serial transmission: Synchronous and Asynchronous. The asynchronous mode uses two lines for data transfer: RX for the reception and TX for the transmission or of the bits. The synchronous mode uses an additional clock signal used to strobe input and output data.

The transmitter and receiver module of USART/USART are connected to the Lite Direct Memory Access Controller. The Direct Memory Access Controller allows the transmission or reception of data bytes independently of the microcontroller. See Direct Memory Access Controller chapter for more information.

24.1.1 Features

- Programmable baud rate generator
- Parity, framing and overrun error detection
- Idle flag for J1587 protocol
- Line break generation and detection
- Automatic echo, local loopback and remote loopback channel modes
- Multi-drop mode: address detection and generation
- Interrupt generation
- 2 dedicated LDMA channels per USART
- 5 to 9 bit character length
- Configurable start bit of data transmission
- Support the LIN protocol: LIN1.2 or LIN2.0 configurable release
- Smart–Card protocol: error signaling and re-transmission
- Asynchronous mode maximum baud rate: PCLK/16
- Synchronous mode maximum baud rate when providing SCK clock: PCLK/2
- Synchronous mode maximum baud rate when receiving SCK clock: PCLK/4

24.1.2 Pin Description

Table 24-1 USART Pin Description

Pin Name	Function	I/O Type	Active Level	Comments
USARTTXD[3:0]	USART Transmit Data Line	O	–	–
USARTRXD[3:0]	USART Reception Data Line	I	–	–
USARTCLK[3:0]	USART transmission clock	Bi-Direction	–	–

24.2 Functional Description

24.2.1 Block Diagram

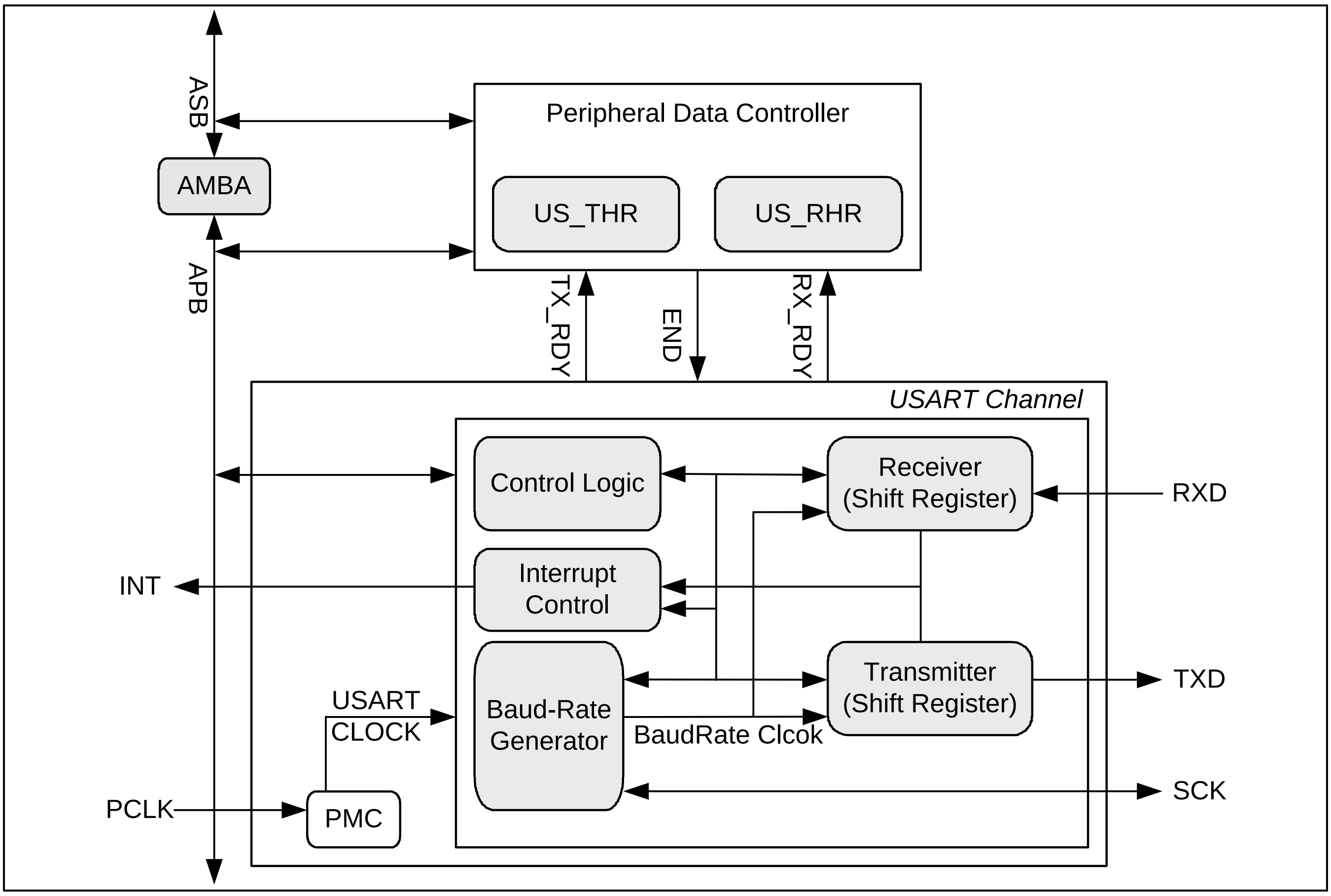


Figure 24-1 Template_REV1.08 USART Block Diagram

24.2.2 Baud Rate Generator

24.2.2.1 General Description

The baud rate generator provides the bit period clock named the baud rate clock to both the receiver and the transmitter. The baud rate generator can select between external and internal clock sources. The external clock source is USARTCLK[3:0]. The internal clock sources can be either PCLK or PCLK divided by 8 (PCLK/8).

NOTE: In all cases, if an external clock is used, the duration of each of its levels must be longer than the PCLK period. The external clock frequency must be less than 40% of PCLK frequency.

24.2.2.2 Asynchronous mode

When the USART is programmed to operate in asynchronous mode (SYNC = 0 in the Mode Register US_MR), the selected clock is divided by 16 times the value (CD) written in US_BRGR (Baud Rate Generator Register). If US_BRGR is set to 0, the baud rate clock is disabled.

- Baud Rate = Selected Clock/ 16 × CD where selected clock is either PCLK, PCLK/ 8 or USARTCLK[3:0]

24.2.2.3 Synchronous mode

When the USART is programmed to operate in synchronous mode (SYNC = 1 in the Mode Register US_MR) and the selected clock is internal (CLKS[1] = 0 in the Mode Register US_MR), the Baud Rate Clock is the internal selected clock divided by the value written in US_BRGR. If US_BRGR is set to 0, the Baud Rate Clock is disabled.

- Baud Rate = Selected Clock/ CD where selected clock is either PCLK, PCLK/ 8 or USARTCLK[3:0]

In synchronous mode with external clock selected (CLKS[1] = 1 in the Mode Register US_MR), the clock is provided directly by the signal on the USARTCLK[3:0] pin. No division is active. The value written in US_BRGR has no effect.

24.2.2.4 Block Diagram

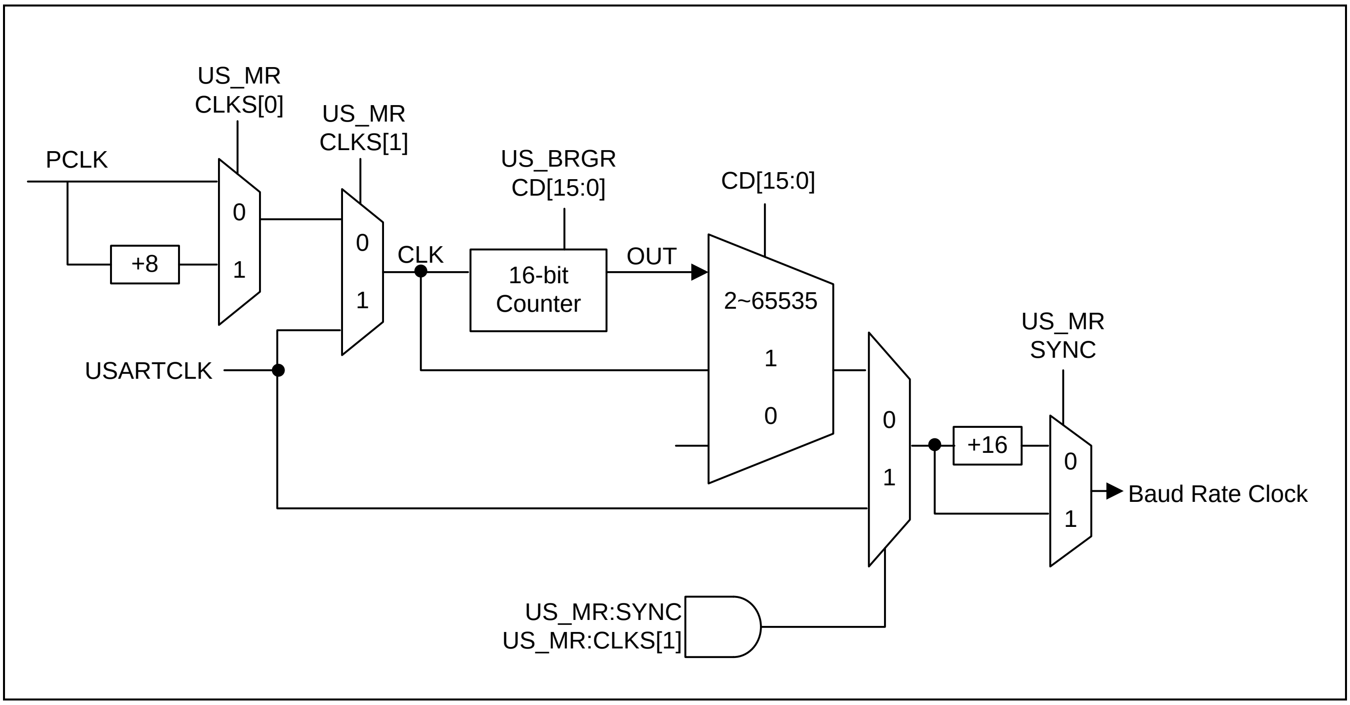


Figure 24-2 USART Baud Rate Generator Block Diagram

24.2.2.5 Baud Rate Configuration Example

The following table shows different register configuration of the US_BRGR register for different core frequency. For each case, the calculated error shows the difference between the real baud rate and the expected one.

In the following table, CLKS[1:0] = 00 (PCLK selected as USART clock USARTCLK) and SYNC = 0 (asynchronous mode) in the US_MR register.

Table 24-2 Asynchronous Mode (SYNC = 0)

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
75	3906	1200	–0.01%
	1953	2400	–0.01%
	977	4800	0.04%
	488	9600	–0.06%
	326	14400	0.15%
	244	19200	–0.06%
	122	38400	–0.06%
	81	57600	–0.47%
72	3750	1200	0.00%
	1875	2400	0.00%
	938	4800	0.05%
	469	9600	0.05%
	313	14400	0.16%
	234	19200	–0.16%
	117	38400	–0.16%
	78	57600	–0.16%
	39	115200	–0.16%
60	3125	1200	0.00%
	1563	2400	0.03%
	781	4800	–0.03%
	391	9600	0.10%
	260	14400	–0.16%
	195	19200	–0.16%
	98	38400	0.35%
	65	57600	–0.16%
40	2083	1200	–0.02%
	1042	2400	0.03%
	521	4800	0.03%
	260	9600	–0.16%
	174	14400	0.22%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
	130	19200	–0.16%
	65	38400	–0.16%
37.5	1953	1200	–0.01%
	977	2400	0.04%
	488	4800	–0.06%
	244	9600	–0.06%
	163	14400	0.15%
	122	19200	–0.06%
	61	38400	–0.06%
36	1875	1200	0.00%
	938	2400	0.05%
	469	4800	0.05%
	234	9600	–0.16%
	156	14400	–0.16%
	117	19200	–0.16%
	39	57600	–0.16%
30	1563	1200	0.03%
	781	2400	–0.03%
	391	4800	0.10%
	195	9600	–0.16%
	130	14400	–0.16%
	98	19200	0.35%
	49	38400	0.35%
20	1042	1200	0.03%
	521	2400	0.03%
	260	4800	–0.16%
	130	9600	–0.16%
	87	14400	0.22%
	65	19200	–0.16%
18.75	977	1200	0.04%
	488	2400	–0.06%
	244	4800	–0.06%
	122	9600	–0.06%
	81	14400	–0.47%
	61	19200	–0.06%
18	938	1200	0.05%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
	469	2400	0.05%
	234	4800	–0.16%
	117	9600	–0.16%
	78	14400	–0.16%
16	833	1200	–0.04%
	417	2400	0.08%
	208	4800	–0.16%
	104	9600	–0.16%
	52	19200	–0.16%
	26	38400	–0.16%
15	781	1200	–0.03%
	391	2400	0.10%
	195	4800	–0.16%
	98	9600	0.3%
	65	14400	–0.16%
	49	19200	0.35%
10	521	1200	0.03%
	260	2400	–0.16%
	130	4800	–0.16%
	65	9600	–0.16%
9.375	488	1200	–0.06%
	244	2400	–0.06%
	122	4800	–0.06%
	61	9600	–0.06%
8	417	1200	0.08%
	208	2400	–0.16%
	104	4800	–0.16%
	52	9600	–0.16%
	26	19200	–0.16%
	13	38400	–0.16%
5	260	1200	–0.16%
	130	2400	–0.16%
	65	4800	–0.16%
4.6875	244	1200	–0.06%
	122	2400	–0.06%
	61	4800	–0.06%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
4	208	1200	–0.16%
	104	2400	–0.16%
	52	4800	–0.16%
	26	9600	–0.16%
	13	19200	–0.16%
2.5	130	1200	–0.16%
	65	2400	–0.16%
2	104	1200	–0.16%
	52	2400	–0.16%
	26	4800	–0.16%
	13	9600	–0.16%
1.25	65	1200	–0.16%
1	52	1200	–0.16%
	26	2400	–0.16%
	13	4800	–0.16%
0.5	26	1200	–0.16%
	13	2400	–0.16%
0.25	13	1200	–0.16%

Table 24-3 Synchronous Mode (SYNC = 1)

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
75	3906 × 16	1200	–0.01%
	1953 × 16	2400	–0.01%
	977 × 16	4800	0.04%
	488 × 16	9600	–0.06%
	326 × 16	14400	0.15%
	244 × 16	19200	–0.06%
	122 × 16	38400	–0.06%
	81 × 16	57600	–0.47%
72	3750 × 16	1200	0.00%
	1875 × 16	2400	0.00%
	938 × 16	4800	0.05%
	469 × 16	9600	0.05%
	313 × 16	14400	0.16%
	234 × 16	19200	–0.16%
	117 × 16	38400	–0.16%
	78 × 16	57600	–0.16%
	39 × 16	115200	–0.16%
60	3125 × 16	1200	0.00%
	1563 × 16	2400	0.03%
	781 × 16	4800	–0.03%
	391 × 16	9600	0.10%
	260 × 16	14400	–0.16%
	195 × 16	19200	–0.16%
	98 × 16	38400	0.35%
	65 × 16	57600	–0.16%
40	2083 × 16	1200	–0.02%
	1042 × 16	2400	0.03%
	521 × 16	4800	0.03%
	260 × 16	9600	–0.16%
	174 × 16	14400	0.22%
	130 × 16	19200	–0.16%
	65 × 16	38400	–0.16%
37.5	1953 × 16	1200	–0.01%
	977 × 16	2400	0.04%
	488 × 16	4800	–0.06%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
	244 × 16	9600	–0.06%
	163 × 16	14400	0.15%
	122 × 16	19200	–0.06%
	61 × 16	38400	–0.06%
36	1875 × 16	1200	0.00%
	938 × 16	2400	0.05%
	469 × 16	4800	0.05%
	234 × 16	9600	–0.16%
	156 × 16	14400	–0.16%
	117 × 16	19200	–0.16%
	39 × 16	57600	–0.16%
30	1563 × 16	1200	0.03%
	781 × 16	2400	–0.03%
	391 × 16	4800	0.10%
	195 × 16	9600	–0.16%
	130 × 16	14400	–0.16%
	98 × 16	19200	0.35%
	49 × 16	38400	0.35%
20	1042 × 16	1200	0.03%
	521 × 16	2400	0.03%
	260 × 16	4800	–0.16%
	130 × 16	9600	–0.16%
	87 × 16	14400	0.22%
	65 × 16	19200	–0.16%
18.75	977 × 16	1200	0.04%
	488 × 16	2400	–0.06%
	244 × 16	4800	–0.06%
	122 × 16	9600	–0.06%
	81 × 16	14400	–0.47%
	61 × 16	19200	–0.06%
18	938 × 16	1200	0.05%
	469 × 16	2400	0.05%
	234 × 16	4800	–0.16%
	117 × 16	9600	–0.16%
	78 × 16	14400	–0.16%
16	833 × 16	1200	–0.04%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
	417 × 16	2400	0.08%
	208 × 16	4800	–0.16%
	104 × 16	9600	–0.16%
	52 × 16	19200	–0.16%
	26 × 16	38400	–0.16%
15	781 × 16	1200	–0.03%
	391 × 16	2400	0.10%
	195 × 16	4800	–0.16%
	98 × 16	9600	0.35%
	65 × 16	14400	–0.16%
	49 × 16	19200	0.35%
10	521 × 16	1200	0.03%
	260 × 16	2400	–0.16%
	130 × 16	4800	–0.16%
	65 × 16	9600	–0.16%
9.375	488 × 16	1200	–0.06%
	244 × 16	2400	–0.06%
	122 × 16	4800	–0.06%
	61 × 16	9600	–0.06%
8	417 × 16	1200	0.08%
	208 × 16	2400	–0.16%
	104 × 16	4800	–0.16%
	52 × 16	9600	–0.16%
	26 × 16	19200	–0.16%
	13 × 16	38400	–0.16%
5	260 × 16	1200	–0.16%
	130 × 16	2400	–0.16%
	65 × 16	4800	–0.16%
4.6875	244 × 16	1200	–0.06%
	122 × 16	2400	–0.06%
	61 × 16	4800	–0.06%
4	208 × 16	1200	–0.16%
	104 × 16	2400	–0.16%
	52 × 16	4800	–0.16%
	26 × 16	9600	–0.16%
	13 × 16	19200	–0.16%

