

MC9S12DG128

Device User Guide

Covers MC9S12DT128E, MC9S12DG128E,
MC9S12DJ128E, MC9S12DT128, MC9S12DJ128,
MC9S12DB128, MC9S12A128, SC515846, SC515847,
SC515848, SC515849, SC101161DT, SC101161DG,
SC101161DJ, SC102202, SC102203, SC102204,
SC102205

HCS12
Microcontrollers

9S12DT128DGV2/D
V02.17
03 Jun 2010

freescale.com



Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V01.00	18 Jun 2001	18 June 2001		Initial version (parent doc v2.03 dug for dp256).
V01.01	23 July 2001	23 July 2001		Updated version after review
V01.02	23 Sep 2001	23 Sep 2001		Changed Partname, added pierce mode, updated electrical characteristics some minor corrections
V01.03	12 Oct 2001	12 Oct 2001		Replaced Star12 by HCS12
V01.04	27 Feb 2002	27 Feb 2002		Updated electrical spec after MC-Qualification (IOL/IOH), Data for Pierce, NVM reliability New document numbering. Corrected Typos
V01.05	4 Mar 2002	4 Mar 2002		Increased VDD to 2.35V, removed min. oscillator startup Removed Document order number except from Cover Sheet
V01.06	8 July 2002	22 July 2002		Added: Pull-up columns to signal table, example for PLL Filter calculation, Thermal values for junction to board and package, BGND pin pull-up Part Order Information Global Register Table Chip Configuration Summary Modified: Reduced Wait and Run IDD values Mode of Operation chapter changed leakage current for ADC inputs down to +-1uA Corrected: Interrupt vector table enable register inconsistencies PCB layout for 80QFP VREGEN position
V02.00	11 Jan 2002	11 Jan 2002		NEW MASKSET Changed part number from DTB128 to DT128 Functional Changes: ROMCTL changes in Emulation Mode 80 Pin Byteflight package Option available Flash with 2 Bit Backdoor Key Enable Additional CAN0 routing to PJ7,6 Improved BDM with sync and acknowledge capabilities New Part ID number Improvements: Significantly improved NVM reliability data Corrections: Interrupt vector Table
V02.01	01 Feb 2002	01 Feb 2002		Updated Block User Guide versions in preface Updated Appendix A Electrical Characteristics

Version Number	Revision Date	Effective Date	Author	Description of Changes
V02.02	08 Mar 2002	08 Mar 2002		<p>Changed XCLKS to PE7 in Table 2-2</p> <p>Updated device part numbers in Figure 2-1</p> <p>Updated BDM clock in Figure 3-1</p> <p>Removed SIM description in overview & n_{UPOSC} spec in Table A-15</p> <p>Updated electrical spec of VDD & VDDPLL (Table A-4), IOL/IOH (Table A-6), C_{INS} (Table A-9), C_{IN} (Table A-6 & A-15),</p> <p>Updated interrupt pulse timing variables in Table A-6</p> <p>Updated device part numbers in Figure 2-1</p> <p>Added document numbers on cover page and Table 0-2</p>
V02.03	14 Mar 2002	14 Mar 2002		<p>Cleaned up Fig. 1-1, 2-1</p> <p>Updated Section 1.5 descriptions</p> <p>Corrected PE assignment in Table 2-2, Fig. 2-5,6,7.</p> <p>Corrected NVM sizes in Sections 16, 17</p> <p>Added I_{REF} spec for 1ATD in Table A-8</p> <p>Added Blank Check in A.3.1.5 and Table A-11</p> <p>Updated CRG spec in Table A-15</p>
V02.04	16 Aug 2002	16 Aug 2002		<p>Added:</p> <ul style="list-style-type: none"> Pull-up columns to signal table, Example for PLL Filter calculation, Thermal values for junction to board and package, BGND pin pull-up Part Order Information Global Register Table Chip Configuration Summary Device specific info on CRG <p>Modified:</p> <ul style="list-style-type: none"> Reduced Wait and Run IDD values Mode of Operation chapter Changed leakage current for ADC inputs down to $\pm 1\mu A$ Minor modification of PLL frequency/ voltage gain values <p>Corrected:</p> <ul style="list-style-type: none"> Pin names/functions on 80 pin packages Interrupt vector table enable register inconsistencies PCB layout for 80QFP VREGEN position
V02.05	12 Sep 2002	12 Sep 2002		<p>Corrected:</p> <ul style="list-style-type: none"> Register address mismatches in 1.5.1
V02.06	06 Nov 2002	06 Nov 2002		<p>Removed document order no. from Revision History pages</p> <p>Renamed "Preface" section to "Derivative Differences and Document references". Added details for derivatives missing CAN0/1/4, BDLC, IIC and/or Byteflight</p> <p>Added 2L40K mask set in section 1.6</p> <p>Added OSC User Guide in Preface, "Document References"</p> <p>Added oscillator clock connection to BDM in S12_CORE in fig 3-1</p> <p>Corrected several register and bit names in "Local Enable" column of Table 5.1 Interrupt Vector Locations</p> <p>Section HCS12 Core Block Description: mentioned alternate clock of BDM to be equivalent to oscillator clock</p> <p>Added new section: "Oscillator (OSC) Block Description"</p> <p>Corrected in footnote of Table "PLL Characteristics": $f_{OSC} = 4MHz$</p>

Version Number	Revision Date	Effective Date	Author	Description of Changes
V02.07	29 Jan 2003	29 Jan 2003		<p>Added 3L40K mask set in section 1.6</p> <p>Corrected register entries in section 1.5.1 “Detailed Memory Map”</p> <p>Updated description for ROMCTL in section 2.3.31</p> <p>Updated section 4.3.3 “Unsecuring the Microcontroller”</p> <p>Corrected and updated device-specific information for OSC (section 8.1) & Byteflight (section 15.1)</p> <p>Updated footnote in Table A-4 “Operating Conditions”</p> <p>Changed reference of VDDM to VDDR in section A.1.8</p> <p>Removed footnote on input leakage current in Table A-6 “5V I/O Characteristics”</p>
V02.08	26 Feb 2003	26 Feb 2003		<p>Added part numbers MC9S12DT128E, MC9S12DG128E, and MC9S12DJ128E in “Preface” and related part number references</p> <p>Removed mask sets 0L40K and 2L40K from Table 1-3</p>
V02.09	15 Oct 2003	15 Oct 2003		<p>Replaced references to HCS12 Core Guide by the individual HCS12 Block guides in Table 0-2, section 1.5.1, and section 6; updated Fig.3-1 “Clock Connections” to show the individual HCS12 blocks</p> <p>Corrected PIM module name and document order number in Table 0-2 “Document References”</p> <p>Corrected ECT pulse accumulators description in section 1.2 “Features”</p> <p>Corrected KWP5 pin name in Fig 2-1 112LQFP pin assignments</p> <p>Corrected pull resistor CTRL/reset states for PE7 and PE4-PE0 in Table 2.1 “Signal Properties”</p> <p>Mentioned “S12LRAE” bootloader in Flash section 17</p> <p>Corrected footnote on clamp of TEST pin under Table A-1 “Absolute Maximum Ratings”</p> <p>Corrected minimum bus frequency to 0.25MHz in Table A-4 “Operating Conditions”</p> <p>Replaced “burst programming” by “row programming” in A.3 “NVM, Flash and EEPROM”</p> <p>Corrected blank check time for EEPROM in Table A-11 “NVM Timing Characteristics”</p> <p>Corrected operating frequency in Table A-18 “SPI Master/Slave Mode Timing Characteristics”</p>
V02.10	6 Feb 2004	6 Feb 2004		<p>Added A128 information in “Derivative Differences”, 2.1 “Device Pinout”, 2.2 “Signal Properties Summary”, Fig 23-2 & Fig 23-4</p> <p>Added lead-free package option (PVE) in Table 0-2 “Derivative Differences for MC9S12DB128” and Fig 0-1 “Order Partnumber Example”</p> <p>Added an “AEC qualified” row in the “Derivative Differences” tables 0-1 & 0-2.</p>
V02.11	3 May 2004	3 May 2004		<p>Added part numbers SC515846, SC515847, SC515848, and SC515849 in “Derivative Differences” tables 0-1 & 0-2, section 2, and section 23.</p> <p>Corrected and added maskset 4L40K in tables 0-1 & 0-2 and section 1.6.</p> <p>Corrected BDLC module availability in DB128 80QFP part in “Derivative Differences” table 0-2.</p>

Version Number	Revision Date	Effective Date	Author	Description of Changes
V02.12	06 Dec 2004	06 Dec 2004		Added maskset 0L94R Added items $V_{IH,EXTAL}$, $V_{IL,EXTAL}$, & $V_{HYS,EXTAL}$ in table A-15 "Oscillator characteristics" Removed item "Oscillator" from table A-4 "Operating Conditions" as it is already covered in table "Oscillator Characteristics"
V02.13	04 Mar 2005	04 Mar 2005		Amended feature list of A128 in Table 0-1 "Derivative Differences"
V02.14	28 Apr 2005	28 Apr 2005		Updated cover page Added part numbers SC101161DT, SC101161DG, SC101161DJ, SC102202, SC102203, SC102204, & SC102205 Added masksets 5L40K & 1L59W Changed T_{Javg} to 85°C in table A-12 "NVM Reliability" & added footnote concerning data retention
V02.15	05 Oct 2005	05 Oct 2005		Updated "NVM Reliability" table A-12 format with added data. Added figure A-2 "Typical Endurance vs Temperature"
V02.16	12 Apr 2008	12 Apr 2008		Added maskset 2L94R
V02.17	3 Jun 2010	3 Jun 2010		Added maskset 1L59W for MC9S12A128

Table of Contents

Section 1 Introduction

1.1	Overview	25
1.2	Features	25
1.3	Modes of Operation	27
1.4	Block Diagram	28
1.5	Device Memory Map	30
1.5.1	Detailed Register Map	32
1.6	Part ID Assignments	55

Section 2 Signal Description

2.1	Device Pinout	57
2.2	Signal Properties Summary	60
2.3	Detailed Signal Descriptions	64
2.3.1	EXTAL, XTAL — Oscillator Pins	64
2.3.2	RESET — External Reset Pin	64
2.3.3	TEST — Test Pin	64
2.3.4	XFC — PLL Loop Filter Pin	64
2.3.5	BKGD / TAGHI / MODC — Background Debug, Tag High, and Mode Pin	64
2.3.6	PAD[15] / AN1[7] / ETRIG1 — Port AD Input Pin [15]	65
2.3.7	PAD[14:8] / AN1[6:0] — Port AD Input Pins [14:8]	65
2.3.8	PAD[7] / AN0[7] / ETRIG0 — Port AD Input Pin [7]	65
2.3.9	PAD[6:0] / AN0[6:0] — Port AD Input Pins [6:0]	65
2.3.10	PA[7:0] / ADDR[15:8] / DATA[15:8] — Port A I/O Pins	65
2.3.11	PB[7:0] / ADDR[7:0] / DATA[7:0] — Port B I/O Pins	65
2.3.12	PE7 / NOACC / XCLKS — Port E I/O Pin 7	65
2.3.13	PE6 / MODB / IPIPE1 — Port E I/O Pin 6	67
2.3.14	PE5 / MODA / IPIPE0 — Port E I/O Pin 5	67
2.3.15	PE4 / ECLK — Port E I/O Pin 4	67
2.3.16	PE3 / LSTRB / TAGLO — Port E I/O Pin 3	67
2.3.17	PE2 / R/W — Port E I/O Pin 2	67
2.3.18	PE1 / IRQ — Port E Input Pin 1	67
2.3.19	PE0 / XIRQ — Port E Input Pin 0	67
2.3.20	PH7 / KWH7 — Port H I/O Pin 7	67

2.3.21	PH6 / KWH6 — Port H I/O Pin 6	68
2.3.22	PH5 / KWH5 — Port H I/O Pin 5	68
2.3.23	PH4 / KWH4 — Port H I/O Pin 2	68
2.3.24	PH3 / KWH3 / SS1 — Port H I/O Pin 3	68
2.3.25	PH2 / KWH2 / SCK1 — Port H I/O Pin 2	68
2.3.26	PH1 / KWH1 / MOSI1 — Port H I/O Pin 1	68
2.3.27	PH0 / KWH0 / MISO1 — Port H I/O Pin 0	68
2.3.28	PJ7 / KWJ7 / TXCAN4 / SCL / TXCAN0 — PORT J I/O Pin 7	68
2.3.29	PJ6 / KWJ6 / RXCAN4 / SDA / RXCAN0 — PORT J I/O Pin 6	69
2.3.30	PJ[1:0] / KWJ[1:0] — Port J I/O Pins [1:0]	69
2.3.31	PK7 / ECS / ROMCTL — Port K I/O Pin 7	69
2.3.32	PK[5:0] / XADDR[19:14] — Port K I/O Pins [5:0]	69
2.3.33	PM7 / BF_PSLM / TXCAN4 — Port M I/O Pin 7	69
2.3.34	PM6 / BF_PERR / RXCAN4 — Port M I/O Pin 6	69
2.3.35	PM5 / BF_PROK / TXCAN0 / TXCAN4 / SCK0 — Port M I/O Pin 5	69
2.3.36	PM4 / BF_PSYN / RXCAN0 / RXCAN4/ MOSI0 — Port M I/O Pin 4	70
2.3.37	PM3 / TX_BF / TXCAN1 / TXCAN0 / SS0 — Port M I/O Pin 3	70
2.3.38	PM2 / RX_BF / RXCAN1 / RXCAN0 / MISO0 — Port M I/O Pin 2	70
2.3.39	PM1 / TXCAN0 / TXB — Port M I/O Pin 1	70
2.3.40	PM0 / RXCAN0 / RXB — Port M I/O Pin 0	70
2.3.41	PP7 / KWP7 / PWM7 — Port P I/O Pin 7	70
2.3.42	PP6 / KWP6 / PWM6 — Port P I/O Pin 6	70
2.3.43	PP5 / KWP5 / PWM5 — Port P I/O Pin 5	71
2.3.44	PP4 / KWP4 / PWM4 — Port P I/O Pin 4	71
2.3.45	PP3 / KWP3 / PWM3 / SS1 — Port P I/O Pin 3	71
2.3.46	PP2 / KWP2 / PWM2 / SCK1 — Port P I/O Pin 2	71
2.3.47	PP1 / KWP1 / PWM1 / MOSI1 — Port P I/O Pin 1	71
2.3.48	PP0 / KWP0 / PWM0 / MISO1 — Port P I/O Pin 0	71
2.3.49	PS7 / SS0 — Port S I/O Pin 7	71
2.3.50	PS6 / SCK0 — Port S I/O Pin 6	71
2.3.51	PS5 / MOSI0 — Port S I/O Pin 5	72
2.3.52	PS4 / MISO0 — Port S I/O Pin 4	72
2.3.53	PS3 / TXD1 — Port S I/O Pin 3	72
2.3.54	PS2 / RXD1 — Port S I/O Pin 2	72
2.3.55	PS1 / TXD0 — Port S I/O Pin 1	72
2.3.56	PS0 / RXD0 — Port S I/O Pin 0	72

2.3.57	PT[7:0] / IOC[7:0] — Port T I/O Pins [7:0]	72
2.4	Power Supply Pins	72
2.4.1	VDDX, VSSX — Power & Ground Pins for I/O Drivers	73
2.4.2	VDDR, VSSR — Power & Ground Pins for I/O Drivers & for Internal Voltage Regulator	73
2.4.3	VDD1, VDD2, VSS1, VSS2 — Internal Logic Power Supply Pins	73
2.4.4	VDDA, VSSA — Power Supply Pins for ATD and VREG	74
2.4.5	VRH, VRL — ATD Reference Voltage Input Pins	74
2.4.6	VDDPLL, VSSPLL — Power Supply Pins for PLL	74
2.4.7	VREGEN — On Chip Voltage Regulator Enable	74

Section 3 System Clock Description

3.1	Overview	75
-----	----------	----

Section 4 Modes of Operation

4.1	Overview	77
4.2	Chip Configuration Summary	77
4.3	Security	78
4.3.1	Securing the Microcontroller	78
4.3.2	Operation of the Secured Microcontroller	78
4.3.3	Unsecuring the Microcontroller	79
4.4	Low Power Modes	79
4.4.1	Stop	79
4.4.2	Pseudo Stop	79
4.4.3	Wait	79
4.4.4	Run	79

Section 5 Resets and Interrupts

5.1	Overview	81
5.2	Vectors	81
5.2.1	Vector Table	81
5.3	Effects of Reset	82
5.3.1	I/O pins	82
5.3.2	Memory	83

Section 6 HCS12 Core Block Description

6.1	CPU Block Description	85
-----	-----------------------	----

6.1.1	Device-specific information	85
6.2	HCS12 Module Mapping Control (MMC) Block Description	85
6.2.1	Device-specific information	85
6.3	HCS12 Multiplexed External Bus Interface (MEBI) Block Description	85
6.3.1	Device-specific information	85
6.4	HCS12 Interrupt (INT) Block Description	86
6.5	HCS12 Background Debug Module (BDM) Block Description	86
6.5.1	Device-specific information	86
6.6	HCS12 Breakpoint (BKP) Block Description	86

Section 7 Clock and Reset Generator (CRG) Block Description

7.1	Device-specific information.	86
-----	--------------------------------------	----

Section 8 Oscillator (OSC) Block Description

8.1	Device-specific information.	86
-----	--------------------------------------	----

Section 9 Enhanced Capture Timer (ECT) Block Description

Section 10 Analog to Digital Converter (ATD) Block Description

Section 11 Inter-IC Bus (IIC) Block Description

Section 12 Serial Communications Interface (SCI) Block Description

Section 13 Serial Peripheral Interface (SPI) Block Description

Section 14 J1850 (BDLC) Block Description

Section 15 Byteflight (BF) Block Description

15.1	Device-specific information.	88
------	--------------------------------------	----

Section 16 Pulse Width Modulator (PWM) Block Description

Section 17 Flash EEPROM 128K Block Description

Section 18 EEPROM 2K Block Description

Section 19 RAM Block Description

Section 20 MSCAN Block Description**Section 21 Port Integration Module (PIM) Block Description****Section 22 Voltage Regulator (VREG) Block Description****Section 23 Printed Circuit Board Layout Proposal****Appendix A Electrical Characteristics**

A.1	General	97
A.1.1	Parameter Classification	97
A.1.2	Power Supply	97
A.1.3	Pins	98
A.1.4	Current Injection	98
A.1.5	Absolute Maximum Ratings	99
A.1.6	ESD Protection and Latch-up Immunity	99
A.1.7	Operating Conditions	100
A.1.8	Power Dissipation and Thermal Characteristics	101
A.1.9	I/O Characteristics	103
A.1.10	Supply Currents	104
A.2	ATD Characteristics	107
A.2.1	ATD Operating Characteristics	107
A.2.2	Factors influencing accuracy	107
A.2.3	ATD accuracy	109
A.3	NVM, Flash and EEPROM	111
A.3.1	NVM timing	111
A.3.2	NVM Reliability	113
A.4	Voltage Regulator	117
A.5	Reset, Oscillator and PLL	119
A.5.1	Startup	119
A.5.2	Oscillator	120
A.5.3	Phase Locked Loop	121
A.6	MSCAN	127
A.7	SPI	129
A.7.1	Master Mode	129
A.7.2	Slave Mode	131

A.8	External Bus Timing	133
A.8.1	General Multiplexed Bus Timing	133

Appendix B Package Information

B.1	General.	137
B.2	112-pin LQFP package.	138
B.3	80-pin QFP package.	139

List of Figures

Figure 0-1	Order Partnumber Example	20
Figure 1-1	MC9S12DT128 Block Diagram	29
Figure 1-2	MC9S12DT128 Memory Map	31
Figure 2-1	Pin assignments 112 LQFP for MC9S12DT128E, MC9S12DT128, MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12DB128, MC9S12A128, SC515846, SC515847, SC515848, SC515849, SC101161DT, SC101161DG, SC101161DJ, SC102202, SC102203, SC102204, and SC102205	58
Figure 2-2	Pin Assignments in 80 QFP for MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204 Bondout	59
Figure 2-3	Pin Assignments in 80 QFP for MC9S12DB128, SC515846, and SC102202 Bond- out	60
Figure 2-4	PLL Loop Filter Connections	64
Figure 2-5	Colpitts Oscillator Connections (PE7=1)	66
Figure 2-6	Pierce Oscillator Connections (PE7=0)	66
Figure 2-7	External Clock Connections (PE7=0)	66
Figure 3-1	Clock Connections.	75
Figure 23-1	Recommended PCB Layout for 112LQFP Colpitts Oscillator	91
Figure 23-2	Recommended PCB Layout for 80QFP (MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204) Colpitts Oscillator	92
Figure 23-3	Recommended PCB Layout for 112LQFP Pierce Oscillator	93
Figure 23-4	Recommended PCB Layout for 80QFP (MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204) Pierce Oscillator	94
Figure 23-5	Recommended PCB Layout for 80QFP (MC9S12DB128, SC515846, and SC102202) Pierce Oscillator	95
Figure A-1	ATD Accuracy Definitions	110
Figure A-2	Typical Endurance vs Temperature.	115
Figure A-3	Basic PLL functional diagram	122
Figure A-4	Jitter Definitions	124
Figure A-5	Maximum bus clock jitter approximation	124
Figure A-6	SPI Master Timing (CPHA = 0)	129
Figure A-7	SPI Master Timing (CPHA =1).	130
Figure A-8	SPI Slave Timing (CPHA = 0)	131
Figure A-9	SPI Slave Timing (CPHA =1).	131

Figure A-10 General External Bus Timing. 134
Figure 23-6 112-pin LQFP mechanical dimensions (case no. 987) 138

List of Tables

Table 0-1	Derivative Differences1	19
Table 0-2	Derivative Differences for MC9S12DB1281.	19
Table 0-3	Document References	23
Table 1-1	Device Memory Map	30
\$0000 - \$000F	MEBI map 1 of 3 (HCS12 Multiplexed External Bus Interface)	32
\$0010 - \$0014	MMC map 1 of 4 (HCS12 Module Mapping Control)	32
\$0015 - \$0016	INT map 1 of 2 (HCS12 Interrupt)	33
\$0017 - \$0017	MMC map 2 of 4 (HCS12 Module Mapping Control)	33
\$0018 - \$0019	Reserved	33
\$001A - \$001B	Device ID Register ((Table 1-3))	33
\$001C - \$001D	MMC map 3 of 4 (HCS12 Module Mapping Control, (Table 1-4))	33
\$001E - \$001E	MEBI map 2 of 3 (HCS12 Multiplexed External Bus Interface)	33
\$001F - \$001F	INT map 2 of 2 (HCS12 Interrupt)	34
\$0020 - \$0027	Reserved	34
\$0028 - \$002F	BKP (HCS12 Breakpoint)	34
\$0030 - \$0031	MMC map 4 of 4 (HCS12 Module Mapping Control)	34
\$0032 - \$0033	MEBI map 3 of 3 (HCS12 Multiplexed External Bus Interface)	34
\$0034 - \$003F	CRG (Clock and Reset Generator)	35
\$0040 - \$007F	ECT (Enhanced Capture Timer 16 Bit 8 Channels)	35
\$0080 - \$009F	ATD0 (Analog to Digital Converter 10 Bit 8 Channel)	38
\$00A0 - \$00C7	PWM (Pulse Width Modulator 8 Bit 8 Channel)	39
\$00C8 - \$00CF	SCI0 (Asynchronous Serial Interface)	41
\$00D0 - \$00D7	SCI1 (Asynchronous Serial Interface)	41
\$00D8 - \$00DF	SPI0 (Serial Peripheral Interface)	42
\$00E0 - \$00E7	IIC (Inter IC Bus)	42
\$00E8 - \$00EF	BDLC (Byte Level Data Link Controller J1850)	43
\$00F0 - \$00F7	SPI1 (Serial Peripheral Interface)	43
\$00F8 - \$00FF	Reserved	43
\$0100 - \$010F	Flash Control Register (fts128k2)	44
\$0110 - \$011B	EEPROM Control Register (eets2k)	44
\$011C - \$011F	Reserved for RAM Control Register	45
\$0120 - \$013F	ATD1 (Analog to Digital Converter 10 Bit 8 Channel)	45
\$0140 - \$017F	CAN0 (Motorola Scalable CAN - MSCAN)	46

Table 1-2	Detailed MSCAN Foreground Receive and Transmit Buffer Layout.	47
\$0180 - \$01BF	CAN1 (Motorola Scalable CAN - MSCAN)	48
\$01C0 - \$01FF	Reserved	49
\$0200 - \$023F	Reserved	49
\$0240 - \$027F	PIM (Port Integration Module)	50
\$0280 - \$02BF	CAN4 (Motorola Scalable CAN - MSCAN)	52
\$02C0 - \$02FF	Reserved	53
\$0300 - \$035F	Byteflight	53
\$0360 - \$03FF	Reserved	55
Table 1-3	Assigned Part ID Numbers	55
Table 1-4	Memory size registers	55
Table 2-1	Signal Properties	61
Table 2-2	MC9S12DT128 Power and Ground Connection Summary	72
Table 4-1	Mode Selection	77
Table 4-2	Clock Selection Based on PE7	77
Table 4-3	Voltage Regulator VREGEN	78
Table 5-1	Interrupt Vector Locations	81
Table 23-1	Suggested External Component Values	89
Table A-1	Absolute Maximum Ratings	99
Table A-2	ESD and Latch-up Test Conditions	100
Table A-3	ESD and Latch-Up Protection Characteristics	100
Table A-4	Operating Conditions	101
Table A-5	Thermal Package Characteristics	103
Table A-6	5V I/O Characteristics	104
Table A-7	Supply Current Characteristics	105
Table A-8	ATD Operating Characteristics	107
Table A-9	ATD Electrical Characteristics	108
Table A-10	ATD Conversion Performance	109
Table A-11	NVM Timing Characteristics	112
Table A-12	NVM Reliability Characteristics	114
Table A-13	Voltage Regulator Recommended Load Capacitances	117
Table A-14	Startup Characteristics	119
Table A-15	Oscillator Characteristics	121
Table A-16	PLL Characteristics	125
Table A-17	MSCAN Wake-up Pulse Characteristics	127
Table A-18	SPI Master Mode Timing Characteristics	130

Table A-19 SPI Slave Mode Timing Characteristics 132
Table A-20 Expanded Bus Timing Characteristics 135

Derivative Differences and Document References

Derivative Differences

(Table 0-1) and (Table 0-2) show the availability of peripheral modules on the various derivatives. For details about the compatibility within the MC9S12D-Family refer also to engineering bulletin EB386.

Table 0-1 Derivative Differences¹

Modules	MC9S12DT128E ³ MC9S12DT128 SC515849 ⁴ SC101161DT ⁵ SC102205 ⁶	MC9S12DG128E ³ MC9S12DG128 SC515847 ⁴ SC101161DG ⁵ SC102203 ⁶	MC9S12DJ128E ³ MC9S12DJ128 SC515848 ⁴ SC101161DJ ⁵ SC102204 ⁶	MC9S12A128
# of CANs	3	2	2	0
CAN4	✓	✓	✓	X
CAN1	✓	X	X	X
CAN0	✓	✓	✓	X
J1850/BDLC	X	X	✓	X
IIC	✓	✓	✓	✓
Byteflight	X	X	X	X
Package	112 LQFP	112 LQFP/80 QFP ²	112 LQFP/80 QFP ²	112 LQFP/80 QFP ²
Package Code	PV	PV/FU	PV/FU	PV/FU
Mask set	1L40K ³ , 3L40K, 0L94R, 4L40K ⁴ , 1L59W ⁵ , 5L40K ⁶ , 2L94R	1L40K ³ , 3L40K, 0L94R, 4L40K ⁴ , 1L59W ⁵ , 5L40K ⁶ , 2L94R	1L40K ³ , 3L40K, 0L94R, 4L40K ⁴ , 1L59W ⁵ , 5L40K ⁶ , 2L94R	3L40K, 0L94R, 2L94R, 1L59W
Temp Options	M, V, C	M, V, C	M, V, C	C
AEC qualified	Yes	Yes	Yes	No
Notes	An errata exists contact Sales Office	An errata exists contact Sales Office	An errata exists contact Sales Office	An errata exists contact Sales Office

Table 0-2 Derivative Differences for MC9S12DB128¹

Modules	MC9S12DB128 SC515846 ⁴ SC102202 ⁶	MC9S12DB128 SC515846 ⁴ SC102202 ⁶
# of CANs	2	0
CAN4	✓	X
CAN1	X	X
CAN0	✓	X
J1850/BDLC	X	X
IIC	X	X
Byteflight	✓	✓
Package	112 LQFP	80 QFP ²

Modules	MC9S12DB128 SC515846 ⁴ SC102202 ⁶	MC9S12DB128 SC515846 ⁴ SC102202 ⁶
Package Code	PV/PVE	FU
Mask set	3L40K, 0L94R, 4L40K ⁴ , 5L40K ⁶ , 2L94R	3L40K, 0L94R, 4L40K ⁴ , 5L40K ⁶ , 2L94R
Temp Options	M, V, C/M, V	M, V, C
AEC qualified	Yes	Yes
Notes	An errata exists contact Sales Office	An errata exists contact Sales Office

NOTE:

- ✓: Available for this device, X: Not available for this device.
- 80 Pin bond-out for MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204 is the same; MC9S12DB128, SC515846, and SC102202 have a different bond-out.
- Part numbers MC9S12DT128E, MC9S12DG128E, and MC9S12DJ128E are associated with the mask set 1L40K.
- Part numbers SC515846, SC515847, SC515848, and SC515849 are associated with the mask set 4L40K.
- Part numbers SC101161DT, SC101161DG, SC101161DJ are associated with the mask set 1L59W.
- Part numbers SC102202, SC102203, SC102204, and SC102205 are associated with the mask set 5L40K which is not for volume production.

The following figure provides an ordering number example for the MC9S12D128 devices.

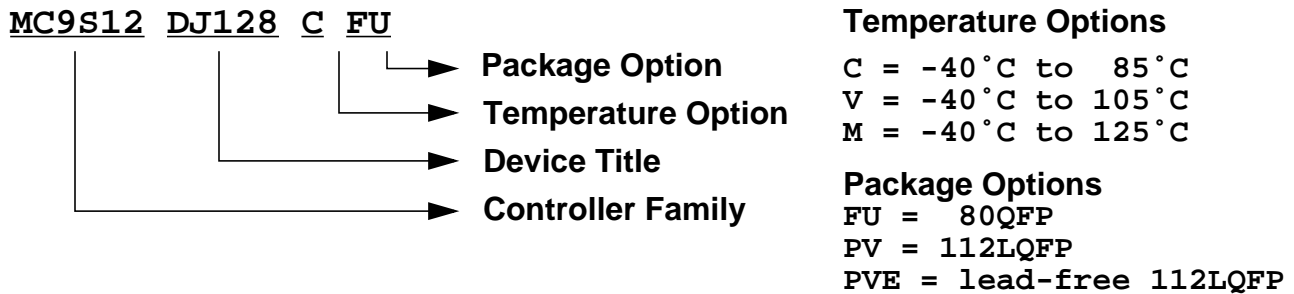


Figure 0-1 Order Partnumber Example

The following items should be considered when using a derivative.

- Registers**

- Do not write or read CAN0 registers (after reset: address range \$0140 - \$017F), if using a derivative without CAN0 (see **(Table 0-1)** and **(Table 0-2)**).
- Do not write or read CAN1 registers (after reset: address range \$0180 - \$01BF), if using a derivative without CAN1 (see **(Table 0-1)** and **(Table 0-2)**).
- Do not write or read CAN4 registers (after reset: address range \$0280 - \$02BF), if using a derivative without CAN4 (see **(Table 0-1)** and **(Table 0-2)**).
- Do not write or read BDLC registers (after reset: address range \$00E8 - \$00EF), if using a derivative without BDLC (see **(Table 0-1)** and **(Table 0-2)**).
- Do not write or read IIC registers (after reset: address range \$00E0 - \$00E7), if using a derivative without IIC (see **(Table 0-1)** and **(Table 0-2)**).