PCLK (MHz)	US_BRGR CD[15:0]	Baud Rate	% Error
2.5	130 × 16	1200	–0.16%
	65 × 16	2400	–0.16%
2	104 × 16	1200	–0.16%
	52 × 16	2400	–0.16%
	26 × 16	4800	–0.16%
	13 × 16	9600	–0.16%
1.25	65 × 16	1200	–0.16%
1	52 × 16	1200	–0.16%
	26 × 16	2400	–0.16%
	13 × 16	4800	–0.16%
0.5	26 × 16	1200	–0.16%
	13 × 16	2400	–0.16%
0.25	13 × 16	1200	–0.16%

24.2.3 Receiver

24.2.3.1 Asynchronous Receiver

The USART is configured for asynchronous operation when SYNC = 0 (bit 8 of US_MR). In asynchronous mode, the USART detects the start of a received character by sampling the RXD signal until it detects a valid start bit. A low level (space) on RXD is interpreted as a valid start bit if it is detected for more than 7 cycles of the sampling clock, which is 16 times the baud rate. Hence a space which is longer than 7/16 of the bit period is detected as a valid start bit. A space that is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the RXD at the theoretical mid-point of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (One bit period) so the sampling point is 8 cycles (0.5 bit periods) after the start of the bit. The first sampling point is therefore 24 cycles (1.5 bit periods) after the falling edge of the start bit was detected. Each subsequent bit is sampled 16 cycles (1 bit period) after the previous one.

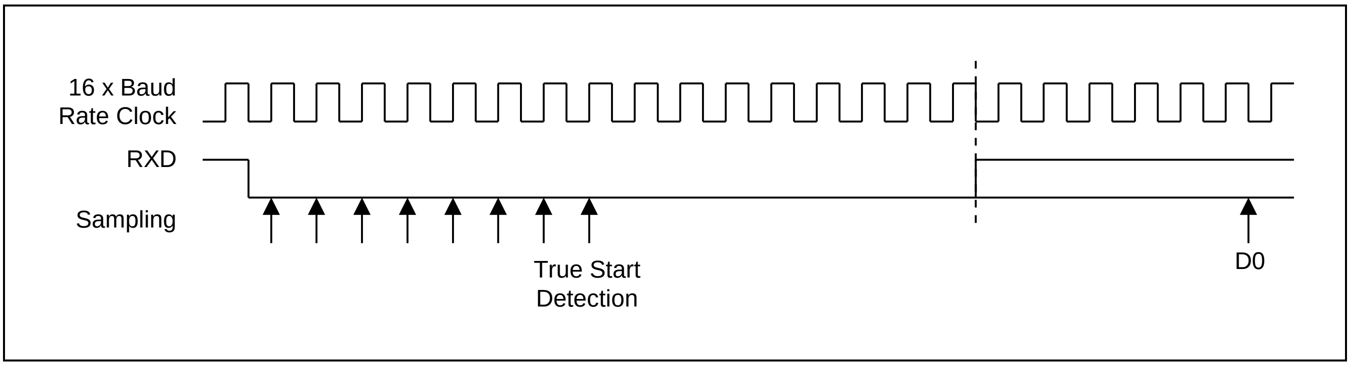


Figure 24-3 Asynchronous Mode, Start Bit Detection

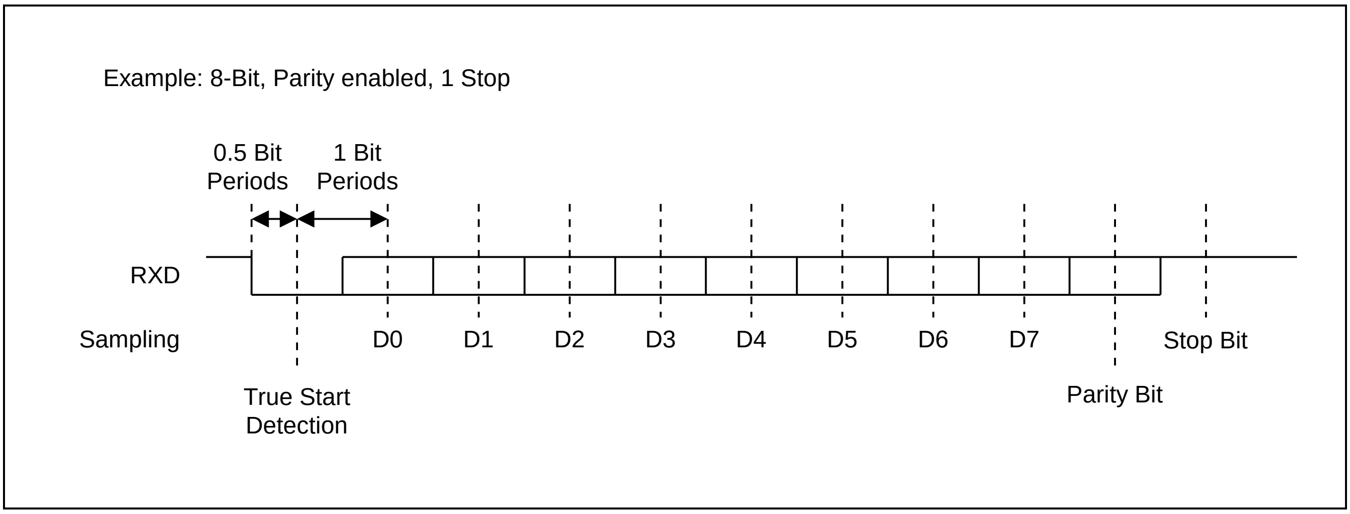


Figure 24-4 Asynchronous Mode, Character Reception

24.2.3.2 Synchronous Receiver

When configured for synchronous operation (SYNC = 1), the receiver samples the RXD signal on each rising edge of USARTCLK[3:0]. If a low level is detected, it is considered as a start. Data bits, parity bit and stop bit are sampled and the receiver waits for the next start bit. See example below.

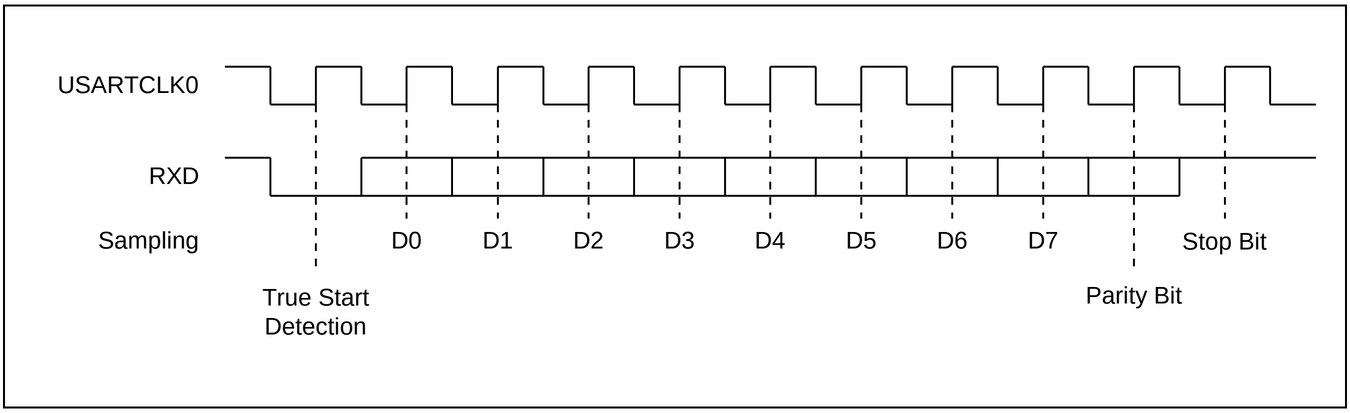


Figure 24-5 Synchronous Mode, Character Reception

24.2.3.3 Receiver Ready

When a complete character is received, it is transferred to the US_RHR and the RXRDY status bit in US_SR is set. The RXRDY is set after the last stop bit.

If the US_RHR register has not been read since the last transfer and before a transfer to the US_RHR register takes place, the OVRE status bit is set in US_SR register.

24.2.3.4 Overrun Error

If US_RHR has not been read since the last transfer, the OVRE status bit in US_SR is set.

24.2.3.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR[2:0] in US_MR(USART Mode Register, Page 1-30). It then compares the result with the received parity bit. If different, the parity error bit PARE in US_SR is set.

24.2.3.6 Framing Error

If a character is received with a stop bit at low level and with at least one data bit at high level, a framing error is generated. This sets FRAME in US_SR.

24.2.3.7 IDLE Flag

The idle flag turns low when USART receive a start bit and turn high at the end of a J1587 protocol frame (After 10 stop bits). An interrupt can be generated on the rising edge of the idle flag.

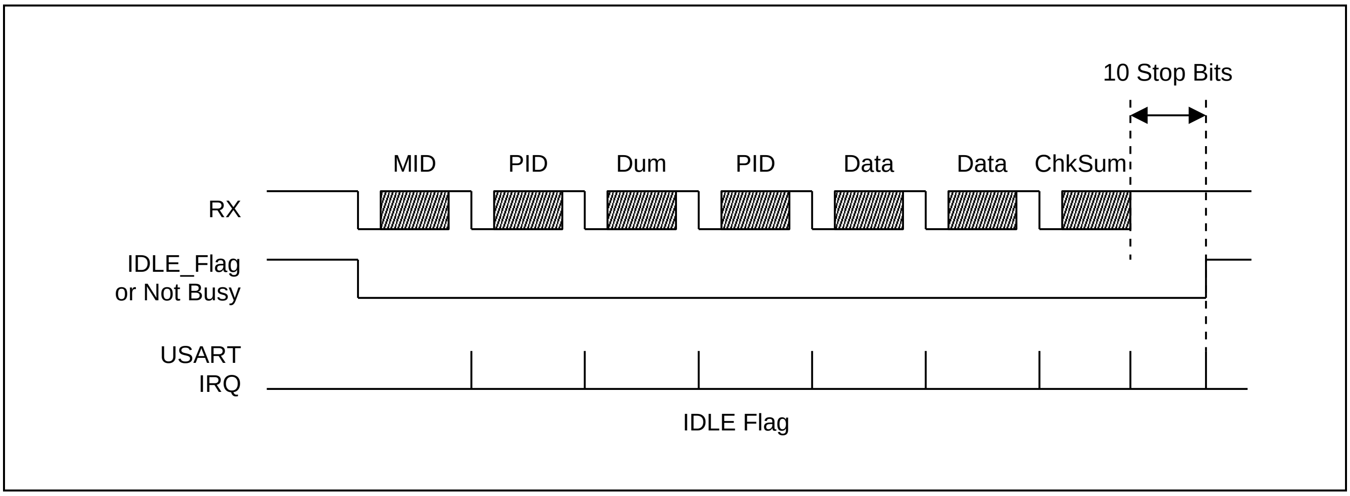


Figure 24-6 IDLE Flag

24.2.3.8 Time-Out

This function allows an idle condition on the RXD line to be detected. The maximum delay for which the USART should wait for a new character to arrive while the RXD line is inactive (High level) is programmed in TO[15:0] of US_RTOR (Receiver Time-out). When this register is set to 0, no time-out is detected.

Otherwise, the receiver waits for a first character and then initializes a counter which is decremented at each bit period and reloaded at each byte reception. When the counter reaches 0, the TIMEOUT bit in US_SR is set. The user starts (or restarts) the wait for a first character by setting the STTTO (Start Time-Out) bit in US_CR register.

In other words, to start a time-out, the following conditions must be met:

- US_RTOR must not be equal to 0
- The time-out must be started by setting to a logical 1 the STTTO (Start time-out) in the US_CR register
- One character must be received

Calculation of Time-Out Duration:

Duration = Value × Bit period in asynchronous mode
Duration = Value × 16 × period in synchronous mode

24.2.4 Transmitter

24.2.4.1 General Description

The transmitter has the same behavior in both synchronous and asynchronous operating modes. Start bit, data bits, parity bit and stop bits are serially shifted, least significant bit first.

The number of data bits is selected in the CHRL[1:0] field in US_MR.

The parity bit is set according to the PAR[2:0] field in US_MR register. If the parity type is even then the parity bit depends on the one bit sum of all data bits. For odd parity, the parity bit is the inverted sum of all data bits.

The number of stop bits is selected in the NBSTOP[1:0] field in US_MR.

When a character is written to US_THR (Transmit Holding), it is transferred to the Shift Register as soon as it is empty.

When the transfer occurs, the TXRDY bit in US_SR is set until a new character is written to US_THR. If Transmit Shift Register and US_THR are both empty, the TXEMPTY bit in US_SR is set (After the last stop bit of the last transfer).

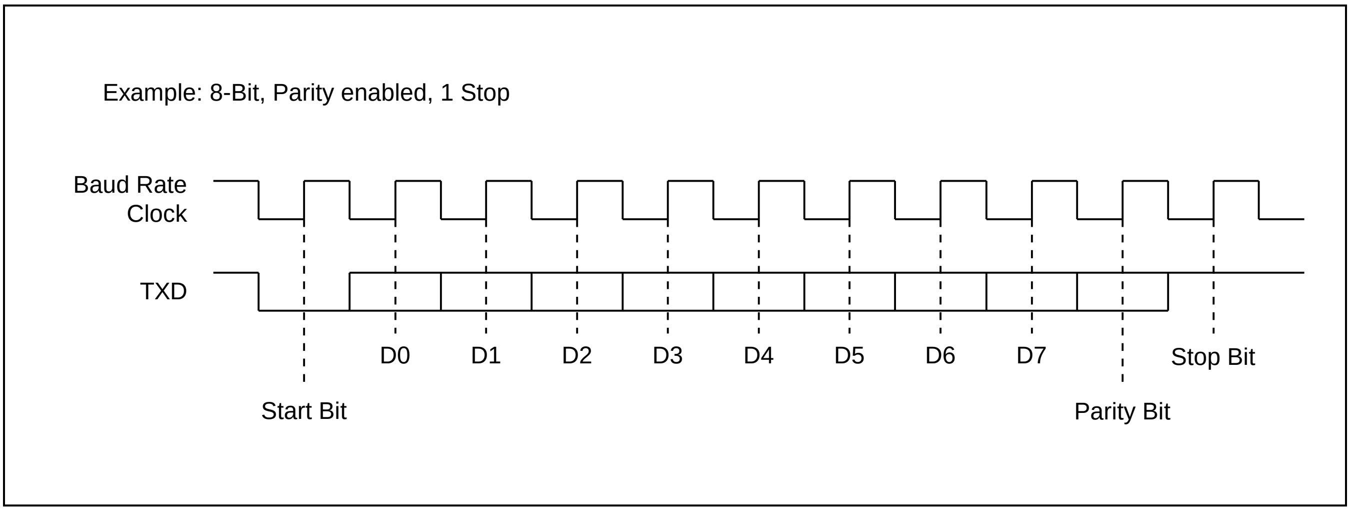


Figure 24-7 Synchronous and Asynchronous Modes, Character Transmission

24.2.4.2 Time-Guard

The Time-guard function allows the transmitter to insert an idle state on the TXD line between 2 characters. The duration of the idle state is programmed in US_TTGR (Transmitter Time-Guard). When this register is set to zero, no time-guard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in US_TTGR.

24.2.4.3 Multi-Drop Mode

When the field PAR field in US_MR equals 11Xb, the USART is configured to run in multi-drop mode for automatic address/ ndata detection and the PARE (Parity error bit in US_SR register) is used to identify a data byte (PARITY bit is detected low) or an address byte (parity bit is high). So, in this mode, the parity error bit (PARE in US_SR) is set when data is identified as an address byte. The PARE status bit is cleared with the PARE bit in the US_CSR (Clear Status Register). If the parity bit is detected low, identifying a data byte, PARE is not set.

The transmitter sends an address byte (Parity bit set) when a Send Address Command (SENDA) is written to US_CR.

In this case, the next byte written to US_THR will be transmitted as an address. After this any byte transmitted will have the parity bit cleared.

24.2.5 Break

24.2.5.1 Transmit Break

The transmitter can generate a break condition on the TXD line when the STTBRK (Start break) command is set in US_CR (Control Register). In this case, the characters present in the Transmit Shift Register are completed before the line is held low.

To remove this break condition on the TXD line, the STPBRK (Stop break) command in US_CR must be set. The USART generates a minimum break duration of one character length.

The TXD line then returns to high level (Idle state) for at least 12 bit periods to ensure that the end of break is correctly detected. Then the transmitter resumes normal operation.

24.2.5.2 Receive Break

The break condition is detected by the receiver when all data, parity and stop bits are low. At the moment of the low stop bit detection, the receiver asserts the RXBRK (Break received) bit in US_SR.

End of receive break is detected by a high level for at least 2/16 of the bit period in asynchronous operating mode or at least one sample in synchronous operating mode. RXBRK is also set after end of break has been detected.

24.2.5.3 Interrupts

Most of status bit in US_SR has a corresponding bit in US_IER (Interrupt Enable) and US_IDR (Interrupt Disable) which controls the generation of interrupts by asserting the USART interrupt line connected to the GIC. US_IMR (Interrupt Mask Register) indicates the status of the corresponding bits.

When a bit is set in US_SR and the same bit is set in US_IMR, the interrupt line is asserted.

24.2.5.4 Test modes

The USART can be programmed to operate in 3 different test modes, using the field CHMODE[1:0] in US_MR.

Automatic echo mode allows bit by bit retransmission.
When a bit is received on the RXD line, it is sent to the TXD line. Programming the transmitter has no effect.

Local loopback mode allows the transmitted characters to be received. TXD and RXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The RXD pin level has no effect and the TXD pin is held high, as in idle state.

Remote loopback mode directly connects the RXD pin to the TXD pin. The Transmitter and the Receiver are disabled and have no effect. This mode allows bit by bit retransmission.

24.2.5.5 LIN protocol (Compliant with LIN1.2 and LIN2.0 Releases)

This section describes the LIN protocol supported by the USART.

The USART works as a "Master Control Unit" (As it is defined on the LIN Protocol Specification) when the LIN bit is set in the US_MR register.

The LIN2_0 bit in the US_MR Mode Register selects the LIN protocol release. By default, USART supports the LIN 1.2 protocol. The software user can not modified this bit during a Message Transfer.

It generates the "HEADER FRAME" automatically if the STHEADER is set in the US_CR Controller Register and the end of header is signaled through the ENDHEADER bit on the Status Register. The header content concerning the "IDENTIFIER" is defined by the Identifier Register (US_LIR on Read/Write access). The parity is automatically calculated. In the header part, the SYNC_BREAK length is configurable through the Sync Break Length Register (US_SBLR) (The SYNC_BREAK could be set from 8Tbit up to 31Tbit).

Setting the STRESP bit in the US_CR register has the effect of sending a “RESPONSE MESSAGE" (See the LIN Protocol Specification). The data to be transmitted are defined in the US_DFWR[1:0] register in the LIN1.2 release, the number of data bytes to be transmitted depends on the IDENTIFIER[5:4] value configured in the US_LIR register. In the LIN2.0 release, the number of data bytes to be transmitted depends on the NDATA[2:0] value configured in US_LIR register. The CHECKSUM field is automatically calculated and sent (For the LIN1.2 release, the checksum is classic. For the LIN2.0 release, the checksum can be enhanced or classic, it depends on the CHK_SEL bit configured in the US_LIR register).

It is also possible to fill the message response to the THR register. In this case, user has to wait the END HEADER interrupt before starting to fill the THR register (LDMAC can be used the data bytes.)

The "RESPONSE MESSAGE" received by the USART is available in the US_DFRR[1:0] register even if the USAT has field the “RESPONSE MESSAGE” in the US_DFWR[1:0]] register.

The USART is able to detect and to signal the end of "LIN MESSAGE" through the ENDMESS bit on the Status Register. Moreover the USART detects the Bit-Error, the Identifier-Parity-Error, Not-Responding-Error, the Checksum-Error and the Wakeup.

Write accesses in the US_LIR and US_DFWR[1:0] registers, LIN2_0 bit in the US_MR register, STHEADER in the US_CR register and SYNC_BRK[4:0] field in the US_SBLR register have no effect during a message transfer.

It is also advised not to write on the Transmitter Holding Register (US_THR) during the Header.

The inter-byte space is configurable through the US_TTGR register (transmit time guard), the default inter-byte space is one Tbit.

24.2.5.6 USART Configuration for LIN

To work in LIN mode, the USART has to be set in normal mode (See the MODE register) with the transmitter and its receiver enabled.

24.2.5.6.1 Message Characteristics

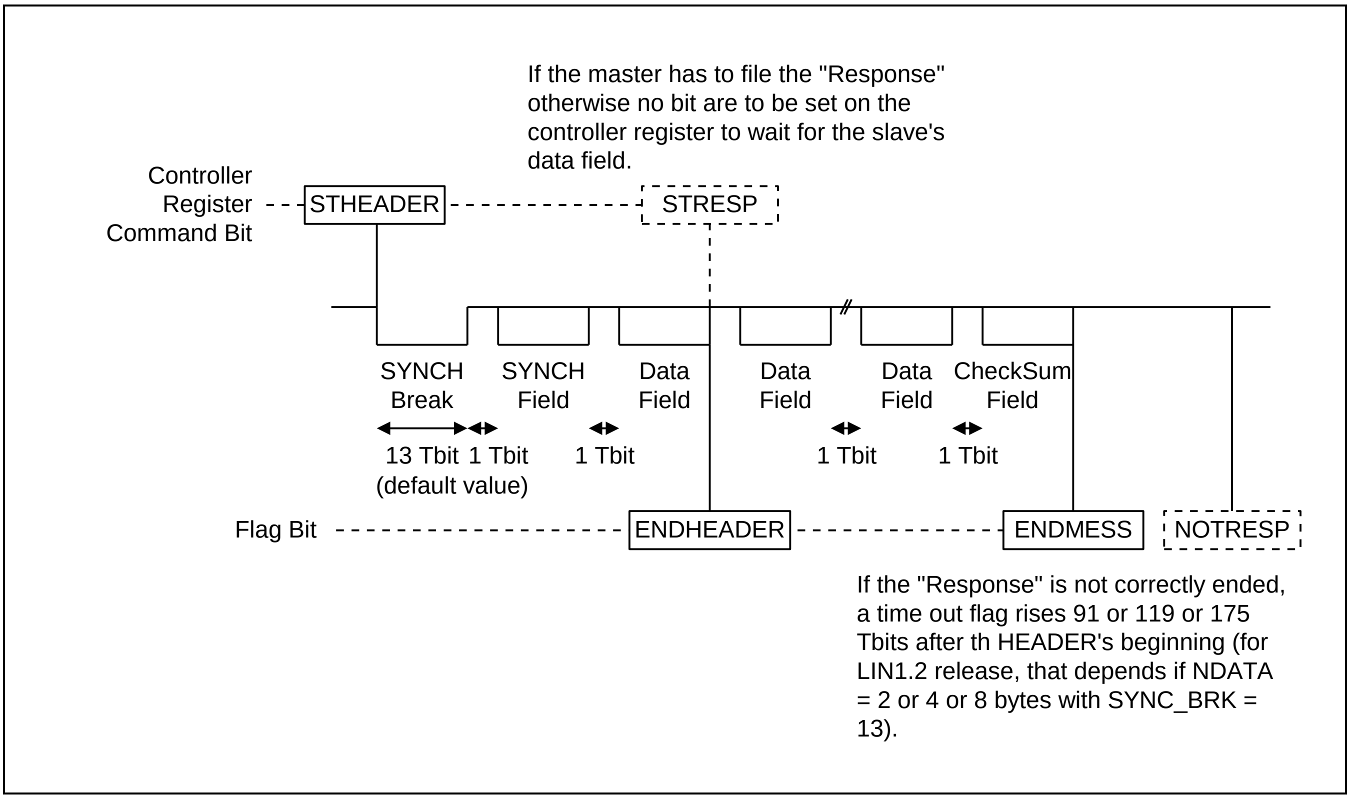

If the master has to file the "Response" otherwise no bit are to be set on the controller register to wait for the slave's data field.
Controller Register Command Bit
STHEADER
STRESP
SYNCH Break
SYNCH Field
Data Field
Data Field
Data Field
CheckSum Field
13 Tbit (default value)
1 Tbit
1 Tbit
1 Tbit
1 Tbit
Flag Bit
ENDHEADER
ENDMESS
NOTRESP
If the "Response" is not correctly ended, a time out flag rises 91 or 119 or 175 Tbits after th HEADER's beginning (for LIN1.2 release, that depends if NDATA = 2 or 4 or 8 bytes with SYNC_BRK = 13).

Figure 24-8 Message Characteristics

24.2.5.6.2 Wake-Up

The USART can wake-up from the sleep mode by sending a Wake-up signal. For the LIN1.2 protocol, this signal consists of the character 0xC0, 0x80 or 0x00. Theses characters will be detected by the master as a valid data byte. For the LIN2.0 protocol, the wake-up request is issued by forcing the line to the dominant state for at least 250us. This will asserts the WAKEUP bit in the US_SR Register.

24.2.5.7 Smart-Card protocol

The USARTs are ISO7816-3 compliant and allow character repetition or error signaling on parity errors.

The following lines describes the functions which are available if the SMCARDPT bit is set on the US_MR register.

The USART smart card protocol requires that both the transmitter and the receiver are enabled.

If PIO block allows it, TXD can be configured as an open drain output and connected to the RXD pin with an additional external pull-up resistor resulting in the smart card COMMS line.

24.2.5.8 Character Transmission to Smart Card

The USART is able to detect that the smart card has not received correctly the last transmitted byte by checking the parity error signal generated by the smart card.

When the card generates the error signal, the last transmitted byte is re-transmitted again as many times as determined in the SENDTIME[2:0] bits of the US_MR register until the error signal is no longer generated by the smart card.

When the error signal is detected by the USART, the FRAME error flag is set in the US_SR register.

The error signal is checked by the USART at t0 + 11 bits where t0 is the falling edge of the start bit (i.e. between the 2 stop bits).

In the following example, a parity error is detected by the smart card. The smart card generates the error signal on the COMMS line. The error signal is detected by the USART which re-transmit the last character.

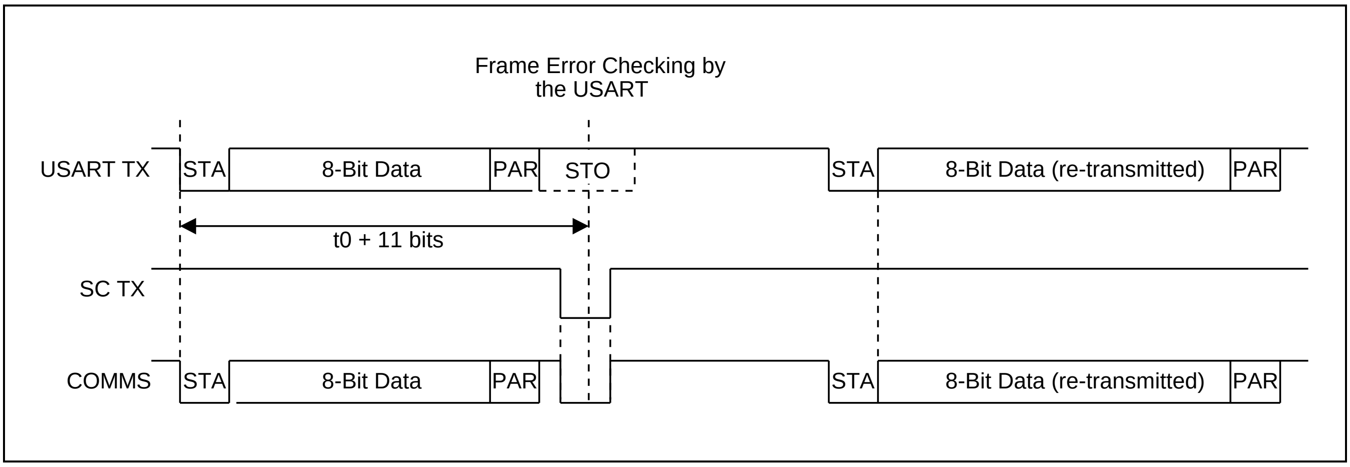

Frame Error Checking by the USART
USART TX
STA
8-Bit Data
PAR
STO
STA
8-Bit Data (re-transmitted)
PAR
t0 + 11 bits
SC TX
COMMS
STA
8-Bit Data
PAR
STA
8-Bit Data (re-transmitted)
PAR

Figure 24-9 Smart-Card Transmission Error

If a LDMA transfer is used to send data byte to the card, the LDMA counter will not be decremented and the LDMA memory pointer will not be incremented until the card has received a correct byte or the maximum repetition time has been reached.

24.2.5.9 Character Reception from Smart Card

The USART is able to generate the error signal (See ISO7816-3 protocol) when last byte received has a parity error.

When a parity error is detected by the USART, the USART transmission line is driven low for 1.0625 bit period starting at t0 + 10.625 + [0:0.0625] (i.e. during the 2 stop bits) to indicate to the smart card that a bad reception has occurred on the USART. With T = 0 protocol type smart cards, the smart card must re-send the last character.

In that case, the USART signals only a parity error by setting the PARE bit in the US_SR register.

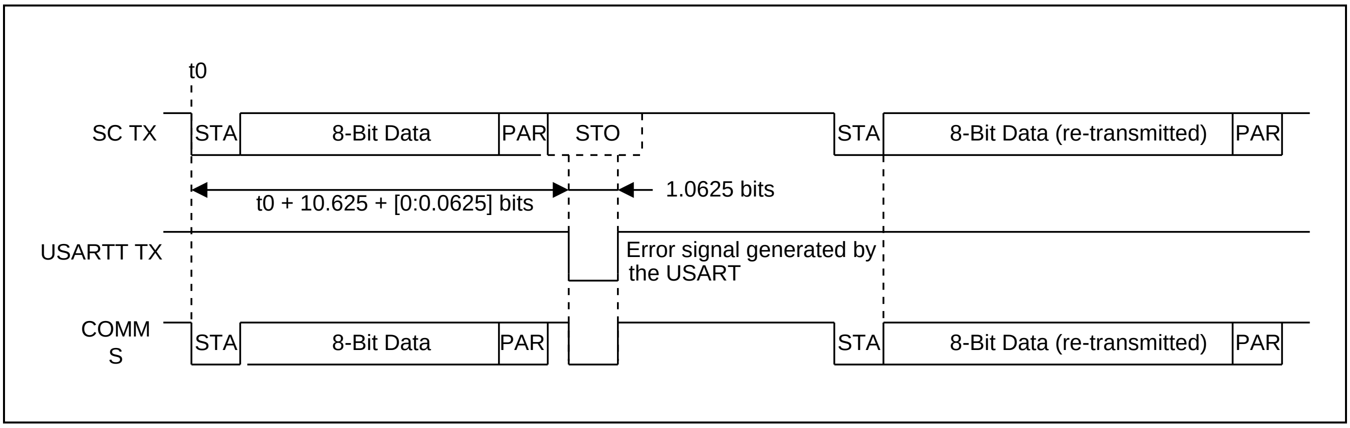

t0
SC TX
STA
8-Bit Data
PAR
STO
STA
8-Bit Data (re-transmitted)
PAR
t0 + 10.625 + [0:0.0625] bits
1.0625 bits
USARTT TX
Error signal generated by the USART
COMM S
STA
8-Bit Data
PAR
STA
8-Bit Data (re-transmitted)
PAR

Figure 24-10 Error Signaling on Reception

If a LDMA transfer is used to receive data byte from the card, the LDMA counter will be decremented and the LDMA memory pointer will be incremented even when a parity error is detected. The user must reconfigure the LDMA memory pointer (Decrement by 1) and counter (Increment by 1) in order to receive all the remaining bytes.

24.2.5.10 USART Configuration in Smart Card Mode

To work in Smart Card mode, the USART has to be set in normal mode and the number of stop bits must be programmed at 2 (see the MODE register).

24.3 Programming Examples

Example of use of the USART with LIN functionality

Send a Header Frame and then send the Message Frame with an identification of 0x30 with a message of 2x16bits at 19600 bps using interrupt.

Configuration:

Enable the clock on USART by writing bit USART in US_ECR.

- Do a software reset of the USART peripheral to be in a known state by writing bit SWRST in US_CR.
- Configuration of US_BRGR: For 19600bps CD field in the US_BRGR=PCLK/(16×19600) (take nearest integer value).
- Enable Transmitter and Receiver bit RXEN and TXEN in the US_CR register.
- Configuration of US_MR register: normal mode, no parity, 8 data bits, 1 stop bit, normal mode, LIN supported, use PCLK clock for USART baud rate generation.
- Configuration of US_SBLR register: Set the syncbreak to 0x13.
- Enter the data to send in DRWR0 and DFWR1.
- Configuration of US_IER register: generate an interrupt at the end of the header and message transmission. This is done by setting the ENDHEADER, ENDMESS in the US_IMR register. Associate interrupt to error like BIT error, Parity error, checksum error … GIC must be configured.
- Configure the identifier by writing 0x30 in the US_LIR register.
- Enable the LIN header transmission by writing bit STHEADER in the US_CR register, this will send the header. Before going to the next step, software should be informed by the interrupt that the header has been sent.
- Enable the LIN response transmission by setting the bit STRESP in the US_CR register, this will send the message response. An interrupt is generated at the end of the transmission.