- Do not write or read Byteflight registers (after reset: address range \$0300 - \$035F), if using a derivative without Byteflight registers (see **(Table 0-1)** and **(Table 0-2)**).
- **Interrupts**
 - Fill the four CAN0 interrupt vectors (\$FFB0 - \$FFB7) according to your coding policies for unused interrupts, if using a derivative without CAN0 (see **(Table 0-1)** and **(Table 0-2)**).
 - Fill the four CAN1 interrupt vectors (\$FFA8 - \$FFAF) according to your coding policies for unused interrupts, if using a derivative without CAN1 (see **(Table 0-1)** and **(Table 0-2)**).
 - Fill the four CAN4 interrupt vectors (\$FF90 - \$FF97) according to your coding policies for unused interrupts, if using a derivative without CAN4 (see **(Table 0-1)** and **(Table 0-2)**).
 - Fill the BDLC interrupt vector (\$FFC2, \$FFC3) according to your coding policies for unused interrupts, if using a derivative without BDLC (see **(Table 0-1)** and **(Table 0-2)**).
 - Fill the IIC interrupt vector (\$FFC0, \$FFC1) according to your coding policies for unused interrupts, if using a derivative without IIC (see **(Table 0-1)** and **(Table 0-2)**).
 - Fill the four Byteflight interrupt vectors (\$FFA0 - \$FFA7) according to your coding policies for unused interrupts, if using a derivative without Byteflight (see **(Table 0-1)** and **(Table 0-2)**).
- **Ports**
 - The CAN0 pin functionality (TXCAN0, RXCAN0) is not available on port PJ7, PJ6, PM5, PM4, PM3, PM2, PM1 and PM0, if using a derivative without CAN0 (see **(Table 0-1)** and **(Table 0-2)**).
 - The CAN1 pin functionality (TXCAN1, RXCAN1) is not available on port PM3 and PM2, if using a derivative without CAN1 (see **(Table 0-1)** and **(Table 0-2)**).
 - The CAN4 pin functionality (TXCAN4, RXCAN4) is not available on port PJ7, PJ6, PM7, PM6, PM5 and PM4, if using a derivative without CAN4 (see **(Table 0-1)** and **(Table 0-2)**).
 - The BDLC pin functionality (TXB, RXB) is not available on port PM1 and PM0, if using a derivative without BDLC (see **(Table 0-1)** and **(Table 0-2)**).
 - The IIC pin functionality (SCL, SCA) is not available on port PJ7 and PJ6, if using a derivative without IIC (see **(Table 0-1)** and **(Table 0-2)**).
 - The Byteflight pin functionality (BF_PSLM, BF_PERR, BF_PROK, BF_PSYN, TX_BF, RX_BF) is not available on port PM7, PM6, PM5, PM4, PM3 and PM2, if using a derivative without Byteflight (see **(Table 0-1)** and **(Table 0-2)**).
 - Do not write MODRR1 and MODRR0 Bit of Module Routing Register (PIM_9DTB128 Block User Guide), if using a derivative without CAN0 (see **(Table 0-1)** and **(Table 0-2)**).
 - Do not write MODRR3 and MODRR2 Bit of Module Routing Register (PIM_9DTB128 Block User Guide), if using a derivative without CAN4 (see **(Table 0-1)** and **(Table 0-2)**).
- **Pins not available in 80 pin QFP package for MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204**

- **Port H**
In order to avoid floating nodes the ports should be either configured as outputs by setting the data direction register (DDRH at Base+\$0262) to \$FF, or enabling the pull resistors by writing a \$FF to the pull enable register (PERH at Base+\$0264).
- **Port J[1:0]**
Port J pull-up resistors are enabled out of reset on all four pins (7:6 and 1:0). Therefore care must be taken not to disable the pull enables on PJ[1:0] by clearing the bits PERJ1 and PERJ0 at Base+\$026C.
- **Port K**
Port K pull-up resistors are enabled out of reset, i.e. Bit 7 = PUKE = 1 in the register PUCR at Base+\$000C. Therefore care must be taken not to clear this bit.
- **Port M[7:6]**
PM7:6 must be configured as outputs or their pull resistors must be enabled to avoid floating inputs.
- **Port P6**
PP6 must be configured as output or its pull resistor must be enabled to avoid a floating input.
- **Port S[7:4]**
PS7:4 must be configured as outputs or their pull resistors must be enabled to avoid floating inputs.
- **PAD[15:8] (ATD1 channels)**
Out of reset the ATD1 is disabled preventing current flows in the pins. Do not modify the ATD1 registers!
- **Pins not available in 80 pin QFP package for MC9S12DB128, SC515846, and SC102202**
 - **Port H**
In order to avoid floating nodes the ports should be either configured as outputs by setting the data direction register (DDRH at Base+\$0262) to \$FF, or enabling the pull resistors by writing a \$FF to the pull enable register (PERH at Base+\$0264).
 - **Port J[7:6, 1:0]**
Port J pull-up resistors are enabled out of reset on all four pins (7:6 and 1:0). Therefore care must be taken not to disable the pull enables on PJ[7:6, 1:0] by clearing the bits PERJ7, PERJ6, PERJ1 and PERJ0 at Base+\$026C.
 - **Port K**
Port K pull-up resistors are enabled out of reset, i.e. Bit 7 = PUKE = 1 in the register PUCR at Base+\$000C. Therefore care must be taken not to clear this bit.
 - **Port M[1:0]**
PM1:0 must be configured as outputs or their pull resistors must be enabled to avoid floating inputs.
 - **Port P6**
PP6 must be configured as output or its pull resistor must be enabled to avoid a floating input.

- **Port S[3:2]**
PS3:2 must be configured as outputs or their pull resistors must be enabled to avoid floating inputs.
- **PAD[15:8] (ATD1 channels)**
Out of reset the ATD1 is disabled preventing current flows in the pins. Do not modify the ATD1 registers!

Document References

The Device User Guide provides information about the MC9S12DT128 device made up of standard HCS12 blocks and the HCS12 processor core.

This document is part of the customer documentation. A complete set of device manuals also includes all the individual Block User Guides of the implemented modules. In a effort to reduce redundancy all module specific information is located only in the respective Block User Guide. If applicable, special implementation details of the module are given in the block description sections of this document.

See **Table 0-3** for names and versions of the referenced documents throughout the Device User Guide.

Table 0-3 Document References

User Guide	Version	Document Order Number
HCS12 CPU Reference Manual	V02	S12CPUV2/D
HCS12 Module Mapping Control (MMC) Block Guide	V04	S12MMCV4/D
HCS12 Multiplexed External Bus Interface (MEBI) Block Guide	V03	S12MEBIV3/D
HCS12 Interrupt (INT) Block Guide	V01	S12INTV1/D
HCS12 Background Debug Module (BDM) Block Guide	V04	S12BDMV4/D
HCS12 Breakpoint (BKP) Block Guide	V01	S12BKPV1/D
Clock and Reset Generator (CRG) Block User Guide	V04	S12CRGV4/D
Oscillator (OSC) Block User Guide	V02	S12OSCV2/D
Enhanced Capture Timer 16 Bit 8 Channel (ECT_16B8C) Block User Guide	V01	S12ECT16B8CV1/D
Analog to Digital Converter 10 Bit 8 Channel (ATD_10B8C) Block User Guide	V02	S12ATD10B8CV2/D
Inter IC Bus (IIC) Block User Guide	V02	S12IICV2/D
Asynchronous Serial Interface (SCI) Block User Guide	V02	S12SCIV2/D
Serial Peripheral Interface (SPI) Block User Guide	V02	S12SPIV2/D
Pulse Width Modulator 8 Bit 8 Channel (PWM_8B8C) Block User Guide	V01	S12PWM8B8CV1/D
128K Byte Flash (FTS128K) Block User Guide	V02	S12FTS128KV2/D
2K Byte EEPROM (EETS2K) Block User Guide	V01	S12EETS2KV1/D
Byte Level Data Link Controller -J1850 (BDLC) Block User Guide	V01	S12BDLCV1/D
Motorola Scalable CAN (MSCAN) Block User Guide	V02	S12MSCANV2/D
Voltage Regulator (VREG) Block User Guide	V01	S12VREGV1/D
Port Integration Module (PIM_9DTB128) Block User Guide	V02	S12DTB128PIMV2/D
Byteflight (BF) Block User Guide	V01	S12BFV1/D

Section 1 Introduction

1.1 Overview

The MC9S12DT128 microcontroller unit (MCU) is a 16-bit device composed of standard on-chip peripherals including a 16-bit central processing unit (HCS12 CPU), 128K bytes of Flash EEPROM, 8K bytes of RAM, 2K bytes of EEPROM, two asynchronous serial communications interfaces (SCI), two serial peripheral interfaces (SPI), an 8-channel IC/OC enhanced capture timer, two 8-channel, 10-bit analog-to-digital converters (ADC), an 8-channel pulse-width modulator (PWM), a digital Byte Data Link Controller (BDLC), 29 discrete digital I/O channels (Port A, Port B, Port K and Port E), 20 discrete digital I/O lines with interrupt and wakeup capability, three CAN 2.0 A, B software compatible modules (MSCAN12), a Byteflight module and an Inter-IC Bus. The MC9S12DT128 has full 16-bit data paths throughout. However, the external bus can operate in an 8-bit narrow mode so single 8-bit wide memory can be interfaced for lower cost systems. The inclusion of a PLL circuit allows power consumption and performance to be adjusted to suit operational requirements.

1.2 Features

- HCS12 Core
 - 16-bit HCS12 CPU
 - i. Upward compatible with M68HC11 instruction set
 - ii. Interrupt stacking and programmer's model identical to M68HC11
 - iii. 20-bit ALU
 - iv. Instruction queue
 - v. Enhanced indexed addressing
 - MEBI (Multiplexed External Bus Interface)
 - MMC (Module Mapping Control)
 - INT (Interrupt control)
 - BKP (Breakpoints)
 - BDM (Background Debug Module)
- CRG (Clock and Reset Generator)
 - Choice of low current Colpitts oscillator or standard Pierce Oscillator
 - PLL
 - COP watchdog
 - real time interrupt
 - clock monitor
- 8-bit and 4-bit ports with interrupt functionality

- Digital filtering
- Programmable rising or falling edge trigger
- Memory
 - 128K Flash EEPROM
 - 2K byte EEPROM
 - 8K byte RAM
- Two 8-channel Analog-to-Digital Converters
 - 10-bit resolution
 - External conversion trigger capability
- Three 1M bit per second, CAN 2.0 A, B software compatible modules
 - Five receive and three transmit buffers
 - Flexible identifier filter programmable as 2 x 32 bit, 4 x 16 bit or 8 x 8 bit
 - Four separate interrupt channels for Rx, Tx, error and wake-up
 - Low-pass filter wake-up function
 - Loop-back for self test operation
- Enhanced Capture Timer
 - 16-bit main counter with 7-bit prescaler
 - 8 programmable input capture or output compare channels
 - Four 8-bit or two 16-bit pulse accumulators
- 8 PWM channels
 - Programmable period and duty cycle
 - 8-bit 8-channel or 16-bit 4-channel
 - Separate control for each pulse width and duty cycle
 - Center-aligned or left-aligned outputs
 - Programmable clock select logic with a wide range of frequencies
 - Fast emergency shutdown input
 - Usable as interrupt inputs
- Serial interfaces
 - Two asynchronous Serial Communications Interfaces (SCI)
 - Two Synchronous Serial Peripheral Interface (SPI)
 - Byteflight
- Byte Data Link Controller (BDLC)

- SAE J1850 Class B Data Communications Network Interface
 - Compatible and ISO Compatible for Low-Speed (<125 Kbps) Serial Data Communications in Automotive Applications
- Inter-IC Bus (IIC)
 - Compatible with I2C Bus standard
 - Multi-master operation
 - Software programmable for one of 256 different serial clock frequencies
- 112-Pin LQFP and 80-Pin QFP package options
 - I/O lines with 5V input and drive capability
 - 5V A/D converter inputs
 - Operation at 50MHz equivalent to 25MHz Bus Speed
 - Development support
 - Single-wire background debug™ mode
 - On-chip hardware breakpoints

1.3 Modes of Operation

User modes

- Normal and Emulation Operating Modes
 - Normal Single-Chip Mode
 - Normal Expanded Wide Mode
 - Normal Expanded Narrow Mode
 - Emulation Expanded Wide Mode
 - Emulation Expanded Narrow Mode
- Special Operating Modes
 - Special Single-Chip Mode with active Background Debug Mode
 - Special Test Mode (**Freescale use only**)
 - Special Peripheral Mode (**Freescale use only**)

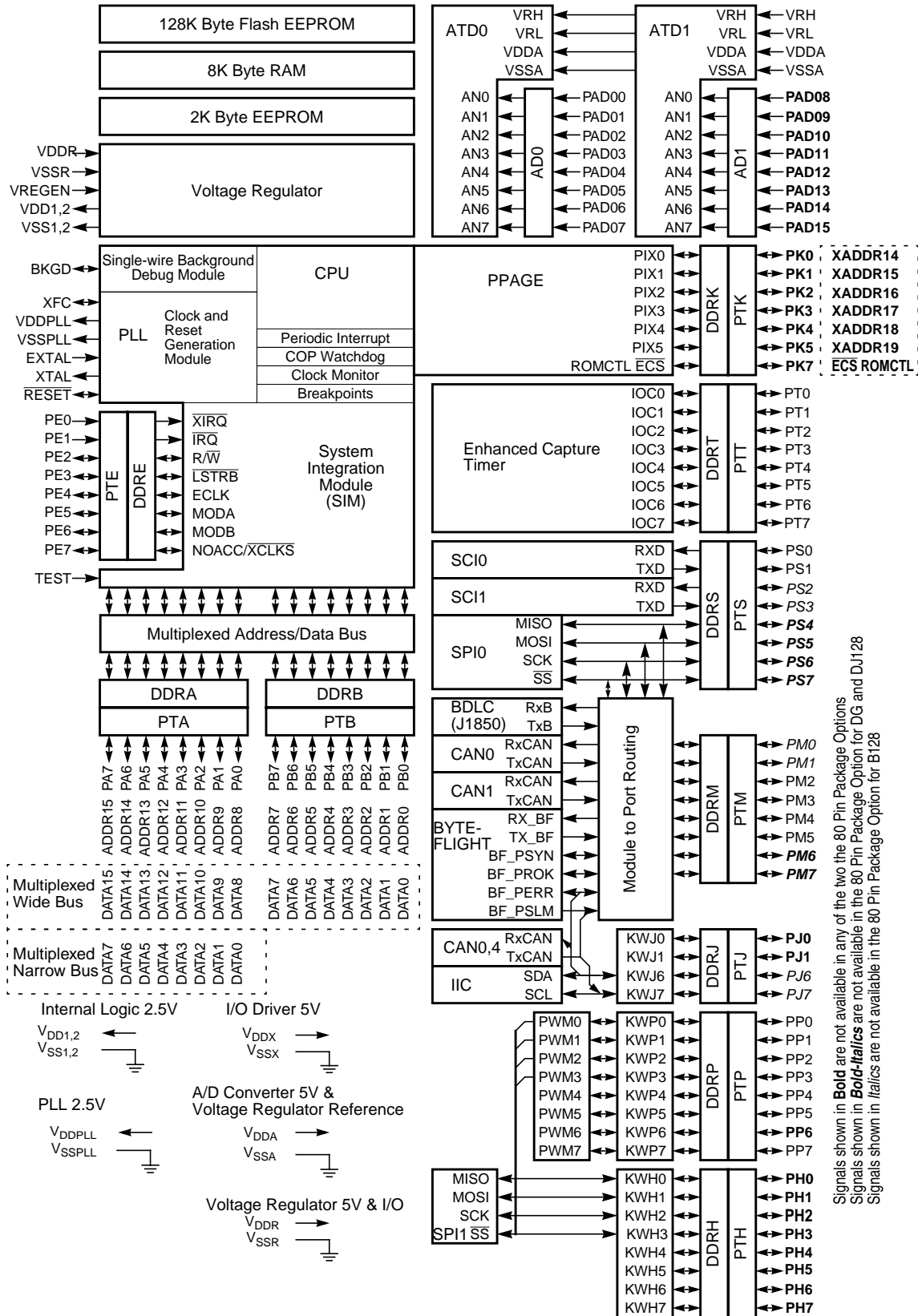
Low power modes

- Stop Mode
- Pseudo Stop Mode
- Wait Mode

1.4 Block Diagram

Figure 1-1 shows a block diagram of the MC9S12DT128 device.

Figure 1-1 MC9S12DT128 Block Diagram



1.5 Device Memory Map

(Table 1-1) and (Figure 1-2) show the device memory map of the MC9S12DT128 after reset. Note that after reset the EEPROM (\$0000 – \$07FF) is hidden by the register space (\$0000 - \$03FF) and the RAM (\$0000 - \$1FFF). The bottom 1K Bytes of RAM (\$0000 - \$03FF) are hidden by the register space.

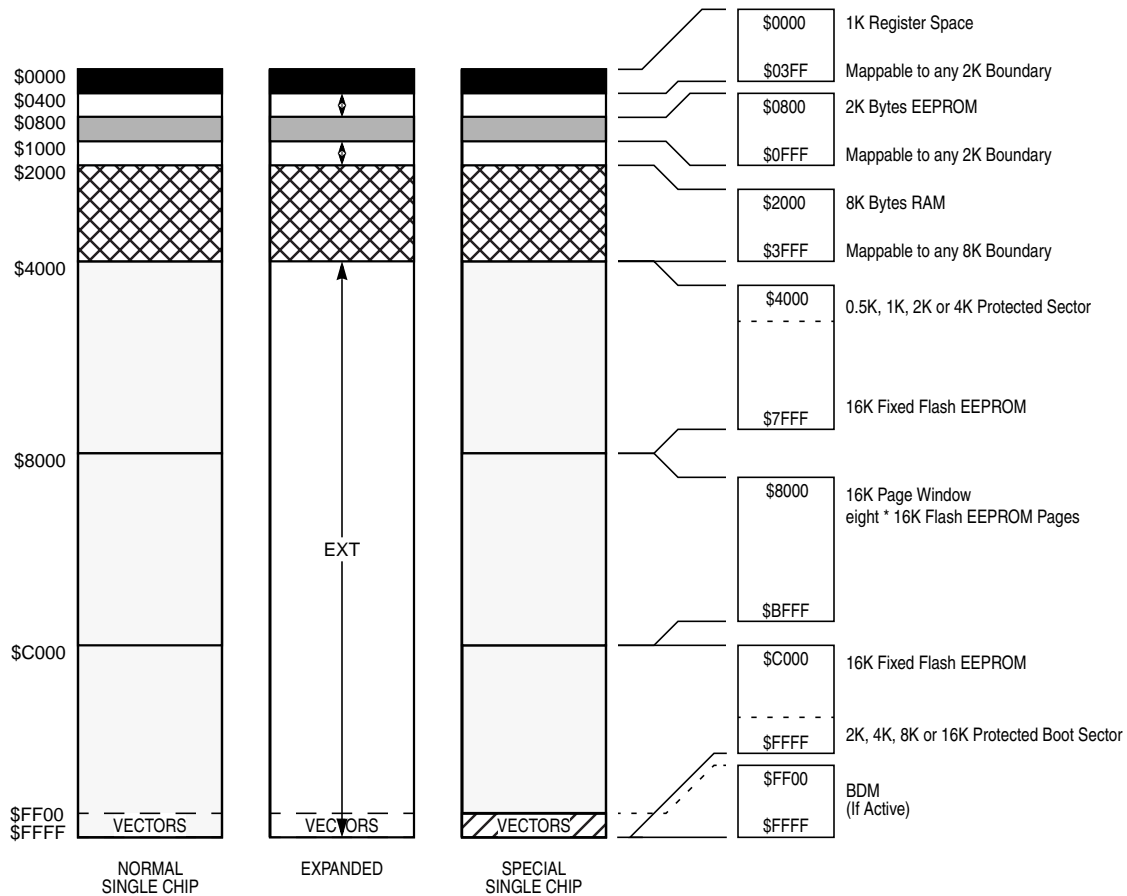
Table 1-1 Device Memory Map

Address	Module	Size (Bytes)
\$0000 – \$0017	CORE (Ports A, B, E, Modes, Inits, Test)	24
\$0018 – \$0019	Reserved	2
\$001A – \$001B	Device ID register (PARTID)	2
\$001C – \$001F	CORE (MEMSIZ, IRQ, HPRIO)	4
\$0020 – \$0027	Reserved	8
\$0028 – \$002F	CORE (Background Debug Module)	8
\$0030 – \$0033	CORE (PPAGE, Port K)	4
\$0034 – \$003F	Clock and Reset Generator (PLL, RTI, COP)	12
\$0040 – \$007F	Enhanced Capture Timer 16-bit 8 channels	64
\$0080 – \$009F	Analog to Digital Converter 10-bit 8 channels (ATD0)	32
\$00A0 – \$00C7	Pulse Width Modulator 8-bit 8 channels (PWM)	40
\$00C8 – \$00CF	Serial Communications Interface (SCI0)	8
\$00D0 – \$00D7	Serial Communications Interface (SCI1)	8
\$00D8 – \$00DF	Serial Peripheral Interface (SPI0)	8
\$00E0 – \$00E7	Inter IC Bus	8
\$00E8 – \$00EF	Byte Level Data Link Controller (BDLC)	8
\$00F0 – \$00F7	Serial Peripheral Interface (SPI1)	8
\$00F8 – \$00FF	Reserved	8
\$0100 – \$010F	Flash Control Register	16
\$0110 – \$011B	EEPROM Control Register	12
\$011C – \$011F	Reserved	4
\$0120 – \$013F	Analog to Digital Converter 10-bit 8 channels (ATD1)	32
\$0140 – \$017F	Motorola Scalable CAN (CAN0)	64
\$0180 – \$01BF	Motorola Scalable CAN (CAN1)	64
\$01C0 – \$01FF	Reserved	64
\$0200 – \$023F	Reserved	64
\$0240 – \$027F	Port Integration Module (PIM)	64
\$0280 – \$02BF	Motorola Scalable CAN (CAN4)	64
\$02C0 – \$02FF	Reserved	64
\$0300 – \$035F	Byteflight (BF)	96
\$0360 – \$03FF	Reserved	160
\$0000 – \$07FF	EEPROM array	2048
\$0000 – \$1FFF	RAM array	8192
\$4000 – \$7FFF	Fixed Flash EEPROM array incl. 0.5K, 1K, 2K or 4K Protected Sector at start	16384
\$8000 – \$BFFF	Flash EEPROM Page Window	16384

Table 1-1 Device Memory Map

Address	Module	Size (Bytes)
\$C000 – \$FFFF	Fixed Flash EEPROM array incl. 0.5K, 1K, 2K or 4K Protected Sector at end and 256 bytes of Vector Space at \$FF80 – \$FFFF	16384

Figure 1-2 MC9S12DT128 Memory Map



The address does not show the map after reset, but a useful map. After reset the map is:
 \$0000 – \$03FF: Register Space
 \$0000 – \$1FFF: 8K RAM
 \$0000 – \$07FF: 2K EEPROM (not visible)

1.5.1 Detailed Register Map

\$0000 - \$000F

MEBI map 1 of 3 (HCS12 Multiplexed External Bus Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0000	PORTA	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$0001	PORTB	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$0002	DDRA	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$0003	DDRB	Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$0004	Reserved	Read: Write:	0	0	0	0	0	0	0	0
\$0005	Reserved	Read: Write:	0	0	0	0	0	0	0	0
\$0006	Reserved	Read: Write:	0	0	0	0	0	0	0	0
\$0007	Reserved	Read: Write:	0	0	0	0	0	0	0	0
\$0008	PORTE	Read: Write:	Bit 7	6	5	4	3	2	Bit 1	Bit 0
\$0009	DDRE	Read: Write:	Bit 7	6	5	4	3	Bit 2	0	0
\$000A	PEAR	Read: Write:	NOACCE	0	PIPOE	NECLK	LSTRE	RDWE	0	0
\$000B	MODE	Read: Write:	MODC	MODB	MODA	0	IVIS	0	EMK	EME
\$000C	PUCR	Read: Write:	PUPKE	0	0	PUPEE	0	0	PUPBE	PUPAE
\$000D	RDRIV	Read: Write:	RDPK	0	0	RDPE	0	0	RDPB	RDPA
\$000E	EBICTL	Read: Write:	0	0	0	0	0	0	0	ESTR
\$000F	Reserved	Read: Write:	0	0	0	0	0	0	0	0

\$0010 - \$0014

MMC map 1 of 4 (HCS12 Module Mapping Control)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0010	INITRM	Read: Write:	RAM15	RAM14	RAM13	RAM12	RAM11	0	0	RAMHAL
\$0011	INITRG	Read: Write:	0	REG14	REG13	REG12	REG11	0	0	0
\$0012	INITEE	Read: Write:	EE15	EE14	EE13	EE12	EE11	0	0	EEON
\$0013	MISC	Read: Write:	0	0	0	0	EXSTR1	EXSTR0	ROMHM	ROMON
\$0014	Reserved	Read: Write:	0	0	0	0	0	0	0	0

\$0015 - \$0016**INT map 1 of 2 (HCS12 Interrupt)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0015	ITCR	Read: 0	0	0	WRINT	ADR3	ADR2	ADR1	ADR0
		Write:							
\$0016	ITEST	Read: INTE	INTC	INTA	INT8	INT6	INT4	INT2	INT0
		Write:							

\$0017 - \$0017**MMC map 2 of 4 (HCS12 Module Mapping Control)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0017	MTST1 Test Only	Read: Bit 7	6	5	4	3	2	1	Bit 0
		Write:							

\$0018 - \$0019**Reserved**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0018 - \$0019	Reserved	Read: 0	0	0	0	0	0	0	0
		Write:							

\$001A - \$001B**Device ID Register ((Table 1-3))**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$001A	PARTIDH	Read: ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
		Write:							
\$001B	PARTIDL	Read: ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		Write:							

**\$001C - \$001D
1-4))****MMC map 3 of 4 (HCS12 Module Mapping Control, (Table**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$001C	MEMSIZ0	Read: reg_sw0	0	eep_sw1	eep_sw0	0	ram_sw2	ram_sw1	ram_sw0
		Write:							
\$001D	MEMSIZ1	Read: rom_sw1	rom_sw0	0	0	0	0	pag_sw1	pag_sw0
		Write:							

\$001E - \$001E**MEBI map 2 of 3 (HCS12 Multiplexed External Bus Interface)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$001E	INTCR	Read: IRQE	IRQEN	0	0	0	0	0	0
		Write:							

\$001F - \$001F

INT map 2 of 2 (HCS12 Interrupt)

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$001F	HPRIO	Write:	PSEL7	PSEL6	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0

\$0020 - \$0027

Reserved

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0020 - \$0027	Reserved	Write:	0	0	0	0	0	0	0	0

\$0028 - \$002F

BKP (HCS12 Breakpoint)

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0028	BKPCT0	Write:	BKEN	BKFULL	BKBDM	BKTAG	0	0	0	0
\$0029	BKPCT1	Write:	BK0MBH	BK0MBL	BK1MBH	BK1MBL	BK0RWE	BK0RW	BK1RWE	BK1RW
\$002A	BKP0X	Write:	0	0	BK0V5	BK0V4	BK0V3	BK0V2	BK0V1	BK0V0
\$002B	BKP0H	Write:	Bit 15	14	13	12	11	10	9	Bit 8
\$002C	BKP0L	Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$002D	BKP1X	Write:	0	0	BK1V5	BK1V4	BK1V3	BK1V2	BK1V1	BK1V0
\$002E	BKP1H	Write:	Bit 15	14	13	12	11	10	9	Bit 8
\$002F	BKP1L	Write:	Bit 7	6	5	4	3	2	1	Bit 0

\$0030 - \$0031

MMC map 4 of 4 (HCS12 Module Mapping Control)

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0030	PPAGE	Write:	0	0	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
\$0031	Reserved	Write:	0	0	0	0	0	0	0	0

\$0032 - \$0033

MEBI map 3 of 3 (HCS12 Multiplexed External Bus Interface)

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0032	PORTK	Write:	Bit 7	6	5	4	3	2	1	Bit 0
\$0033	DDRK	Write:	Bit 7	6	5	4	3	2	1	Bit 0

\$0034 - \$003F**CRG (Clock and Reset Generator)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0034	SYNR	Read:	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
		Write:								
\$0035	REFDV	Read:	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0
		Write:								
\$0036	CTFLG TEST ONLY	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0037	CRGFLG	Read:	RTIF	PORF	0	LOCKIF	LOCK	TRACK	SCMIF	SCM
		Write:								
\$0038	CRGINT	Read:	RTIE	0	0	LOCKIE	0	0	SCMIE	0
		Write:								
\$0039	CLKSEL	Read:	PLLSEL	PSTP	SYSWAI	ROAWAI	PLLWAI	CWAI	RTIWAI	COPWAI
		Write:								
\$003A	PLLCTL	Read:	CME	PLLON	AUTO	ACQ	0	PRE	PCE	SCME
		Write:								
\$003B	RTICTL	Read:	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
		Write:								
\$003C	COPCTL	Read:	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
		Write:								
\$003D	FORBYP TEST ONLY	Read:	0	0	0	0	0	0	0	0
		Write:								
\$003E	CTCTL TEST ONLY	Read:	0	0	0	0	0	0	0	0
		Write:								
\$003F	ARMCOP	Read:	0	0	0	0	0	0	0	0
		Write:	Bit 7	6	5	4	3	2	1	Bit 0

\$0040 - \$007F**ECT (Enhanced Capture Timer 16 Bit 8 Channels)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0040	TIOS	Read:	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
		Write:								
\$0041	CFORC	Read:	0	0	0	0	0	0	0	0
		Write:	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
\$0042	OC7M	Read:	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
		Write:								
\$0043	OC7D	Read:	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
		Write:								
\$0044	TCNT (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0045	TCNT (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0046	TSCR1	Read:	TEN	TSWAI	TSFRZ	TFFCA	0	0	0	0
		Write:								
\$0047	TTOV	Read:	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
		Write:								
\$0048	TCTL1	Read:	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
		Write:								
\$0049	TCTL2	Read:	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		Write:								

\$0040 - \$007F

ECT (Enhanced Capture Timer 16 Bit 8 Channels)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$004A	TCTL3	Read:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
		Write:								
\$004B	TCTL4	Read:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		Write:								
\$004C	TIE	Read:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
		Write:								
\$004D	TSCR2	Read:	TOI	0	0	0	TCRE	PR2	PR1	PR0
		Write:								
\$004E	TFLG1	Read:	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
		Write:								
\$004F	TFLG2	Read:	TOF	0	0	0	0	0	0	0
		Write:								
\$0050	TC0 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0051	TC0 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0052	TC1 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0053	TC1 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0054	TC2 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0055	TC2 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0056	TC3 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0057	TC3 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0058	TC4 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0059	TC4 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$005A	TC5 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$005B	TC5 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$005C	TC6 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$005D	TC6 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$005E	TC7 (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$005F	TC7 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0060	PACTL	Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
		Write:								
\$0061	PAFLG	Read:	0	0	0	0	0	0	PAOVF	PAIF
		Write:								
\$0062	PACN3 (hi)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$0040 - \$007F**ECT (Enhanced Capture Timer 16 Bit 8 Channels)**

Address	Name	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0063	PACN2 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0064	PACN1 (hi)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0065	PACN0 (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0066	MCCTL	Read:	MCZI	MODMC	RDMCL	0	0	MCEN	MCPR1	MCPR0
		Write:				ICLAT	FLMC			
\$0067	MCFLG	Read:	MCZF	0	0	0	POLF3	POLF2	POLF1	POLF0
		Write:								
\$0068	ICPAR	Read:	0	0	0	0	PA3EN	PA2EN	PA1EN	PA0EN
		Write:								
\$0069	DLYCT	Read:	0	0	0	0	0	0	DLY1	DLY0
		Write:								
\$006A	ICOVW	Read:	NOVW7	NOVW6	NOVW5	NOVW4	NOVW3	NOVW2	NOVW1	NOVW0
		Write:								
\$006B	ICSYS	Read:	SH37	SH26	SH15	SH04	TFMOD	PACMX	BUFEN	LATQ
		Write:								
\$006C	Reserved	Read:								
		Write:								
\$006D	TIMTST	Read:	0	0	0	0	0	0	TCBYP	0
	Test Only	Write:								
\$006E	Reserved	Read:								
		Write:								
\$006F	Reserved	Read:								
		Write:								
\$0070	PBCTL	Read:	0	PBEN	0	0	0	0	PBOVI	0
		Write:								
\$0071	PBFLG	Read:	0	0	0	0	0	0	PBOVF	0
		Write:								
\$0072	PA3H	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0073	PA2H	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0074	PA1H	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0075	PA0H	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0076	MCCNT (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0077	MCCNT (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$0078	TC0H (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$0079	TC0H (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$007A	TC1H (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$007B	TC1H (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$0040 - \$007F