Interruption Handling:

- IRQ Entry and call C function.
- Read the US_SR and verify the source of the interrupt. This register is read and clear for some status field. Care should be taken to keep status information to be able to proceed with all cases. Bits present in the US_CSR are not read and clear and then should be cleared in the next step.
- Clear the corresponding interrupt at peripheral level by writing in the US_CSR register.
- Interrupt treatment: inform background software that header or message has been transmitted.
- IRQ Exit.

24.4 Register Description

24.4.1 Register Map Summary

- Base Address: 0x4008_0000
- Base Address: 0x4008_1000
- Base Address: 0x4008_2000
- Base Address: 0x4008_3000

Register	Offset	Description	Reset Value
US_IDR	0x0000	ID Register	0x0001_001B
US_CEDR	0x0004	Clock Enable/ Disable Register	0x0000_0000
US_SRR	0x0008	Software Reset Register	0x0000_0000
US_CR	0x000C	Control Register	0x0000_0000
US_MR	0x0010	Mode Register	0x0000_0000
US_IMSCR	0x0014	Interrupt Mask Set/ Clear Register	0x0000_0000
US_RISR	0x0018	Raw Interrupt Status Register	0x0000_0000
US_MISR	0x001C	Masked Interrupt Status Register	0x0000_0000
US_ICR	0x0020	Interrupt Clear Register	0x0000_0000
US_SR	0x0024	Status Register	0x0000_0800
US_RHR	0x0028	Receiver Holding Register	0x0000_0000
US_THR	0x002C	Transmitter Holding Register	0x0000_0000
US_BRGR	0x0030	Baud rate Generator Register	0x0000_0000
US_RTOR	0x0034	Receiver Time-Out Register	0x0000_0000
US_TTGR	0x0038	Transmitter Time-Guard Register	0x0000_0000
US_LIR	0x003C	LIN Identifier Register	0x3AD4_0000
US_DFWR0	0x0040	Data Field Write 0 Register	0x0000_0000
US_DFWR1	0x0044	Data Field Write 1 Register	0x0000_0000
US_DFRR0	0x0048	Data Field Read 0 Register	0x0000_0000
US_DFRR1	0x004C	Data Field Read 1 Register	0x0000_0000
US_SBLR	0x0050	Synchronous Break Length Register	0x0000_000D
US_LCP1	0x0054	Limit Counter Protocol 1	0x7768_5B4C
US_LCP2	0x0058	Limit Counter Protocol 2	0xAFA0_9284
US_DMACR	0x005C	DMA Control Register	0x0000_0000

24.4.1.1 US_IDR (USART ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_001B

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

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_001B

24.4.1.2 US_CEDR (USART Clock Enable Disable Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD																														CLKEN
0	0	0	0	0	0	0	0	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	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW

Name	Bit	Type	Description	Reset Value
CLKEN	[0]	RW	Clock Enable/Disable Control Bit. 0 = USART Clock is disabled. 1 = USART Clock is enabled.	0
DBGEN	[31]	RW	Debug Mode Enable. 1 = Disable debug mode. 0 = Enable debug mode. Read) 1 = The debug acknowledge generated by the ICE interface (dbgack_sclk input signal) has no influence on the USART function. 0 = The debug acknowledge freezes the USART function when the ICE interface is activated (High level on input pin). However full read/write access to internal register is kept for the debug purpose.	0

24.4.1.3 US_SRR (USART Software Reset Register)

- Address = Base Address + 0x0008, Reset Value = 0x0000_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
RSVD																															SWRST
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W

Name	Bit	Type	Description	Reset Value
SWRST	[0]	W	Software Reset. 0 = No Effect. 1 = USART Software Reset.	0

24.4.1.4 US_CR (USART Control Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_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
RSVD												RSTLIN	STMESSAGE	STREPS	STHEADER	RSVD			SENDA	STTTO	STPBRK	STTBRK	RSVD	TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	RSVD	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	R	R	R	W	W	W	W	W	W	W	W	W	W	W	R	R

Name	Bit	Type	Description	Reset Value
RSTRX	[2]	W	Reset Receiver. 0 = No effect. 1 = The receiver logic is reset.	0
RSTTX	[3]	W	Reset Transmitter. 0 = No effect. 1 = The transmitter logic is reset.	0
RXEN	[4]	W	Receiver Enable. 0 = No effect. 1 = The receiver is enabled if RXDIS is 0.	0
RXDIS	[5]	W	Receiver Disable. 0 = No effect. 1 = The receiver is disabled	0
TXEN	[6]	W	Transmitter Enable. 0 = No effect. 1 = The transmitter is enabled if TXDIS is 0.	0
TXDIS	[7]	W	Transmitter Disabled. 0 = No effect. 1 = The transmitter is disabled.	0
STTBRK	[9]	W	Start Break. 0 = No effect. 1 = If break is not being transmitted, start transmission of a break after the characters present in the Transmit Shift Register have been transmitted	0
STPBRK	[10]	W	Stop Break. 0 = No effect. 1 = If a break is being transmitted, stop transmission of the break after a minimum of one character length and transmit a high level during 12 bit periods.	0
STTTO	[11]	W	Start Time-out. 0 = No effect.	0

Name	Bit	Type	Description	Reset Value
			1 = Start waiting for a character before clocking the time-out counter.	
SENDA	[12]	W	Send Address. 0 = No effect. 1 = In Multi-drop Mode only, the next character written to the US_THR is sent with the address bit set.	0
STHEADER	[16]	W	Start Header. 0 = No effect. 1 = If the LIN bit is set on the Mode Register, send the LIN's Header Frame.	0
STRESP	[17]	W	Start Response. 0 = No effect. 1 = Send a part or the Data Field 0 Register content and Data Field 1 Register content.	0
STMESSAGE	[18]	W	Start Message. 0 = No effect. 1 = If the LIN bit is set in the Mode Register, send the Header and the Response consecutively	0
RSTLIN	[19]	W	Reset the LIN. 0 = No effect. 1 =Reset the LIN logic. This bit resets the LIN logic except the APB registers configuration. The line TX/RX is in IDLE state (ie. any current transfer on the RX/TX line is aborted).	0

24.4.1.5 US_MR (USART Mode Register)

- Address = Base Address + 0x0010, Reset Value = 0x0000_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
RSVD											DSB	LIN2_0	CLKO	MODE9	SMCARDPT	CHMODE		NBSTOP		PAR			SYNC	CHRL		CLKS		SENDTIME			LIN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>LIN</td><td>[0]</td><td>RW</td><td>Local Interconnect Network Mode.
0 = Disable LIN protocol on USART.
1 = Enable LIN protocol on USART.</td><td>0</td></tr>
<tr><td>SENDTIME</td><td>[3:1]</td><td>RW</td><td>Indicates the maximum number of repetitions a character has to be transmitted by the USART when configured in smart card protocol.
• SENDTIME Configuration Field.
<table>
<tr><th colspan="3">SENDTIME[2:0]</th><th>Time Number</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td colspan="4">–</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>7</td></tr>
</table></td><td>000'b</td></tr>
<tr><td>CLKS</td><td>[5:4]</td><td>RW</td><td>Clock selection (Baud rate generator input clock).
• CLKS Clock Selection Field.
<table>
<tr><th colspan="2">CLKS[1:0]</th><th>Selected Clock</th></tr>
<tr><td>0</td><td>0</td><td>PCLK</td></tr>
<tr><td>0</td><td>1</td><td>PCLK/8</td></tr>
<tr><td>1</td><td>x</td><td>External clock (USARTCLK0)</td></tr>
</table></td><td>00'b</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>CHRL</td><td>[7:6]</td><td>RW</td><td>Character Length.
Start, stop and parity bits are added to the character length.
• Character Length Field.
<table>
<tr><th colspan="2">CHRL[1:0]</th><th>Character Length</th></tr>
<tr><td>0</td><td>0</td><td>5 bits</td></tr>
<tr><td>0</td><td>1</td><td>6 bits</td></tr>
<tr><td>1</td><td>0</td><td>7 bits</td></tr>
<tr><td>1</td><td>1</td><td>8 bits</td></tr>
</table></td><td>00'b</td></tr>
<tr><td>SYNC</td><td>[8]</td><td>RW</td><td>Synchronous Mode Select.
0 = USART operates in Asynchronous Mode.
1 = USART operates in Synchronous Mode.</td><td>0</td></tr>
<tr><td>PAR</td><td>[11:9]</td><td>RW</td><td>Parity Type.
• Parity Type Field.
<table>
<tr><th colspan="3">PAR[2:0]</th><th>Parity Type</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Even Parity</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Odd Parity</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Parity forced to 0 (Space)</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Parity forced to 1 (Mark)</td></tr>
<tr><td>1</td><td>0</td><td>X</td><td>No parity</td></tr>
<tr><td>1</td><td>1</td><td>X</td><td>Multi-drop mode</td></tr>
</table>
<b>NOTE:</b> For LIN, PAR[2:0] must be set to '10X'.</td><td>000'b</td></tr>
<tr><td>NBSTOP</td><td>[13:12]</td><td>RW</td><td>Number of stop bits.
The interpretation of the number of stop bits depends on SYNC.
• NBSTOP Configuration Field.
<table>
<tr><th colspan="2">NBSTOP [1:0]</th><th>Asynchronous (SYNC = 0)</th><th>Synchronous (SYNC = 1)</th></tr>
<tr><td>0</td><td>0</td><td>1 stop bit</td><td>1 stop bit</td></tr>
<tr><td>0</td><td>1</td><td>1.5 stop bits</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>2 stop bits</td><td>2 stop bits</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td><td>Reserved</td></tr>
</table></td><td>00'b</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>CHMODE</td><td>[15:14]</td><td>RW</td><td>Channel Mode.
• Channel Mode Field.
<table>
<tr><th colspan="2">CHMODE[1:0]</th><th>Mode Description</th></tr>
<tr><td>0</td><td>0</td><td>Normal Mode.
The USART channel operates as a Rx/Tx USART.</td></tr>
<tr><td>0</td><td>1</td><td>Automatic Echo.
Receiver data input is connected to TXD pin.</td></tr>
<tr><td>1</td><td>0</td><td>Local Loopback.
Transmitter output signal is connected to receiver input signal.</td></tr>
<tr><td>1</td><td>1</td><td>Remote Loopback.
RXD pin is internally connected to TXD pin.</td></tr>
</table></td><td>00'b</td></tr>
<tr><td>SMCARDPT</td><td>[16]</td><td>RW</td><td>Smart Card Protocol.
0 = Disable smart card protocol on USART.
1 = Enable LIN protocol on USART.</td><td>0</td></tr>
<tr><td>MODE9</td><td>[17]</td><td>RW</td><td>9-bit Character Length.
0 = The CHRL field defines the character length.
1 = 9-Bit character length.</td><td>0</td></tr>
<tr><td>CLKO</td><td>[18]</td><td>RW</td><td>Clock Output Select.
0 = The USART does not drive the SCK pin.
1 = The USART drives the SCK pin if CLKS[1] is 0.</td><td>0</td></tr>
<tr><td>LIN2_0</td><td>[19]</td><td>RW</td><td>Select the LIN Protocol Release.
This bit is significant when the LIN bit is set. The software user can not changed this value during a Message Transfer (i.e. LIN is busy).
0 = USART supports the LIN1.2 protocol.
1 = USART supports the LIN2.0 protocol.</td><td>0</td></tr>
<tr><td>DSB</td><td>[20]</td><td>RW</td><td>Data Start Bit Selection.
0 = Data transmission starts from LSB, and ends to MSB.
1 = Data transmission starts from MSB, and ends to LSB.</td><td>0</td></tr>
</table>

Caution: Is not possible to set the LIN2_0 bit during a Message Transfer.

24.4.1.6 US_IMSCR (USART Interrupt Mask Set and Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD	WAKEUP	CHECKSUM	IPERROR	BITERROR	NOTRESP	ENDMESS	ENDHEADER	RSVD													IDLE	TXEMPTY	TIMEOUT	PARE	FRAME	OVRE	RSVD		RXBRK	TXRDY	RXRDY
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	RW	RW	RW	RW	RW	RW	RW	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	R	R	RW	RW	RW

Name	Bit	Type	Description	Reset Value
RXRDY	[0]	RW	Receiver Ready Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
TXRDY	[1]	RW	Transmitter Ready Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
RXBRK	[2]	RW	Receiver Break Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
OVRE	[5]	RW	Overrun Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
FRAME	[6]	RW	Framing Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
PARE	[7]	RW	Parity Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
TIMEOUT	[8]	RW	Time-Out Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
TXEMPTY	[9]	RW	Transmitter Empty Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
IDLE	[10]	RW	IDLE Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
ENDHEADER	[24]	RW	Ended Header Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt)	0

Name	Bit	Type	Description	Reset Value
			1 = This interrupt is not masked. (Enable an interrupt)	
ENDMESS	[25]	RW	Ended Message Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
NOTREPS	[26]	RW	Not Responding Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
BITERROR	[27]	RW	Bit Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
IPERROR	[28]	RW	Identity Parity Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
CHECKSUM	[29]	RW	Checksum Error Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0
WAKEUP	[30]	RW	Wake Up Interrupt Mask. 0 = This interrupt is masked. (Disable an interrupt) 1 = This interrupt is not masked. (Enable an interrupt)	0

On a read this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit sets the mask, enabling the interrupt to be read. A write of 0 clears the corresponding mask.

24.4.1.7 US_RISR (USART Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_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
RSVD	WAKEUP	CHECKSUM	IPERROR	BITERROR	NOTRESP	ENDMESS	ENDHEADER	RSVD													IDLE	TXEMPTY	TIMEOUT	PARE	FRAME	OVRE	RSVD		RXBRK	TXRDY	RXRDY
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
RXRDY	[0]	R	Receiver ready Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the RXRDY interrupt.	0
TXRDY	[1]	R	Transmitter ready Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the TXRDY interrupt.	0
RXBRK	[2]	R	Receiver break Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the RXBRK interrupt.	0
OVRE	[5]	R	Overrun error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the OVRE interrupt.	0
FRAME	[6]	R	Framing error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the FRAME interrupt.	0
PARE	[7]	R	Parity error Raw Interrupt State. Gives the raw interrupt state (prior to masking) of the PARE interrupt.	0
TIMEOUT	[8]	R	Time-out Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the TIMEOUT interrupt.	0
TXEMPTY	[9]	R	Transmitter empty Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the TXEMPTY interrupt.	0
IDLE	[10]	R	IDLE Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the IDLE interrupt.	0
ENDHEADER	[24]	R	Ended header Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the ENDHEADER interrupt.	0

Name	Bit	Type	Description	Reset Value
ENDMESS	[25]	R	Ended message Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the ENDMESS interrupt.	0
NOTREPS	[26]	R	Not responding error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the NOTREPS interrupt	0
BITERROR	[27]	R	Bit error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the BITERROR interrupt.	0
IPERROR	[28]	R	Identity parity error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the IPERROR interrupt.	0
CHECKSUM	[29]	R	Checksum error Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the CHECKSUM interrupt.	0
WAKEUP	[30]	R	Wake up Raw Interrupt State. Gives the raw interrupt state (Prior to masking) of the WAKEUP interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

24.4.1.8 US_MISR (USART Masked Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD	WAKEUP	CHECKSUM	IPERROR	BITERROR	NOTRESP	ENDMESS	ENDHEADER	RSVD													IDLE	TXEMPTY	TIMEOUT	PARE	FRAME	OVRE	RSVD		RXBRK	TXRDY	RXRDY
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
RXRDY	[0]	R	Receiver ready Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the RXRDY interrupt.	0
TXRDY	[1]	R	Transmitter ready Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the TXRDY interrupt.	0
RXBRK	[2]	R	Receiver break Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the RXBRK interrupt.	0
OVRE	[5]	R	Overrun error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the OVRE interrupt.	0
FRAME	[6]	R	Framing error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the FRAME interrupt.	0
PARE	[7]	R	Parity error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the PARE interrupt.	0
TIMEOUT	[8]	R	Time-out Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the TIMEOUT interrupt.	0
TXEMPTY	[9]	R	Transmitter empty Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the TXEMPTY interrupt.	0
IDLE	[10]	R	IDLE Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the IDLE interrupt.	0
ENDHEADER	[24]	R	Ended header Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the ENDHEADER interrupt.	0

Name	Bit	Type	Description	Reset Value
ENDMESS	[25]	R	Ended message Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the ENDMESS interrupt.	0
NOTREPS	[26]	R	Not responding error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the NOTREPS interrupt.	0
BITERROR	[27]	R	Bit error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the BITERROR interrupt.	0
IPERROR	[28]	R	Identity parity error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the IPERROR interrupt.	0
CHECKSUM	[29]	R	Checksum error Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the CHECKSUM interrupt.	0
WAKEUP	[30]	R	Wake up Masked Interrupt State. Gives the masked interrupt state (Prior to masking) of the WAKEUP interrupt.	0

On a read this register gives the current masked status value of the corresponding interrupt. A write has no effect.