ECT (Enhanced Capture Timer 16 Bit 8 Channels)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$007C	TC2H (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$007D	TC2H (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$007E	TC3H (hi)	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$007F	TC3H (lo)	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$0080 - \$009F

ATD0 (Analog to Digital Converter 10 Bit 8 Channel)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0080	ATD0CTL0	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0081	ATD0CTL1	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0082	ATD0CTL2	Read:	ADPU	AFFC	AWAI	ETRIGLE	ETRIGP	ETRIG	ASCIE	ASCIF
		Write:								
\$0083	ATD0CTL3	Read:	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
		Write:								
\$0084	ATD0CTL4	Read:	SRES8	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
\$0085	ATD0CTL5	Read:	DJM	DSGN	SCAN	MULT	0	CC	CB	CA
		Write:								
\$0086	ATD0STAT0	Read:	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
		Write:								
\$0087	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0088	ATD0TEST0	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0089	ATD0TEST1	Read:	0	0	0	0	0	0	0	SC
		Write:								
\$008A	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$008B	ATD0STAT1	Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		Write:								
\$008C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$008D	ATD0DIEN	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$008E	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$008F	PORTAD0	Read:	Bit7	6	5	4	3	2	1	BIT 0
		Write:								
\$0090	ATD0DR0H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0091	ATD0DR0L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								

\$0080 - \$009F**ATD0 (Analog to Digital Converter 10 Bit 8 Channel)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0092	ATD0DR1H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0093	ATD0DR1L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0094	ATD0DR2H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0095	ATD0DR2L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0096	ATD0DR3H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0097	ATD0DR3L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0098	ATD0DR4H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0099	ATD0DR4L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$009A	ATD0DR5H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$009B	ATD0DR5L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$009C	ATD0DR6H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$009D	ATD0DR6L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$009E	ATD0DR7H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$009F	ATD0DR7L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								

\$00A0 - \$00C7**PWM (Pulse Width Modulator 8 Bit 8 Channel)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00A0	PWME	Read:	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
		Write:								
\$00A1	PWMPOL	Read:	PPOL7	PPOL6	PPOL5	PPOL4	PPOL3	PPOL2	PPOL1	PPOL0
		Write:								
\$00A2	PWMCLK	Read:	PCLK7	PCLK6	PCLK5	PCLK4	PCLK3	PCLK2	PCLK1	PCLK0
		Write:								
\$00A3	PWMPRCLK	Read:	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
		Write:								
\$00A4	PWMCAE	Read:	CAE7	CAE6	CAE5	CAE4	CAE3	CAE2	CAE1	CAE0
		Write:								
\$00A5	PWMCTL	Read:	CON67	CON45	CON23	CON01	PSWAI	PFRZ	0	0
		Write:								
\$00A6	PWMTST Test Only	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00A7	PWMPRSC Test Only	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00A8	PWMSCLA	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$00A0 - \$00C7

PWM (Pulse Width Modulator 8 Bit 8 Channel)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00A9	PWMSCLB	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00AA	PWMSCNTA Test Only	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00AB	PWMSCNTB Test Only	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00AC	PWMCNT0	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00AD	PWMCNT1	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00AE	PWMCNT2	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00AF	PWMCNT3	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00B0	PWMCNT4	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00B1	PWMCNT5	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00B2	PWMCNT6	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00B3	PWMCNT7	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:	0	0	0	0	0	0	0	0
\$00B4	PWMPER0	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00B5	PWMPER1	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00B6	PWMPER2	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00B7	PWMPER3	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00B8	PWMPER4	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00B9	PWMPER5	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BA	PWMPER6	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BB	PWMPER7	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BC	PWMDTY0	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BD	PWMDTY1	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BE	PWMDTY2	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00BF	PWMDTY3	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00C0	PWMDTY4	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00C1	PWMDTY5	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$00A0 - \$00C7
PWM (Pulse Width Modulator 8 Bit 8 Channel)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00C2	PWMDTY6	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00C3	PWMDTY7	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$00C4	PWMSDN	Read:	PWMIF	PWMIE	PWMRSTRT	PWMLVL	0	PWM7IN	PWM7INL	PWM7ENA
		Write:								
\$00C5	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00C6	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00C7	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$00C8 - \$00CF
SCI0 (Asynchronous Serial Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00C8	SCI0BDH	Read:	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
\$00C9	SCI0BDL	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
\$00CA	SCI0CR1	Read:	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		Write:								
\$00CB	SCI0CR2	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
\$00CC	SCI0SR1	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								
\$00CD	SCI0SR2	Read:	0	0	0	0	0	BRK13	TXDIR	RAF
		Write:								
\$00CE	SCI0DRH	Read:	R8	T8	0	0	0	0	0	0
		Write:								
\$00CF	SCI0DRL	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0

\$00D0 - \$00D7
SCI1 (Asynchronous Serial Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00D0	SCI1BDH	Read:	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
		Write:								
\$00D1	SCI1BDL	Read:	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		Write:								
\$00D2	SCI1CR1	Read:	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		Write:								
\$00D3	SCI1CR2	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
\$00D4	SCI1SR1	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		Write:								

\$00D0 - \$00D7

SCI1 (Asynchronous Serial Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00D5	SCI1SR2	Read:	0	0	0	0	0	BRK13	TXDIR	RAF
		Write:								
\$00D6	SCI1DRH	Read:	R8	T8	0	0	0	0	0	0
		Write:								
\$00D7	SCI1DRL	Read:	R7	R6	R5	R4	R3	R2	R1	R0
		Write:	T7	T6	T5	T4	T3	T2	T1	T0

\$00D8 - \$00DF

SPI0 (Serial Peripheral Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00D8	SPI0CR1	Read:	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		Write:								
\$00D9	SPI0CR2	Read:	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		Write:								
\$00DA	SPI0BR	Read:	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		Write:								
\$00DB	SPI0SR	Read:	SPIF	0	SPTEF	MODF	0	0	0	0
		Write:								
\$00DC	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00DD	SPI0DR	Read:	Bit7	6	5	4	3	2	1	Bit0
		Write:								
\$00DE	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00DF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$00E0 - \$00E7

IIC (Inter IC Bus)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00E0	IBAD	Read:	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	0
		Write:								
\$00E1	IBFD	Read:	IBC7	IBC6	IBC5	IBC4	IBC3	IBC2	IBC1	IBC0
		Write:								
\$00E2	IBCR	Read:	IBEN	IBIE	MS/ \overline{SL}	TX/ \overline{RX}	TXAK	0	0	IBSWAI
		Write:						RSTA		
\$00E3	IBSR	Read:	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
		Write:								
\$00E4	IBDR	Read:	D7	D6	D5	D4	D3	D2	D1	D0
		Write:								
\$00E5	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00E6	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00E7	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$00E8 - \$00EF

BDLC (Byte Level Data Link Controller J1850)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00E8	DLCBCR1	Read:	IMSG	CLKS	0	0	0	0	IE	WCM
		Write:								
\$00E9	DLCBSVR	Read:	0	0	I3	I2	I1	I0	0	0
		Write:								
\$00EA	DLCBCR2	Read:	SMRST	DLOOP	RX4XE	NBFS	TEOD	TSIFR	TMIFR1	TMIFR0
		Write:								
\$00EB	DLCBDR	Read:	D7	D6	D5	D4	D3	D2	D1	D0
		Write:								
\$00EC	DLCBARD	Read:	0	RXPOL	0	0	BO3	BO2	BO1	BO0
		Write:								
\$00ED	DLCBRSR	Read:	0	0	R5	R4	R3	R2	R1	R0
		Write:								
\$00EE	DLCSCR	Read:	0	0	0	BDLCE	0	0	0	0
		Write:								
\$00EF	DLCBSTAT	Read:	0	0	0	0	0	0	0	IDLE
		Write:								

\$00F0 - \$00F7

SPI1 (Serial Peripheral Interface)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00F0	SPI1CR1	Read:	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		Write:								
\$00F1	SPI1CR2	Read:	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		Write:								
\$00F2	SPI1BR	Read:	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		Write:								
\$00F3	SPI1SR	Read:	SPIF	0	SPTEF	MODF	0	0	0	0
		Write:								
\$00F4	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00F5	SPI1DR	Read:	Bit7	6	5	4	3	2	1	Bit0
		Write:								
\$00F6	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$00F7	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$00F8 - \$00FF

Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$00F8 - \$00FF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0100 - \$010F

Flash Control Register (fts128k2)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0100	FCLKDIV	Read:	FDIVLD	PRDIV8	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
		Write:								
\$0101	FSEC	Read:	KEYEN1	KEYEN0	NV5	NV4	NV3	NV2	SEC1	SEC0
		Write:								
\$0102	FTSTMOD	Read:	0	0	0	WRALL	0	0	0	0
		Write:								
\$0103	FCNFG	Read:	CBEIE	CCIE	KEYACC	0	0	0	BKSEL1	BKSEL0
		Write:								
\$0104	FPROT	Read:	FPOPEN	NV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
		Write:								
\$0105	FSTAT	Read:	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
		Write:								
\$0106	FCMD	Read:	0	CMDB6	CMDB5	0	0	CMDB2	0	CMDB0
		Write:								
\$0107	Reserved for Factory Test	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0108	FADDRHI	Read:	0	Bit 14	13	12	11	10	9	Bit 8
		Write:								
\$0109	FADDRLO	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$010A	FDATAHI	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$010B	FDATALO	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$010C - \$010F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0110 - \$011B

EEPROM Control Register (eets2k)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0110	ECLKDIV	Read:	EDIVLD	PRDIV8	EDIV5	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
		Write:								
\$0111	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0112	Reserved for Factory Test	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0113	ECNFG	Read:	CBEIE	CCIE	0	0	0	0	0	0
		Write:								
\$0114	EPROT	Read:	EPOPEN	NV6	NV5	NV4	EPDIS	EP2	EP1	EP0
		Write:								
\$0115	ESTAT	Read:	CBEIF	CCIF	PVIOL	ACCERR	0	BLANK	0	0
		Write:								
\$0116	ECMD	Read:	0	CMDB6	CMDB5	0	0	CMDB2	0	CMDB0
		Write:								
\$0117	Reserved for Factory Test	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0118	EADDRHI	Read:	0	0	0	0	0	0	Bit 9	Bit 8
		Write:								

\$0110 - \$011B**EEPROM Control Register (eets2k)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0119	EADDRLO	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$011A	EDATAHI	Read:	Bit 15	14	13	12	11	10	9	Bit 8
		Write:								
\$011B	EDATALO	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								

\$011C - \$011F**Reserved for RAM Control Register**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$011C - \$011F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0120 - \$013F**ATD1 (Analog to Digital Converter 10 Bit 8 Channel)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0120	ATD1CTL0	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0121	ATD1CTL1	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0122	ATD1CTL2	Read:	ADPU	AFFC	AWAI	ETRIGLE	ETRIGP	ETRIG	ASCIE	ASCIF
		Write:								
\$0123	ATD1CTL3	Read:	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
		Write:								
\$0124	ATD1CTL4	Read:	SRES8	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
\$0125	ATD1CTL5	Read:	DJM	DSGN	SCAN	MULT	0	CC	CB	CA
		Write:								
\$0126	ATD1STAT0	Read:	SCF	0	ETORF	FIFOR	0	CC2	CC1	CC0
		Write:								
\$0127	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0128	ATD1TEST0	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0129	ATD1TEST1	Read:	0	0	0	0	0	0	0	SC
		Write:								
\$012A	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$012B	ATD1STAT1	Read:	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		Write:								
\$012C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$012D	ATD1DIEN	Read:	Bit 7	6	5	4	3	2	1	Bit 0
		Write:								
\$012E	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$012F	PORTAD1	Read:	Bit7	6	5	4	3	2	1	BIT 0
		Write:								

\$0120 - \$013F

ATD1 (Analog to Digital Converter 10 Bit 8 Channel)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0130	ATD1DR0H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0131	ATD1DR0L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0132	ATD1DR1H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0133	ATD1DR1L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0134	ATD1DR2H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0135	ATD1DR2L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0136	ATD1DR3H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0137	ATD1DR3L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$0138	ATD1DR4H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$0139	ATD1DR4L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$013A	ATD1DR5H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$013B	ATD1DR5L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$013C	ATD1DR6H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$013D	ATD1DR6L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								
\$013E	ATD1DR7H	Read:	Bit15	14	13	12	11	10	9	Bit8
		Write:								
\$013F	ATD1DR7L	Read:	Bit7	Bit6	0	0	0	0	0	0
		Write:								

\$0140 - \$017F

CAN0 (Motorola Scalable CAN - MSCAN)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0140	CANOCTL0	Read:	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
		Write:								
\$0141	CANOCTL1	Read:	CANE	CLKSRC	LOOPB	LISTEN	0	WUPM	SLPAK	INITAK
		Write:								
\$0142	CAN0BTR0	Read:	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		Write:								
\$0143	CAN0BTR1	Read:	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
		Write:								
\$0144	CAN0RFLG	Read:	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
		Write:								
\$0145	CAN0RIER	Read:	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
		Write:								

\$0140 - \$017F**CAN0 (Motorola Scalable CAN - MSCAN)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0146	CAN0TFLG	Read:	0	0	0	0	0	TXE2	TXE1	TXE0
		Write:								
\$0147	CAN0TIER	Read:	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
		Write:								
\$0148	CAN0TARQ	Read:	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
		Write:								
\$0149	CAN0TAAK	Read:	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
		Write:								
\$014A	CAN0TBSEL	Read:	0	0	0	0	0	TX2	TX1	TX0
		Write:								
\$014B	CAN0IDAC	Read:	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
		Write:								
\$014C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$014D	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$014E	CAN0RXERR	Read:	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
		Write:								
\$014F	CAN0TXERR	Read:	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
		Write:								
\$0150 - \$0153	CAN0IDAR0 - CAN0IDAR3	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$0154 - \$0157	CAN0IDMR0 - CAN0IDMR3	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$0158 - \$015B	CAN0IDAR4 - CAN0IDAR7	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$015C - \$015F	CAN0IDMR4 - CAN0IDMR7	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$0160 - \$016F	CAN0RXFG	Read:	FOREGROUND RECEIVE BUFFER see (Table 1-2)							
		Write:								
\$0170 - \$017F	CAN0TXFG	Read:	FOREGROUND TRANSMIT BUFFER see (Table 1-2)							
		Write:								

Table 1-2 Detailed MSCAN Foreground Receive and Transmit Buffer Layout

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$xxx0	Extended ID	Read:	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
		Standard ID	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
\$xxx1	CANxRIDR0	Write:								
		Extended ID	ID20	ID19	ID18	SRR=1	IDE=1	ID17	ID16	ID15
\$xxx2	CANxRIDR1	Read:	ID2	ID1	ID0	RTR	IDE=0			
		Write:								
\$xxx3	CANxRIDR2	Read:	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
		Write:								
\$xxx4	CANxRIDR3	Read:	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
		Write:								
\$xxx4- \$xxxB	CANxRDSR0 - CANxRDSR7	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
		Write:								

Table 1-2 Detailed MSCAN Foreground Receive and Transmit Buffer Layout

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$xxxC	CANRxDLR	Read:					DLC3	DLC2	DLC1	DLC0
		Write:								
\$xxxD	Reserved	Read:								
		Write:								
\$xxxE	CANxRTSRH	Read:	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
		Write:								
\$xxxF	CANxRTSRL	Read:	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
		Write:								
\$xx10	Extended ID CANxTIDR0	Read:	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
		Write:								
\$xx10	Standard ID	Read:	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
		Write:								
\$xx11	Extended ID CANxTIDR1	Read:	ID20	ID19	ID18	SRR=1	IDE=1	ID17	ID16	ID15
		Write:								
\$xx11	Standard ID	Read:	ID2	ID1	ID0	RTR	IDE=0			
		Write:								
\$xx12	Extended ID CANxTIDR2	Read:	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
		Write:								
\$xx12	Standard ID	Read:								
		Write:								
\$xx13	Extended ID CANxTIDR3	Read:	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
		Write:								
\$xx13	Standard ID	Read:								
		Write:								
\$xx14- \$xx1B	CANxTDSR0 - CANxTDSR7	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
		Write:								
\$xx1C	CANxTDLR	Read:					DLC3	DLC2	DLC1	DLC0
		Write:								
\$xx1D	CONxTTBPR	Read:	PRI07	PRI06	PRI05	PRI04	PRI03	PRI02	PRI01	PRI00
		Write:								
\$xx1E	CANxTTSRH	Read:	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
		Write:								
\$xx1F	CANxTTSRL	Read:	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
		Write:								

\$0180 - \$01BF

CAN1 (Motorola Scalable CAN - MSCAN)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0180	CAN1CTL0	Read:	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
		Write:								
\$0181	CAN1CTL1	Read:	CANE	CLKSRC	LOOPB	LISTEN	0	WUPM	SLPAK	INITAK
		Write:								
\$0182	CAN1BTR0	Read:	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		Write:								
\$0183	CAN1BTR1	Read:	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
		Write:								
\$0184	CAN1RFLG	Read:	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
		Write:								

\$0180 - \$01BF

CAN1 (Motorola Scalable CAN - MSCAN)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0185	CAN1RIER	Read:	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
		Write:								
\$0186	CAN1TFLG	Read:	0	0	0	0	0	TXE2	TXE1	TXE0
		Write:								
\$0187	CAN1TIER	Read:	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
		Write:								
\$0188	CAN1TARQ	Read:	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
		Write:								
\$0189	CAN1TAAK	Read:	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
		Write:								
\$018A	CAN1TBSEL	Read:	0	0	0	0	0	TX2	TX1	TX0
		Write:								
\$018B	CAN1IDAC	Read:	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
		Write:								
\$018C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$018D	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$018E	CAN1RXERR	Read:	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
		Write:								
\$018F	CAN1TXERR	Read:	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
		Write:								
\$0190 - \$0193	CAN1IDAR0 - CAN1IDAR3	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$0194 - \$0197	CAN1IDMR0 - CAN1IDMR3	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$0198 - \$019B	CAN1IDAR4 - CAN1IDAR7	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$019C - \$019F	CAN1IDMR4 - CAN1IDMR7	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$01A0 - \$01AF	CAN0RXFG	Read:	FOREGROUND RECEIVE BUFFER see (Table 1-2)							
		Write:								
\$01B0 - \$01BF	CAN0TXFG	Read:	FOREGROUND TRANSMIT BUFFER see (Table 1-2)							
		Write:								

\$01C0 - \$01FF

Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$01C0 - \$01FF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0200 - \$023F

Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$020C - \$023F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0240 - \$027F

PIM (Port Integration Module)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0240	PTT	Read:	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
		Write:								
\$0241	PTIT	Read:	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
		Write:								
\$0242	DDRT	Read:	DDRT7	DDRT7	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
		Write:								
\$0243	RDRT	Read:	RDRT7	RDRT6	RDRT5	RDRT4	RDRT3	RDRT2	RDRT1	RDRT0
		Write:								
\$0244	PERT	Read:	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
		Write:								
\$0245	PPST	Read:	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
		Write:								
\$0246	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0247	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0248	PTS	Read:	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		Write:								
\$0249	PTIS	Read:	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		Write:								
\$024A	DDRS	Read:	DDRS7	DDRS7	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		Write:								
\$024B	RDRS	Read:	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
		Write:								
\$024C	PERS	Read:	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		Write:								
\$024D	PPSS	Read:	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		Write:								
\$024E	WOMS	Read:	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		Write:								
\$024F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0250	PTM	Read:	PTM7	PTM6	PTM5	PTM4	PTM3	PTM2	PTM1	PTM0
		Write:								
\$0251	PTIM	Read:	PTIM7	PTIM6	PTIM5	PTIM4	PTIM3	PTIM2	PTIM1	PTIM0
		Write:								
\$0252	DDRM	Read:	DDRM7	DDRM7	DDRM5	DDRM4	DDRM3	DDRM2	DDRM1	DDRM0
		Write:								
\$0253	RDRM	Read:	RDRM7	RDRM6	RDRM5	RDRM4	RDRM3	RDRM2	RDRM1	RDRM0
		Write:								
\$0254	PERM	Read:	PERM7	PERM6	PERM5	PERM4	PERM3	PERM2	PERM1	PERM0
		Write:								
\$0255	PPSM	Read:	PPSM7	PPSM6	PPSM5	PPSM4	PPSM3	PPSM2	PPSM1	PPSM0
		Write:								
\$0256	WOMM	Read:	WOMM7	WOMM6	WOMM5	WOMM4	WOMM3	WOMM2	WOMM1	WOMM0
		Write:								
\$0257	MODRR	Read:	0	0	MODRR5	MODRR4	MODRR3	MODRR2	MODRR1	MODRR0
		Write:								
\$0258	PTP	Read:	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
		Write:								

\$0240 - \$027F**PIM (Port Integration Module)**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0259	PTIP	Read:	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
		Write:								
\$025A	DDRP	Read:	DDRP7	DDRP6	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
		Write:								
\$025B	RDRP	Read:	RDRP7	RDRP6	RDRP5	RDRP4	RDRP3	RDRP2	RDRP1	RDRP0
		Write:								
\$025C	PERP	Read:	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
		Write:								
\$025D	PPSP	Read:	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSS0
		Write:								
\$025E	PIEP	Read:	PIEP7	PIEP6	PIEP5	PIEP4	PIEP3	PIEP2	PIEP1	PIEP0
		Write:								
\$025F	PIFP	Read:	PIFP7	PIFP6	PIFP5	PIFP4	PIFP3	PIFP2	PIFP1	PIFP0
		Write:								
\$0260	PTH	Read:	PTH7	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
		Write:								
\$0261	PTIH	Read:	PTIH7	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
		Write:								
\$0262	DDRH	Read:	DDRH7	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
		Write:								
\$0263	RDRH	Read:	RDRH7	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
		Write:								
\$0264	PERH	Read:	PERH7	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
		Write:								
\$0265	PPSH	Read:	PPSH7	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
		Write:								
\$0266	PIEH	Read:	PIEH7	PIEH6	PIEH5	PIEH4	PIEH3	PIEH2	PIEH1	PIEH0
		Write:								
\$0267	PIFH	Read:	PIFH7	PIFH6	PIFH5	PIFH4	PIFH3	PIFH2	PIFH1	PIFH0
		Write:								
\$0268	PTJ	Read:	PTJ7	PTJ6	0	0	0	0	PTJ1	PTJ0
		Write:								
\$0269	PTIJ	Read:	PTIJ7	PTIJ6	0	0	0	0	PTIJ1	PTIJ0
		Write:								
\$026A	DDRJ	Read:	DDRJ7	DDRJ6	0	0	0	0	DDRJ1	DDRJ0
		Write:								
\$026B	RDRJ	Read:	RDRJ7	RDRJ6	0	0	0	0	RDRJ1	RDRJ0
		Write:								
\$026C	PERJ	Read:	PERJ7	PERJ6	0	0	0	0	PERJ1	PERJ0
		Write:								
\$026D	PPSJ	Read:	PPSJ7	PPSJ6	0	0	0	0	PPSJ1	PPSJ0
		Write:								
\$026E	PIEJ	Read:	PIEJ7	PIEJ6	0	0	0	0	PIEJ1	PIEJ0
		Write:								
\$026F	PIFJ	Read:	PIFJ7	PIFJ6	0	0	0	0	PIFJ1	PIFJ0
		Write:								
\$0270 - \$027F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

\$0280 - \$02BF

CAN4 (Motorola Scalable CAN - MSCAN)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0280	CAN4CTL0	Read:	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
		Write:								
\$0281	CAN4CTL1	Read:	CANE	CLKSRC	LOOPB	LISTEN	0	WUPM	SLPAK	INITAK
		Write:								
\$0282	CAN4BTR0	Read:	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		Write:								
\$0283	CAN4BTR1	Read:	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
		Write:								
\$0284	CAN4RFLG	Read:	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
		Write:								
\$0285	CAN4RIER	Read:	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
		Write:								
\$0286	CAN4TFLG	Read:	0	0	0	0	0	TXE2	TXE1	TXE0
		Write:								
\$0287	CAN4TIER	Read:	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
		Write:								
\$0288	CAN4TARQ	Read:	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
		Write:								
\$0289	CAN4TAAK	Read:	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
		Write:								
\$028A	CAN4TBSEL	Read:	0	0	0	0	0	TX2	TX1	TX0
		Write:								
\$028B	CAN4IDAC	Read:	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
		Write:								
\$028C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$028D	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$028E	CAN4RXERR	Read:	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
		Write:								
\$028F	CAN4TXERR	Read:	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
		Write:								
\$0290 - \$0293	CAN0IDAR0 - CAN0IDAR3	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$0294 - \$0297	CAN0IDMR0 - CAN0IDMR3	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$0298 - \$029B	CAN0IDAR4 - CAN0IDAR7	Read:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		Write:								
\$029C - \$029F	CAN0IDMR4 - CAN0IDMR7	Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		Write:								
\$02A0 - \$02AF	CAN4RXFG	Read:	FOREGROUND RECEIVE BUFFER see (Table 1-2)							
		Write:								
\$02B0 - \$02BF	CAN4TXFG	Read:	FOREGROUND TRANSMIT BUFFER see (Table 1-2)							
		Write:								

\$02C0 - \$02FF

Reserved

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$02C0 - \$02FF	Reserved	0	0	0	0	0	0	0	0

\$0300 - \$035F

Byteflight

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0300	BFMCR	Read: INITRQ	Read: MASTER	Read: ALARM	Read: SLPK	Read: SLPRQ	Read: WPULSE	Read: SSWAI	Read: INITAK
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0301	BFFSIZR	Read: 0	Read: 0	Read: 0	Read: FSIZ4	Read: FSIZ3	Read: FSIZ2	Read: FSIZ1	Read: FSIZ0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0302	BFTCR1	Read: TWX0T7	Read: TWX0T6	Read: TWX0T5	Read: TWX0T4	Read: TWX0T3	Read: TWX0T2	Read: TWX0T1	Read: TWX0T0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0303	BFTCR2	Read: TWX0R7	Read: TWX0R6	Read: TWX0R5	Read: TWX0R4	Read: TWX0R3	Read: TWX0R2	Read: TWX0R1	Read: TWX0R0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0304	BFTCR3	Read: TWX0D7	Read: TWX0D6	Read: TWX0D5	Read: TWX0D4	Read: TWX0D3	Read: TWX0D2	Read: TWX0D1	Read: TWX0D0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0305	Reserved	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0306	BFRISR	Read: RCVFIF	Read: RXIF	Read: SYNAIF	Read: SYNIF	Read: SLMMIF	Read: 0	Read: XSYNIF	Read: OPTDF
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0307	BFGISR	Read: TXIF	Read: OVRNIF	Read: ERRIF	Read: SYNEIF	Read: SYNLIF	Read: ILLPIF	Read: LOCKIF	Read: WAKEIF
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0308	BFRIER	Read: RCVFIE	Read: RXIE	Read: SYNAIE	Read: SYNIE	Read: SLMMIE	Read: 0	Read: XSYNIE	Read: 0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0309	BFGIER	Read: TXIE	Read: OVRNIE	Read: ERRIE	Read: SYNEIE	Read: SYNLIE	Read: ILLPIE	Read: LOCKIE	Read: WAKEIE
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030A	BFRIVEC	Read: 0	Read: 0	Read: 0	Read: 0	Read: RIVEC3	Read: RIVEC2	Read: RIVEC1	Read: RIVEC0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030B	BFTIVEC	Read: 0	Read: 0	Read: 0	Read: 0	Read: TIVEC3	Read: TIVEC2	Read: TIVEC1	Read: TIVEC0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030C	BFFIDAC	Read: FIDAC7	Read: FIDAC6	Read: FIDAC5	Read: FIDAC4	Read: FIDAC3	Read: FIDAC2	Read: FIDAC1	Read: FIDAC0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030D	BFFIDMR	Read: FIDMR7	Read: FIDMR6	Read: FIDMR5	Read: FIDMR4	Read: FIDMR3	Read: FIDMR2	Read: FIDMR1	Read: FIDMR0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030E	BFMVR	Read: MVR7	Read: MVR6	Read: MVR5	Read: MVR4	Read: MVR3	Read: MVR2	Read: MVR1	Read: MVR0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$030F	Reserved	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0310	BFPCTLBF	Read: PMEREN	Read: 0	Read: PSLMEN	Read: PERREN	Read: PROKEN	Read: PSYNEN	Read: 0	Read: BFEN
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0311	Reserved	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0312	BFBUFLOCK	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: TXBUFL OCK	Read: RXBUFL OCK
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0313	Reserved	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0	Read: 0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:
\$0314	BFFIDRJ	Read: FIDRJ7	Read: FIDRJ6	Read: FIDRJ5	Read: FIDRJ4	Read: FIDRJ3	Read: FIDRJ2	Read: FIDRJ1	Read: FIDRJ0
		Write:	Write:	Write:	Write:	Write:	Write:	Write:	Write:

\$0300 - \$035F

Byteflight

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0315	BFFIDRMR	Read:	FIDRMR 7	FIDRMR 6	FIDRMR 5	FIDRMR 5	FIDRMR 4	FIDRMR 3	FIDRMR 2	FIDRMR 1
		Write:								
\$0316	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0317	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0318	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0319	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031A	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031B	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031C	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031D	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031E	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$031F	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								
\$0320	BFTIDENT	Read:	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		Write:								
\$0321	BFTLEN	Read:					LEN3	LEN2	LEN1	LEN0
		Write:								
\$0322 - \$032D	BFTDATA0- BFTDATA11	Read:	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
		Write:								
\$032E - \$032F	Reserved	Read:								
		Write:								
\$0330	BFRIDENT	Read:	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		Write:								
\$0331	BFRLEN	Read:					LEN3	LEN2	LEN1	LEN0
		Write:								
\$0332 - \$033D	BFRDATA0- BFRDATA11	Read:	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA 0
		Write:								
\$033E- \$033F	Reserved	Read:								
		Write:								
\$0340	BFFIDENT	Read:	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		Write:								
\$0341	BFFLEN	Read:					LEN3	LEN2	LEN1	LEN0
		Write:								
\$0342 - \$034D	BFFDATA0- BFFDATA11	Read:	DATA 7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
		Write:								
\$034E - \$034F	Reserved	Read:								
		Write:								
\$0350 - \$035F	BFBUFCTL0 - BFBUFCTL15	Read:	IFLG	IENA	LOCK	ABTAK	ABTRQ	0	0	CFG
		Write:								

\$0360 - \$03FF**Reserved**

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
\$0360 - \$03FF	Reserved	Read:	0	0	0	0	0	0	0	0
		Write:								

1.6 Part ID Assignments

The part ID is located in two 8-bit registers PARTIDH and PARTIDL (addresses \$001A and \$001B after reset). The read-only value is a unique part ID for each revision of the chip. **(Table 1-3)** shows the assigned part ID number.

Table 1-3 Assigned Part ID Numbers

Device	Mask Set Number	Part ID ¹
MC9S12DT128	1L40K	\$0111
MC9S12DT128	3L40K	\$0113
MC9S12DT128	4L40K	\$0114
MC9S12DT128	0L94R	\$0110
MC9S12DT128	1L59W	\$0115
MC9S12DT128	5L40K	\$0115
MC9S12DT128	2L94R	\$0115

NOTES:

- The coding is as follows:
 Bit 15-12: Major family identifier
 Bit 11-8: Minor family identifier
 Bit 7-4: Major mask set revision number including FAB transfers
 Bit 3-0: Minor - non full - mask set revision

The device memory sizes are located in two 8-bit registers MEMSIZ0 and MEMSIZ1 (addresses \$001C and \$001D after reset). **Table 1-4** shows the read-only values of these registers. Refer to HCS12 Module Mapping Control (MMC) Block Guide for further details.