US_MISR = US_IMSCR & US_RISR

24.4.1.9 US_ICR (USART Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD	WAKEUP	CHECKSUM	IPERROR	BITERROR	NOTRESP	ENDMESS	ENDHEADER	RSVD													IDLE	TXEMPTY	TIMEOUT	PARE	FRAME	OVRE	RSVD		RXBRK	RSVD	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	W	W	W	W	W	W	W	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	R	R	W	R	R

Name	Bit	Type	Description	Reset Value
RXBRK	[2]	W	Receiver break Masked Interrupt State. 0 = No effect. 1 = Clears the RXBRK interrupt.	0
OVRE	[5]	W	Overrun error Masked Interrupt State. 0 = No effect. 1 = Clears the OVRE interrupt.	0
FRAME	[6]	W	Framing error Masked Interrupt State. 0 = No effect. 1 = Clears the FRAME interrupt.	0
PARE	[7]	W	Parity error Masked Interrupt State. 0 = No effect. 1 = Clears the PARE interrupt.	0
TIMEOUT	[8]	W	Time-out Masked Interrupt State. 0 = No effect. 1 = Clears the TIMEOUT interrupt.	0
TXEMPTY	[9]	W	Transmitter empty Masked Interrupt State. 0 = No effect. 1 = Clears the TXEMPTY interrupt.	0
IDLE	[10]	W	IDLE Masked Interrupt State. 0 = No effect. 1 = Clears the IDLE interrupt.	0
ENDHEADER	[24]	W	Ended header Masked Interrupt State. 0 = No effect. 1 = Clears the ENDHEADER interrupt.	0
ENDMESS	[25]	W	Ended message Masked Interrupt State. 0 = No effect. 1 = Clears the ENDMESS interrupt.	0
NOTREPS	[26]	W	Not responding error Masked Interrupt State. 0 = No effect. 1 = Clears the NOTREPS interrupt.	0

Name	Bit	Type	Description	Reset Value
BITERROR	[27]	W	Bit error Masked Interrupt State. 0 = No effect. 1 = Clears the BITERROR interrupt.	0
IPERROR	[28]	W	Identity parity error Masked Interrupt State. 0 = No effect. 1 = Clears the IPERROR interrupt.	0
CHECKSUM	[29]	W	Checksum error Masked Interrupt State. 0 = No effect. 1 = Clears the CHECKSUM interrupt.	0
WAKEUP	[30]	W	Wake up Masked Interrupt State. 0 = No effect. 1 = Clears the WAKEUP interrupt.	0

O n a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

24.4.1.10 US_SR (USART Status Register)

- Address = Base Address + 0x0024, Reset Value = 0x0000_0800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LINBUSY	WAKEUP	CHECKSU	IPERROR	BITERROR	NOTRESP	ENDMESS	ENDHEADER	RSVD												IDLEFLAG	IDLE	TXEMPTY	TIMEOUT	PARE	FRAME	OVRE	RSVD		RXBRK	TXRDY	RXRDY
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
RXRDY	[0]	R	Receiver Ready. 0 = No complete character has been received since the last read of the US_RHR or the receiver is disabled. 1 = At least one complete character has been received and the US_RHR has not yet been read.	0
TXRDY	[1]	R	Transmitter Ready. 0 = A character is in US_THR waiting to be transferred to the Transmit Shift Register or the transmitter is disabled. 1 = There is no character in the US_THR. Equal to zero when the USART is disabled or at reset. Transmitter Enable command (in US_CR) sets this bit to one.	0
RXBRK	[2]	R	Receiver Break. 0 = "No Break Received" has been detected since the last "Reset Status Bits" command in the Control Register. 1 = "Break Received" has been detected since the last "Reset Status Bits" command in the Control Register.	0
OVRE	[5]	R	Overrun Error. 0 = No byte has been transferred from the Receive Shift Register to the US_RHR when RxRDY was asserted since the last "Reset Status Bits" command. 1 = At least one byte has been transferred from the Receive Shift Register to the US_RHR when RxRDY was asserted since the last "Reset Status Bits" command.	0
FRAME	[6]	R	Framing Error. 0 = No stop bit has been detected low since the last "Reset Status Bits" command. 1 = At least one stop bit has been detected low since the last "Reset Status Bits" command.	0
PARE	[7]	R	Parity Error. 0 = No parity bit has been detected false (or a parity bit high in multi-drop mode) since the last "Reset Status Bits"	0

Name	Bit	Type	Description	Reset Value
			command. 1 = At least one parity bit has been detected false (or a parity bit high in multi-drop mode) since the last "Reset Status Bits" command.	
TIMEOUT	[8]	R	Time-Out. 0 = There has not been a time-out since the last "Start Time-out" command or the Time-out Register is 0. 1 = There has been a time-out since the last "Start Time-out" command.	0
TXEMPTY	[9]	R	Transmitter Empty. 0 = There are characters in either US_THR or the Transmit Shift Register. 1 = There are no characters in both US_THR and the Transmit Shift Register. Equal to zero when the USART is disabled or after reset. Transmitter Enable command (in US_CR) sets this bit to one.	0
IDLE	[10]	R	IDLE. 0 = No end of J1587 protocol frame. 1 = An end of J1587 protocol frame occurred.	0
IDLEFLAG	[11]	R	0 = A frame is being received by the USART. 1 = No frame is being received by the USART. This bit indicates a frame transmission in J1587 protocol. It's turn low when a reception start and turn high when a reception is followed by at least 10 stop bits (10 bits at high level).	1
ENDHEADER	[24]	R	Ended Header. 0 = No end of Header has occurred on a LIN Frame. 1 = An end of Header has occurred on a LIN Frame.	0
ENDMESS	[25]	R	Ended Message. 0 = No end of Message occurred on a LIN Frame. 1 = An end of Message has occurred on a LIN Frame.	0
NOTREPS	[26]	R	Not Responding Error. 0 = No Slave-not-responding-error has been detected LIN Frame. 1 = A Slave-not-responding-error has been detected LIN Frame.	0
BITERROR	[27]	R	Bit Error. 0 = No Bit-error has been detected on a LIN Frame. 1 = A Bit-error has been detected on a LIN Frame.	0
IPERROR	[28]	R	Identity Parity Error. 0 = No Identity-Parity-Error has been detected on a LIN Frame. 1 = A Identity-Parity-Error has been detected on a LIN Frame.	0

Name	Bit	Type	Description	Reset Value
CHECKSUM	[29]	R	Checksum Error. 0 = No Checksum-error has been detected LIN Frame. 1 = A Checksum-error has been detected LIN Frame.	0
WAKEUP	[30]	R	Wake Up. 0 = No Wakeup has been detected. 1 = A Wakeup has been detected.	0
LIN	[31]	R	0 = LIN is in IDLE. 1 = LIN is BUSY.	0

24.4.1.11 US_RHR (USART Receiver Holding Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_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
RSVD																							RXCHR								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
RXCHR	[8:0]	R	Received character. Last character received if RXRDY is set. The value is maintained in the register until a new byte is received. When number of data bits is less than 9 bits, the bits are right aligned.	0x000

Caution: When reading this register, the RXRDY bit is clear in the US_RHR register.
During the debug mode, users should use the ghost registers to avoid clearing RXRDY bit.

24.4.1.12 US_THR (USART Transmit Holding Register)

- Address = Base Address + 0x002C, Reset Value = 0x0000_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
RSVD																							TXCHR								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
TXCHR	[8:0]	W	Character to be transmitted. If TXRDY is at a logical 1, this will be the next character to be transmitted (After current one if already a character is present in the transmit shift register). If TXRDY is at a logical 0, the current character in the US_THR register will be overwritten. When number of data bits is less than 9 bits, the bits are right aligned.	0x000

24.4.1.13 US_BRGR (USART Baud Rate Generator Register)

- Address = Base Address + 0x0030, Reset Value = 0x0000_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
RSVD																CD															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>CD</td><td>[15:0]</td><td>RW</td><td>Clock Divider.
This register has no effect if the synchronous mode is selected with an external clock.
• Clock Divisor Field.<table>
<tr><th>CD[15:0]</th><th>Action</th></tr>
<tr><td>0</td><td>Disables clock</td></tr>
<tr><td>1</td><td>Clock divider bypassed</td></tr>
<tr><td>2 to 65535</td><td>Baud Rate (Asynchronous Mode) = Selected clock/(16 × CD)
Baud Rate (Synchronous Mode) = Selected clock/CD</td></tr>
</table></td><td>0x0000</td></tr>
</table>

Caution: In the synchronous mode, the programmed value must be even to ensure a 50:50 mark/space ratio and the software user must enable the clock (i.e. CD is different to zero) after configuring the Baud Rate clock via the US_MR register.
CD = 1 must not be used when internal clock (PCLK) is selected (i.e USCLKS[1:0] = 0).

24.4.1.14 US_RTOR (USART Receiver Time-Out Register)

- Address = Base Address + 0x0034, Reset Value = 0x0000_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
RSVD																TO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>TO</td><td>[15:0]</td><td>RW</td><td>Time-out value.
When a value is written to this register, a time-out start command is automatically performed.
• Time-Out Configuration Field
<table>
<tr><th>TO[15:0]</th><th>Action</th></tr>
<tr><td>0</td><td>Disables the receiver time-out function.</td></tr>
<tr><td>1–65565</td><td>The time-out counter is loaded with TO[15:0] when the time-out start command is given or when each new data character is received (after reception has started).</td></tr>
</table>
In asynchronous mode:
Time-out duration = TO[15:0] × Bit period.
In synchronous mode:
Time-out duration = TO[15:0] × 16 × Bit period.</td><td>0x0000</td></tr>
</table>

Caution: When the receiver is disabled by setting the RXDIS bit in the US_CR register, the time-out is stopped. If the receiver is re-enabled by setting the RXEN bit in the US_CR register, the timeout restarts where it left off (i.e. it is not reset).

24.4.1.15 US_TTGR (USART Transmit Time-Guard Register)

- Address = Base Address + 0x0038, Reset Value = 0x0000_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
RSVD																								TG							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>TG</td><td>[7:0]</td><td>RW</td><td>Time-Guard Value.
• Time-Guard Configuration Field.
<table>
<tr><th>TO[15:0]</th><th>Action</th></tr>
<tr><td>0</td><td>Disables the transmitter time-guard function.</td></tr>
<tr><td>1–255</td><td>TXD is inactive high after the transmission of each character for the time-guard duration.</td></tr>
</table>
Time-guard duration = TG[7:0] × Bit period.</td><td>0x00</td></tr>
</table>

24.4.1.16 US_LIR (USART Transmit Time-Guard Register)

- Address = Base Address + 0x0040, Reset Value = 0x3AD4_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
RSVD		WAKE_UP_TIME														RSVD						CHK_SEL	NDATA			IDENTIFIER					
0	0	1	1	1	0	1	0	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R	R	R	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
IDENTIFIER	[5:0]	RW	LIN Identifier. Indicates the LIN's identifier content to be transmitted on the next "HEADER MESSAGE". For the LIN1.2 release, the IDENTIFIER[5:4] field specifies the number of data field. • Number of Data Field for the LIN1.2 Release. IDENTIFIER[5:4] : Number of Data Field 0 : 2 1 : 2 2 : 4 3 : 8	0x3AD4
NDATA	[8:6]	RW	Number of data field for the LIN2.0 release. Specifies the number of data field to be sent or received. The range is from 0 up to 7 which corresponds to 1 up to 8 data field.	0x0
CHK_SEL	[9]	RW	Checksum Selection. 0 = Classic checksum. 1 = Enhanced checksum. (compliant to LIN2.0 release)	0
WAKE_UP _TIME	[29:16]	RW	Wake up time for the LIN2.0 release. Sets the wake up time counter. After reset, the counter is loaded with the 0 x 3AD4 value which corresponds at least 753us for PCLK = 20MHz.	0x0000

Nested table in the IDENTIFIER description:

IDENTIFIER[5:4]	Number of Data Field
0	2
1	2
2	4
3	8

Caution: It is not possible to write on this register during a Message Transfer.

24.4.1.17 US_DFWR0 (USART Data Field Write 0 Register)

- Address = Base Address + 0x0040, Reset Value = 0x0000_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
DATA3								DATA2								DATA1								DATA0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA0 DATA1 DATA2 DATA3	[7:0] [15:8] [23:16] [31:24]	RW	LIN's byte field to be transmitted. Data to be transmitted on the "LIN's RESPONSE MESSAGE" when the STRESP will be set on the Controller Register.	0x00 0x00 0x00 0x00

Caution: It is not possible to write on this register during a Message Transfer.

24.4.1.18 US_DFWR1 (USART Data Field Write 1 Register)

- Address = Base Address + 0x0044, Reset Value = 0x0000_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
DATA7								DATA6								DATA5								DATA4							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
DATA4 DATA5 DATA6 DATA7	[7:0] [15:8] [23:16] [31:24]	RW	LIN's byte field to be transmitted. Data to be transmitted on the "LIN RESPONSE MESSAGE" when the STRESP will be set on the Controller Register.	0x00 0x00 0x00 0x00

Caution: It is not possible to write on this register during a Message Transfer.

24.4.1.19 US_DFRR0 (USART Data Field Read 0 Register)

- Address = Base Address + 0x0048, Reset Value = 0x0000_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
DATA3								DATA2								DATA1								DATA0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DATA0 DATA1 DATA2 DATA3	[7:0] [15:8] [23:16] [31:24]	R	LIN's byte field to be received. Data received in the "LIN RESPONSE MESSAGE" when the ENDMESS is set in the US_SR register.	0x00 0x00 0x00 0x00

24.4.1.20 US_DFRR1 (USART Data Field Read 1 Register)

- Address = Base Address + 0x004C, Reset Value = 0x0000_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
DATA7								DATA6								DATA5								DATA4							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
DATA4 DATA5 DATA6 DATA7	[7:0] [15:8] [23:16] [31:24]	R	LIN's byte field to be received. Data received in the "LIN RESPONSE MESSAGE" when the ENDMESS is set in the US_SR register.	0x00 0x00 0x00 0x00

24.4.1.21 US_SBLR (USART Synchronous Break Length Register)

- Address = Base Address + 0x0050, Reset Value = 0x0000_000D

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

Name	Bit	Type	Description	Reset Value
SYNC_BRK	[4:0]	RW	Synchronous Break Length. Values from 0 to 7 cannot be written in the register. (Register keeps pervious value)	0x0D

Caution: It is not possible to write on this register during a message transfer.

24.4.1.22 US_LCP1 (USART Synchronous Break Length Register 1)

- Address = Base Address + 0x0054, Reset Value = 0x7768_5B4C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCP3								LCP2								LCP1								LCP0							
0	1	1	1	0	1	1	1	0	1	1	0	1	0	0	0	0	1	0	0	1	0	1	1	0	1	0	0	1	1	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
LCP0 LCP1 LCP2 LCP3	[7:0] [15:8] [23:16] [31:24]	RW	Limit Counter Protocol. x is the corresponding number of byte defined (NDATA) in the US_LIR register. This parameters allows to define the time-out counter for the "not responding error" flag following the calculation of the time-out limit. For the LIN1.2 or LIN2.1 release, the time-out limits is defined as follows: limit_cpt = LCPndata + SYNC_BRK – 13.	0x77 0x68 0x5B 0x4C

Caution: It is not possible to write on this register during a message transfer.

24.4.1.23 US_LCP2 (USART Synchronous Break Length Register 2)

- Address = Base Address + 0x0058, Reset Value = 0XAFA0_9284

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCP7								LCP6								LCP5								LCP4							
1	0	1	0	1	1	1	1	1	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W

Name	Bit	Type	Description	Reset Value
LCP4 LCP5 LCP6 LCP7	[7:0] [15:8] [23:16] [31:24]	RW	Limit Counter Protocol. x is the corresponding number of byte defined (NDATA) in the US_LIR register. This parameters allows to define the time-out counter for the "not responding error" flag following the calculation of the time-out limit. For the LIN1.2 or LIN2.1 release, the time-out limits is defined as follows: limit_cpt = LCPndata + SYNC_BRK – 13.	0xAF 0xA0 0x92 0x84

Caution: It is not possible to write on this register during a message transfer.

24.4.1.24 US_DMACR (USART DMA Control Register)

- Address = Base Address + 0x0058, Reset Value = 0X0000_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
RSVD																														TXDMAE	RXDMAE
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
RXDMAE	[0]	RW	DMA for the receive Enable/ Disable Control Bit. 0 = Disable. 1 = Enable.	0
TXDMAE	[1]	RW	DMA for the transmit Enable/ Disable Control Bit. 0 = Disable. 1 = Enable .	0

24.5 4MHz–40MHz Asynchronous Mode (SYNC = 0)

Table 24-4 Asynchronous Mode (SYNC = 0)

SYSCLK	PDIV	PCLK	US_BRGR CD[15:0]	Baud Rate	Result	% Error
4MHz	1	4	208	1200	1201.92	–0.16%
			104	2400	2403.85	–0.16%
			52	4800	4807.69	–0.16%
			26	9600	9615.38	–0.16%
			13	19200	19230.77	–0.16%
	2	2	104	1200	1201.92	–0.16%
			52	2400	2403.85	–0.16%
			26	4800	4807.69	–0.16%
			13	9600	9615.38	–0.16%
	4	1	52	1200	1201.92	–0.16%
			26	2400	2403.85	–0.16%
			13	4800	4807.69	–0.16%
	8	0.5	26	1200	1201.92	–0.16%
			13	2400	2403.85	–0.16%
	16	0.25	13	1200	1201.92	–0.16%
8MHz	1	8	417	1200	1199.04	0.08%
			208	2400	2403.85	–0.16%
			104	4800	4807.69	–0.16%
			52	9600	9615.38	–0.16%
			26	19200	19230.77	–0.16%
			13	38400	38461.54	–0.16%
	2	4	208	1200	1201.92	–0.16%
			104	2400	2403.85	–0.16%
			52	4800	4807.69	–0.16%
			26	9600	9615.38	–0.16%
			13	19200	19230.77	–0.16%
	4	2	104	1200	1201.92	–0.16%
			52	2400	2403.85	–0.16%
			26	4800	4807.69	–0.16%
			13	9600	9615.38	–0.16%
	8	1	52	1200	1201.92	–0.16%
			26	2400	2403.85	–0.16%
			13	4800	4807.69	–0.16%
	16	0.5	26	1200	1201.92	–0.16%

SYSCLK	PDIV	PCLK	US_BRGR CD[15:0]	Baud Rate	Result	% Error
			13	2400	2403.85	–0.16%
16MHz	1	16	833	1200	1200.48	–0.04%
			417	2400	2398.08	0.08%
			208	4800	4807.69	–0.16%
			104	9600	9615.38	–0.16%
			52	19200	19230.77	–0.16%
			26	38400	38461.54	–0.16%
	2	8	417	1200	1199.04	0.08%
			208	2400	2403.85	–0.16%
			104	4800	4807.69	–0.16%
			52	9600	9615.38	–0.16%
			26	19200	19230.77	–0.16%
			13	38400	38461.54	–0.16%
	4	4	208	1200	1201.92	–0.16%
			104	2400	2403.85	–0.16%
			52	4800	4807.69	–0.16%
			26	9600	9615.38	–0.16%
			13	19200	19230.77	–0.16%
	8	2	104	1200	1201.92	–0.16%
			52	2400	2403.85	–0.16%
			26	4800	4807.69	–0.16%
			13	9600	9615.38	–0.16%
	16	1	52	1200	1201.92	–0.16%
			26	2400	2403.85	–0.16%
			13	4800	4807.69	–0.16%
20MHz	1	20	1042	1200	1199.62	0.03%
			521	2400	2399.23	0.03%
			260	4800	4807.69	–0.16%
			130	9600	9615.38	–0.16%
			87	14400	14367.82	0.22%
			65	19200	19230.77	–0.16%
	2	10	521	1200	1199.62	0.03%
			260	2400	2403.85	–0.16%
			130	4800	4807.69	–0.16%
			65	9600	9615.38	–0.16%
	4	5	260	1200	1201.92	–0.16%

SYSCLK	PDIV	PCLK	US_BRGR CD[15:0]	Baud Rate	Result	% Error
			130	2400	2403.85	–0.16%
			65	4800	4807.69	–0.16%
	8	2.5	130	1200	1201.92	–0.16%
			65	2400	2403.85	–0.16%
	16	1.25	65	1200	1201.92	–0.16%
40MHz	1	40	2083	1200	1200.19	–0.02%
			1042	2400	2399.23	0.03%
			521	4800	4798.46	0.03%
			260	9600	9615.38	–0.16%
			174	14400	14367.82	0.22%
			130	19200	19230.77	–0.16%
			65	38400	38461.54	–0.16%
	2	20	1042	1200	1199.62	0.03%
			521	2400	2399.23	0.03%
			260	4800	4807.69	–0.16%
			130	9600	9615.38	–0.16%
			87	14400	14367.82	0.22%
			65	19200	19230.77	–0.16%
	4	10	521	1200	1199.62	0.03%
			260	2400	2403.85	–0.16%
			130	4800	4807.69	–0.16%
			65	9600	9615.38	–0.16%
	8	5	260	1200	1201.92	–0.16%
			130	2400	2403.85	–0.16%
			65	4800	4807.69	–0.16%
	16	2.5	130	1200	1201.92	–0.16%
			65	2400	2403.85	–0.16%

25 Watchdog Timer

25.1 Overview

The watchdog timer is used to prevent the system from locking-up (for example in infinite software loops). If the software does not write to the watchdog during the programmed time, then it can generate an interrupt (WDTOVF) or an internal reset.

25.1.1 Feature

The watchdog timer has a programmable 16-bit down counter.

The software can control the action to perform when the WDT counter overflows (i.e. reaches 0):

- If the RSTEN bit is set in the WDT_OMR register, an internal reset is generated.
- If the WDTOVF bit is set in the WDT_IMSCR register, an interrupt is generated on the Interrupt Controller.

The input frequency clock (EMCLK, ESCLK, IMCLK, ISCLK, or PLLCLK) from the clock manager supplies the watchdog counter through the programmable divider WDTPDIV.

There is a possibility to set a programmable pending window where users can restart the watchdog counter only within this window + 1. This protection is set with the RSTALW bit, otherwise users can restart the watchdog counter whenever. When the pending window is reached, the WDTPEND bit is set followed by the PENDING bit.

The supplied clock (FIN) is divided by the WDTPDIV[2:0] divider and provided to the down-counter input WDTCLK.

All write accesses are protected by control access keys to help prevent corruption of the watchdog.

To update the contents of the mode and control registers, it is necessary to write the correct bit pattern to the control access key bits at the same time as the control bits are written (the same write access).

25.2 Functional Description

25.2.1 Block Diagram

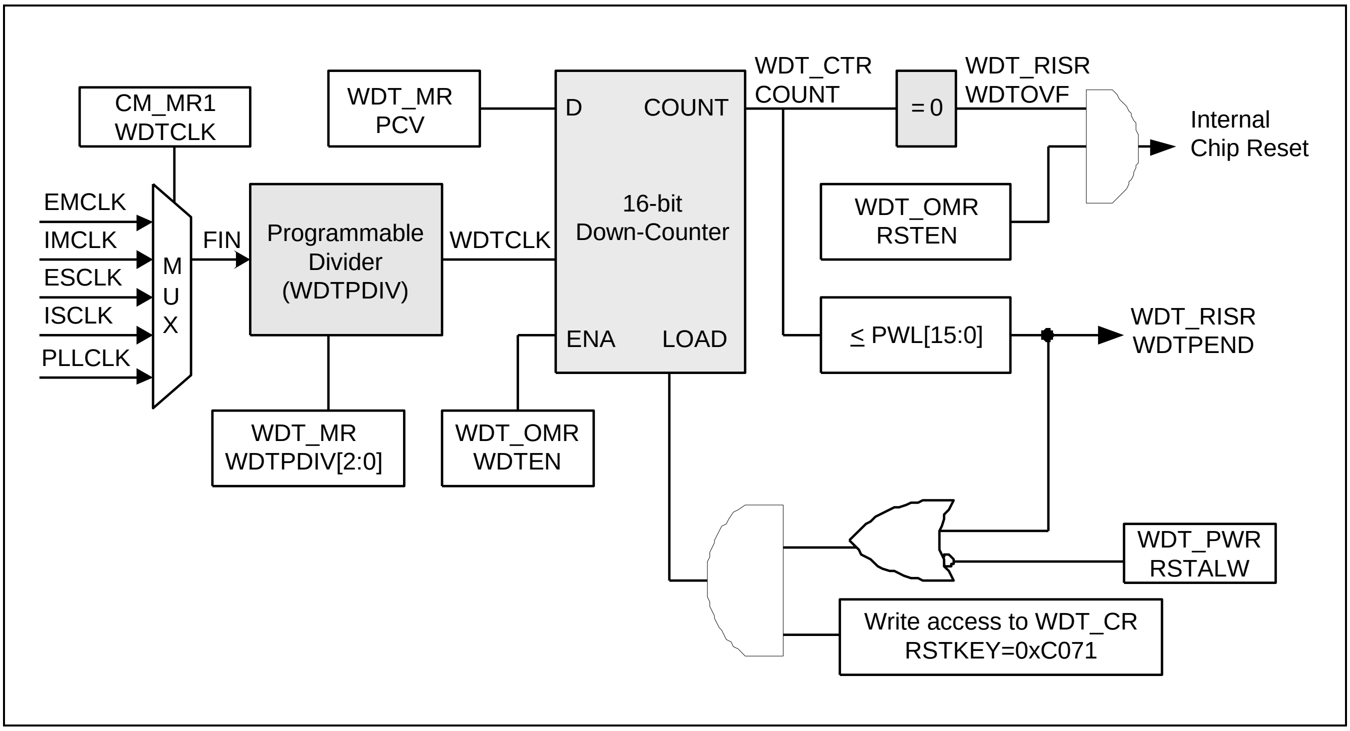


Figure 25-1 Watchdog Timer Block Diagram

25.2.2 Watchdog Timer Functionality

25.2.2.1 General Description

The watchdog timer contains a programmable length down-counter. The down-counter input clock is a subdivision of the input frequency clock (EMCLK, IMCLK, ESCLK, ISCLK or PLLCLK) from the clock manager:

NOTE: The count value for overflow should be greater than (3 PCLK + 5 FIN).

- WDTCLK = FIN/WDTPDIV[2:0]

WDTPDIV[2:0]			WDTCLK
0	0	0	FIN/2
0	0	1	FIN/4
0	1	0	FIN/8
0	1	1	FIN/16
1	0	0	FIN/64
1	0	1	FIN/128
1	1	0	FIN/256
1	1	1	FIN/512

The count length determines the timeout period, and is controlled by loading PCV field of WDT_MR register.
The time out period (in seconds) is:

- (PCV[15:0] + 1)/WDCLK freq.

When the counter reaches the value programmed in the pending windows PWL[15:0] of WDT_PWR register, the watchdog can generate a watchdog pending interrupt. The pending interrupt occurs after:

- {(PCV[15:0]) – (PWL[15:0])}/WDCLK freq.

If PWL[15:0] is greater than PCV[15:0](the previous time is negative), the WDT pending interrupt should not be used.

In order to prevent an internal chip reset (if RSTEN bit is set in the WDT_OMR) or interrupt (if bit WDTOVF is set in the WDT_IMSCR), the software must reset the counter before it reaches 0 by writing the correct key in the WDT_CR register (0xC071). The time (in seconds) between the WDT pending interrupt and the WDT overflow is:

- (PWL[15:0])/WDTCLK freq.

When the counter reaches 0, it triggers the programmed action (internal chip reset or overflow interrupt).

If no WDT reset is programmed (i.e. RSTEN is at a logical 0) when the WDT reaches 0, it is reset to the programmed value and continues to down count, unless it is disabled. This is to be used to generate periodic interrupts.

25.2.3 Watchdog Timer Events

25.2.3.1 Internal Chip Reset Pulse Generation

If the RSTEN bit is set in the WDT_OMR register, an internal system (chip) reset pulse is generated when the overflow occurs.

After a reset, the clock selected by the watchdog timer is FIN/512.

25.2.3.2 Internal Interrupt Request

The watchdog can generate an internal interrupt request when the overflow occurs. The software can enable or disable this interrupt either in the WDT module (WDT_IMSCR register).

25.2.4 Example

Example of use of the watchdog timer:

Use of the windows to generate an interrupt and reload the watchdog counter within the windows only.
If the interruption is not called due to a bug, a reset occurs when the watchdog counter reaches 0.

25.2.4.1 Configuration

- Configuration of WDT_MR: Choice of the clock to decrease counter, preload value from which counter starts to decrease.
- Configuration of WDT_ PWR: Upper limit of the window from which it generates an interrupt when reached and the bit which allows to restart counter only within this window.
- Configuration of WDT_IMSCR: Enable Interrupt at the peripheral level when the window is reached (bit WDTPEND) or when the counter overflow (bit WDTOVF if watchdog reset is not enabled), interrupt mask must be configured.
- Configuration of WDT_OMR: Enable the watchdog (start decrement the counter) and enable the watchdog reset if counter overflow.

25.2.4.2 Interrupt Handing

- IRQ Entry and call C function.
- Read WDT_MISR and verify the source of the interrupt.
- Clear the corresponding interrupt at peripheral level by writing in the WDT_ICR.
- Interrupt treatment: If this is a windows pending interrupt, restart the watchdog by writing in WDT_CR.
- Exit IRQ.

25.3 Register Description

25.3.1 Register Map Summary

- Base Address: 0x4003_0000

Register	Offset	Description	Reset Value
WDT_IDR	0x0000	Watchdog ID Register	0x0001_0000
WDT_CR	0x0004	Watchdog Control Register	0x0000_0000
WDT_MR	0x0008	Watchdog Mode Register	0x00FF_FF07
WDT_OMR	0x000C	Watchdog Overflow Mode Register	0x0000_0003
WDT_SR	0x0010	Watchdog Status Register	0x8000_0100
WDT_IMSCR	0x0014	Watchdog Interrupt Mask Set /Clear Register	0x0000_0000
WDT_RISR	0x0018	Watchdog Raw Interrupt Status Register	0x0000_0001
WDT_MISR	0x001C	Watchdog Masked Interrupt Status Register	0x0000_0000
WDT_ICR	0x0020	Watchdog Interrupt Clear Register	0x0000_0000
WDT_PWR	0x0024	Watchdog Pending Window Register	0x00FF_FF00
WDT_CTR	0x0028	Watchdog Counter Test Register	0x0000_FFFF

25.3.1.1 WDT_IDR (Watchdog Timer ID Register)

- Address = Base Address + 0x0000, Reset Value = 0x0001_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
RSVD						IDCODE																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
IDCODE	[25:0]	R	ID Code Register. This field stores the ID code for the corresponding IP.	0x0001_0000

25.3.1.2 WDT_CR (Watchdog Timer Control Register)

- Address = Base Address + 0x0004, Reset Value = 0x0000_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
DBGEN	RSVD															RSTKEY															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Name	Bit	Type	Description	Reset Value
RSTKEY	[15:0]	W	Restart Key Field. 0xC071 = Watchdog counter is restarted if its value is equal or less than the length of a pending window or if pending window is disabled. Other values = No effect. **NOTE:** A restart command (write the restart key in WDT_CR) will not be effective, if it occurs less than (2 WDTCLK+1/2 FIN) periods after a previous start command.	0x0000
DBGEN	[31]	W	Debug Enable/Disable Control Bit. 0 = The debug mode is disabled. The WDT counter keeps running when a debug is request. 1 = The debug mode is enabled. The WDT counter stops running when a debug is request.	0

25.3.1.3 WDT_MR (Watchdog Timer Mode Register)

- Address = Base Address + 0x0008, Reset Value = 0x00FF_FF07

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

<table>
<tr><th>Name</th><th>Bit</th><th>Type</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>WDTPDIV</td><td>[2:0]</td><td>RW</td><td>WDT Clock Divider Field.
This field determines the clock frequency of watchdog timer.
<table>
<tr><th colspan="3">WDPDIV[2:0]</th><th>WDCLK</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>FIN/2</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>FIN/4</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>FIN/8</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>FIN/16</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>FIN/64</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>FIN/128</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>FIN/256</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>FIN/512</td></tr>
</table>
<b>NOTE:</b> FIN is an input clock asserted into a watchdog timer, supplied by clock manager.</td><td>111'b</td></tr>
<tr><td>PCV</td><td>[23:8]</td><td>RW</td><td>Preload Counter Value.
Counter is preloaded when watchdog counter is restarted.
Time to generate overflow = PCV[15:0]/WDTCLK freq.</td><td>0xFFFF</td></tr>
<tr><td>CKEY</td><td>[31:24]</td><td>W</td><td>Clock Access Key Field.
Used only when writing in WDT_MR. CKEY is read as 0.
Write access in WDT_MR is allowed only if CKEY[7:0] = 0x37.</td><td>0x00</td></tr>
</table>

25.3.1.4 WDT_OMR (Watchdog Timer Overflow Mode Register)

- Address = Base Address + 0x000C, Reset Value = 0x0000_0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																OKEY												RSVD	LOCKRSTEN	RSTEN	WDTEN
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W	W	W	W	W	R	R W	R W	R W

Name	Bit	Type	Description	Reset Value
WDTEN	[0]	RW	Watchdog Enable/Disable Control Bit. 0 = Disable Watchdog Timer 1 = Enable Watchdog Timer	1
RSTEN	[1]	RW	System (Chip) Reset Enable/Disable Control Bit. 0 = Disable the generation of a system (chip) reset by the Watchdog. 1 = When overflow occurs, the Watchdog generates the system (chip) reset.	1
LOCKRSTEN	[2]	RW	CPU Lock-up Reset Enable/Disable Control Bit. 0 = Disable the generation of an internal reset on CPU Lock-up. 1 = Enable the generation of an internal reset on CPU Lock-up. Cortex-M3 supports the flag of CPU Lock-up status and reset signal. In case of malfunction, this can be used.	0
OKEY	[15:4]	W	Overflow Access Key Field. Used only when writing WDT_OMR. OKEY is read as 0. 0x234 = Write access in WDT_OMR is allowed. Other values = Write access in WDT_OMR is prohibited.	0x000

25.3.1.5 WDT_SR (Watchdog Timer Status Register)

- Address = Base Address + 0x0010, Reset Value = 0x8000_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
DBGEN	RSVD																					CLEAR_STATUS	PENDING	RSVD							
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
PENDING	[8]	R	Watchdog Pending status. 0 = Watchdog counter is over pending window length. 1 = Watchdog counter is equal or less than pending window length.	1
CLEAR_ STATUS	[9]	R	Clear Status. 0 = Finish Watchdog Counter Reset. 1 = Watchdog Counter Reset operation starts and doesn't end.	0
DBGEN	[31]	R	DBGEN Status. 0 = Disabled debug mode 1 = Enabled debug mode	1

25.3.1.6 WDT_IMSCR (Watchdog Timer Interrupt Mask Set and Clear Register)

- Address = Base Address + 0x0014, Reset Value = 0x0000_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
RSVD																														WDTOVF	WDTPEND
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW

Name	Bit	Type	Description	Reset Value
WDTPEND	[0]	RW	Watchdog Timer Pending Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0
WDTOVF	[1]	RW	Watchdog Timer Overflow Interrupt Mask. 0 = This interrupt is masked. (Disable the interrupt) 1 = This interrupt is not masked. (Enable the interrupt)	0

This register is the interrupt mask set or clear register. It is a read/write register.
On a read this register gives the current value of the mask on the relevant interrupt. A write of ‘1’ to the particular bit sets the mask, enabling the interrupt to be read. A write of ‘0’ clears the corresponding mask.