Table 1-4 Memory size registers

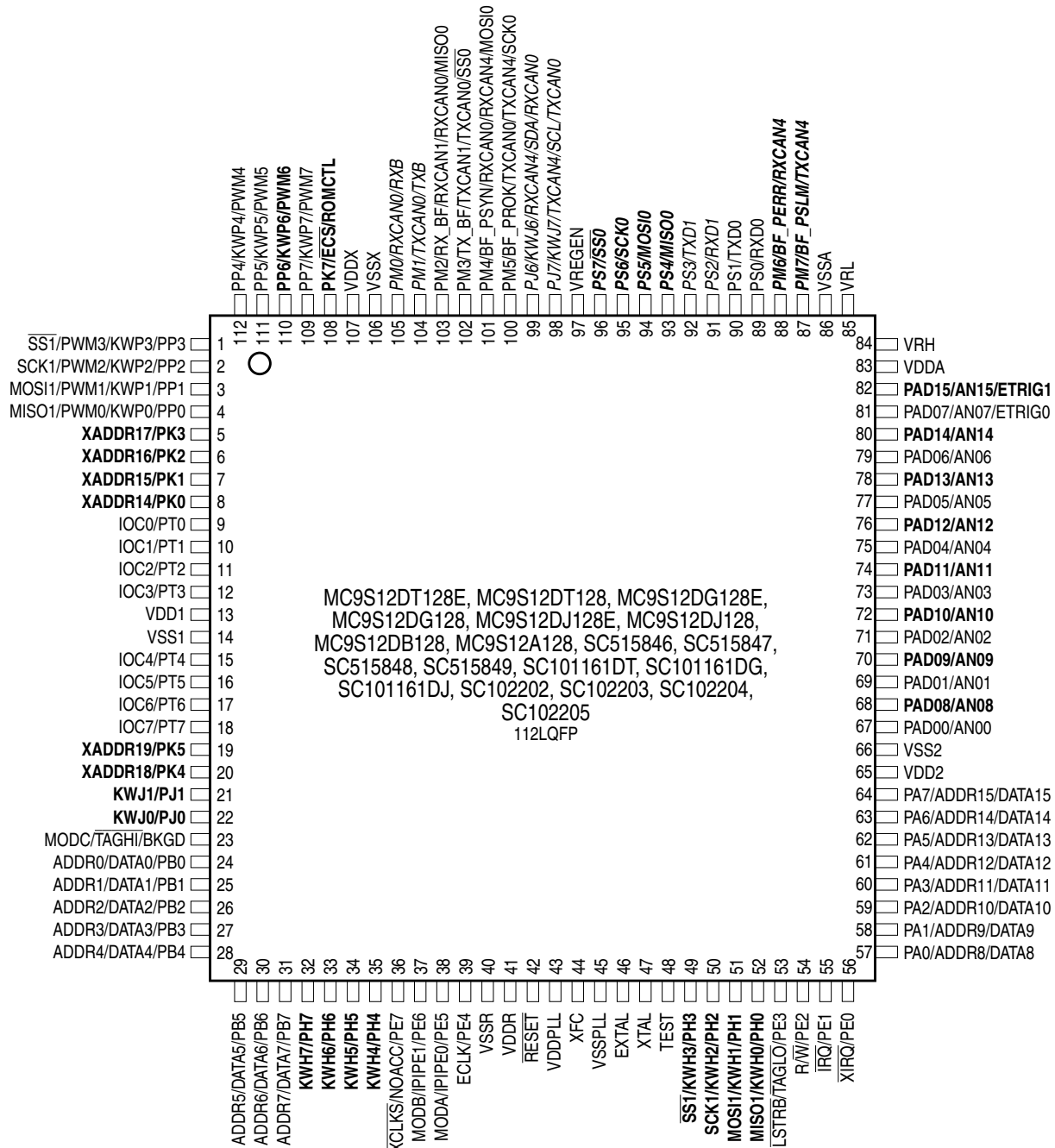
Register name	Value
MEMSIZ0	\$13
MEMSIZ1	\$80

Section 2 Signal Description

This section describes signals that connect off-chip. It includes a pinout diagram, a table of signal properties, and detailed discussion of signals. It is built from the signal description sections of the Block User Guides of the individual IP blocks on the device.

2.1 Device Pinout

The MC9S12DT128 and its derivatives are available in a 112-pin low profile quad flat pack (LQFP) and in a 80-pin quad flat pack (QFP). Most pins perform two or more functions, as described in the Signal Descriptions. **Figure 2-1**, **Figure 2-2**, and **Figure 2-3** show the pin assignments for different packages.



Signals shown in **Bold** are not available on all the 80 pin package options
 Signals shown in **Bold-Italics** are not available on the MC9S12DJ128E, MC9S12DJ128, MC9S12DG128E, MC9S12DG128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204 80 pin package options
 Signals shown in *Italics* are not available on the MC9S12DB128, SC515846, and SC102202 80 pin package options

Figure 2-1 Pin assignments 112 LQFP for MC9S12DT128E, MC9S12DT128, MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12DB128 MC9S12A128, SC515846, SC515847, SC515848, SC515849, SC101161DT, SC101161DG, SC101161DJ, SC102202, SC102203, SC102204, and SC102205

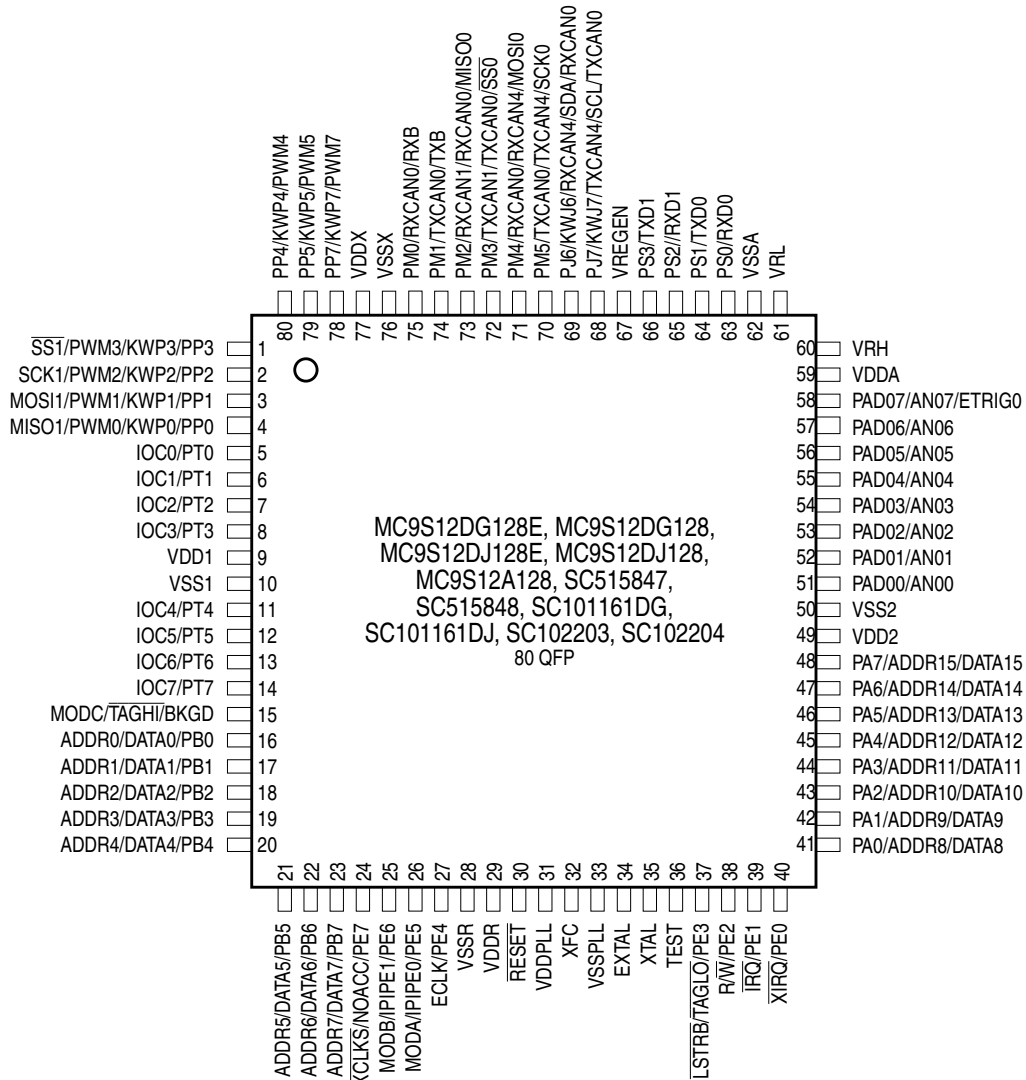


Figure 2-2 Pin Assignments in 80 QFP for MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204 Bondout

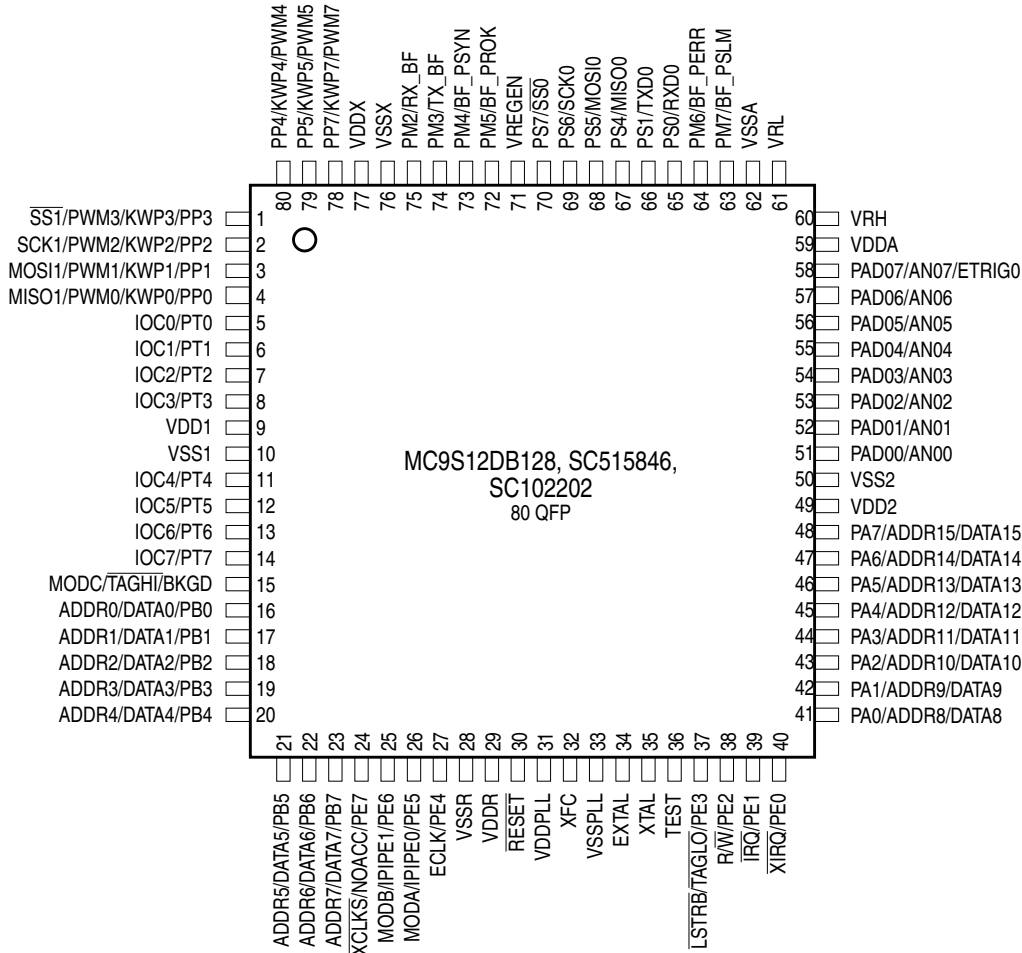


Figure 2-3 Pin Assignments in 80 QFP for MC9S12DB128, SC515846, and SC102202 Bondout

2.2 Signal Properties Summary

(Table 2-1) summarizes the pin functionality. Signals shown in **Bold** are not available on all the 80-pin package options. Signals shown in *Bold-Italics* are not available on the MC9S12DG128E, MC9S12DG128, MC9S12DJ128E, MC9S12DJ128, MC9S12A128, SC515847, SC515848, SC101161DG, SC101161DJ, SC102203, and SC102204 80-pin package options. Signals shown in *Italics* are not available on MC9S12DB128, SC515846, and SC102202 80-pin package options.

Table 2-1 Signal Properties

Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Pin Name Function 4	Pin Name Function 5	Powered by	Internal Pull Resistor		Description
						CTRL	Reset State	
EXTAL	—	—	—	—	VDDPLL	NA	NA	Oscillator Pins
XTAL	—	—	—	—	VDDPLL	NA	NA	
RESET	—	—	—	—	VDDR	None	None	External Reset
TEST	—	—	—	—	N.A.	None	None	Test Input
VREGEN	—	—	—	—	VDDX	NA	NA	Voltage Regulator Enable Input
XFC	—	—	—	—	VDDPLL	NA	NA	PLL Loop Filter
BKGD	$\overline{\text{TAGHI}}$	MODC	—	—	VDDR	Always Up	Up	Background Debug, Tag High, Mode Input
PAD[15]	AN1[7]	ETRIG1	—	—	VDDA	None	None	Port AD Input, Analog Inputs, External Trigger Input (ATD1)
PAD[14:8]	AN1[6:0]	—	—	—	VDDA	None	None	Port AD Input, Analog Inputs (ATD1)
PAD[7]	AN0[7]	ETRIG0	—	—	VDDA	None	None	Port AD Input, Analog Inputs, External Trigger Input (ATD0)
PAD[6:0]	AN0[6:0]	—	—	—	VDDA	None	None	Port AD Input, Analog Inputs (ATD0)
PA[7:0]	ADDR[15:8]/ DATA[15:8]	—	—	—	VDDR	PUCR/ PUPAE	Disabled	Port A I/O, Multiplexed Address/Data
PB[7:0]	ADDR[7:0]/ DATA[7:0]	—	—	—	VDDR	PUCR/ PUPBE	Disabled	Port B I/O, Multiplexed Address/Data
PE7	NOACC	$\overline{\text{XCLKS}}$	—	—	VDDR	PUCR/ PUPEE	Mode depend- ant ¹	Port E I/O, Access, Clock Select
PE6	IPIPE1	MODB	—	—	VDDR	While $\overline{\text{RESET}}$ pin low: Down		Port E I/O, Pipe Status, Mode Input
PE5	IPIPE0	MODA	—	—	VDDR			Port E I/O, Pipe Status, Mode Input
PE4	ECLK	—	—	—	VDDR	PUCR/ PUPEE	Mode depend- ant ¹	Port E I/O, Bus Clock Output
PE3	$\overline{\text{LSTRB}}$	$\overline{\text{TAGLO}}$	—	—	VDDR			Port E I/O, Byte Strobe, Tag Low
PE2	R/ $\overline{\text{W}}$	—	—	—	VDDR			Port E I/O, R/ $\overline{\text{W}}$ in expanded modes
PE1	$\overline{\text{IRQ}}$	—	—	—	VDDR		Up	Port E Input, Maskable Interrupt
PE0	$\overline{\text{XIRQ}}$	—	—	—	VDDR	Port E Input, Non Maskable Interrupt		
PH7	KWH7	---	—	—	VDDR	PERH/ PPSH	Disabled	Port H I/O, Interrupt

Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Pin Name Function 4	Pin Name Function 5	Powered by	Internal Pull Resistor		Description
						CTRL	Reset State	
PH6	KWH6	---	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt
PH5	KWH5	---	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt
PH4	KWH4	---	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt
PH3	KWH3	$\overline{SS1}$	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt, \overline{SS} of SPI1
PH2	KWH2	SCK1	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt, SCK of SPI1
PH1	KWH1	MOSI1	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt, MOSI of SPI1
PH0	KWH0	MISO1	—	—	VDDR	PERH/PPSH	Disabled	Port H I/O, Interrupt, MISO of SPI1
PJ7	KWJ7	TXCAN4	SCL	TXCAN0	VDDX	PERJ/PPSJ	Up	Port J I/O, Interrupt, TX of CAN4, SCL of IIC
PJ6	KWJ6	RXCAN4	SDA	RXCAN0	VDDX	PERJ/PPSJ	Up	Port J I/O, Interrupt, RX of CAN4, SDA of IIC
PJ[1:0]	KWJ[1:0]	—	—	—	VDDX	PERJ/PPSJ	Up	Port J I/O, Interrupts
PK7	\overline{ECS}	ROMCTL	—	—	VDDX	PUCR/PUPKE	Up	Port K I/O, Emulation Chip Select, ROM Control
PK[5:0]	XADDR[19:14]	—	—	—	VDDX	PUCR/PUPKE	Up	Port K I/O, Extended Addresses
PM7	BF_PSLM	TXCAN4	—	—	VDDX	PERM/PPSM	Disabled	Port M I/O, BF slot mismatch pulse, TX of CAN4
PM6	BF_PERR	RXCAN4	—	—	VDDX	PERM/PPSM	Disabled	Port M I/O, BF illegal pulse/message format error pulse, RX of CAN4
PM5	BF_PROK	TXCAN0	TXCAN4	SCK0	VDDX	PERM/PPSM	Disabled	Port M I/O, BF reception ok pulse, TX of CAN0, CAN4, SCK of SPI0
PM4	BF_PSYN	RXCAN0	RXCAN4	MOSI0	VDDX	PERM/PPSM	Disabled	Port M I/O, BF sync pulse (Rx/Tx) OK pulse o/p, RX of CAN0, CAN4, MOSI of SPI0
PM3	TX_BF	TXCAN1	TXCAN0	$\overline{SS0}$	VDDX	PERM/PPSM	Disabled	Port M I/O, TX of BF, CAN1, CAN0, \overline{SS} of SPI0
PM2	RX_BF	RXCAN1	RXCAN0	MISO0	VDDX	PERM/PPSM	Disabled	Port M I/O, RX of BF, CAN1, CAN0, MISO of SPI0

Pin Name Function 1	Pin Name Function 2	Pin Name Function 3	Pin Name Function 4	Pin Name Function 5	Powered by	Internal Pull Resistor		Description
						CTRL	Reset State	
PM1	TXCAN0	TXB	—	—	VDDX	PERM/PPSM	Disabled	Port M I/O, TX of CAN0, RX of BDLC
PM0	RXCAN0	RXB	—	—	VDDX	PERM/PPSM	Disabled	Port M I/O, RX of CAN0, RX of BDLC
PP7	KWP7	PWM7	—	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 7 of PWM
PP6	KWP6	PWM6	—	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 6 of PWM
PP5	KWP5	PWM5	—	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 5 of PWM
PP4	KWP4	PWM4	—	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 4 of PWM
PP3	KWP3	PWM3	$\overline{SS1}$	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 3 of PWM, \overline{SS} of SPI1
PP2	KWP2	PWM2	SCK1	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 2 of PWM, SCK of SPI1
PP1	KWP1	PWM1	MOSI1	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 1 of PWM, MOSI of SPI1
PP0	KWP0	PWM0	MISO1	—	VDDX	PERP/PPSP	Disabled	Port P I/O, Interrupt, Channel 0 of PWM, MISO2 of SPI1
PS7	$\overline{SS0}$	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, \overline{SS} of SPI0
PS6	SCK0	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, SCK of SPI0
PS5	MOSI0	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, MOSI of SPI0
PS4	MISO0	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, MISO of SPI0
PS3	TXD1	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, TXD of SCI1
PS2	RXD1	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, RXD of SCI1
PS1	TXD0	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, TXD of SCI0
PS0	RXD0	—	—	—	VDDX	PERS/PPSS	Up	Port S I/O, RXD of SCI0
PT[7:0]	IOC[7:0]	—	—	—	VDDX	PERT/PPST	Disabled	Port T I/O, Timer channels

NOTES:

1. Refer to PEAR register description in HCS12 Multiplexed External Bus Interface (MEBI) Block Guide.

2.3 Detailed Signal Descriptions

2.3.1 EXTAL, XTAL — Oscillator Pins

EXTAL and XTAL are the crystal driver and external clock pins. On reset all the device clocks are derived from the EXTAL input frequency. XTAL is the crystal output.

2.3.2 $\overline{\text{RESET}}$ — External Reset Pin

An active low bidirectional control signal, it acts as an input to initialize the MCU to a known start-up state, and an output when an internal MCU function causes a reset.

2.3.3 TEST — Test Pin

This input only pin is reserved for test.

NOTE: *The TEST pin must be tied to VSS in all applications.*

2.3.4 XFC — PLL Loop Filter Pin

PLL loop filter. Please ask your Freescale representative for the interactive application note to compute PLL loop filter elements. Any current leakage on this pin must be avoided.

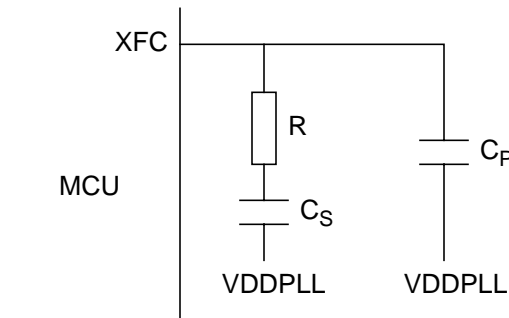


Figure 2-4 PLL Loop Filter Connections

2.3.5 BKGD / $\overline{\text{TAGHI}}$ / MODC — Background Debug, Tag High, and Mode Pin

The BKGD/ $\overline{\text{TAGHI}}$ /MODC pin is used as a pseudo-open-drain pin for the background debug communication. In MCU expanded modes of operation when instruction tagging is on, an input low on this pin during the falling edge of E-clock tags the high half of the instruction word being read into the instruction queue. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODC bit at the rising edge of $\overline{\text{RESET}}$. This pin has a permanently enabled pull-up device.

2.3.6 PAD[15] / AN1[7] / ETRIG1 — Port AD Input Pin [15]

PAD15 is a general purpose input pin and analog input of the analog to digital converter ATD1. It can act as an external trigger input for the ATD1.

2.3.7 PAD[14:8] / AN1[6:0] — Port AD Input Pins [14:8]

PAD14 - PAD8 are general purpose input pins and analog inputs of the analog to digital converter ATD1.

2.3.8 PAD[7] / AN0[7] / ETRIG0 — Port AD Input Pin [7]

PAD7 is a general purpose input pin and analog input of the analog to digital converter ATD0. It can act as an external trigger input for the ATD0.

2.3.9 PAD[6:0] / AN0[6:0] — Port AD Input Pins [6:0]

PAD6 - PAD0 are general purpose input pins and analog inputs of the analog to digital converter ATD0.

2.3.10 PA[7:0] / ADDR[15:8] / DATA[15:8] — Port A I/O Pins

PA7-PA0 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the multiplexed external address and data bus.

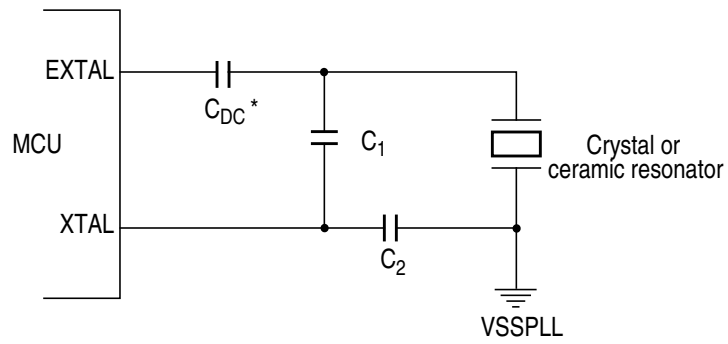
2.3.11 PB[7:0] / ADDR[7:0] / DATA[7:0] — Port B I/O Pins

PB7-PB0 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the multiplexed external address and data bus.

2.3.12 PE7 / NOACC / \overline{XCLKS} — Port E I/O Pin 7

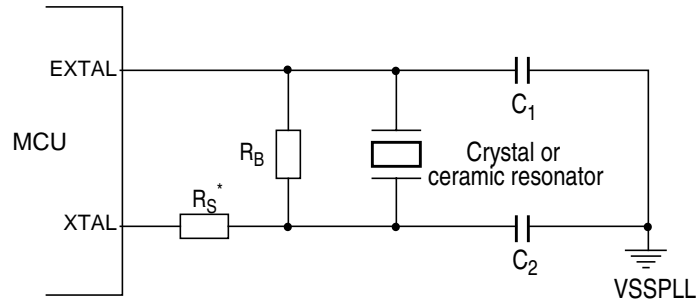
PE7 is a general purpose input or output pin. During MCU expanded modes of operation, the NOACC signal, when enabled, is used to indicate that the current bus cycle is an unused or “free” cycle. This signal will assert when the CPU is not using the bus.

The \overline{XCLKS} is an input signal which controls whether a crystal in combination with the internal Colpitts (low power) oscillator is used or whether Pierce oscillator/external clock circuitry is used. The state of this pin is latched at the rising edge of \overline{RESET} . If the input is a logic low the EXTAL pin is configured for an external clock drive. If input is a logic high an oscillator circuit is configured on EXTAL and XTAL. Since this pin is an input with a pull-up device during reset, if the pin is left floating, the default configuration is an oscillator circuit on EXTAL and XTAL.



* Due to the nature of a translated ground Colpitts oscillator a DC voltage bias is applied to the crystal
 Please contact the crystal manufacturer for crystal DC bias conditions and recommended capacitor value C_{DC} .

Figure 2-5 Colpitts Oscillator Connections (PE7=1)



* R_S can be zero (shorted) when used with higher frequency crystals. Refer to manufacturer's data.

Figure 2-6 Pierce Oscillator Connections (PE7=0)

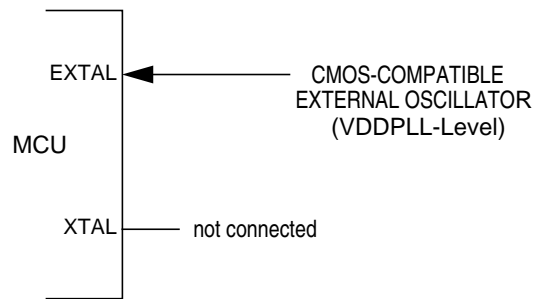


Figure 2-7 External Clock Connections (PE7=0)

2.3.13 PE6 / MODB / IPIPE1 — Port E I/O Pin 6

PE6 is a general purpose input or output pin. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODB bit at the rising edge of $\overline{\text{RESET}}$. This pin is shared with the instruction queue tracking signal IPIPE1. This pin is an input with a pull-down device which is only active when $\overline{\text{RESET}}$ is low.

2.3.14 PE5 / MODA / IPIPE0 — Port E I/O Pin 5

PE5 is a general purpose input or output pin. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODA bit at the rising edge of $\overline{\text{RESET}}$. This pin is shared with the instruction queue tracking signal IPIPE0. This pin is an input with a pull-down device which is only active when $\overline{\text{RESET}}$ is low.

2.3.15 PE4 / ECLK — Port E I/O Pin 4

PE4 is a general purpose input or output pin. It can be configured to drive the internal bus clock ECLK. ECLK can be used as a timing reference.

2.3.16 PE3 / $\overline{\text{LSTRB}}$ / $\overline{\text{TAGLO}}$ — Port E I/O Pin 3

PE3 is a general purpose input or output pin. In MCU expanded modes of operation, $\overline{\text{LSTRB}}$ can be used for the low-byte strobe function to indicate the type of bus access and when instruction tagging is on, $\overline{\text{TAGLO}}$ is used to tag the low half of the instruction word being read into the instruction queue.

2.3.17 PE2 / $\overline{\text{R/W}}$ — Port E I/O Pin 2

PE2 is a general purpose input or output pin. In MCU expanded modes of operations, this pin drives the read/write output signal for the external bus. It indicates the direction of data on the external bus.

2.3.18 PE1 / $\overline{\text{IRQ}}$ — Port E Input Pin 1

PE1 is a general purpose input pin and the maskable interrupt request input that provides a means of applying asynchronous interrupt requests. This will wake up the MCU from STOP or WAIT mode.

2.3.19 PE0 / $\overline{\text{XIRQ}}$ — Port E Input Pin 0

PE0 is a general purpose input pin and the non-maskable interrupt request input that provides a means of applying asynchronous interrupt requests. This will wake up the MCU from STOP or WAIT mode.

2.3.20 PH7 / KWH7 — Port H I/O Pin 7

PH7 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode.

2.3.21 PH6 / KWH6 — Port H I/O Pin 6

PH6 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode.

2.3.22 PH5 / KWH5 — Port H I/O Pin 5

PH5 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode.

2.3.23 PH4 / KWH4 — Port H I/O Pin 2

PH4 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode.

2.3.24 PH3 / KWH3 / $\overline{SS1}$ — Port H I/O Pin 3

PH3 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as slave select pin \overline{SS} of the Serial Peripheral Interface 1 (SPI1).

2.3.25 PH2 / KWH2 / SCK1 — Port H I/O Pin 2

PH2 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as serial clock pin SCK of the Serial Peripheral Interface 1 (SPI1).

2.3.26 PH1 / KWH1 / MOSI1 — Port H I/O Pin 1

PH1 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as master output (during master mode) or slave input pin (during slave mode) MOSI of the Serial Peripheral Interface 1 (SPI1).

2.3.27 PH0 / KWH0 / MISO1 — Port H I/O Pin 0

PH0 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as master input (during master mode) or slave output (during slave mode) pin MISO of the Serial Peripheral Interface 1 (SPI1).

2.3.28 PJ7 / KWJ7 / TXCAN4 / SCL / TXCAN0 — PORT J I/O Pin 7

PJ7 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as the transmit pin TXCAN for the Motorola Scalable Controller Area Network controller 0 or 4 (CAN0, CAN4) or the serial clock pin SCL of the IIC module.

2.3.29 PJ6 / KWJ6 / RXCAN4 / SDA / RXCAN0 — PORT J I/O Pin 6

PJ6 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as the receive pin RXCAN for the Motorola Scalable Controller Area Network controller 0 or 4 (CAN0, CAN4) or the serial data pin SDA of the IIC module.

2.3.30 PJ[1:0] / KWJ[1:0] — Port J I/O Pins [1:0]

PJ1 and PJ0 are general purpose input or output pins. They can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode.

2.3.31 PK7 / $\overline{\text{ECS}}$ / ROMCTL — Port K I/O Pin 7

PK7 is a general purpose input or output pin. During MCU expanded modes of operation, this pin is used as the emulation chip select output ($\overline{\text{ECS}}$). While configuring MCU expanded modes, this pin is used to enable the Flash EEPROM memory in the memory map (ROMCTL). At the rising edge of $\overline{\text{RESET}}$, the state of this pin is latched to the ROMON bit. For a complete list of modes refer to **4.2 Chip Configuration Summary**.

2.3.32 PK[5:0] / XADDR[19:14] — Port K I/O Pins [5:0]

PK5-PK0 are general purpose input or output pins. In MCU expanded modes of operation, these pins provide the expanded address XADDR[19:14] for the external bus.

2.3.33 PM7 / BF_PSLM / TXCAN4 — Port M I/O Pin 7

PM7 is a general purpose input or output pin. It can be configured as the slot mismatch output pulse pin of Byteflight. It can be configured as the transmit pin TXCAN of the Motorola Scalable Controller Area Network controllers 4 (CAN4).

2.3.34 PM6 / BF_PERR / RXCAN4 — Port M I/O Pin 6

PM6 is a general purpose input or output pin. It can be configured as the illegal pulse or message format error output pulse pin of Byteflight. It can be configured as the receive pin RXCAN of the Motorola Scalable Controller Area Network controllers 4 (CAN4).

2.3.35 PM5 / BF_PROK / TXCAN0 / TXCAN4 / SCK0 — Port M I/O Pin 5

PM5 is a general purpose input or output pin. It can be configured as the reception OK output pulse pin of Byteflight. It can be configured as the transmit pin TXCAN of the Motorola Scalable Controller Area Network controllers 0 or 4 (CAN0 or CAN4). It can be configured as the serial clock pin SCK of the Serial Peripheral Interface 0 (SPI0).

2.3.36 PM4 / BF_PSYN / RXCAN0 / RXCAN4/ MOSI0 — Port M I/O Pin 4

PM4 is a general purpose input or output pin. It can be configured as the correct synchronisation pulse reception/transmission output pulse pin of Byteflight. It can be configured as the receive pin RXCAN of the Motorola Scalable Controller Area Network controllers 0 or 4 (CAN0 or CAN4). It can be configured as the master output (during master mode) or slave input pin (during slave mode) MOSI for the Serial Peripheral Interface 0 (SPI0).

2.3.37 PM3 / TX_BF / TXCAN1 / TXCAN0 / \overline{SS} — Port M I/O Pin 3

PM3 is a general purpose input or output pin. It can be configured as the transmit pin TX_BF of Byteflight. It can be configured as the transmit pin TXCAN of the Motorola Scalable Controller Area Network controllers 1 or 0 (CAN1 or CAN0). It can be configured as the slave select pin \overline{SS} of the Serial Peripheral Interface 0 (SPI0).

2.3.38 PM2 / RX_BF / RXCAN1 / RXCAN0 / MISO0 — Port M I/O Pin 2

PM2 is a general purpose input or output pin. It can be configured as the receive pin RX_BF of Byteflight. It can be configured as the receive pin RXCAN of the Motorola Scalable Controller Area Network controllers 1 or 0 (CAN1 or CAN0). It can be configured as the master input (during master mode) or slave output pin (during slave mode) MISO for the Serial Peripheral Interface 0 (SPI0).

2.3.39 PM1 / TXCAN0 / TXB — Port M I/O Pin 1

PM1 is a general purpose input or output pin. It can be configured as the transmit pin TXCAN of the Motorola Scalable Controller Area Network controller 0 (CAN0). It can be configured as the transmit pin TXB of the BDLC.

2.3.40 PM0 / RXCAN0 / RXB — Port M I/O Pin 0

PM0 is a general purpose input or output pin. It can be configured as the receive pin RXCAN of the Motorola Scalable Controller Area Network controller 0 (CAN0). It can be configured as the receive pin RXB of the BDLC.

2.3.41 PP7 / KWP7 / PWM7 — Port P I/O Pin 7

PP7 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 7 output.

2.3.42 PP6 / KWP6 / PWM6 — Port P I/O Pin 6

PP6 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 6 output.

2.3.43 PP5 / KWP5 / PWM5 — Port P I/O Pin 5

PP5 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 5 output.

2.3.44 PP4 / KWP4 / PWM4 — Port P I/O Pin 4

PP4 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 4 output.

2.3.45 PP3 / KWP3 / PWM3 / $\overline{SS1}$ — Port P I/O Pin 3

PP3 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 3 output. It can be configured as slave select pin \overline{SS} of the Serial Peripheral Interface 1 (SPI1).

2.3.46 PP2 / KWP2 / PWM2 / SCK1 — Port P I/O Pin 2

PP2 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 2 output. It can be configured as serial clock pin SCK of the Serial Peripheral Interface 1 (SPI1).

2.3.47 PP1 / KWP1 / PWM1 / MOSI1 — Port P I/O Pin 1

PP1 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 1 output. It can be configured as master output (during master mode) or slave input pin (during slave mode) MOSI of the Serial Peripheral Interface 1 (SPI1).

2.3.48 PP0 / KWP0 / PWM0 / MISO1 — Port P I/O Pin 0

PP0 is a general purpose input or output pin. It can be configured to generate an interrupt causing the MCU to exit STOP or WAIT mode. It can be configured as Pulse Width Modulator (PWM) channel 0 output. It can be configured as master input (during master mode) or slave output (during slave mode) pin MISO of the Serial Peripheral Interface 1 (SPI1).

2.3.49 PS7 / $\overline{SS0}$ — Port S I/O Pin 7

PS7 is a general purpose input or output pin. It can be configured as the slave select pin \overline{SS} of the Serial Peripheral Interface 0 (SPI0).

2.3.50 PS6 / SCK0 — Port S I/O Pin 6

PS6 is a general purpose input or output pin. It can be configured as the serial clock pin SCK of the Serial Peripheral Interface 0 (SPI0).

2.3.51 PS5 / MOSI0 — Port S I/O Pin 5

PS5 is a general purpose input or output pin. It can be configured as master output (during master mode) or slave input pin (during slave mode) MOSI of the Serial Peripheral Interface 0 (SPI0).

2.3.52 PS4 / MISO0 — Port S I/O Pin 4

PS4 is a general purpose input or output pin. It can be configured as master input (during master mode) or slave output pin (during slave mode) MOSI of the Serial Peripheral Interface 0 (SPI0).

2.3.53 PS3 / TXD1 — Port S I/O Pin 3

PS3 is a general purpose input or output pin. It can be configured as the transmit pin TXD of Serial Communication Interface 1 (SCI1).

2.3.54 PS2 / RXD1 — Port S I/O Pin 2

PS2 is a general purpose input or output pin. It can be configured as the receive pin RXD of Serial Communication Interface 1 (SCI1).

2.3.55 PS1 / TXD0 — Port S I/O Pin 1

PS1 is a general purpose input or output pin. It can be configured as the transmit pin TXD of Serial Communication Interface 0 (SCI0).

2.3.56 PS0 / RXD0 — Port S I/O Pin 0

PS0 is a general purpose input or output pin. It can be configured as the receive pin RXD of Serial Communication Interface 0 (SCI0).

2.3.57 PT[7:0] / IOC[7:0] — Port T I/O Pins [7:0]

PT7-PT0 are general purpose input or output pins. They can be configured as input capture or output compare pins IOC7-IOC0 of the Enhanced Capture Timer (ECT).

2.4 Power Supply Pins

MC9S12DT128 power and ground pins are described below.

Table 2-2 MC9S12DT128 Power and Ground Connection Summary

Mnemonic	Pin Number	Nominal Voltage	Description
	112-pin QFP		
VDD1, 2	13, 65	2.5V	Internal power and ground generated by internal regulator
VSS1, 2	14, 66	0V	

Mnemonic	Pin Number	Nominal Voltage	Description
	112-pin QFP		
VDDR	41	5.0V	External power and ground, supply to pin drivers and internal voltage regulator.
VSSR	40	0V	
VDDX	107	5.0V	External power and ground, supply to pin drivers.
VSSX	106	0V	
VDDA	83	5.0V	Operating voltage and ground for the analog-to-digital converters and the reference for the internal voltage regulator, allows the supply voltage to the A/D to be bypassed independently.
VSSA	86	0V	
VRL	85	0V	Reference voltages for the analog-to-digital converter.
VRH	84	5.0V	
VDDPLL	43	2.5V	Provides operating voltage and ground for the Phased-Locked Loop. This allows the supply voltage to the PLL to be bypassed independently. Internal power and ground generated by internal regulator.
VSSPLL	45	0V	
VREGEN	97	5V	Internal Voltage Regulator enable/disable

NOTE: All VSS pins must be connected together in the application.

2.4.1 VDDX, VSSX — Power & Ground Pins for I/O Drivers

External power and ground for I/O drivers. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible. Bypass requirements depend on how heavily the MCU pins are loaded.

2.4.2 VDDR, VSSR — Power & Ground Pins for I/O Drivers & for Internal Voltage Regulator

External power and ground for I/O drivers and input to the internal voltage regulator. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible. Bypass requirements depend on how heavily the MCU pins are loaded.

2.4.3 VDD1, VDD2, VSS1, VSS2 — Internal Logic Power Supply Pins

Power is supplied to the MCU through VDD and VSS. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible. This 2.5V supply is derived from the internal voltage regulator. There is no static load on those pins allowed. The internal voltage regulator is turned off, if VREGEN is tied to ground.

NOTE: No load allowed except for bypass capacitors.

2.4.4 VDDA, VSSA — Power Supply Pins for ATD and VREG

VDDA, VSSA are the power supply and ground input pins for the voltage regulator and the analog to digital converter. It also provides the reference for the internal voltage regulator. This allows the supply voltage to the ATD and the reference voltage to be bypassed independently.

2.4.5 VRH, VRL — ATD Reference Voltage Input Pins

VRH and VRL are the reference voltage input pins for the analog to digital converter.

2.4.6 VDDPLL, VSSPLL — Power Supply Pins for PLL

Provides operating voltage and ground for the Oscillator and the Phased-Locked Loop. This allows the supply voltage to the Oscillator and PLL to be bypassed independently. This 2.5V voltage is generated by the internal voltage regulator.

NOTE: *No load allowed except for bypass capacitors.*

2.4.7 VREGEN — On Chip Voltage Regulator Enable

Enables the internal 5V to 2.5V voltage regulator. If this pin is tied low, VDD1,2 and VDDPLL must be supplied externally.

Section 3 System Clock Description

3.1 Overview

The Clock and Reset Generator provides the internal clock signals for the core and all peripheral modules.

Figure 3-1 shows the clock connections from the CRG to all modules.

Consult the CRG Block User Guide for details on clock generation.

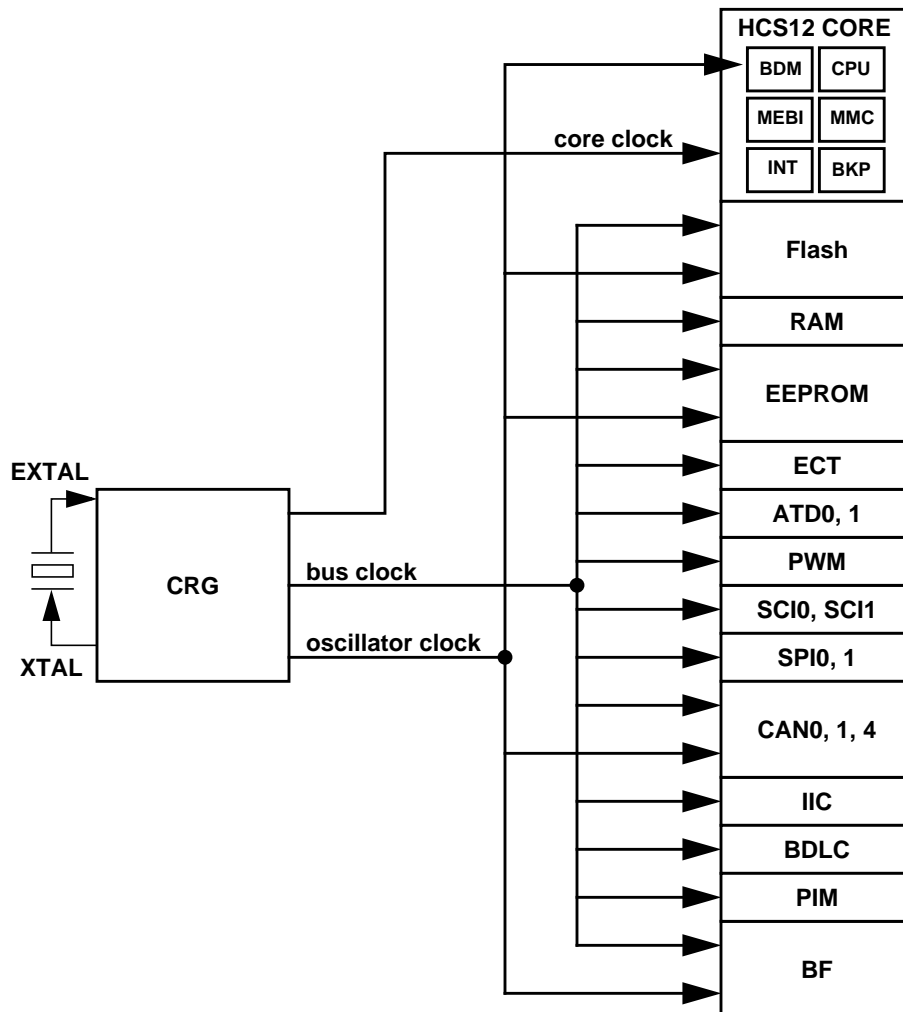


Figure 3-1 Clock Connections

Section 4 Modes of Operation

4.1 Overview

Eight possible modes determine the operating configuration of the MC9S12DT128. Each mode has an associated default memory map and external bus configuration controlled by a further pin.

Three low power modes exist for the device.

4.2 Chip Configuration Summary

The operating mode out of reset is determined by the states of the MODC, MODB, and MODA pins during reset ((**Table 4-1**)). The MODC, MODB, and MODA bits in the MODE register show the current operating mode and provide limited mode switching during operation. The states of the MODC, MODB, and MODA pins are latched into these bits on the rising edge of the reset signal. The ROMCTL signal allows the setting of the ROMON bit in the MISC register thus controlling whether the internal Flash is visible in the memory map. ROMON = 1 mean the Flash is visible in the memory map. The state of the ROMCTL pin is latched into the ROMON bit in the MISC register on the rising edge of the reset signal.

Table 4-1 Mode Selection

BKGD = MODC	PE6 = MODB	PE5 = MODA	PK7 = ROMCTL	ROMON Bit	Mode Description
0	0	0	X	1	Special Single Chip, BDM allowed and ACTIVE. BDM is allowed in all other modes but a serial command is required to make BDM active.
0	0	1	0	1	Emulation Expanded Narrow, BDM allowed
			1	0	
0	1	0	X	0	Special Test (Expanded Wide), BDM allowed
0	1	1	0	1	Emulation Expanded Wide, BDM allowed
			1	0	
1	0	0	X	1	Normal Single Chip, BDM allowed
1	0	1	0	0	Normal Expanded Narrow, BDM allowed
			1	1	
1	1	0	X	1	Special Peripheral; BDM allowed but bus operations would cause bus conflicts (must not be used)
1	1	1	0	0	Normal Expanded Wide, BDM allowed
			1	1	

For further explanation on the modes refer to the HCS12 Multiplexed External Bus Interface Block Guide.

Table 4-2 Clock Selection Based on PE7

PE7 = XCLKS	Description
1	Colpitts Oscillator selected
0	Pierce Oscillator/external clock selected

Table 4-3 Voltage Regulator VREGEN

VREGEN	Description
1	Internal Voltage Regulator enabled
0	Internal Voltage Regulator disabled, VDD1,2 and VDDPLL must be supplied externally with 2.5V

4.3 Security

The device will make available a security feature preventing the unauthorized read and write of the memory contents. This feature allows:

- Protection of the contents of FLASH,
- Protection of the contents of EEPROM,
- Operation in single-chip mode, No BDM possible
- Operation from external memory with internal FLASH and EEPROM disabled.

The user must be reminded that part of the security must lie with the user's code. An extreme example would be user's code that dumps the contents of the internal program. This code would defeat the purpose of security. At the same time the user may also wish to put a back door in the user's program. An example of this is the user downloads a key through the SCI which allows access to a programming routine that updates parameters stored in EEPROM.

4.3.1 Securing the Microcontroller

Once the user has programmed the FLASH and EEPROM (if desired), the part can be secured by programming the security bits located in the FLASH module. These non-volatile bits will keep the part secured through resetting the part and through powering down the part.

The security byte resides in a portion of the Flash array.

Check the Flash Block User Guide for more details on the security configuration.

4.3.2 Operation of the Secured Microcontroller

4.3.2.1 Normal Single Chip Mode

This will be the most common usage of the secured part. Everything will appear the same as if the part was not secured with the exception of BDM operation. The BDM operation will be blocked.

4.3.2.2 Executing from External Memory

The user may wish to execute from external space with a secured microcontroller. This is accomplished by resetting directly into expanded mode. The internal FLASH and EEPROM will be disabled. BDM operations will be blocked.

4.3.3 Unsecuring the Microcontroller

In order to unsecure the microcontroller, the internal FLASH and EEPROM must be erased. This can be done through an external program in expanded mode or via a .sequence of BDM commands. Unsecuring is also possible via the Backdoor Key Access. Refer to Flash Block Guide for details.

Once the user has erased the FLASH and EEPROM, the part can be reset into special single chip mode. This invokes a program that verifies the erasure of the internal FLASH and EEPROM. Once this program completes, the user can erase and program the FLASH security bits to the unsecured state. This is generally done through the BDM, but the user could also change to expanded mode (by writing the mode bits through the BDM) and jumping to an external program (again through BDM commands). Note that if the part goes through a reset before the security bits are reprogrammed to the unsecure state, the part will be secured again.

4.4 Low Power Modes

The microcontroller features three main low power modes. Consult the respective Block User Guide for information on the module behavior in Stop, Pseudo Stop, and Wait Mode. An important source of information about the clock system is the Clock and Reset Generator User Guide (CRG).

4.4.1 Stop

Executing the CPU STOP instruction stops all clocks and the oscillator thus putting the chip in fully static mode. Wake up from this mode can be done via reset or external interrupts.

4.4.2 Pseudo Stop

This mode is entered by executing the CPU STOP instruction. In this mode the oscillator is still running and the Real Time Interrupt (RTI) or Watchdog (COP) sub module can stay active. Other peripherals are turned off. This mode consumes more current than the full STOP mode, but the wake up time from this mode is significantly shorter.

4.4.3 Wait

This mode is entered by executing the CPU WAI instruction. In this mode the CPU will not execute instructions. The internal CPU signals (address and databus) will be fully static. All peripherals stay active. For further power consumption the peripherals can individually turn off their local clocks.

4.4.4 Run

Although this is not a low power mode, unused peripheral modules should not be enabled in order to save power.

Section 5 Resets and Interrupts

5.1 Overview

Consult the Exception Processing section of the CPU Reference Manual for information on resets and interrupts.

5.2 Vectors

5.2.1 Vector Table

(Table 5-1) lists interrupt sources and vectors in default order of priority.

Table 5-1 Interrupt Vector Locations

Vector Address	Interrupt Source	CCR Mask	Local Enable	HPRIO Value to Elevate
\$FFFE, \$FFFF	Reset	None	None	–
\$FFFC, \$FFFD	Clock Monitor fail reset	None	COPCTL (CME, FCME)	–
\$FFFA, \$FFFB	COP failure reset	None	COP rate select	–
\$FFF8, \$FFF9	Unimplemented instruction trap	None	None	–
\$FFF6, \$FFF7	SWI	None	None	–
\$FFF4, \$FFF5	XIRQ / BF High Priority Sync Pulse	X-Bit	None / BFRIER (XSYNIE)	–
\$FFF2, \$FFF3	IRQ	I-Bit	INTCR (IRQEN)	\$F2
\$FFF0, \$FFF1	Real Time Interrupt	I-Bit	CRGIN (RTIE)	\$F0
\$FFEE, \$FFEF	Enhanced Capture Timer channel 0	I-Bit	TIE (C0I)	\$EE
\$FFEC, \$FFED	Enhanced Capture Timer channel 1	I-Bit	TIE (C1I)	\$EC
\$FFEA, \$FFEB	Enhanced Capture Timer channel 2	I-Bit	TIE (C2I)	\$EA
\$FFE8, \$FFE9	Enhanced Capture Timer channel 3	I-Bit	TIE (C3I)	\$E8
\$FFE6, \$FFE7	Enhanced Capture Timer channel 4	I-Bit	TIE (C4I)	\$E6
\$FFE4, \$FFE5	Enhanced Capture Timer channel 5	I-Bit	TIE (C5I)	\$E4
\$FFE2, \$FFE3	Enhanced Capture Timer channel 6	I-Bit	TIE (C6I)	\$E2
\$FFE0, \$FFE1	Enhanced Capture Timer channel 7	I-Bit	TIE (C7I)	\$E0
\$FFDE, \$FFDF	Enhanced Capture Timer overflow	I-Bit	TSCR2 (TOF)	\$DE
\$FFDC, \$FFDD	Pulse accumulator A overflow	I-Bit	PACTL (PAOVI)	\$DC
\$FFDA, \$FFDB	Pulse accumulator input edge	I-Bit	PACTL (PAI)	\$DA
\$FFD8, \$FFD9	SPI0	I-Bit	SPICR1 (SPIE, SPTIE)	\$D8
\$FFD6, \$FFD7	SCI0	I-Bit	SCICR2 (TIE, TCIE, RIE, ILIE)	\$D6
\$FFD4, \$FFD5	SCI1	I-Bit	SCICR2 (TIE, TCIE, RIE, ILIE)	\$D4
\$FFD2, \$FFD3	ATD0	I-Bit	ATDCTL2 (ASCIE)	\$D2
\$FFD0, \$FFD1	ATD1	I-Bit	ATDCTL2 (ASCIE)	\$D0
\$FFCE, \$FFCF	Port J	I-Bit	PIEJ (PIEJ7, PIEJ6, PIEJ1, PIEJ0)	\$CE
\$FFCC, \$FFCD	Port H	I-Bit	PIEH (PIEH7-0)	\$CC

\$FFCA, \$FFCB	Modulus Down Counter underflow	I-Bit	MCCTL (MCZI)	\$CA
\$FFC8, \$FFC9	Pulse Accumulator B Overflow	I-Bit	PBCTL (PBOVI)	\$C8
\$FFC6, \$FFC7	CRG PLL lock	I-Bit	PLLCR (LOCKIE)	\$C6
\$FFC4, \$FFC5	CRG Self Clock Mode	I-Bit	PLLCR (SCMIE)	\$C4
\$FFC2, \$FFC3	BDLC	I-Bit	DLCBCR1 (IE)	\$C2
\$FFC0, \$FFC1	IIC Bus	I-Bit	IBCR (IBIE)	\$C0
\$FFBE, \$FFBF	SPI1	I-Bit	SPICR1 (SPIE, SPTIE)	\$BE
\$FFBC, \$FFBD	Reserved			
\$FFBA, \$FFBB	EEPROM	I-Bit	ECNFG (CCIE, CBEIE)	\$BA
\$FFB8, \$FFB9	FLASH	I-Bit	FCNFG (CCIE, CBEIE)	\$B8
\$FFB6, \$FFB7	CAN0 wake-up	I-Bit	CANRIER (WUPIE)	\$B6
\$FFB4, \$FFB5	CAN0 errors	I-Bit	CANRIER (CSCIE, OVRIE)	\$B4
\$FFB2, \$FFB3	CAN0 receive	I-Bit	CANRIER (RXFIE)	\$B2
\$FFB0, \$FFB1	CAN0 transmit	I-Bit	CANTIER (TXEIE[2:0])	\$B0
\$FFAE, \$FFAF	CAN1 wake-up	I-Bit	CANRIER (WUPIE)	\$AE
\$FFAC, \$FFAD	CAN1 errors	I-Bit	CANRIER (CSCIE, OVRIE)	\$AC
\$FFAA, \$FFAB	CAN1 receive	I-Bit	CANRIER (RXFIE)	\$AA
\$FFA8, \$FFA9	CAN1 transmit	I-Bit	CANTIER (TXEIE[2:0])	\$A8
\$FFA6, \$FFA7	BF Receive FIFO not empty	I-Bit	BFRIER (RCVFIE)	\$A6
\$FFA4, \$FFA5	BF receive	I-Bit	BFBUFCTL[15:0] (IENA)	\$A4
\$FFA2, \$FFA3	BF Synchronization	I-Bit	BFRIER (SYNAIE, SYNIE)	\$A2
\$FFA0, \$FFA1	BF general	I-Bit	BFBUFCTL[15:0] (IENA), BFGIER (OVRNIE, ERRIE, SYNEIE, SYNLIE, ILLPIE, LOCKIE, WAKEIE) BFRIER (SLMMIE)	\$A0
\$FF98, \$FF9F	Reserved			
\$FF96, \$FF97	CAN4 wake-up	I-Bit	CANRIER (WUPIE)	\$96
\$FF94, \$FF95	CAN4 errors	I-Bit	CANRIER (CSCIE, OVRIE)	\$94
\$FF92, \$FF93	CAN4 receive	I-Bit	CANRIER (RXFIE)	\$92
\$FF90, \$FF91	CAN4 transmit	I-Bit	CANTIER (TXEIE[2:0])	\$90
\$FF8E, \$FF8F	Port P Interrupt	I-Bit	PIEP (PIEP7-0)	\$8E
\$FF8C, \$FF8D	PWM Emergency Shutdown	I-Bit	PWMSDN (PWWIE)	\$8C
\$FF80 to \$FF8B	Reserved			

5.3 Effects of Reset

When a reset occurs, MCU registers and control bits are changed to known start-up states. Refer to the respective module Block User Guides for register reset states.

5.3.1 I/O pins

Refer to the HCS12 Multiplexed External Bus Interface (MEBI) Block Guide for mode dependent pin configuration of port A, B, E and K out of reset.

Refer to the PIM Block User Guide for reset configurations of all peripheral module ports.

NOTE: *For devices assembled in 80-pin QFP packages all non-bonded out pins should be configured as outputs after reset in order to avoid current drawn from floating inputs. Refer to **Table 2-1** for affected pins.*

5.3.2 Memory

Refer to **Table 1-1** for locations of the memories depending on the operating mode after reset.

The RAM array is not automatically initialized out of reset.

ATD_10B8C

Block User Guide

V02.12

Original Release Date: 27 OCT 2000
Revised: 28 June 2005

Motorola Inc.

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
00.00	27-10-2000	-		Initial SRS2 release.
01.00	06-06-2001	-		Updated the description of ATDDIEN and PORTAD1 register.
01.10	16-06-2001	-		Made SRS2 Compliant
V02.00	20 June 2001	20 June 2001		Reworked whole document to make it more user friendly
V02.01	26 July 2001	-		Added document names Variable definitions and names have been hidden
V02.02	5 Sept 2001	5 Sept 2001		Corrected sampling phase description, other minor corrections
V02.03	8 Nov 2001	8 Nov 2001		Corrected AWAI bit description
V02.04	16 Jan 2002	16 Jan 2002		Syntax corrections
V02.05	8 Mar 2002	8 Mar 2002		Removed document number from all pages except cover sheet
V02.06	11 Apr 2002	11 Apr 2002		Documented special channel conversion in ATDTEST1 register
V02.07	22 Apr 2002	22 Apr 2002		Corrected Table "Available Result Data Formats"
V02.08	16 Aug 2002	16 Aug 2002		FIFOR flag: corrected clearing mechanism B)
V02.09	23 Aug 2002	23 Aug 2002		Detailed AWAI Bit description. Functional Description: Detailed and corrected Low power modes Table "Available Result Data Formats": Re-corrected
V02.10	21 Feb 2003	21 Feb 2003		Formal corrections on ATDTEST0/1 and ATDDRHx/ATDDRLx register descriptions
V02.11	24 Mar 2005	24 Mar 2005		Corrected PAD7-0 port description
V02.12	28 June 2005	28 June 2005		Enhanced FIFO bit description

Table 0-1 Revision History

Table of Contents

Section 1 Introduction

1.1	Overview	9
1.2	Features	9
1.3	Modes of Operation	9
1.3.1	Conversion modes	9
1.3.2	MCU Operating Modes	9
1.4	Block Diagram	10

Section 2 Signal Description

2.1	Overview	11
2.2	Detailed Signal Descriptions	11
2.2.1	AN7 / ETRIG / PAD7	11
2.2.2	AN6 / PAD6	11
2.2.3	AN5 / PAD5	11
2.2.4	AN4 / PAD4	11
2.2.5	AN3 / PAD3	11
2.2.6	AN2 / PAD2	11
2.2.7	AN1 / PAD1	11
2.2.8	AN0 / PAD0	11
2.2.9	VRH, VRL	12
2.2.10	VDDA, VSSA	12

Section 3 Memory Map and Register Definition

3.1	Overview	13
3.2	Module Memory Map	13
3.3	Register Descriptions	14
3.3.1	Reserved Register (ATDCTL0)	14
3.3.2	Reserved Register (ATDCTL1)	14
3.3.3	ATD Control Register 2 (ATDCTL2)	14
3.3.4	ATD Control Register 3 (ATDCTL3)	16
3.3.5	ATD Control Register 4 (ATDCTL4)	18
3.3.6	ATD Control Register 5 (ATDCTL5)	20
3.3.7	ATD Status Register 0 (ATDSTAT0)	23

3.3.8	Reserved Register (ATDTEST0)	24
3.3.9	ATD Test Register 1 (ATDTEST1).	24
3.3.10	ATD Status Register 1 (ATDSTAT1)	25
3.3.11	ATD Input Enable Register (ATDDIEN)	26
3.3.12	Port Data Register (PORTAD)	27
3.3.13	ATD Conversion Result Registers (ATDDRHx/ATDDRLx)	27

Section 4 Functional Description

4.1	General.	31
4.2	Analog Sub-block	31
4.2.1	Sample and Hold Machine	31
4.2.2	Analog Input Multiplexer.	31
4.2.3	Sample Buffer Amplifier	31
4.2.4	Analog-to-Digital (A/D) Machine.	31
4.3	Digital Sub-block.	32
4.3.1	External Trigger Input (ETRIG)	32
4.3.2	General Purpose Digital Input Port Operation	33
4.3.3	Low Power Modes	33

Section 5 Resets

5.1	General.	35
-----	------------------	----

Section 6 Interrupts

6.1	General.	37
-----	------------------	----

List of Figures

Figure 1-1	ATD_10B8C Block Diagram	10
Figure 3-1	Reserved Register (ATDCTL0)	14
Figure 3-2	Reserved Register (ATDCTL1)	14
Figure 3-3	ATD Control Register 2 (ATDCTL2)	15
Figure 3-4	ATD Control Register 3 (ATDCTL3)	16
Figure 3-5	ATD Control Register 4 (ATDCTL4)	18
Figure 3-6	ATD Control Register 5 (ATDCTL5)	21
Figure 3-7	ATD Status Register 0 (ATDSTAT0)	23
Figure 3-8	Reserved Register (ATDTEST0)	24
Figure 3-9	ATD Test Register 1 (ATDTEST1)	25
Figure 3-10	ATD Status Register 1 (ATDSTAT1)	26
Figure 3-11	ATD Input Enable Register (ATDDIEN)	26
Figure 3-12	Port Data Register (PORTAD)	27
Figure 3-13	Left Justified, ATD Conversion Result Register, High Byte (ATDDRxH)	28
Figure 3-14	Left Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)	28
Figure 3-15	Right Justified, ATD Conversion Result Register, High Byte (ATDDRxH)	28
Figure 3-16	Right Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)	29

.

List of Tables

Table 0-1	Revision History	2
Table 3-1	Module Memory Map	13
Table 3-2	External Trigger Configurations	16
Table 3-3	Conversion Sequence Length Coding	17
Table 3-4	ATD Behavior in Freeze Mode (breakpoint)	18
Table 3-5	Sample Time Select.	19
Table 3-6	Clock Prescaler Values	20
Table 3-7	Available Result Data Formats	21
Table 3-8	Left Justified, Signed and Unsigned ATD Output Codes.	22
Table 3-9	Analog Input Channel Select Coding	22
Table 3-10	Special Channel Select Coding	25
Table 4-1	External Trigger Control Bits	32
Table 6-1	ATD_10B8C Interrupt Vectors	37

.

Section 1 Introduction

1.1 Overview

The ATD_10B8C is an 8-channel, 10-bit, multiplexed input successive approximation analog-to-digital converter. Refer to device electrical specifications for ATD accuracy.

The block is designed to be upwards compatible with the 68HC11 standard 8-bit A/D converter. In addition, there are new operating modes that are unique to the HC12 design.

1.2 Features

- 8/10 Bit Resolution.
- 7 μ sec, 10-Bit Single Conversion Time.
- Sample Buffer Amplifier.
- Programmable Sample Time.
- Left/Right Justified, Signed/Unsigned Result Data.
- External Trigger Control.
- Conversion Completion Interrupt Generation.
- Analog Input Multiplexer for 8 Analog Input Channels.
- Analog/Digital Input Pin Multiplexing.
- 1 to 8 Conversion Sequence Lengths.
- Continuous Conversion Mode.
- Multiple Channel Scans.

1.3 Modes of Operation

1.3.1 Conversion modes

There is software programmable selection between performing **single** or **continuous conversion** on a **single channel** or **multiple channels**.

1.3.2 MCU Operating Modes

- **Stop Mode**
Entering Stop Mode causes all clocks to halt and thus the system is placed in a minimum power standby mode. This aborts any conversion sequence in progress. During recovery from Stop Mode, there must be a minimum delay for the Stop Recovery Time t_{SR} before initiating a new ATD conversion sequence.

- Wait Mode**
 Entering Wait Mode the ATD conversion either continues or aborts for low power depending on the logical value of the AWAIT bit.
- Freeze Mode**
 In Freeze Mode the ATD_10B8C will behave according to the logical values of the FRZ1 and FRZ0 bits. This is useful for debugging and emulation.