25.3.1.7 WDT_RISR (Watchdog Timer Raw Interrupt Status Register)

- Address = Base Address + 0x0018, Reset Value = 0x0000_0001

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

Name	Bit	Type	Description	Reset Value
WDTPEND	[0]	R	Watchdog Pending interrupt raw state. Gives the raw interrupt state (Prior to masking) of the WDTPEND interrupt.	1'b
WDTOVF	[1]	R	Watchdog Overflow interrupt raw state. Gives the raw interrupt state (Prior to masking) of the WDTOVF interrupt.	0

On a read this register gives the current raw status value of the corresponding interrupt prior to masking.

25.3.1.8 WDT_MISR (Watchdog Timer Masked Interrupt Status Register)

- Address = Base Address + 0x001C, Reset Value = 0x0000_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
RSVD																														WDTOVF	WDTPEND
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Name	Bit	Type	Description	Reset Value
WDTPEND	[0]	R	Watchdog Pending masked interrupt Status. Gives the masked interrupt status (After masking) of the WDTPEND interrupt. 0 = The WDTPEND interrupt doesn't occur. 1 = The WDTPEND interrupt occurs.	0
WDTOVF	[1]	R	Watchdog Overflow interrupt mask. Gives the masked interrupt status (After masking) of the WDTOVF interrupt. 0 = The WDTOVF interrupt doesn't occur. 1 = The WDTOVF interrupt occurs.	0

On a read this register gives the current masked status value of the corresponding interrupt.

25.3.1.9 WDT_ICR (Watchdog Timer Interrupt Clear Register)

- Address = Base Address + 0x0020, Reset Value = 0x0000_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
RSVD																														WDTOVF	WDTPEND
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W

Name	Bit	Type	Description	Reset Value
WDTPEND	[0]	W	WDPEND: Watchdog Pending Clear. 0 = No effect. 1 = Clear Watchdog pending interrupt.	0
WDTOVF	[1]	W	WDOVF: Watchdog Overflow Clear. 0 = No effect. 1 = Clear Watchdog overflow interrupt.	0

25.3.1.10 WDT_PWR (Watchdog Pending Windows Register)

- Address = Base Address + 0x0024, Reset Value = 0x00FF_FF00

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

Name	Bit	Type	Description	Reset Value
RSTALW	[0]	RW	Restart allowed. This bit does not disable the bit WDTPEND in the interrupt register. 0 = Restart allowed every time. 1 = Restart allowed only within (pending window +1).	0
PWL	[23:8]	RW	Pending Window Length. Length of the window. The time from preload value to watchdog timer pending is: (PCV[15:0] – PWL[15:0])/WDTCLK freq.	0xFFFF
PWKEY	[31:24]	W	Pending window access key. Used only when writing in WDT_PWR. PWKEY is read as 0. Write access in WDT_PWR is allowed only if PWKEY[7:0] = 0x91.	0x00

25.3.1.11 WDT_CTR (Watchdog Timer Counter Test Register)

- Address = Base Address + 0x0028, Reset Value = 0x0000_FFFF

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

Name	Bit	Type	Description	Reset Value
COUNT	[15:0]	R	Value of Watchdog timer counter.	0xFFFF

26 Electrical Data

26.1 Absolute Maximum Ratings

Stresses above those listed in "Absolute Maximum Ratings" may cause permanent damage to the device. The functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

Table 26-1 Absolute Maximum Ratings

Parameter	Symbol	Conditions	Rating	Unit
DC Supply Voltage for V_{DDCORE}	V_{DDCORE}	–	–0.3 to 6.0	V
DC Supply Voltage for V_{DDIO}	V_{DDIO}	–	–0.3 to 6.0	V
DC Supply Voltage for AV_{DD}	AV_{DD}	–	–0.3 to 6.0	V
DC Supply Voltage for AV_{REF}	AV_{REF}	–	–0.3 to 6.0	V
Digital I/O Input Voltage	V_{IN}	–	–0.3 to V_{DDIO}+0.3	V
Analog I/O Input Voltage	AV_{IN}	–	–0.3 to AV_{DD}+0.3	V
DC Digital Input Current	I_{IN_D}	All Input Pins	–	mA
	–	Per Pin	–	mA
DC Analog Input Current	I_{IN_A}	All Input Pins	–	mA
		Per Pin	–	mA
Output Current Low	I_{O_LOW}	All Output Pins	–	mA
	–	Per Pin	–	mA
Output Current High	I_{O_HIGH}	All Output Pins	–	mA
	–	Per Pin	–	mA
Output Voltage	V_O	All Output Pins	–0.3 to V_{DDIO}+0.3	V
Operating Temperature	T_A	–	–40 to 85	°C
Storage Temperature	T_{STG}	–	–65 to 155	°C

NOTE: The device is not guaranteed to operate properly above those listed in 'Absolute Maximum Ratings'.

26.2 Recommended Operation Conditions

The recommended operating conditions are required in order to ensure the normal operation of the semiconductor device. All of the device's electrical characteristics are warranted when the device is operated within these ranges. Always use semiconductor devices within their recommended operating condition ranges. Operation outside these ranges may adversely affect reliability and could result in device failure.

Table 26-2 Recommended Operating Conditions

Parameter	Symbol	Conditions	Rating	Unit
DC Supply Voltage for V_{DD} Core	V_{DDCORE}	–	2.7 to 5.5	V
DC Supply Voltage for I/O	V_{DDIO}	–	2.7 to 5.5	V
DC Supply Voltage for AV_{DD}	AV_{DD}	–	2.7 to 5.5	V
DC Supply Voltage for AV_{REF}	AV_{REF}	–	0.0 to 5.5	V
Operating Temperature	T_A	–	–40 to 85	°C

26.3 I/O D.C. Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V_{DD}	X_{IN} = 4MHz ~ 8MHz, PLLCLK = 75MHz	2.7	–	5.5	V
Input High Voltage	V_{IH1}	All input pins except V_{IH2}	0.8V_{DD}	–	V_{DD}	V
	V_{IH2}	X_{IN}, $XT_{IN,}$ MODE[2:0], nRESET	V_{DD} –0.3	–	V_{DD}	
Input Low Voltage	V_{IL1}	All input pins except V_{IL2}	–	–	0.2 V_{DD}	V
	V_{IL2}	X_{IN}, $XT_{IN,}$ MODE[2:0], nRESET	–	–	0.3	
Output High Voltage	V_{OH1}	I_{OH} = –1.6mA, V_{DD}=5.0V	V_{DD} –0.4	–	–	V
	V_{OH2}	I_{OH}=-20mA, 12 IMC pads, V_{DD}=5.0V (PWM[1:0] U[2:0], PWM[1:0] D[2:0])	V_{DD} –1.0	–	–	V
Output Low Voltage	V_{OL1}	I_{OL} = 1.6mA , V_{DD}=5.0V	–	–	0.4	V
	V_{OL2}	I_{OL}=20mA, 12 IMC pads, V_{DD}=5.0V (PWM[1:0] U[2:0], PWM[1:0] D[2:0])	–	–	1.0	V
Input High Leakage Current	I_{LIH1}	V_{IN}=V_{DD}, All input pins except MODE[2:0] and I_{LIH2}	–	–	1	μA
	I_{LIH2}	V_{IN}=V_{DD}, X_{IN}, XT_{IN}	–	–	20	
Input Low Leakage Current	I_{LIL1}	V_{IN}=0V, All input pins except nRESET and I_{LIL2}	–	–	–1	μA
	I_{LIL2}	V_{IN}=0V, X_{IN}, XT_{IN}	–	–	–20	
Output High Leakage Current	I_{LOH}	V_{OUT}=V_{DD}, All output pins	–	–	3	μA
Output Low Leakage Current	ILOL	V_{OUT}=0V, All output pins	–	–	–3	μA
Pull-up Resistor	R_L	V_{IN}=0V, V_{DD}=5.0V All ports	16	30	60	kΩ

NOTE: All pins are schmitt trigger type

26.4 I/O Capacitance

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
I/O Capacitance	C_{IO}	F = 1MHz, unmeasured pins are connected to V_{SS}	–	–	10	pF
Input Capacitance	C_{IN}		–	–	10	
Output Capacitance	C_{OUT}		–	–	10	

26.5 RESET Input Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Pull-up Resistor	R_{NRST}	V_{IN}=0V, V_{DD}=5.0V	16	30	60	kΩ
Input Low Width	t_{RSL}	–	0.8	1.2	2	µs

NOTE: If the width of reset pulse is greater than minimum value, the pulse is always recognized as a valid pulse.

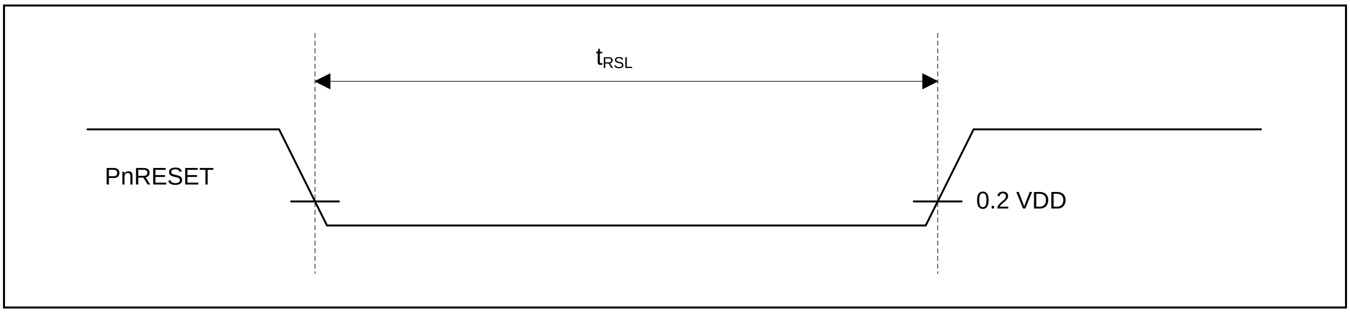


Figure 26-1 Input Timing for nRESET

26.6 External Interrupt Input Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Interrupt Input High Width	t_{INTH}	V_{DD}=5.0V	100	200	300	ns
Interrupt Input Low Width	t_{INTL}	V_{DD}=5.0V	100	200	300	ns

NOTE: If the width of interrupt pulse is greater than minimum value, the pulse is always recognized as a valid pulse.

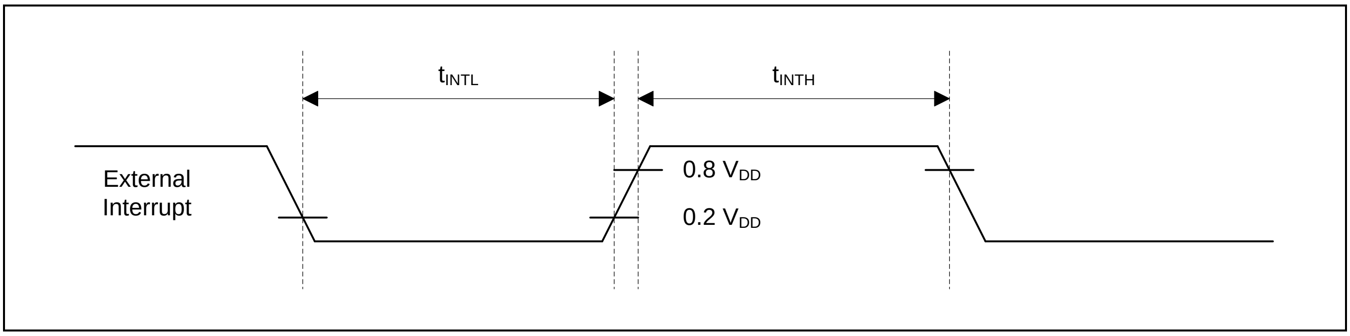
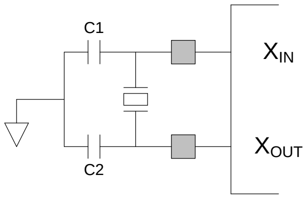
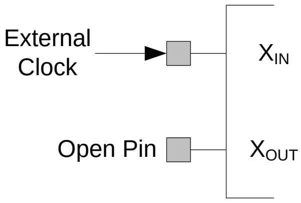


Figure 26-2 Input Timing for External Interrupt

26.7 Oscillator Characteristics

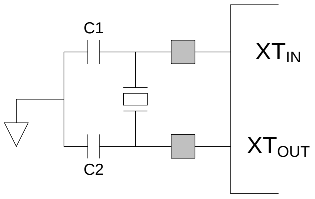
26.7.1 External Main Clock Oscillator Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Oscillator	Symbol	Conditions	Min	Typ	Max	Unit
Oscillator frequency	EMCLK	–	4	–	8	MHz
Stabilization Time	T_{STA}	–	–	–	10	ms
Crystal/Resonator/ Ceramic	EMCLK		4	–	8	MHz
External Clock	EMCLK		4	–	8	MHz

26.7.2 External Sub Clock Oscillator Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Oscillator	Symbol	Conditions	Min	Typ	Max	Unit
Oscillator frequency	ESCLK	–	–	32.768	–	KHz
Stabilization Time	T_{STA}	–	–	–	10	s
Crystal/Resonator/ Ceramic	ESCLK		–	32.768	–	KHz

26.7.3 Internal Main Clock Oscillator Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Oscillator	Symbol	Conditions	Min	Typ	Max	Unit
Oscillator frequency	IMCLK16	V_{DD}=5.0V	–	8/16	–	MHz
	IMCLK20		–	20	–	MHz
Output clock duty ratio	–	V_{DD}=5.0V	40	–	60	%
Accuracy	–	V_{DD}=5.0V, T_A = 25'C	–	–	±1	%
	–	V_{DD}=2.7~5.5V, T_A = 0~70'C	–	–	±2	%
	–	V_{DD}=5.0V, T_A = –40~85'C	–	–	±3	%
	–	V_{DD}=2.7~5.5V, T_A = –40~85'C	–	–	±5	%
Stabilization Time	T_{STA}	–	–	–	2	ms

26.7.4 Internal Sub Clock Oscillator Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Oscillator	Symbol	Conditions	Min	Typ	Max	Unit
Oscillator frequency	ISCLK	V_{DD}=5.0V	–	32.768	–	KHz
Output clock duty ratio	–	V_{DD}=5.0V	40	–	60	%
Accuracy	–	T_A = –40 to 85 °C	–	–	±50	%
Stabilization Time	T_{STA}	–	–	–	500	µs

26.7.5 PLL Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input frequency	F_{IN}	–	4	–	8	MHz
Output frequency	F_{OUT}		8	–	75	MHz
Clock Duty Ratio	T_{OD}		40	50	60	%
Locking Time	T_{LT}		–	–	200	µs

26.8 Current Consumption

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 5.5V)

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
Supply Current	I_{DD111}	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 8MHz	Normal operating 11	–	13	26	mA
	I_{DD112} (3)	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 4MHz	Normal operating 11	–	10	20	mA
	I_{DD121} (3)	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 20MHz	Normal operating 12	–	20	40	mA
	I_{DD122} (3)	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 8MHz	Normal operating 12	–	10	20	mA
	I_{DD2}	Enable PLL Enable EMCLK, IMCLK, ESCLK, and ISCLK Run CPU by PLLCLK	High-speed operating	–	50	90	mA
	I_{DD31}	CPU stops in I_{DD11} Condition.	Normal idle 1	–	6	12	mA
	I_{DD32} (3)	CPU stops in I_{DD12} Condition.	Normal idle 2	–	6	12	mA
	I_{DD4}	CPU stops in I_{DD2} Condition.	High-speed idle	–	10	20	mA
	I_{DD51} (3)	Disable EMCLK, and IMCLK Enable ESCLK, and ISCLK Enable all peripherals Run CPU by ESCLK	Sub-operating 1	–	5	10	mA
	I_{DD52} (3)	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK Enable all peripherals	Sub-operating 2	–	5	10	mA

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
		Run CPU by ISCLK					
	I_{DD61} (3)	CPU stops in I_{DD51} Condition. LVD OFF, LCD OFF	Sub-idle 1	–	4	8	mA
	I_{DD62} (3)	CPU stops in I_{DD52} Condition. LVD OFF, LCD OFF	Sub-idle 2	–	4	8	mA
	I_{DD71}	Disable EMCLK, IMCLK, ESCLK, and ISCLK All peripherals stop. LVD OFF, LCD OFF	Stop 1	–	0.5	1	mA
	I_{DD72} (3)	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK, FRT LVD OFF, LCD OFF	Stop 2	–	0.6	1	mA

NOTE:

1. Supply Current does not include current drawn through internal pull-up resistor, LCD voltage dividing resistors, and external output current loads.
2. Above tables, the current is based on LVD-OFF condition.
3. These values are only characterization data, and not tested in the mass production.