1.4 Block Diagram

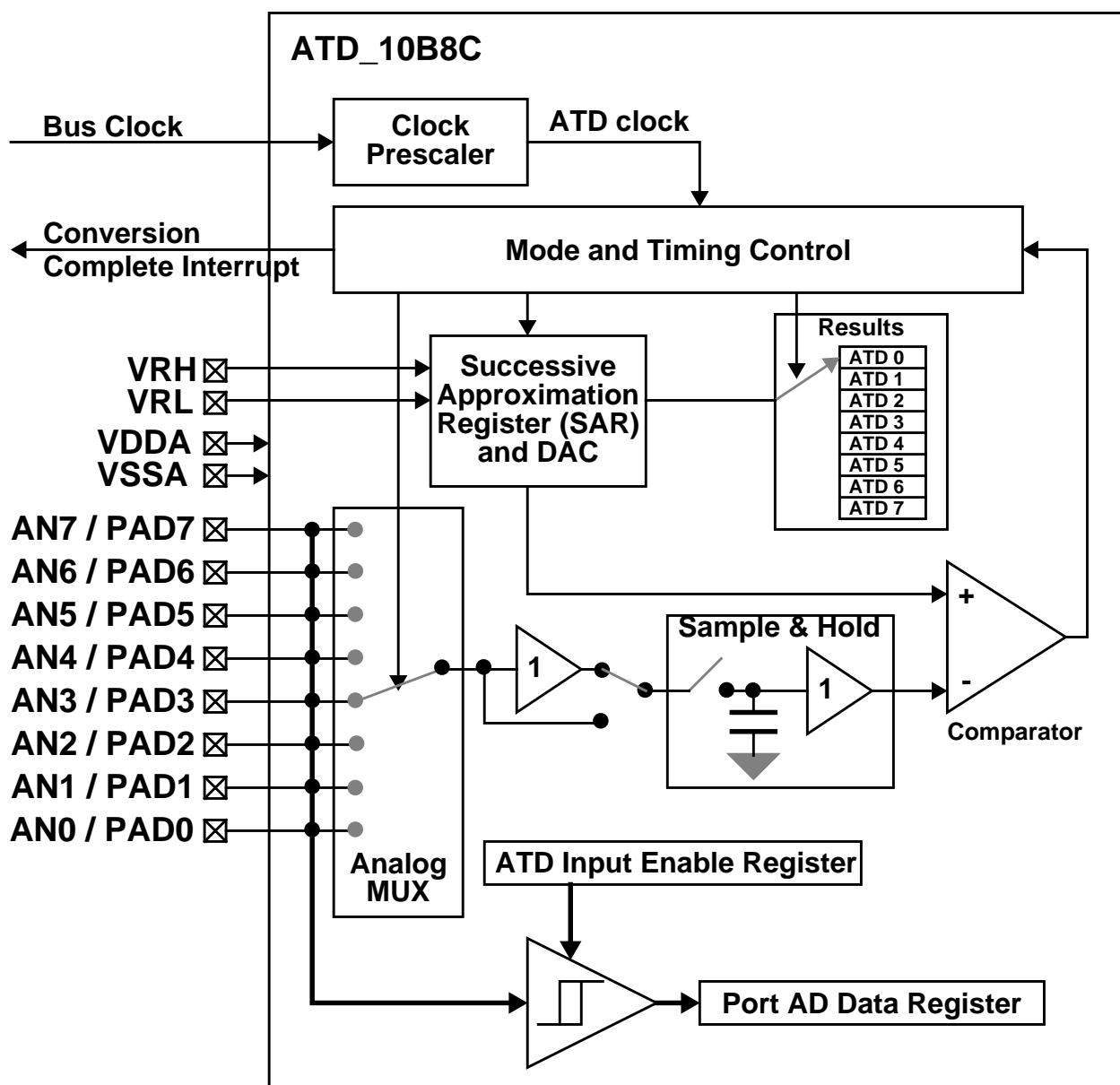


Figure 1-1 ATD_10B8C Block Diagram

Section 2 Signal Description

2.1 Overview

The ATD_10B8C has a total of 12 external pins.

2.2 Detailed Signal Descriptions

2.2.1 AN7 / ETRIG / PAD7

This pin serves as the analog input Channel 7. It can be configured to provide an external trigger for the ATD conversion. It can be configured as digital port pin.

2.2.2 AN6 / PAD6

This pin serves as the analog input Channel 6. It can be configured as digital port pin.

2.2.3 AN5 / PAD5

This pin serves as the analog input Channel 5. It can be configured as digital port pin.

2.2.4 AN4 / PAD4

This pin serves as the analog input Channel 4. It can be configured as digital port pin.

2.2.5 AN3 / PAD3

This pin serves as the analog input Channel 3. It can be configured as digital port pin.

2.2.6 AN2 / PAD2

This pin serves as the analog input Channel 2. It can be configured as digital port pin.

2.2.7 AN1 / PAD1

This pin serves as the analog input Channel 1. It can be configured as digital port pin.

2.2.8 AN0 / PAD0

This pin serves as the analog input Channel 0. It can be configured as digital port pin.

2.2.9 VRH, VRL

VRH is the high reference voltage and VRL is the low reference voltage for ATD conversion.

2.2.10 VDDA, VSSA

These pins are the power supplies for the analog circuitry of the ATD_10B8C block.

Section 3 Memory Map and Register Definition

3.1 Overview

This section provides a detailed description of all registers accessible in the ATD_10B8C.

3.2 Module Memory Map

Table 3-1 gives an overview on all ATD_10B8C registers.

Table 3-1 Module Memory Map

Address Offset	Use	Access
\$_00	ATD Control Register 0 (ATDCTL0) ¹	R
\$_01	ATD Control Register 1 (ATDCTL1) ²	R
\$_02	ATD Control Register 2 (ATDCTL2)	R/W
\$_03	ATD Control Register 3 (ATDCTL3)	R/W
\$_04	ATD Control Register 4 (ATDCTL4)	R/W
\$_05	ATD Control Register 5 (ATDCTL5)	R/W
\$_06	ATD Status Register 0 (ATDSTAT0)	R/W
\$_07	Unimplemented	
\$_08	ATD Test Register 0 (ATDTEST0) ³	R
\$_09	ATD Test Register 1 (ATDTEST1)	R/W
\$_0A	Unimplemented	
\$_0B	ATD Status Register 1 (ATDSTAT1)	R
\$_0C	Unimplemented	
\$_0D	ATD Input Enable Register (ATDDIEN)	R/W
\$_0E	Unimplemented	
\$_0F	Port Data Register (PORTAD)	R
\$_10, \$_11	ATD Result Register 0 (ATDDR0H, ATDDR0L)	R/W
\$_12, \$_13	ATD Result Register 1 (ATDDR1H, ATDDR1L)	R/W
\$_14, \$_15	ATD Result Register 2 (ATDDR2H, ATDDR2L)	R/W
\$_16, \$_17	ATD Result Register 3 (ATDDR3H, ATDDR3L)	R/W
\$_18, \$_19	ATD Result Register 4 (ATDDR4H, ATDDR4L)	R/W
\$_1A, \$_1B	ATD Result Register 5 (ATDDR5H, ATDDR5L)	R/W
\$_1C, \$_1D	ATD Result Register 6 (ATDDR6H, ATDDR6L)	R/W
\$_1E, \$_1F	ATD Result Register 7 (ATDDR7H, ATDDR7L)	R/W

NOTES:

1. ATDCTL0 is intended for factory test purposes only.
2. ATDCTL1 is intended for factory test purposes only.
3. ATDTEST0 is intended for factory test purposes only.

NOTE: *Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.*

3.3 Register Descriptions

This section describes in address order all the ATD_10B8C registers and their individual bits.

3.3.1 Reserved Register (ATDCTL0)

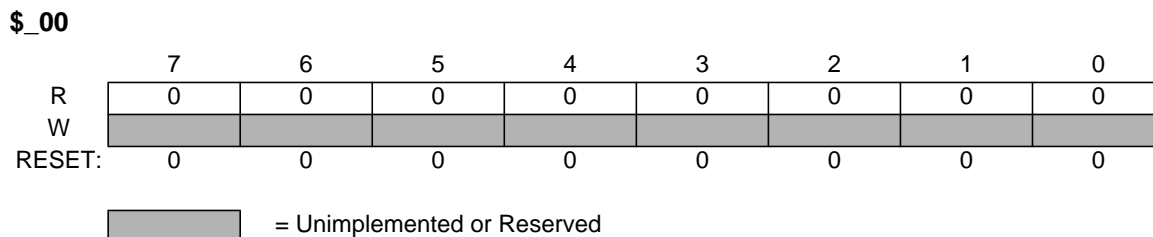


Figure 3-1 Reserved Register (ATDCTL0)

Read: always read \$00 in normal modes

Write: unimplemented in normal modes

3.3.2 Reserved Register (ATDCTL1)

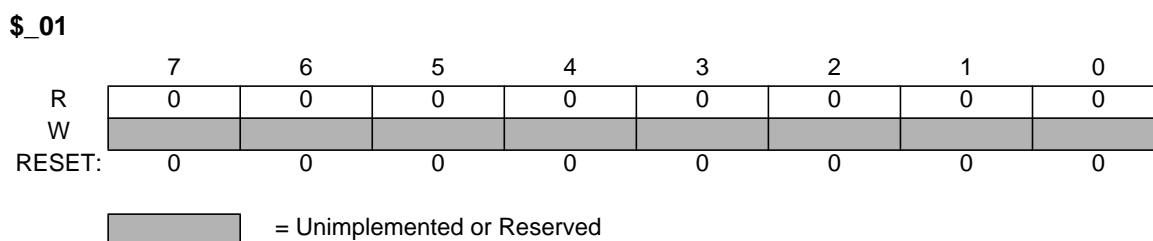


Figure 3-2 Reserved Register (ATDCTL1)

Read: always read \$00 in normal modes

Write: unimplemented in normal modes

NOTE: *Writing to this registers when in special modes can alter functionality.*

3.3.3 ATD Control Register 2 (ATDCTL2)

This register controls power down, interrupt and external trigger. Writes to this register will abort current conversion sequence but will not start a new sequence.

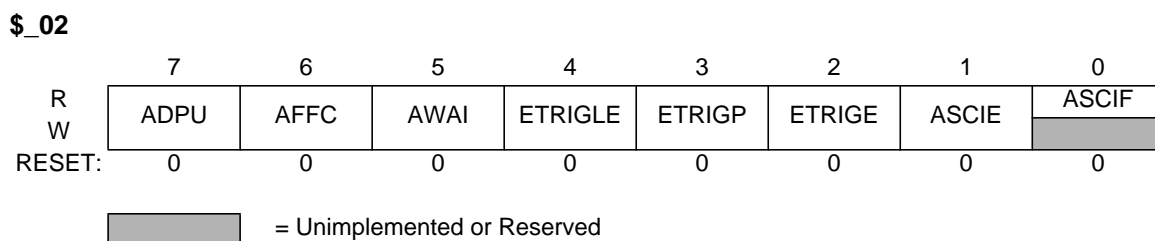


Figure 3-3 ATD Control Register 2 (ATDCTL2)

Read: anytime

Write: anytime

ADPU — ATD Power Up

This bit provides on/off control over the ATD_10B8C block allowing reduced MCU power consumption. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after ADPU bit is enabled.

- 1 = Normal ATD functionality
- 0 = Power down ATD

AFFC — ATD Fast Flag Clear All

- 1 = Changes all ATD conversion complete flags to a fast clear sequence. Any access to a result register will cause the associate CCF flag to clear automatically.
- 0 = ATD flag clearing operates normally (read the status register ATDSTAT1 before reading the result register to clear the associate CCF flag).

AWAI — ATD Power Down in Wait Mode

When entering Wait Mode this bit provides on/off control over the ATD_10B8C block allowing reduced MCU power. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after exit from Wait mode.

- 1 = Halt conversion and power down ATD during Wait mode
After exiting Wait mode with an interrupt conversion will resume. But due to the recovery time the result of this conversion should be ignored.
- 0 = ATD continues to run in Wait mode

ETRIGLE — External Trigger Level/Edge Control

This bit controls the sensitivity of the external trigger signal. See **Table 3-2** for details.

ETRIGP — External Trigger Polarity

This bit controls the polarity of the external trigger signal. See **Table 3-2** for details.

Table 3-2 External Trigger Configurations

ETRIGLE	ETRIGP	External Trigger Sensitivity
0	0	falling edge
0	1	rising edge
1	0	low level
1	1	high level

ETRIGLE — External Trigger Mode Enable

This bit enables the external trigger on ATD channel 7. The external trigger allows to synchronize sample and ATD conversions processes with external events.

- 1 = Enable external trigger
- 0 = Disable external trigger

NOTE: *The conversion results for the external trigger ATD channel 7 have no meaning while external trigger mode is enabled.*

ASCIE — ATD Sequence Complete Interrupt Enable

- 1 = ATD Interrupt will be requested whenever ASCIF=1 is set.
- 0 = ATD Sequence Complete interrupt requests are disabled.

ASCIF — ATD Sequence Complete Interrupt Flag

If ASCIE=1 the ASCIF flag equals the SCF flag (see **3.3.7**), else ASCIF reads zero. Writes have no effect.

- 1 = ATD sequence complete interrupt pending
- 0 = No ATD interrupt occurred

3.3.4 ATD Control Register 3 (ATDCTL3)

This register controls the conversion sequence length, FIFO for results registers and behavior in Freeze Mode. Writes to this register will abort current conversion sequence but will not start a new sequence.

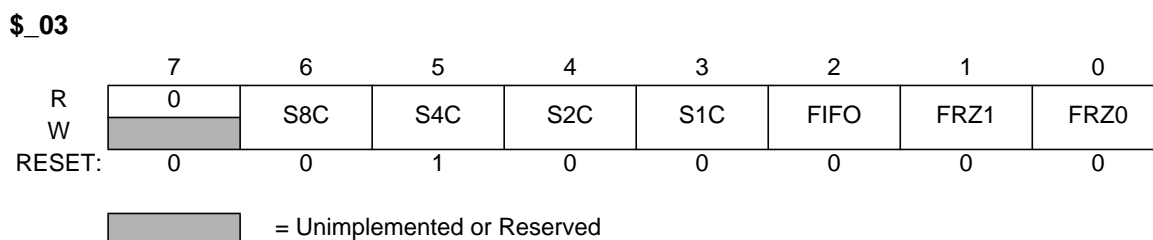


Figure 3-4 ATD Control Register 3 (ATDCTL3)

Read: anytime

Write: anytime

S8C, S4C, S2C, S1C — Conversion Sequence Length

These bits control the number of conversions per sequence. **Table 3-3** shows all combinations. At reset, S4C is set to 1 (sequence length is 4). This is to maintain software continuity to HC12 family.

Table 3-3 Conversion Sequence Length Coding

S8C	S4C	S2C	S1C	Number of Conversions per Sequence
0	0	0	0	8
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	X	X	X	8

FIFO — Result Register FIFO Mode

If this bit is zero (non-FIFO mode), the A/D conversion results map into the result registers based on the conversion sequence; the result of the first conversion appears in the first result register, the second result in the second result register, and so on.

If this bit is one (FIFO mode) the conversion counter is not reset at the beginning or ending of a conversion sequence; sequential conversion results are placed in consecutive result registers. In a continuously scanning conversion sequence, the result register counter will wrap around when it reaches the end of the result register file. The conversion counter value (CC2-0 in ATDSTAT0) can be used to determine where in the result register file, the current conversion result will be placed. Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1. So the first result of a new conversion sequence, started by writing to ATDCTL5, will always be placed in the first result register (ATDDDR0). Intended usage of FIFO mode is continuous conversion (SCAN=1) or triggered conversion (ETRIG=1).

Which result registers hold valid data can be tracked using the conversion complete flags. Fast flag clear mode may or may not be useful in a particular application to track valid data.

1 = Conversion results are placed in consecutive result registers (wrap around at end).

0 = Conversion results are placed in the corresponding result register up to the selected sequence length.

FRZ1, FRZ0 — Background Debug Freeze Enable

When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint (Freeze Mode) is encountered. These 2 bits determine how the ATD will respond to a breakpoint as shown in **Table 3-4**. Leakage onto the storage node and comparator reference capacitors may compromise the accuracy of an immediately frozen conversion depending on the length of the freeze period.

Table 3-4 ATD Behavior in Freeze Mode (breakpoint)

FRZ1	FRZ0	Behavior in Freeze mode
0	0	Continue conversion
0	1	Reserved
1	0	Finish current conversion, then freeze
1	1	Freeze Immediately

3.3.5 ATD Control Register 4 (ATDCTL4)

This register selects the conversion clock frequency, the length of the second phase of the sample time and the resolution of the A/D conversion (i.e.: 8-bits or 10-bits). Writes to this register will abort current conversion sequence but will not start a new sequence.

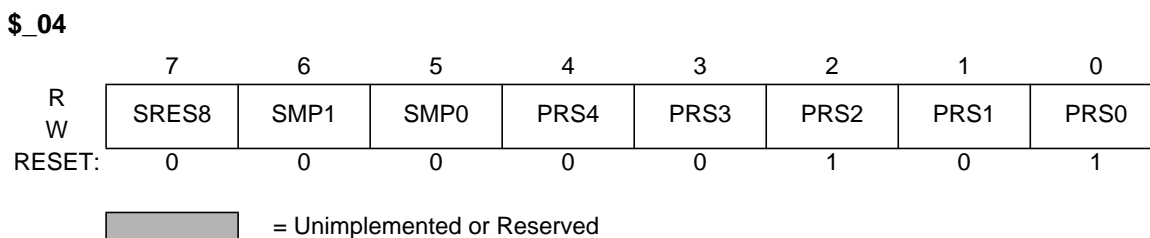


Figure 3-5 ATD Control Register 4 (ATDCTL4)

Read: anytime

Write: anytime

SRES8 — A/D Resolution Select

This bit selects the resolution of A/D conversion results as either 8 or 10 bits. The A/D converter has an accuracy of 10 bits; however, if low resolution is required, the conversion can be speeded up by selecting 8-bit resolution.

1 = 8 bit resolution

0 = 10 bit resolution

SMP1, SMP0 — Sample Time Select

These two bits select the length of the second phase of the sample time in units of ATD conversion clock cycles. Note that the ATD conversion clock period is itself a function of the prescaler value (bits PRS4-0). The sample time consists of two phases. The first phase is two ATD conversion clock cycles long and transfers the sample quickly (via the buffer amplifier) onto the A/D machine’s storage node. The second phase attaches the external analog signal directly to the storage node for final charging and high accuracy. **Table 3-5** lists the lengths available for the second sample phase.

Table 3-5 Sample Time Select

SMP1	SMP0	Length of 2nd phase of sample time
0	0	2 A/D conversion clock periods
0	1	4 A/D conversion clock periods
1	0	8 A/D conversion clock periods
1	1	16 A/D conversion clock periods

PRS4, PRS3, PRS2, PRS1, PRS0 — ATD Clock Prescaler

These 5 bits are the binary value prescaler value PRS. The ATD conversion clock frequency is calculated as follows:

$$\text{ATDclock} = \frac{[\text{BusClock}]}{[\text{PRS} + 1]} \times 0.5$$

Note that the maximum ATD conversion clock frequency is half the Bus Clock. The default (after reset) prescaler value is 5 which results in a default ATD conversion clock frequency that is Bus Clock divided by 12. **Table 3-6** illustrates the divide-by operation and the appropriate range of the Bus Clock.

Table 3-6 Clock Prescaler Values

Prescale Value	Total Divisor Value	Max. Bus Clock ¹	Min. Bus Clock ²
00000	divide by 2	4 MHz	1 MHz
00001	divide by 4	8 MHz	2 MHz
00010	divide by 6	12 MHz	3 MHz
00011	divide by 8	16 MHz	4 MHz
00100	divide by 10	20 MHz	5 MHz
00101	divide by 12	24 MHz	6 MHz
00110	divide by 14	28 MHz	7 MHz
00111	divide by 16	32 MHz	8 MHz
01000	divide by 18	36 MHz	9 MHz
01001	divide by 20	40 MHz	10 MHz
01010	divide by 22	44 MHz	11 MHz
01011	divide by 24	48 MHz	12 MHz
01100	divide by 26	52 MHz	13 MHz
01101	divide by 28	56 MHz	14 MHz
01110	divide by 30	60 MHz	15 MHz
01111	divide by 32	64 MHz	16 MHz
10000	divide by 34	68 MHz	17 MHz
10001	divide by 36	72 MHz	18 MHz
10010	divide by 38	76 MHz	19 MHz
10011	divide by 40	80 MHz	20 MHz
10100	divide by 42	84 MHz	21 MHz
10101	divide by 44	88 MHz	22 MHz
10110	divide by 46	92 MHz	23 MHz
10111	divide by 48	96 MHz	24 MHz
11000	divide by 50	100 MHz	25 MHz
11001	divide by 52	104 MHz	26 MHz
11010	divide by 54	108 MHz	27 MHz
11011	divide by 56	112 MHz	28 MHz
11100	divide by 58	116 MHz	29 MHz
11101	divide by 60	120 MHz	30 MHz
11110	divide by 62	124 MHz	31 MHz
11111	divide by 64	128 MHz	32 MHz

NOTE:

1. Maximum ATD conversion clock frequency is 2MHz. The maximum allowed Bus Clock frequency is shown in this column.
2. Minimum ATD conversion clock frequency is 500KHz. The minimum allowed Bus Clock frequency is shown in this column.

3.3.6 ATD Control Register 5 (ATDCTL5)

This register selects the type of conversion sequence and the analog input channels sampled. Writes to this register will abort current conversion sequence and start a new conversion sequence.

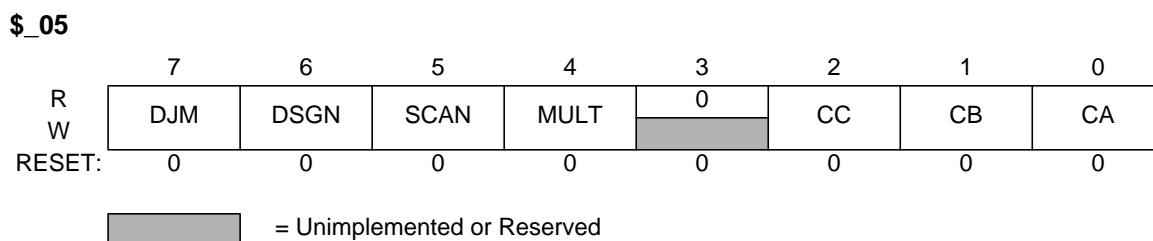


Figure 3-6 ATD Control Register 5 (ATDCTL5)

Read: anytime

Write: anytime

DJM — Result Register Data Justification

This bit controls justification of conversion data in the result registers. See **3.3.13 ATD Conversion Result Registers (ATDDRHx/ATDDRLx)** for details.

- 1 = Right justified data in the result registers
- 0 = Left justified data in the result registers

DSGN — Result Register Data Signed or Unsigned Representation

This bit selects between signed and unsigned conversion data representation in the result registers. Signed data is represented as 2's complement. Signed data is not available in right justification. See **3.3.13 ATD Conversion Result Registers (ATDDRHx/ATDDRLx)** for details.

- 1 = Signed data representation in the result registers
- 0 = Unsigned data representation in the result registers

Table 3-7 summarizes the result data formats available and how they are set up using the control bits.

Table 3-8 illustrates the difference between the signed and unsigned, left justified output codes for an input signal range between 0 and 5.12 Volts.

Table 3-7 Available Result Data Formats

SRES8	DJM	DSGN	Result Data Formats Description and Bus Bit Mapping
1	0	0	8-bit / left justified / unsigned - bits 8-15
1	0	1	8-bit / left justified / signed - bits 8-15
1	1	X	8-bit / right justified / unsigned - bits 0-7
0	0	0	10-bit / left justified / unsigned - bits 6-15
0	0	1	10-bit / left justified / signed - bits 6-15
0	1	X	10-bit / right justified / unsigned - bits 0-9

Table 3-8 Left Justified, Signed and Unsigned ATD Output Codes.

Input Signal Vrl = 0 Volts Vrh = 5.12 Volts	Signed 8-Bit Codes	Unsigned 8-Bit Codes	Signed 10-Bit Codes	Unsigned 10-Bit Codes
5.120 Volts	7F	FF	7FC0	FFC0
5.100	7F	FF	7F00	FF00
5.080	7E	FE	7E00	FE00
2.580	01	81	0100	8100
2.560	00	80	0000	8000
2.540	FF	7F	FF00	7F00
0.020	81	01	8100	0100
0.000	80	00	8000	0000

SCAN — Continuous Conversion Sequence Mode

This bit selects whether conversion sequences are performed continuously or only once.

- 1 = Continuous conversion sequences (scan mode)
- 0 = Single conversion sequence

MULT — Multi-Channel Sample Mode

When MULT is 0, the ATD sequence controller samples only from the specified analog input channel for an entire conversion sequence. The analog channel is selected by channel selection code (control bits CC/CB/CA located in ATDCTL5). When MULT is 1, the ATD sequence controller samples across channels. The number of channels sampled is determined by the sequence length value (S8C, S4C, S2C, S1C). The first analog channel examined is determined by channel selection code (CC, CB, CA control bits); subsequent channels sampled in the sequence are determined by incrementing the channel selection code.

- 1 = Sample across several channels
- 0 = Sample only one channel

CC, CB, CA — Analog Input Channel Select Code

These bits select the analog input channel(s) whose signals are sampled and converted to digital codes. **Table 3-9** lists the coding used to select the various analog input channels. In the case of single channel scans (MULT=0), this selection code specified the channel examined. In the case of multi-channel scans (MULT=1), this selection code represents the first channel to be examined in the conversion sequence. Subsequent channels are determined by incrementing channel selection code; selection codes that reach the maximum value wrap around to the minimum value.

Table 3-9 Analog Input Channel Select Coding

CC	CB	CA	Analog Input Channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3

Table 3-9 Analog Input Channel Select Coding

CC	CB	CA	Analog Input Channel
1	0	0	AN4
1	0	1	AN5
1	1	0	AN6
1	1	1	AN7

3.3.7 ATD Status Register 0 (ATDSTAT0)

This read-only register contains the Sequence Complete Flag, overrun flags for external trigger and FIFO mode, and the conversion counter.

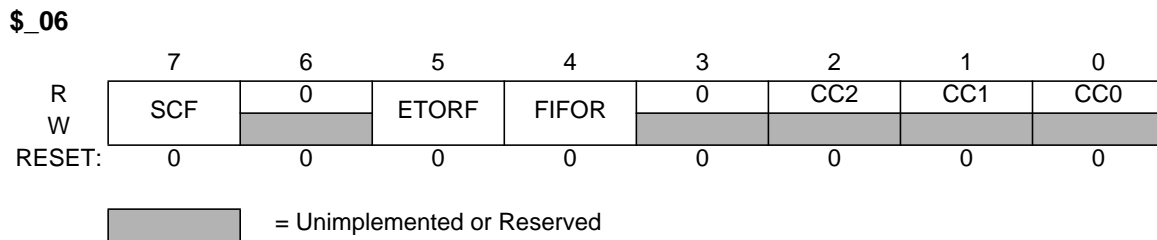


Figure 3-7 ATD Status Register 0 (ATDSTAT0)

Read: anytime

Write: anytime (No effect on (CC2, CC1, CC0))

SCF — Sequence Complete Flag

This flag is set upon completion of a conversion sequence. If conversion sequences are continuously performed (SCAN=1), the flag is set after each one is completed. This flag is cleared when one of the following occurs:

- A) Write “1” to SCF
- B) Write to ATDCTL5 (a new conversion sequence is started)
- C) If AFFC=1 and read of a result register
 - 1 = Conversion sequence has completed
 - 0 = Conversion sequence not completed

ETORF — External Trigger Overrun Flag

While in edge trigger mode (ETRIGLE=0), if additional active edges are detected while a conversion sequence is in process the overrun flag is set. This flag is cleared when one of the following occurs:

- A) Write “1” to ETORF
- B) Write to ATDCTL2, ATDCTL3 or ATDCTL4 (a conversion sequence is aborted)
- C) Write to ATDCTL5 (a new conversion sequence is started)
 - 1 = External trigger over run error has occurred

0 = No External trigger over run error has occurred

FIFOR - FIFO Over Run Flag.

This bit indicates that a result register has been written to before its associated conversion complete flag (CCF) has been cleared. This flag is most useful when using the FIFO mode because the flag potentially indicates that result registers are out of sync with the input channels. However, it is also practical for non-FIFO modes, and indicates that a result register has been over written before it has been read (i.e. the old data has been lost). This flag is cleared when one of the following occurs:

- A) Write “1” to FIFOR
- B) Start a new conversion sequence (write to ATDCTL5 or external trigger)
 - 1 = An over run condition exists
 - 0 = No over run has occurred

CC2, CC1, CC0 — Conversion Counter

These 3 read-only bits are the binary value of the conversion counter. The conversion counter points to the result register that will receive the result of the current conversion. E.g. CC2=1, CC1=1, CC0=0 indicates that the result of the current conversion will be in ATD Result Register 6. If in non-FIFO mode (FIFO=0) the conversion counter is initialized to zero at the begin and end of the conversion sequence. If in FIFO mode (FIFO=1) the register counter is not initialized. The conversion counters wraps around when its maximum value is reached.

Aborting a conversion or starting a new conversion by write to an ATDCTL register (ATDCTL5-0) clears the conversion counter even if FIFO=1.

3.3.8 Reserved Register (ATDTEST0)

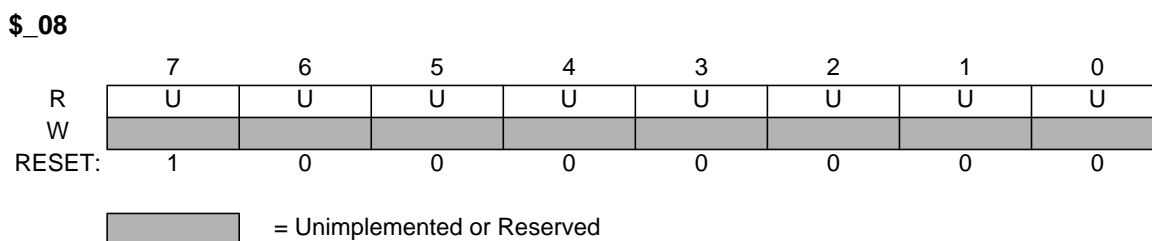


Figure 3-8 Reserved Register (ATDTEST0)

Read: anytime, returns unpredictable values

Write: anytime in special modes, unimplemented in normal modes

NOTE: Writing to this registers when in special modes can alter functionality.

3.3.9 ATD Test Register 1 (ATDTEST1)

This register contains the SC bit used to enable special channel conversions.

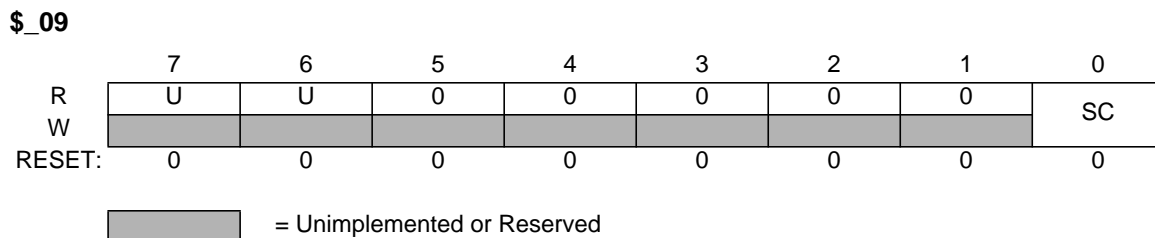


Figure 3-9 ATD Test Register 1 (ATDTEST1)

Read: anytime, returns unpredictable values for Bit7 and Bit6

Write: anytime

SC - Special Channel Conversion Bit

If this bit is set, then special channel conversion can be selected using CC, CB and CA of ATDCTL5.

Table 3-10 lists the coding.

1 = Special channel conversions enabled

0 = Special channel conversions disabled

NOTE: Always write remaining bits of ATDTEST1 (Bit7 to Bit1) zero when writing SC bit. Not doing so might result in unpredictable ATD behavior.

Table 3-10 Special Channel Select Coding

SC	CC	CB	CA	Analog Input Channel
1	0	X	X	Reserved
1	1	0	0	V _{RH}
1	1	0	1	V _{RL}
1	1	1	0	(V _{RH} +V _{RL}) / 2
1	1	1	1	Reserved

3.3.10 ATD Status Register 1 (ATDSTAT1)

This read-only register contains the Conversion Complete Flags.

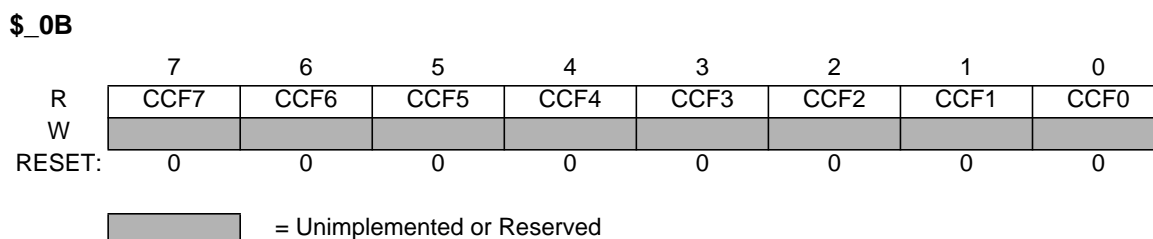


Figure 3-10 ATD Status Register 1 (ATDSTAT1)

Read: anytime

Write: anytime, no effect

CCF_x — Conversion Complete Flag x (x=7,6,5,4,3,2,1,0)

A conversion complete flag is set at the end of each conversion in a conversion sequence. The flags are associated with the conversion position in a sequence (and also the result register number). Therefore, CCF0 is set when the first conversion in a sequence is complete and the result is available in result register ATDDR0; CCF1 is set when the second conversion in a sequence is complete and the result is available in ATDDR1, and so forth. A flag CCF_x (x=7,6,5,4,3,2,1,0) is cleared when one of the following occurs:

- A) Write to ATDCTL5 (a new conversion sequence is started)
- B) If AFFC=0 and read of ATDSTAT1 followed by read of result register ATDDR_x
- C) If AFFC=1 and read of result register ATDDR_x
 - 1 = Conversion number x has completed, result ready in ATDDR_x
 - 0 = Conversion number x not completed

3.3.11 ATD Input Enable Register (ATDDIEN)

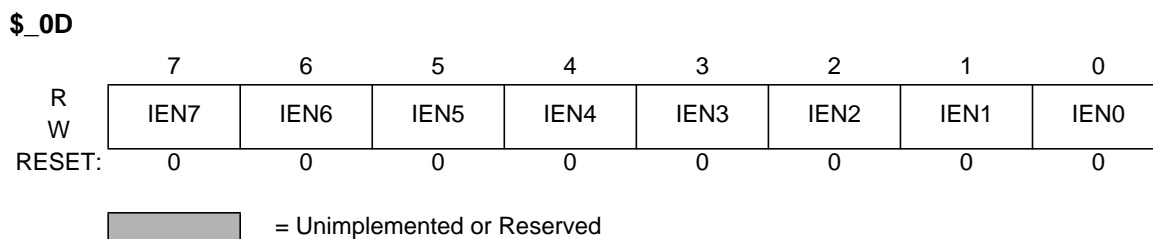


Figure 3-11 ATD Input Enable Register (ATDDIEN)

Read: anytime

Write: anytime

IEN_x — ATD Digital Input Enable on channel x (x= 7, 6, 5, 4, 3, 2, 1, 0)

This bit controls the digital input buffer from the analog input pin (AN_x) to PTAD_x data register.
 1 = Enable digital input buffer to PTAD_x.

0 = Disable digital input buffer to PTADx

NOTE: *Setting this bit will enable the corresponding digital input buffer continuously. If this bit is set while simultaneously using it as an analog port, there is potentially increased power consumption because the digital input buffer maybe in the linear region.*

3.3.12 Port Data Register (PORTAD)

The digital port pins are shared with the analog A/D inputs AN7-0.

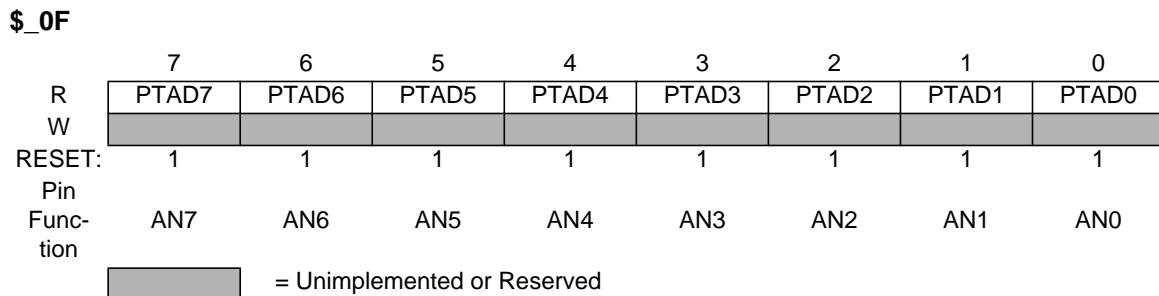


Figure 3-12 Port Data Register (PORTAD)

Read: anytime

Write: anytime, no effect

PTADx — A/D Channel x (ANx) Digital Input (x= 7,6,5,4,3,2,1,0)

If the digital input buffer on the ANx pin is enabled (IENx=1) read returns the logic level on ANx pin (signal potentials not meeting VIL or VIH specifications will have an indeterminate value)).

If the digital input buffers are disabled (IENx=0), read returns a “1”.

Reset sets all PORTAD bits to “1”.

3.3.13 ATD Conversion Result Registers (ATDDRHx/ATDDRLx)

The A/D conversion results are stored in 8 read-only result registers ATDDRHx/ATDDRLx. The result data is formatted in the result registers based on two criteria. First there is left and right justification; this selection is made using the DJM control bit in ATDCTL5. Second there is signed and unsigned data; this selection is made using the DSGN control bit in ATDCTL5. Signed data is stored in 2’s complement format and only exists in left justified format. Signed data selected for right justified format is ignored.

Read: anytime

Write: anytime, no effect in normal modes

3.3.13.1 Left Justified Result Data

\$_10 = ATDDR0H, \$_12 = ATDDR1H, \$_14 = ATDDR2H, \$_16 = ATDDR3H
\$_18 = ATDDR4H, \$_1A = ATDDR5H, \$_1C = ATDDR6H, \$_1E = ATDDR7H

	7	6	5	4	3	2	1	0	
R	BIT 9 MSB	BIT 8	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	10-bit data
W	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	8-bit data
RESET:	0	0	0	0	0	0	0	0	


 = Unimplemented or Reserved

Figure 3-13 Left Justified, ATD Conversion Result Register, High Byte (ATDDRxH)

\$_11 = ATDDR0L, \$_13 = ATDDR1L, \$_15 = ATDDR2L, \$_17 = ATDDR3L
\$_19 = ATDDR4L, \$_1B = ATDDR5L, \$_1D = ATDDR6L, \$_1F = ATDDR7L

	7	6	5	4	3	2	1	0	
R	BIT 1	BIT 0	0	0	0	0	0	0	10-bit data
W	U	U	0	0	0	0	0	0	8-bit data
RESET:	0	0	0	0	0	0	0	0	

 = Unimplemented or Reserved

Figure 3-14 Left Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)

3.3.13.2 Right Justified Result Data

\$_10 = ATDDR0H, \$_12 = ATDDR1H, \$_14 = ATDDR2H, \$_16 = ATDDR3H
\$_18 = ATDDR4H, \$_1A = ATDDR5H, \$_1C = ATDDR6H, \$_1E = ATDDR7H

	7	6	5	4	3	2	1	0	
R	0	0	0	0	0	0	BIT 9 MSB	BIT 8	10-bit data
W	0	0	0	0	0	0	0	0	8-bit data
RESET:	0	0	0	0	0	0	0	0	

 = Unimplemented or Reserved

Figure 3-15 Right Justified, ATD Conversion Result Register, High Byte (ATDDRxH)

\$_11 = ATDDR0L, \$_13 = ATDDR1L, \$_15 = ATDDR2L, \$_17 = ATDDR3L
 \$_19 = ATDDR4L, \$_1B = ATDDR5L, \$_1D = ATDDR6L, \$_1F = ATDDR7L

	7	6	5	4	3	2	1	0	
R	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	10-bit data
W	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	8-bit data
RESET:	0	0	0	0	0	0	0	0	


 = Unimplemented or Reserved

Figure 3-16 Right Justified, ATD Conversion Result Register, Low Byte (ATDDRxL)

Section 4 Functional Description

4.1 General

The ATD_10B8C is structured in an analog and a digital sub-block.

4.2 Analog Sub-block

The analog sub-block contains all analog electronics required to perform a single conversion. Separate power supplies VDDA and VSSA allow to isolate noise of other MCU circuitry from the analog sub-block.

4.2.1 Sample and Hold Machine

The Sample and Hold (S/H) Machine accepts analog signals from the external surroundings and stores them as capacitor charge on a storage node.

The sample process uses a two stage approach. During the first stage, the sample amplifier is used to quickly charge the storage node. The second stage connects the input directly to the storage node to complete the sample for high accuracy.

When not sampling, the sample and hold machine disables its own clocks. The analog electronics still draw their quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

The input analog signals are unipolar and must fall within the potential range of VSSA to VDDA.

4.2.2 Analog Input Multiplexer

The analog input multiplexer connects one of the 8 external analog input channels to the sample and hold machine.

4.2.3 Sample Buffer Amplifier

The sample amplifier is used to buffer the input analog signal so that the storage node can be quickly charged to the sample potential.

4.2.4 Analog-to-Digital (A/D) Machine

The A/D Machine performs analog to digital conversions. The resolution is program selectable at either 8 or 10 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the stored analog sample potential with a series of digitally generated analog potentials. By following a binary search algorithm, the A/D machine locates the approximating potential that is nearest to the sampled potential.

When not converting the A/D machine disables its own clocks. The analog electronics still draws quiescent current. The power down (ADPU) bit must be set to disable both the digital clocks and the analog power consumption.

Only analog input signals within the potential range of V_{RL} to V_{RH} (A/D reference potentials) will result in a non-railed digital output codes.

4.3 Digital Sub-block

This subsection explains some of the digital features in more detail. See register descriptions for all details.

4.3.1 External Trigger Input (ETRIG)

The external trigger feature allows the user to synchronize ATD conversions to the external environment events rather than relying on software to signal the ATD module when ATD conversions are to take place. The input signal (ATD channel 7) is programmable to be edge or level sensitive with polarity control. **Table 4-1** gives a brief description of the different combinations of control bits and their affect on the external trigger function.

Table 4-1 External Trigger Control Bits

ETRIGLE	ETRIGP	ETRIGE	SCAN	Description
X	X	0	0	Ignores external trigger. Performs one conversion sequence and stops.
X	X	0	1	Ignores external trigger. Performs continuous conversion sequences.
0	0	1	X	Falling edge triggered. Performs one conversion sequence per trigger.
0	1	1	X	Rising edge triggered. Performs one conversion sequence per trigger.
1	0	1	X	Trigger active low. Performs continuous conversions while trigger is active.
1	1	1	X	Trigger active high. Performs continuous conversions while trigger is active.

During a conversion, if additional active edges are detected the overrun error flag ETORF is set.

In either level or edge triggered modes, the first conversion begins when the trigger is received. In both cases, the maximum latency time is one Bus Clock cycle plus any skew or delay introduced by the trigger circuitry.

NOTE: *The conversion results for the external trigger ATD channel 7 have no meaning while external trigger mode is enabled.*

Once ETRIGE is enabled, conversions cannot be started by a write to ATDCTL5, but rather must be triggered externally.

If the level mode is active and the external trigger both de-asserts and re-asserts itself during a conversion sequence, this does not constitute an overrun; therefore, the flag is not set. If the trigger is left asserted in level mode while a sequence is completing, another sequence will be triggered immediately.

4.3.2 General Purpose Digital Input Port Operation

The input channel pins can be multiplexed between analog and digital data. As analog inputs, they are multiplexed and sampled to supply signals to the A/D converter. As digital inputs, they supply external input data that can be accessed through the digital port register PORTAD (input-only).

The analog/digital multiplex operation is performed in the input pads. The input pad is always connected to the analog inputs of the ATD_10B8C. The input pad signal is buffered to the digital port registers. This buffer can be turned on or off with the ATDDIEN register. This is important so that the buffer does not draw excess current when analog potentials are presented at its input.

4.3.3 Low Power Modes

The ATD_10B8C can be configured for lower MCU power consumption in 3 different ways:

- **Stop Mode:** This halts A/D conversion. Exit from Stop mode will resume A/D conversion, But due to the recovery time the result of this conversion should be ignored.
- **Wait Mode with AWAI=1:** This halts A/D conversion. Exit from Wait mode will resume A/D conversion, but due to the recovery time the result of this conversion should be ignored.
- **Writing ADPU=0 (Note that all ATD registers remain accessible.):** This aborts any A/D conversion in progress.

Note that the reset value for the ADPU bit is zero. Therefore, when this module is reset, it is reset into the power down state.

Section 5 Resets

5.1 General

At reset the ATD_10B8C is in a power down state. The reset state of each individual bit is listed within the Register Description section (see **Section 3 Memory Map and Register Definition**) which details the registers and their bit-field.

Section 6 Interrupts

6.1 General

The interrupt requested by the ATD_10B8C is listed in **Table 6-1**. Refer to MCU specification for related vector address and priority.

Table 6-1 ATD_10B8C Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Sequence Complete Interrupt	I bit	ASCIE in ATDCTL2

See register descriptions for further details.

User Guide End Sheet

**FINAL PAGE OF
40
PAGES**


MSCAN

Block Guide

V02.15

Original Release Date: 19 MAY 1998
Revised: 15 JUL 2004

Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

©Motorola, Inc., 2001

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V02.08	17 JUL 2001	17 JUL 2001		- 1st official version by Technical Publishing
V02.09	10 JUL 2001	10 JUL 2001		- Updated according to requirements of SRSv3 - Corrected footnote 1 in INITRQ description.
V02.10	10 OCT 2001	10 OCT 2001		- Replaced all references w.r.t. new family name HCS12.
V02.11	22 OCT 2001	22 OCT 2001		- Corrected figure title and note of CANTIER. - Corrected local enable register names in table 4-4 'CRG Interrupt Vectors'. - Updated block diagram. - Corrected section 'Description of Interrupt Operation'.
V02.12	04 MAR 2002	04 MAR 2002		- Document format updates.
V02.13	22 JUL 2002	22 JUL 2002		- Corrected TBPR register offset. - Corrected Table 'Message Buffer Organization'. - Corrected SLPRQ bit description. - Corrected MSCAN Sleep Mode description. - Updated WUPE bit description. - Updated Simplified State Transitions figure. - Updated Recovery from STOP or WAIT description and CPU vs. MSCAN Modes table.
V02.14	18 SEP 2002	18 SEP 2002		- Added Initialization/Application information. - Replaced 'MCU' with 'CPU' in several places. - Cleaned up Mode descriptions. - General cleanup.
V02.15	15 JUL 2004	15 JUL 2004		- Corrected buffer read/write access definitions. - Corrected bit time equation.

Table of Contents

Section 1 Introduction

1.1	Overview	11
1.2	Features	12

Section 2 External Signal Description

2.1	Overview	13
2.2	Detailed Signal Description	13
2.2.1	RXCAN — CAN Receiver Input Pin	13
2.2.2	TXCAN — CAN Transmitter Output Pin	13
2.3	CAN System	13

Section 3 Memory Map/Register Definition

3.1	Overview	15
3.2	Module Memory Map	15
3.3	Register Descriptions	16
3.3.1	Programmer's Model of Control Registers	17
3.3.2	Programmer's Model of Message Storage	38

Section 4 Functional Description

4.1	General	45
4.2	Message Storage	45
4.2.1	Message Transmit Background	46
4.2.2	Transmit Structures	46
4.2.3	Receive Structures	47
4.3	Identifier Acceptance Filter	48
4.3.1	Protocol Violation Protection	52
4.3.2	Clock System	52
4.4	Timer Link	55
4.5	Modes of Operation	55
4.5.1	Normal Modes	55
4.5.2	Special Modes	55
4.5.3	Emulation Modes	55
4.5.4	Listen-Only Mode	55

- 4.5.5 Security Modes 56
- 4.6 Low Power Options 56
 - 4.6.1 CPU Run Mode 56
 - 4.6.2 CPU Wait Mode 57
 - 4.6.3 CPU Stop Mode 57
 - 4.6.4 MSCAN Sleep Mode 57
 - 4.6.5 MSCAN Initialization Mode 59
 - 4.6.6 MSCAN Power Down Mode 60
 - 4.6.7 Programmable Wake-Up Function 61
- 4.7 Reset Initialization 61
- 4.8 General 61
- 4.9 Description of Interrupt Operation 62
 - 4.9.1 Transmit Interrupt 62
 - 4.9.2 Receive Interrupt 62
 - 4.9.3 Wake-Up Interrupt 62
 - 4.9.4 Error Interrupt 62
- 4.10 Interrupt Acknowledge 62
- 4.11 Recovery from STOP or WAIT 63

Section 5 Initialization/Application Information

- 5.1 MSCAN initialization 63

List of Figures

Figure 1-1	MSCAN Block Diagram	11
Figure 2-1	The CAN System	14
Figure 3-1	MSCAN Control 0 Register (CANCTL0)	17
Figure 3-2	MSCAN Control 1 Register (CANCTL1)	20
Figure 3-3	MSCAN Bus Timing Register 0 (CANBTR0)	21
Figure 3-4	MSCAN Bus Timing Register 1 (CANBTR1)	22
Figure 3-5	MSCAN Receiver Flag Register (CANRFLG)	24
Figure 3-6	MSCAN Receiver Interrupt Enable Register (CANRIER)	26
Figure 3-7	MSCAN Transmitter Flag Register (CANTFLG)	28
Figure 3-8	MSCAN Transmitter Interrupt Enable Register (CANTIER)	29
Figure 3-9	MSCAN Transmitter Message Abort Request (CANTARQ)	29
Figure 3-10	MSCAN Transmitter Message Abort Control (CANTAACK)	30
Figure 3-11	MSCAN Transmitter Flag Register (CANTBSEL)	31
Figure 3-12	MSCAN Identifier Acceptance Control Register (CANIDAC)	32
Figure 3-13	Reserved Registers	33
Figure 3-14	MSCAN Receive Error Counter Register (CANRXERR)	34
Figure 3-15	MSCAN Transmit Error Counter Register (CANTXERR)	34
Figure 3-16	MSCAN Identifier Acceptance Registers (1st Bank)	35
Figure 3-17	MSCAN Identifier Acceptance Registers (2nd Bank)	36
Figure 3-18	MSCAN Identifier Mask Registers (1st Bank)	37
Figure 3-19	MSCAN Identifier Mask Registers (2nd Bank)	37
Figure 3-20	Receive / Transmit Message Buffer Extended Identifier	39
Figure 3-21	Standard Identifier Mapping	40
Figure 3-22	Transmit Buffer Priority Register (TBPR)	43
Figure 3-23	Time Stamp Register (TSRH - High Byte)	43
Figure 3-24	Time Stamp Register (TSRL - Low Byte)	44
Figure 4-1	User Model for Message Buffer Organization	45
Figure 4-2	32-bit Maskable Identifier Acceptance Filter	50
Figure 4-3	16-bit Maskable Identifier Acceptance Filters	50
Figure 4-4	8-bit Maskable Identifier Acceptance Filters	51
Figure 4-5	MSCAN Clocking Scheme	52
Figure 4-6	Segments within the Bit Time	54
Figure 4-7	Sleep Request / Acknowledge Cycle	57

Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode 59

Figure 4-9 Initialization Request/Acknowledge Cycle 60

List of Tables

Table 3-1	MSCAN Register Organization	15
Table 3-2	Module Memory Map	15
Table 3-3	Synchronization Jump Width	22
Table 3-4	Baud Rate Prescaler	22
Table 3-5	Time Segment 2 Values.	23
Table 3-6	Time Segment 1 Values.	24
Table 3-7	Identifier Acceptance Mode Settings	32
Table 3-8	Identifier Acceptance Hit Indication	33
Table 3-9	Message Buffer Organization	39
Table 3-10	Data length codes	42
Table 4-1	Time Segment Syntax	54
Table 4-2	CAN Standard Compliant Bit Time Segment Settings.	55
Table 4-3	CPU vs. MSCAN Operating Modes	56
Table 4-4	CRG Interrupt Vectors	61

Preface

Terminology

Acronyms and Abbreviations	
ACK	Acknowledge
CAN	Controller Area Network
CRC	Cyclic Redundancy Code
EOF	End of Frame
FIFO	First-In-First-Out Memory
IFS	Inter-Frame Sequence
MSCAN	Motorola Scalable CAN Module
SOF	Start of Frame

Section 1 Introduction

1.1 Overview

The Motorola Scalable Controller Area Network (MSCAN) definition is based on the MSCAN12 definition which is the specific implementation of the Motorola Scalable CAN concept targeted for the Motorola MC68HC12 Microcontroller Family.

The module is a communication controller implementing the CAN 2.0 A/B protocol as defined in the BOSCH specification dated September 1991. For users to fully understand the MSCAN specification, it is recommended that the Bosch specification be read first to familiarize the reader with the terms and concepts contained within this document.

The CAN protocol was primarily, but not only, designed to be used as a vehicle serial data bus, meeting the specific requirements of this field: real-time processing, reliable operation in the EMI environment of a vehicle, cost-effectiveness and required bandwidth.

MSCAN utilizes an advanced buffer arrangement resulting in a predictable real-time behavior and simplifies the application software.

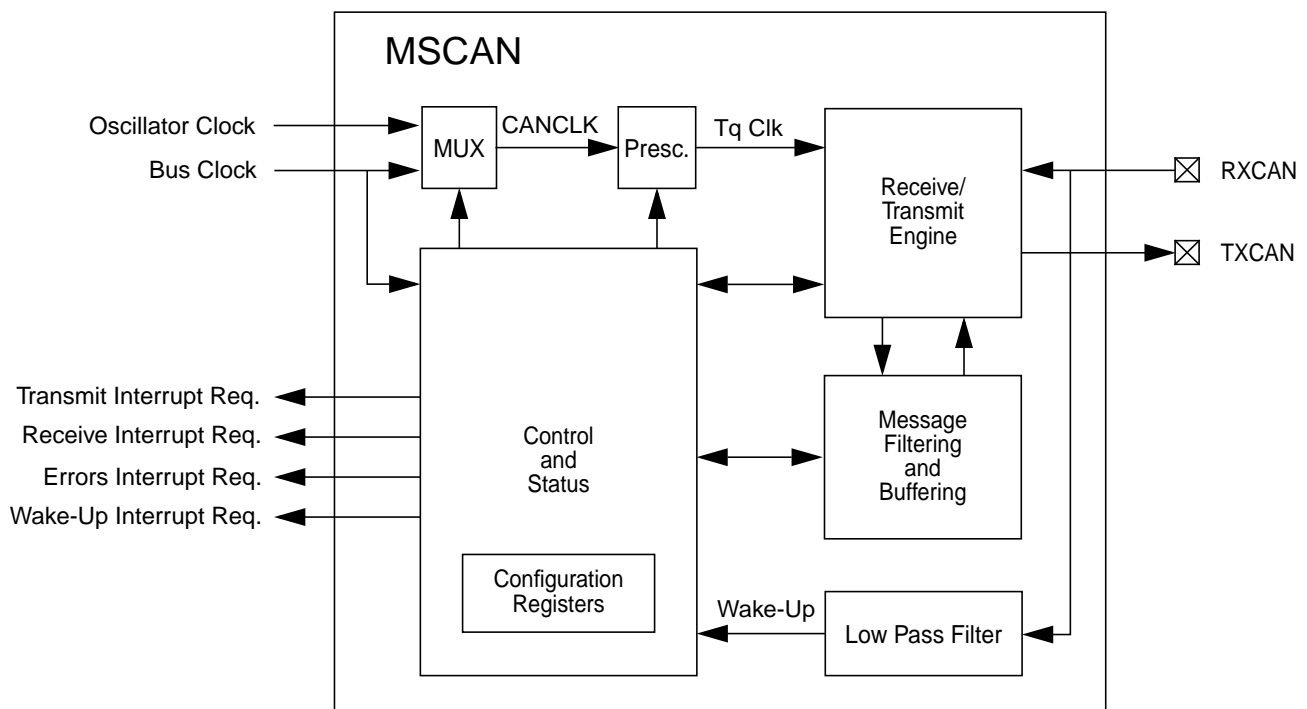


Figure 1-1 MSCAN Block Diagram

1.2 Features

The basic features of the MSCAN are as follows:

- Implementation of the CAN protocol - Version 2.0A/B
 - Standard and extended data frames
 - 0 - 8 bytes data length
 - Programmable bit rate up to 1 Mbps¹
 - Support for remote frames
 - 5 receive buffers with FIFO storage scheme
- 3 transmit buffers with internal prioritization using a “local priority” concept
- Flexible maskable identifier filter supports two full size extended identifier filters (two 32-bit) or four 16-bit filters or eight 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter
- Programmable loop back mode supports self-test operation
- Programmable listen-only mode for monitoring of CAN bus
- Separate signalling and interrupt capabilities for all CAN receiver and transmitter error states (Warning, Error Passive, Bus-Off)
- Programmable MSCAN clock source either Bus Clock or Oscillator Clock
- Internal timer for time-stamping of received and transmitted messages
- Three low power modes: Sleep, Power Down and MSCAN Enable
- Global initialization of configuration registers

NOTES:

1. Depending on the actual bit timing and the clock jitter of the PLL.

Section 2 External Signal Description

2.1 Overview

This section lists and describes the signals that connect off chip.

2.2 Detailed Signal Description

The MSCAN uses two external pins.

2.2.1 RXCAN — CAN Receiver Input Pin

RXCAN is the MSCAN receiver input pin.

2.2.2 TXCAN — CAN Transmitter Output Pin

TXCAN is the MSCAN transmitter output pin. The TXCAN output pin represents the logic level on the CAN bus:

0 = Dominant state

1 = Recessive state

2.3 CAN System

A typical CAN system with MSCAN is shown in **Figure 2-1**. Each CAN station is connected physically to the CAN bus lines through a transceiver chip. The transceiver is capable of driving the large current needed for the CAN bus and has current protection against defected CAN or defected stations.

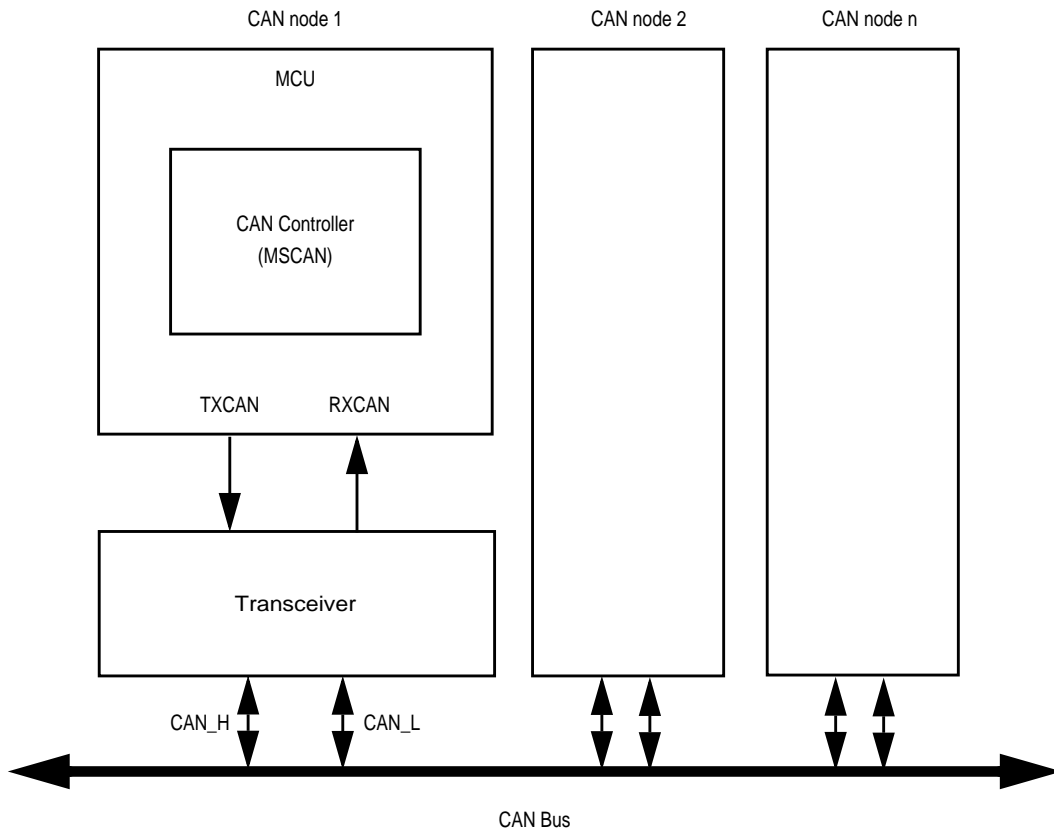


Figure 2-1 The CAN System

Section 3 Memory Map/Register Definition

3.1 Overview

This section provides a detailed description of all registers accessible in the MSCAN.

3.2 Module Memory Map

Table 3-1 and **Table 3-2** give an overview on all registers and their individual bits in the MSCAN memory map. The *register address* results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level. The *address offset* is defined at the module level.

The MSCAN occupies 64 bytes in the memory space. The base address of the MSCAN module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset.

Address Offset	
\$__00	CONTROL REGISTERS 12 BYTES
\$__0B	
\$__0C	RESERVED 2 BYTES
\$__0D	
\$__0E	ERROR COUNTERS 2 BYTES
\$__0F	
\$__10	IDENTIFIER FILTER 16 BYTES
\$__1F	
\$__20	RECEIVE BUFFER 16 BYTES (Window)
\$__2F	
\$__30	TRANSMIT BUFFER 16 BYTES (Window)
\$__3F	

Table 3-1 MSCAN Register Organization

Table 3-1 shows the individual registers associated with the MSCAN and their relative offset from the base address. The detailed register descriptions follow in the order they appear in the register map (see **Table 3-2**).

Table 3-2 Module Memory Map

Address	Use	Access
\$__00	MSCAN Control Register 0 (CANCTL0)	R/W ¹
\$__01	MSCAN Control Register 1 (CANCTL1)	R/W ¹
\$__02	MSCAN Bus Timing Register 0 (CANBTR0)	R/W
\$__03	MSCAN Bus Timing Register 1 (CANBTR1)	R/W

\$__04	MSCAN Receiver Flag Register (CANRFLG)	R/W ¹
\$__05	MSCAN Receiver Interrupt Enable Register (CANRIER)	R/W
\$__06	MSCAN Transmitter Flag Register (CANTFLG)	R/W ¹
\$__07	MSCAN Transmitter Interrupt Enable Register (CANTIER)	R/W ¹
\$__08	MSCAN Transmitter Message Abort Control (CANTARQ)	R/W ¹
\$__09	MSCAN Transmitter Message Abort Control (CANTAACK)	R
\$__0A	MSCAN Transmit Buffer Selection (CANTBSEL)	R/W ¹
\$__0B	MSCAN Identifier Acceptance Control Register (CANIDAC)	R/W ¹
\$__0C -\$__0D	RESERVED	
\$__0E	MSCAN Receive Error Counter Register (CANRXERR)	R
\$__0F	MSCAN Transmit Error Counter Register (CANTXERR)	R
\$__10	MSCAN Identifier Acceptance Register 0 (CANIDAR0)	R/W
\$__11	MSCAN Identifier Acceptance Register 1 (CANIDAR1)	R/W
\$__12	MSCAN Identifier Acceptance Register 2 (CANIDAR2)	R/W
\$__13	MSCAN Identifier Acceptance Register 3 (CANIDAR3)	R/W
\$__14	MSCAN Identifier Mask Register 0 (CANIDMR0)	R/W
\$__15	MSCAN Identifier Mask Register 1 (CANIDMR1)	R/W
\$__16	MSCAN Identifier Mask Register 2 (CANIDMR2)	R/W
\$__17	MSCAN Identifier Mask Register 3 (CANIDMR3)	R/W
\$__18	MSCAN Identifier Acceptance Register 4 (CANIDAR4)	R/W
\$__19	MSCAN Identifier Acceptance Register 5 (CANIDAR5)	R/W
\$__1A	MSCAN Identifier Acceptance Register 6 (CANIDAR6)	R/W
\$__1B	MSCAN Identifier Acceptance Register 7 (CANIDAR7)	R/W
\$__1C	MSCAN Identifier Mask Register 4 (CANIDMR4)	R/W
\$__1D	MSCAN Identifier Mask Register 5 (CANIDMR5)	R/W
\$__1E	MSCAN Identifier Mask Register 6 (CANIDMR6)	R/W
\$__1F	MSCAN Identifier Mask Register 7 (CANIDMR7)	R/W
\$__20 -\$__2F	Foreground Receive Buffer (CANRXFG)	R ²
\$__30 -\$__3F	Foreground Transmit Buffer (CANTXFG)	R ² /W

NOTES:

1. Refer to detailed register description for write access restrictions on per bit basis.
2. Reserved bits and unused bits within the TX- and RX-Buffers (CANTXFG, CANRXFG) will be read as "X", because of RAM based implementation.