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 3.3V) [3]

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
Supply Current	I_{DD111}	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 8MHz	Normal operating 11	–	10	20	mA
	I_{DD112}	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 4MHz	Normal operating 11	–	7	14	mA
	I_{DD121}	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 20MHz	Normal operating 12	–	17	34	mA
	I_{DD122}	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 8MHz	Normal operating 12	–	8	16	mA
	I_{DD2}	Enable PLL Enable EMCLK, IMCLK, ESCLK, and ISCLK Run CPU by PLLCLK	High-speed operating	–	45	85	mA
	I_{DD31}	CPU stops in I_{DD11} Condition.	Normal idle 1	–	3	6	mA
	I_{DD32}	CPU stops in I_{DD12} Condition.	Normal idle 2	–	3	6	mA
	I_{DD4}	CPU stops in I_{DD2} Condition.	High-speed idle	–	7	14	mA
	I_{DD51}	Disable EMCLK, and IMCLK Enable ESCLK, and ISCLK Enable all peripherals Run CPU by ESCLK	Sub-operating 1	–	3	6	mA
	I_{DD52}	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK Enable all peripherals Run CPU by ISCLK	Sub-operating 2	–	3	6	mA
	I_{DD61}	CPU stops in I_{DD51} Condition. LVD OFF, LCD OFF	Sub-idle 1	–	2	4	mA
	I_{DD62}	CPU stops in I_{DD52} Condition. LVD OFF, LCD OFF	Sub-idle 2	–	2	4	mA
	I_{DD71}	Disable EMCLK, IMCLK, ESCLK, and ISCLK All peripherals stop.	Stop 1	–	0.5	1	mA

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
		LVD OFF, LCD OFF					
	I_{DD72}	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK, FRT LVD OFF, LCD OFF	Stop 2	–	0.5	1	mA

NOTE:

1. Supply Current does not include current drawn through internal pull-up resistor, LCD voltage dividing resistors, and external output current loads.
2. Above tables, the current is based on LVD-OFF condition.
3. 3.3V data is only characteristic data. These values are not tested in the mass production

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V) (3)

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
Supply Current	I_{DD111}	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 8MHz	Normal operating 11	–	9	18	mA
	I_{DD112}	Enable EMCLK, IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by EMCLK = 4MHz	Normal operating 11	–	6	12	mA
	I_{DD121}	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 20MHz	Normal operating 12	–	17	34	mA
	I_{DD122}	Disable EMCLK Enable IMCLK, ESCLK, and ISCLK Enable all peripherals Run CPU by IMCLK = 8MHz	Normal operating 12	–	8	16	mA
	I_{DD2}	Enable PLL Enable EMCLK, IMCLK, ESCLK, and ISCLK Run CPU by PLLCLK	High-speed operating	–	44	85	mA
	I_{DD31}	CPU stops in I_{DD11} Condition.	Normal idle 1	–	2	5	mA
	I_{DD32}	CPU stops in I_{DD12} Condition.	Normal idle 2	–	2	5	mA
	I_{DD4}	CPU stops in I_{DD2} Condition.	High-speed idle	–	7	14	mA
	I_{DD51}	Disable EMCLK, and IMCLK Enable ESCLK, and ISCLK Enable all peripherals Run CPU by ESCLK	Sub-operating 1	–	2	5	mA
	I_{DD52}	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK Enable all peripherals Run CPU by ISCLK	Sub-operating 2	–	2	5	mA
	I_{DD61}	CPU stops in I_{DD51}	Sub-idle 1	–	1	3	mA

Parameter	Symbol	Condition	Mode	Min.	Typ.	Max.	Unit
		Condition. LVD OFF, LCD OFF					
	I_{DD62}	CPU stops in I_{DD52} Condition. LVD OFF, LCD OFF	Sub-idle 2	–	1	3	mA
	I_{DD71}	Disable EMCLK, IMCLK, ESCLK, and ISCLK All peripherals stop. LVD OFF, LCD OFF	Stop 1	–	0.5	1	mA
	I_{DD72}	Disable EMCLK, IMCLK, and ESCLK Enable ISCLK, FRT LVD OFF, LCD OFF	Stop 2	–	0.5	1	mA

NOTE:

1. Supply Current does not include current drawn through internal pull-up resistor, LCD voltage dividing resistors, and external output current loads.
2. Above tables, the current is based on LVD-OFF condition.
3. 2.7V data is only characteristic data. These values are not tested in the mass production

26.9 LVD Characteristics

(T_A = –40 to 85 °C, $V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD}$ = 2.7V to 5.5V)

Oscillator	Symbol	Conditions	Min	Typ	Max	Unit
LVD Detect Voltage	V_{LVD0}	Reset Default	2.5	2.6	2.7	V
	V_{LVD1}		2.6	2.8	3.0	
	V_{LVD2}		3.5	3.75	4.0	
	V_{LVD3}		4.0	4.25	4.5	
Hysteresis of V_{LVD} (Slew Rate of LVD)	ΔV	–	–	100	200	mV

NOTE: User can select the level for reset and interrupt voltage by LVDRL[2:0] and LVDIL[2:0] in CM_MR0 register

26.10 12-Bit ADC0 Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution	–	–	–	12	–	Bit
Supply Voltage	AV_{DD0}	–	2.7	5	5.5	V
Reference Voltage	AV_{REF}	–	2.7	–	AV_{DD0}	V
Input Voltage Range	V_{AIN0}	–	0	–	AV_{REF}	V
Clock Frequency	F_{ADC0}	50% duty cycle	–	–	5	MHz
Maximum Conversion Time	t_{ADC0}	Max. F_{ADC0} AV_{DD0} = AV_{REF}	–	–	1	MSPS
Differential nonlinearity	DNL	AV_{DD0} = AV_{REF} AV_{SS0} = 0.0V	–	–	±1.5	LSB
Integral nonlinearity	INL	AV_{DD0} = AV_{REF} AV_{SS0} = 0.0V	–	–	±3.5	LSB
Offset Error (unadjusted)[1]	TOPOFF	AV_{REF} = 5.5V	–	–	AV_{REF} - 43	mV
		AV_{REF} = 2.7V	–	–	AV_{REF} - 22	mV
	BOTOFF	AV_{REF} = 5.5V	–	–	43	mV
		AV_{REF} = 2.7V	–	–	22	mV
Operation Current	I_{OP}	AV_{DD0} = AV_{REF} = 5.0V AV_{SS0} = 0.0V	–	7.5	10	mA
Power down Current	I_{PD}	AV_{DD0} = AV_{REF} = 5.0V AV_{SS0} = 0.0V	–	20	30	µA

NOTE: When Internal Calibration mode is set, the 1/4 and 3/4 values of V_{REF} are stored in ADC_OCR registers but these values can't be used for getting accurate compensation (calibration) because of the offset difference in the customer board environment. Therefore external calibration is better to be used for more accurate calibration. More detail information is shown in ADC0 section.

26.10.1 OP-AMP Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input Offset Voltage	$\lvert V_{IO} \rvert$	–	–	–	±9	mV
Input Voltage Range	V_I	Gain = 2.32 to 4.09	$0.04AV_{DD0}$	–	$0.36AV_{DD0}$	V
		Gain = 4.53 to 5.92	$0.02AV_{DD0}$	–	$0.18AV_{DD0}$	V
		Gain = 7.09, 8.86	$0.01AV_{DD0}$	–	$0.085AV_{DD0}$	V
Slew Rate	S_R	@CL = 10pF	10	15	-	V/µs
		@CL = 50pF	–	10	-	V/µs
Gain Error	G_E	Gain = 2.32 to 4.53, T_A = 25°C	–	±1.0	±2.0	%
		Gain = 5.04 to 8.86, T_A = 25°C	–	±1.5	±3.0	%
		Gain = 2.32 to 4.53, T_A = –40 to 85 °C	–	±2.0	±4.0	%
		Gain = 5.04 to 8.86, T_A = –40 to 85 °C	–	±3.0	±6.0	%

26.11 10-Bit ADC1 Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution	–	–	–	10	–	Bit
Supply Voltage	AV_{DD1}	–	2.7	5	5.5	V
Reference Voltage	AV_{REF}	–	2.7	–	AV_{DD1}	V
Input Voltage Range	V_{AIN1}	–	0	–	AV_{REF}	V
Clock Frequency	F_{ADC1}	50% duty cycle	–	–	700	㎑
Maximum Conversion Time	t_{ADC1}	Max. F_{ADC1} AV_{DD1} = AV_{REF}	–	–	50	KSPS
Differential nonlinearity	DNL	AV_{DD1} = AV_{REF} AV_{SS1} = 0.0V	–	–	±1	LSB
Integral nonlinearity	INL	AV_{DD1} = AV_{REF} AV_{SS1} = 0.0V	–	–	±1.5	LSB
Offset Error (unadjusted)	TOPOFF	AV_{REF} = 5.5V	–	–	AV_{REF} - 172	mV
		AV_{REF} = 2.7V	–	–	AV_{REF} - 85	mV
	BOTOFF	AV_{REF} = 5.5V	–	–	172	mV
		AV_{REF} = 2.7V	–	–	85	mV
Operation Current	I_{OP}	AV_{DD1} = AV_{REF} = 5.0V AV_{SS1} = 0.0V	–	0.2	1	mA
Power down Current	I_{PD}	AV_{DD1} = AV_{REF} = 5.0V AV_{SS1} = 0.0V	–	1	2	µA

26.12 LCD Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
LCD voltage dividing resistor	R_{LCD1}	–	LCDC_CR.17=0	40	60	80	kΩ
	R_{LCD2}		LCDC_CR.17=1	20	30	40	kΩ
$\|V_{LCD}$ - COMi$\|$ voltage drop (i=0 ~ 3)	V_{DC}	–15μA per common pin		–	–	120	mV
$\|V_{LCD}$ - SEGi$\|$ voltage drop (i=0 ~ 39)	V_{DS}	–15μA per common pin		–	–	120	mV
Middle Output Voltage	V_{LCD1}	–		$0.67V_{DD}$-0.2	–	$0.67V_{DD}$+0.2	V
	V_{LCD2}			$0.33V_{DD}$-0.2	–	$0.33V_{DD}$+0.2	V

26.13 Memory Characteristics

26.13.1 Program Flash Memory Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Total Size	F_{SIZE}	–	–	384	–	KB
Program Size	F_{WSIZE}	–	–	4	–	B
Page Size	F_{PSIZE}	–	–	1024	–	B
Sector Size	F_{SSIZE}	–	–	32	–	KB
Programming Time for 1 Word	F_{TW}	–	20	25	30	µs
Page Erase Time	F_{TPERA}	–	4	8	12	ms
Sector Erase Time	F_{TSERA}	–	12	20	28	ms
Chip Erase Time	F_{TCERA}	–	32	50	70	ms
Read Command to valid data	F_{RA}	–	–	–	50	ns
Endurance Number of writing/erasing	F_{NWE}	–	10,000	–	–	Times
Data Retention	F_{TDR}	–	10	–	–	Years

NOTE: Flash hardware operating times: Total flash operating (program/erase) time may depend upon the software

26.13.2 Data FLASH Memory Characteristics

(T_A = –40 to 85 °C, V_{DD} = V_{DDCORE} = V_{DDIO} = AV_{DD} = 2.7V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Total Size	F_{SIZE}	–	–	16	–	KB
Program Size	F_{WSIZE}	–	1	2	4	B
Page Size	F_{PSIZE}	–	–	256	–	B
Sector Size	F_{SSIZE}	–	–	1	–	KB
Programming Time for 1 Word	F_{TW}	–	20	25	30	µs
Page Erase Time	F_{TPERA}	–	4	8	12	ms
Sector Erase Time	F_{TSERA}	–	12	20	28	ms
Chip Erase Time	F_{TCERA}	–	32	50	70	ms
Read Command to valid data	F_{RA}	–	–	–	100	ns
Endurance Number of writing/erasing	F_{NWE}	–	100,000	–	–	Times
Data Retention	F_{TDR}	–	10	–	–	Years

NOTE: Flash hardware operating times: Total flash operating (program/erase) time may depend upon the software

26.14 ESD Characteristics

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Electrostatic discharge	V_{ESD}	HBM	2000	–	–	V
		MM	200	–	–	V
		CDM	500	–	–	V

27 Package Specification

27.1 Overview

This chapter describes the package information of S3FM02G. It is available in a 128-ETQFP-1414 package type.

Table 27-1 Absolute Maximum Ratings

Package Number	128-ETQFP-1414
Package Width / Package Length	14.0 / 14.0 mm
Mounting Height	1.20 mm MAX
Lead Pitch	0.40 mm

27.2 Package Dimension

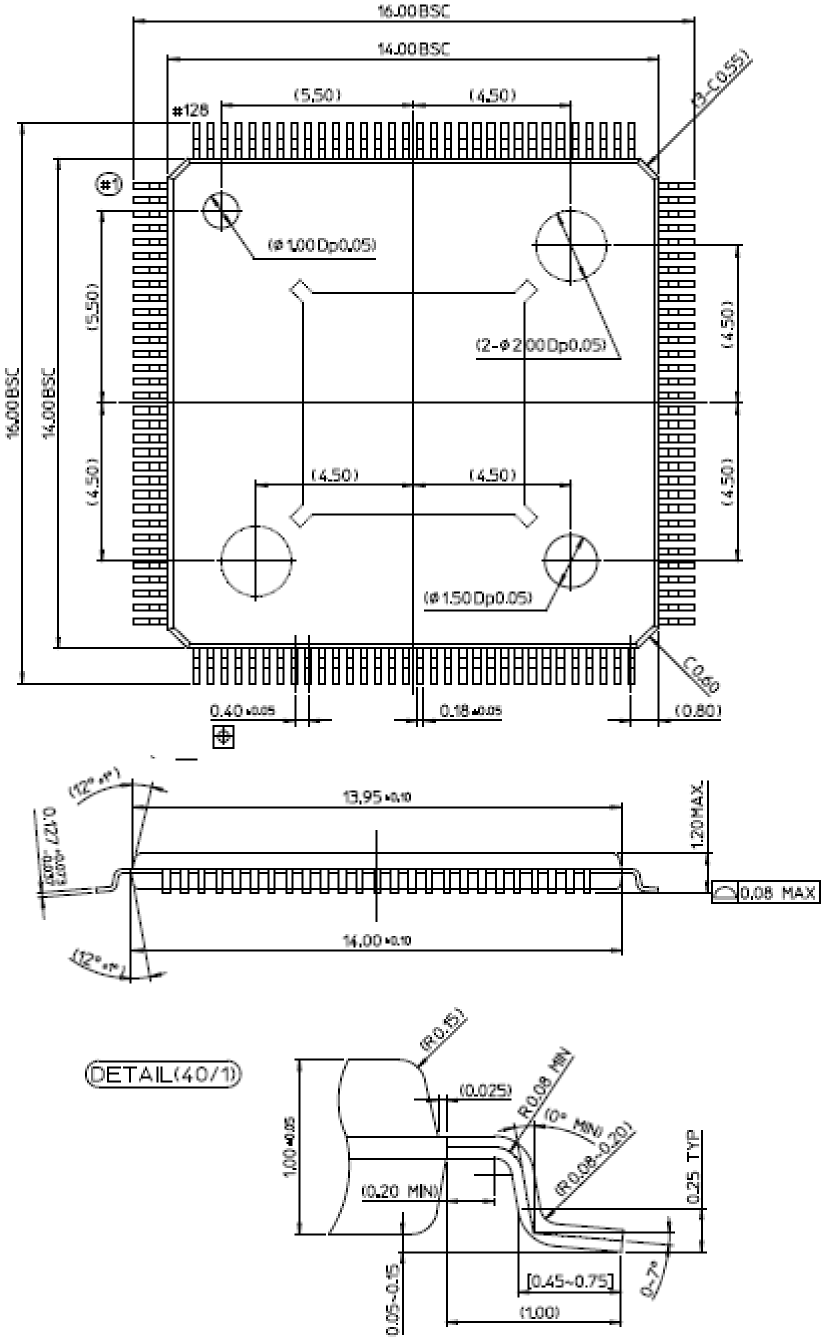


Figure 27-1 128ETQFP-1414 Package Dimension