3.3 Register Descriptions

This section describes in detail all the registers and register bits in the MSCAN module. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order. All bits of all registers in this module are completely synchronous to internal clocks during a register read.

3.3.1 Programmer’s Model of Control Registers

The programmer’s model is laid out for maximum simplicity and efficiency. **Table 3-2** provides an overview of the control registers for the MSCAN.

3.3.1.1 MSCAN Control 0 Register (CANCTL0)

The CANCTL0 register provides for various control of the MSCAN module as described below.

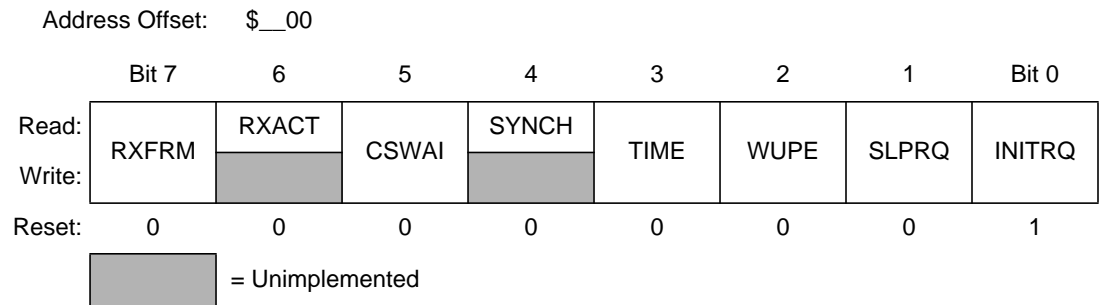


Figure 3-1 MSCAN Control 0 Register (CANCTL0)

NOTE: *The CANCTL0 register, except the WUPE, INITRQ and SLPRQ bits, is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when out of Initialization Mode; exceptions are read-only bits RXACT and SYNCH, bit RXFRM which is set by the module only and bit INITRQ which is also writable in Initialization Mode.

RXFRM — Received Frame Flag

This bit is read and clear only. It is set when a receiver has received a valid message correctly, independently of the filter configuration. Once set, it remains set until cleared by software or reset. Clearing is done by writing a ‘1’ to the bit. A write ‘0’ is ignored. This bit is not valid in loop back mode.

- 1 = A valid message was received since last clearing of this flag
- 0 = No valid message was received since last clearing this flag.

NOTE: *The MSCAN must be in Normal Mode for this bit to become set.*

RXACT — Receiver Active Status

This read-only flag indicates the MSCAN is receiving a message. The flag is controlled by the receiver front end. This bit is not valid in loop back mode.

- 1 = MSCAN is receiving a message (including when arbitration is lost)¹
- 0 = MSCAN is transmitting or idle¹.

CSWAI — CAN Stops in Wait Mode

Enabling this bit allows for lower power consumption in Wait Mode by disabling all the clocks at the bus interface to the MSCAN module.

- 1 = The module ceases to be clocked during Wait Mode.
- 0 = The module is not affected during Wait Mode.

NOTE: *In order to protect from accidentally violating the CAN protocol the TXCAN pin is immediately forced to a recessive state when the CPU enters Wait (CSWAI=1) or Stop Mode (see **4.6.2 CPU Wait Mode** and **4.6.3 CPU Stop Mode**)*

SYNCH — Synchronized Status

This read-only flag indicates whether the MSCAN is synchronized to the CAN bus and, as such, can participate in the communication process. It is set and cleared by the MSCAN.

- 1 = MSCAN is synchronized to the CAN bus.
- 0 = MSCAN is not synchronized to the CAN bus.

TIME - Timer Enable

This bit activates an internal 16-bit wide free running timer which is clocked by the bit clock. If the timer is enabled, a 16-bit time stamp will be assigned to each transmitted/received message within the active TX/RX buffer. As soon as a message is acknowledged on the CAN bus, the time stamp will be written to the highest bytes (\$_E, \$_F) in the appropriate buffer **3.3.2 Programmer's Model of Message Storage**. The internal timer is reset (all bits set to "0") when Initialization Mode is active.

- 1 = Enable internal MSCAN timer.
- 0 = Disable internal MSCAN timer.

WUPE — Wake-Up Enable

This configuration bit allows the MSCAN to restart from Sleep Mode when traffic on CAN is detected (see **4.6.4 MSCAN Sleep Mode**).

- 1 = Wake-Up enabled— The MSCAN is able to restart.
- 0 = Wake-Up disabled— The MSCAN ignores traffic on CAN.

NOTE: *The CPU has to make sure that the WUPE register and the WUPIE Wake-Up interrupt enable register **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)** is enabled, if the recovery mechanism from STOP or WAIT is required.*

SLPRQ — Sleep Mode Request

This bit requests the MSCAN to enter Sleep Mode, which is an internal power saving mode (see **4.6.4 MSCAN Sleep Mode**). The Sleep Mode request is serviced when the CAN bus is idle, i.e. the module is not receiving a message and all transmit buffers are empty. The module indicates entry to Sleep Mode by setting SLPK=1 (**3.3.1.2 MSCAN Control 1 Register (CANCTL1)**). Sleep Mode will be active until SLPRQ is cleared by the CPU or, depending on the setting of WUPE bit, the MSCAN detects bus activity on CAN and clears the SLPRQ itself.

NOTES:

1. See the Bosch CAN 2.0A/B protocol specification dated September 1991 for a detailed definition of transmitter and receiver states.

1 = Sleep Mode Request – The MSCAN enters Sleep Mode when CAN bus idle.

0 = Running – The MSCAN functions normally.

NOTE: *The CPU cannot clear the SLPRQ bit before the MSCAN has entered Sleep Mode (SLPRQ=1 and SLPAK=1)*

INITRQ — Initialization Mode Request

When this bit is set by the CPU, the MSCAN skips to Initialization Mode (see **4.6.5 MSCAN Initialization Mode**). Any ongoing transmission or reception is aborted and synchronization to the bus is lost. The module indicates entry to Initialization Mode by setting INITAK=1 (**3.3.1.2 MSCAN Control 1 Register (CANCTL1)**).

The following registers enter their hard reset state and restore their default values: CANCTL0¹, CANRFLG², CANRIER³, CANTFLG, CANTIER, CANTARQ, CANTAACK, CANTBSEL.

The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, CANIDMR0-7 can only be written by the CPU when the MSCAN is in Initialization Mode (INITRQ=1 and INITAK=1). The values of the error counters are not affected by Initialization Mode.

When this bit is cleared by the CPU, the MSCAN restarts and then tries to synchronize to the CAN bus. If the MSCAN is not in Bus-Off state, it synchronizes after 11 consecutive recessive bits on the bus; if the MSCAN is in Bus-Off state it continues to wait for 128 occurrences of 11 consecutive recessive bits.

Writing to other bits in CANCTL0, CANRFLG, CANRIER, CANTFLG or CANTIER must only be done after Initialization Mode is left, which is INITRQ=0 and INITAK=0.

1 = MSCAN in Initialization Mode.

0 = Normal operation.

NOTE: *The CPU cannot clear the INITRQ bit before the MSCAN has entered Initialization Mode (INITRQ=1 and INITAK=1)*

NOTE: *In order to protect from accidentally violating the CAN protocol the TXCAN pin is immediately forced to a recessive state when the Initialization Mode is requested by the CPU. Thus the recommended procedure is to bring the MSCAN into Sleep Mode (SLPRQ=1 and SLPAK=1) before.*

3.3.1.2 MSCAN Control 1 Register (CANCTL1)

The CANCTL1 register provides for various control and handshake status information of the MSCAN module as described below.

Address Offset: \$__01

NOTES:

1. Except the WUPE, INITRQ and SLPRQ bits
2. The TSTAT1, TSTAT0 bits are not affected by Initialization Mode
3. The RSTAT1, RSTAT0 bits are not affected by Initialization Mode

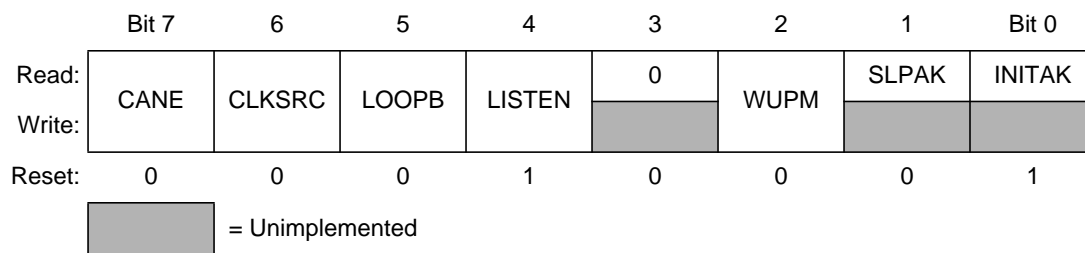


Figure 3-2 MSCAN Control 1 Register (CANCTL1)

Read: Anytime

Write: Anytime when INITRQ=1 and INITAK=1, except CANE which is write once in normal and anytime in special system operation modes when the MSCAN is in Initialization Mode (INITRQ=1 and INITAK=1).

CANE — MSCAN Enable

- 1 = The MSCAN module is enabled.
- 0 = The MSCAN module is disabled.

CLKSRC — MSCAN Clock Source

This bit defines the clock source for the MSCAN module (only for systems with a clock generation module; **4.3.2 Clock System** and **Figure 4-5**).

- 1 = The MSCAN clock source is the Bus Clock.
- 0 = The MSCAN clock source is the Oscillator Clock.

LOOPB — Loop Back Self Test Mode

When this bit is set, the MSCAN performs an internal loop back which can be used for self test operation. The bit stream output of the transmitter is fed back to the receiver internally. The RXCAN input pin is ignored and the TXCAN output goes to the recessive state (logic ‘1’). The MSCAN behaves as it does normally when transmitting and treats its own transmitted message as a message received from a remote node. In this state, the MSCAN ignores the bit sent during the ACK slot in the CAN frame Acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- 1 = Loop Back Self Test enabled
- 0 = Loop Back Self Test disabled

LISTEN — Listen Only Mode

This bit configures the MSCAN as a bus monitor. When the bit is set, all valid CAN messages with matching ID are received, but no acknowledgement or error frames are sent out **4.5.4 Listen-Only Mode**. In addition the error counters are frozen.

Listen Only Mode supports applications which require “hot plugging” or throughput analysis. The MSCAN is unable to transmit any messages, when Listen Only Mode is active.

- 1 = Listen Only Mode activated
- 0 = Normal operation

WUPM — Wake-Up Mode

This bit defines whether the integrated low-pass filter is applied to protect the MSCAN from spurious wake-up **4.6.4 MSCAN Sleep Mode**.

- 1 = MSCAN wakes-up the CPU only in case of a dominant pulse on the bus which has a length of T_{wup} and WUPE=1 in CANCTL0 (see **3.3.1.1 MSCAN Control 0 Register (CANCTL0)**).
- 0 = MSCAN wakes-up the CPU after any recessive to dominant edge on the CAN bus and WUPE=1 in CANCTL0.

SLPAK — Sleep Mode Acknowledge

This flag indicates whether the MSCAN module has entered Sleep Mode **4.6.4 MSCAN Sleep Mode**. It is used as a handshake flag for the SLPRQ Sleep Mode request. Sleep Mode is active when SLPRQ=1 and SLPAK=1. Depending on the setting of the WUPE bit the MSCAN will clear the flag if it detects bus activity on CAN while in Sleep Mode.

- 1 = Sleep Mode Active – The MSCAN has entered Sleep Mode.
- 0 = Running – The MSCAN operates normally.

INITAK — Initialization Mode Acknowledge

This flag indicates whether the MSCAN module is in Initialization Mode **4.6.5 MSCAN Initialization Mode**. It is used as a handshake flag for the INITRQ Initialization Mode request. Initialization Mode is active when INITRQ=1 and INITAK=1.

The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, CANIDMR0-7 can only be written by the CPU when the MSCAN is in Initialization Mode.

- 1 = Initialization Mode Active – The MSCAN has entered Initialization Mode.
- 0 = Running – The MSCAN operates normally.

3.3.1.3 MSCAN Bus Timing Register 0 (CANBTR0)

The CANBTR0 register provides for various bus timing control of the MSCAN module as described below.

Address Offset: $\$_{02}$

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 3-3 MSCAN Bus Timing Register 0 (CANBTR0)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

SJW1, SJW0 — Synchronization Jump Width

The synchronization jump width defines the maximum number of time quanta (Tq) clock cycles a bit can be shortened or lengthened to achieve resynchronization to data transitions on the bus (see **Table 3-3**).

Table 3-3 Synchronization Jump Width

SJW1	SJW0	Synchronization jump width
0	0	1 Tq clock cycle
0	1	2 Tq clock cycles
1	0	3 Tq clock cycles
1	1	4 Tq clock cycles

BRP[5-0] — Baud Rate Prescaler

These bits determine the time quanta (Tq) clock which is used to build up the individual bit timing, as shown in **Table 3-4**.

Table 3-4 Baud Rate Prescaler

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	Prescaler value (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	0	63
1	1	1	1	1	1	64

3.3.1.4 MSCAN Bus Timing Register 1 (CANBTR1)

The CANBTR1 register provides for various bus timing control of the MSCAN module as described below.

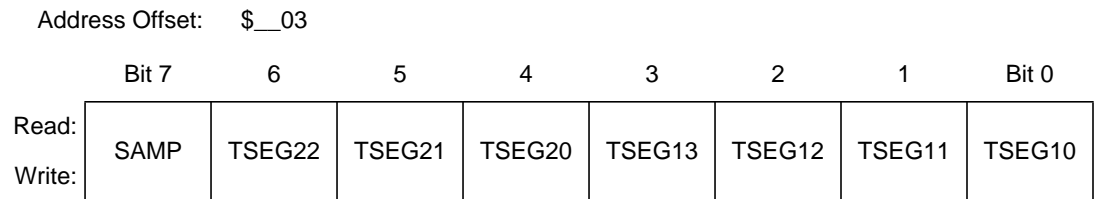


Figure 3-4 MSCAN Bus Timing Register 1 (CANBTR1)

Reset: 0 0 0 0 0 0 0 0

Figure 3-4 MSCAN Bus Timing Register 1 (CANBTR1)

Read: Anytime

Write: Anytime in Initialization Mode (INTRQ=1 and INITAK=1)

SAMP — Sampling

This bit determines the number of samples of the serial bus to be taken per bit time. If set, three samples per bit are taken; the regular one (sample point) and two preceding samples using a majority rule. For higher bit rates, it is recommended that SAMP be cleared which means that only one sample is taken per bit.

- 1 = Three samples per bit¹.
- 0 = One sample per bit.

TSEG22 – TSEG20 — Time Segment 2

Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see **Figure 4-6 Segments within the Bit Time**).

Time segment 2 (TSEG2) values are programmable as shown in **Table 3-5**.

Table 3-5 Time Segment 2 Values

TSEG22	TSEG21	TSEG20	Time segment 2
0	0	0	1 Tq clock cycle ¹
0	0	1	2 Tq clock cycles
.	.	.	.
1	1	0	7 Tq clock cycles
1	1	1	8 Tq clock cycles

NOTES:

1. This setting is not valid. Please refer to **Table 4-2 CAN Standard Compliant Bit Time Segment Settings** for valid settings.

TSEG13 – TSEG10 — Time Segment 1

Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see **Figure 4-6 Segments within the Bit Time**).

Time segment 1 (TSEG1) values are programmable as shown in **Table 3-6**.

NOTES:

1. In this case, PHASE_SEG1 must be at least 2 Time Quanta.

Table 3-6 Time Segment 1 Values

TSEG13	TSEG12	TSEG11	TSEG10	Time segment 1
0	0	0	0	1 Tq clock cycle ¹
0	0	0	1	2 Tq clock cycles ¹
0	0	1	0	3 Tq clock cycles ¹
0	0	1	1	4 Tq clock cycles
.
1	1	1	0	15 Tq clock cycles
1	1	1	1	16 Tq clock cycles

NOTES:

1. This setting is not valid. Please refer to **Table 4-2 CAN Standard Compliant Bit Time Segment Settings** for valid settings.

The bit time is determined by the oscillator frequency, the baud rate prescaler, and the number of time quanta (Tq) clock cycles per bit (as shown in **Table 3-5** and **Table 3-6** above).

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{\text{CANCLK}}} \cdot (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)

A flag can only be cleared when the condition which caused the setting is no longer valid and can only be cleared by software (writing a ‘1’ to the corresponding bit position). Every flag has an associated interrupt enable bit in the CANRIER register.

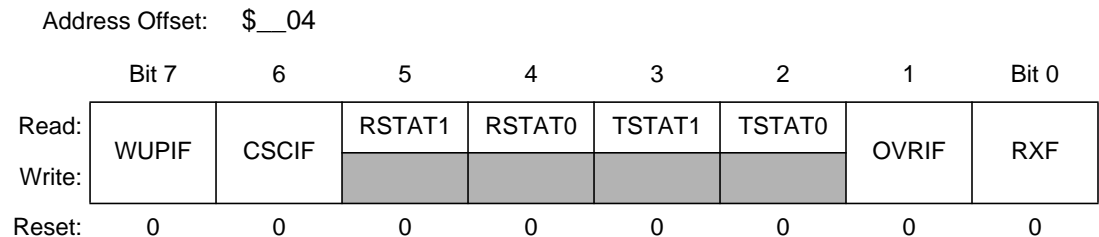


Figure 3-5 MSCAN Receiver Flag Register (CANRFLG)

NOTE: The *CANRFLG* register is held in the reset state¹ when the Initialization Mode is active (*INITRQ*=1 and *INITAK*=1). This register is writable again as soon as the Initialization Mode is left (*INITRQ*=0 and *INITAK*=0).

Read: Anytime

Write: Anytime when out of Initialization Mode, except *RSTAT*[1:0] and *TSTAT*[1:0] flags which are read-only; write of '1' clears flag; write of '0' ignored

WUPIF — Wake-Up Interrupt Flag

If the MSCAN detects bus activity while in Sleep Mode **4.6.4 MSCAN Sleep Mode** and the *WUPE*=1 in *CANTCTL0* (see **3.3.1.1 MSCAN Control 0 Register (CANCTL0)**), it will set the *WUPIF* flag. If not masked, a Wake-Up interrupt is pending while this flag is set.

- 1 = MSCAN detected activity on the bus and requested wake-up.
- 0 = No wake-up activity observed while in Sleep Mode.

CSCIF — CAN Status Change Interrupt Flag

This flag is set when the MSCAN changes its current bus status due to the actual value of the Transmit Error Counter (TEC) and the Receive Error Counter (REC). An additional 4-bit (*RSTAT*[1:0], *TSTAT*[1:0]) status register, which is split into separate sections for TEC/REC, informs the system on the actual bus status **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)**. If not masked, an Error interrupt is pending while this flag is set. *CSCIF* provides a blocking interrupt. That guarantees that the Receiver / Transmitter status bits (*RSTAT*/*TSTAT*) are only updated when no CAN Status Change interrupt is pending. If the TECs/RECs change their current value after the *CSCIF* is asserted and therefore would cause an additional state change in the *RSTAT*/*TSTAT* bits, these bits keep their old state bits until the current *CSCIF* interrupt is cleared again.

- 1 = MSCAN changed current bus status.
- 0 = No change in bus status occurred since last interrupt.

*RSTAT*1, *RSTAT*0 — Receiver Status Bits

The values of the error counters control the actual bus status of the MSCAN. As soon as the Status Change Interrupt Flag (*CSCIF*) is set these bits indicate the appropriate receiver related bus status of the MSCAN. The coding for the bits *RSTAT*1, *RSTAT*0 is:

- 00 = RxOK: 0 ≤ Receive Error Counter ≤ 96
- 01 = RxWRN: 96 < Receive Error Counter ≤ 127
- 10 = RxERR: 127 < Receive Error Counter
- 11 = Bus-Off²: Transmit Error Counter > 255

*TSTAT*1, *TSTAT*0 — Transmitter Status Bits

The values of the Error Counters control the actual bus status of the MSCAN. As soon as the Status Change Interrupt Flag (*CSCIF*) is set these bits indicate the appropriate transmitter related bus status of the MSCAN. The coding for the bits *TSTAT*1, *TSTAT*0 is:

NOTES:

1. The *RSTAT*[1:0], *TSTAT*[1:0] bits are not affected by Initialization Mode
2. Redundant Information for the most critical bus status which is "CAN Bus-Off". This only occurs if the Tx Error Counter exceeds a number of 255 errors. CAN Bus-Off affects the receiver state. As soon as the transmitter leaves its Bus-Off state the receiver state skips to RxOK too. Refer also to *TSTAT*[1:0] coding.

- 00 = TxOK: 0 ≤ Transmit Error Counter ≤ 96
- 01 = TxWRN: 96 < Transmit Error Counter ≤ 127
- 10 = TxERR: 127 < Transmit Error Counter ≤ 255
- 11 = Bus-Off: Transmit Error Counter > 255

OVRIF — Overrun Interrupt Flag

This flag is set when a data overrun condition occurs. If not masked, an Error interrupt is pending while this flag is set.

- 1 = A data overrun detected.
- 0 = No data overrun condition.

RXF — Receive Buffer Full Flag

The RXF flag is set by the MSCAN when a new message is shifted in the receiver FIFO. This flag indicates whether the shifted buffer is loaded with a correctly received message (matching identifier, matching Cyclic Redundancy Code (CRC) and no other errors detected). After the CPU has read that message from the RxFG buffer in the receiver FIFO, the RXF flag must be cleared to release the buffer. A set RXF flag prohibits the shifting of the next FIFO entry into the foreground buffer (RxFG). If not masked, a Receive interrupt is pending while this flag is set.

- 1 = The receiver FIFO is not empty. A new message is available in the RxFG.
- 0 = No new message available within the RxFG.

NOTE: *To ensure data integrity, do not read the receive buffer registers while the RXF flag is cleared.*

For MCUs with dual CPUs, reading the receive buffer registers while the RXF flag is cleared may result in a CPU fault condition.

3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)

This register contains the interrupt enable bits for the interrupt flags described above.

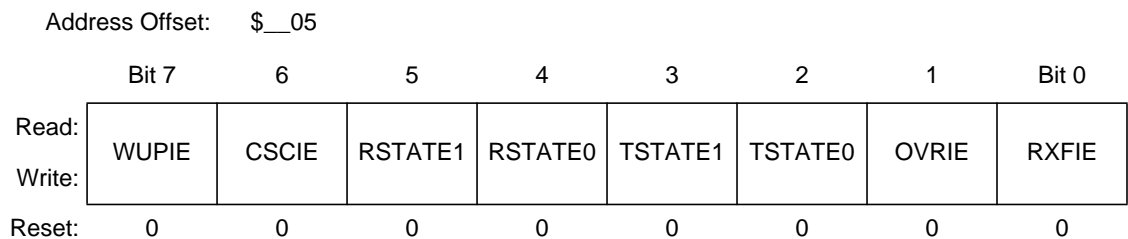


Figure 3-6 MSCAN Receiver Interrupt Enable Register (CANRIER)

NOTE: *The CANRIER register is held in the reset state¹ when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

NOTES:

- 1. The RSTATE[1:0], TSTATE[1:0] bits are not affected by Initialization Mode

Read: Anytime

Write: Anytime when out of Initialization Mode

WUPIE — Wake-Up Interrupt Enable

- 1 = A wake-up event causes a Wake-Up interrupt request.
- 0 = No interrupt request is generated from this event.

NOTE: *The CPU has to make sure that the Wake-Up interrupt register and the WUPE register 3.3.1.1 MSCAN Control 0 Register (CANCTL0) is enabled, if the recovery mechanism from STOP or WAIT is required.*

CSCIE — CAN Status Change Interrupt Enable

- 1 = A CAN Status Change event causes an error interrupt request.
- 0 = No interrupt request is generated from this event.

RSTATE1, RSTATE0— Receiver Status Change Enable

These RSTAT enable bits control the sensitivity level in which receiver state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the RSTAT flags still indicate the actual receiver state and are only updated if no CSCIF interrupt is pending.

- 11 = generate CSCIF interrupt on all state changes
- 10 = generate CSCIF interrupt only if the receiver enters or leaves “RxErr” or “Bus-Off”¹ state. Discard other receiver state changes for generating CSCIF interrupt.
- 01 = generate CSCIF interrupt only if the receiver enters or leaves “Bus-Off” state. Discard other receiver state changes for generating CSCIF interrupt.
- 00 = do not generate any CSCIF interrupt caused by receiver state changes.

TSTATE1, TSTATE0— Transmitter Status Change Enable

These TSTAT enable bits control the sensitivity level in which transmitter state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the TSTAT flags still indicate the actual transmitter state and are only updated if no CSCIF interrupt is pending.

- 11 = generate CSCIF interrupt on all state changes
- 10 = generate CSCIF interrupt only if the transmitter enters or leaves “TxErr” or “Bus-Off” state. Discard other transmitter state changes for generating CSCIF interrupt.
- 01 = generate CSCIF interrupt only if the transmitter enters or leaves “Bus-Off” state. Discard other transmitter state changes for generating CSCIF interrupt.
- 00 = do not generate any CSCIF interrupt caused by transmitter state changes.

OVRIE — Overrun Interrupt Enable

- 1 = An overrun event causes an error interrupt request.
- 0 = No interrupt request is generated from this event.

RXFIE — Receiver Full Interrupt Enable

NOTES:

1. Bus-Off state is only defined by the CAN standard for transmitters. Because the only possible state change for the transmitter from Bus-Off to TxOK also forces the receiver to skip its current state to RxOK, the coding of the RXSTAT[1:0] flags define an additional Bus-Off state for the receiver 3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)

- 1 = A receive buffer full (successful message reception) event causes a receiver interrupt request.
- 0 = No interrupt request is generated from this event.

3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)

The Transmit Buffer Empty flags each have an associated interrupt enable bit in the CANTIER register.

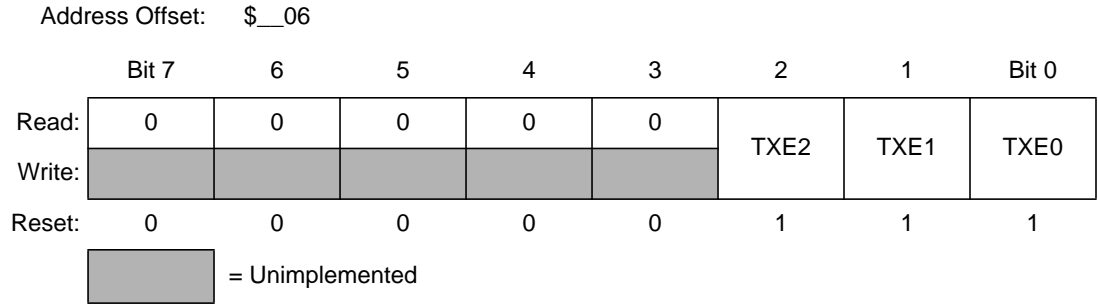


Figure 3-7 MSCAN Transmitter Flag Register (CANTFLG)

NOTE: *The CANTFLG register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime for TXEx flags when not in Initialization Mode; write of ‘1’ clears flag, write of ‘0’ ignored

TXE2 - TXE0 —Transmitter Buffer Empty

This flag indicates that the associated transmit message buffer is empty, and thus not scheduled for transmission. The CPU must clear the flag after a message is set up in the transmit buffer and is due for transmission. The MSCAN sets the flag after the message is sent successfully. The flag is also set by the MSCAN when the transmission request is successfully aborted due to a pending abort request (see **3.3.1.9 MSCAN Transmitter Message Abort Control (CANTARQ)**). If not masked, a Transmit interrupt is pending while this flag is set.

Clearing a TXEx flag also clears the corresponding ABTAKx (see **3.3.1.10 MSCAN Transmitter Message Abort Control (CANTAACK)**). When a TXEx flag is set, the corresponding ABTRQx bit is cleared (see **3.3.1.9 MSCAN Transmitter Message Abort Control (CANTARQ)**).

When Listen-Mode is active (see **3.3.1.2 MSCAN Control 1 Register (CANCTL1)**) the TXEx flags cannot be cleared and no transmission is started.

Read and write accesses to the transmit buffer will be blocked, if the corresponding TXEx bit is cleared (TXEx=’0’) and the buffer is scheduled for transmission.

- 1 = The associated message buffer is empty (not scheduled).
- 0 = The associated message buffer is full (loaded with a message due for transmission).

3.3.1.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

This register contains the interrupt enable bits for the Transmit Buffer Empty interrupt flags.

Address Offset: \$__07

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
Write:	[Unimplemented]							
Reset:	0	0	0	0	0	0	0	0

[Unimplemented] = Unimplemented

Figure 3-8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

NOTE: The CANTIER register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).

Read: Anytime

Write: Anytime when not in Initialization Mode

TXEIE2 - TXEIE0 — Transmitter Empty Interrupt Enable

1 = A transmitter empty (transmit buffer available for transmission) event causes a transmitter empty interrupt request.

0 = No interrupt request is generated from this event.

3.3.1.9 MSCAN Transmitter Message Abort Control (CANTARQ)

The CANTARQ register provides for abort request of queued messages as described below.

Address Offset: \$__08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
Write:	[Unimplemented]							
Reset:	0	0	0	0	0	0	0	0

[Unimplemented] = Unimplemented

Figure 3-9 MSCAN Transmitter Message Abort Request (CANTARQ)

NOTE: The CANTARQ register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).

Read: Anytime

Write: Anytime when not in Initialization Mode

ABTRQ2 - ABTRQ0 — Abort Request

The CPU sets the ABTRQ_x bit to request that a scheduled message buffer (TXE_x=0) be aborted. The MSCAN grants the request if the message has not already started transmission, or if the transmission is not successful (lost arbitration or error). When a message is aborted, the associated TXE (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and Abort Acknowledge flags (ABTAK, see **3.3.1.10 MSCAN Transmitter Message Abort Control (CANTAACK)**) are set and a transmit interrupt occurs if enabled. The CPU cannot reset ABTRQ_x. ABTRQ_x is reset whenever the associated TXE flag is set.

- 1 = Abort request pending.
- 0 = No abort request.

3.3.1.10 MSCAN Transmitter Message Abort Control (CANTAACK)

The CANTAACK register indicates the successful abort of a queued message, if requested by the appropriate bits in the CANTARQ register

Address Offset: \$_09

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 3-10 MSCAN Transmitter Message Abort Control (CANTAACK)

NOTE: *The CANTAACK register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1).*

Read: Anytime

Write: Unimplemented for ABTAK_x flags;

ABTAK2 - ABTAK0 — Abort Acknowledge

This flag acknowledges that a message was aborted due to a pending abort request from the CPU. After a particular message buffer is flagged empty, this flag can be used by the application software to identify whether the message was aborted successfully or was sent anyway. The ABTAK_x flag is cleared whenever the corresponding TXE flag is cleared.

- 1 = The message was aborted.
- 0 = The message was not aborted.

3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)

The CANTBSEL register allows the selection of the actual transmit message buffer, which will be then accessible in the CANTXFG register space (**3.3.1 Programmer's Model of Control Registers**).

Address Offset: \$ _0A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0			
Write:						TX2	TX1	TX0
Reset:	0	0	0	0	0	0	0	0


 = Unimplemented

Figure 3-11 MSCAN Transmitter Flag Register (CANTBSEL)

NOTE: The CANTBSEL register is held in the reset state when the Initialization Mode is active ($INITRQ=1$ and $INITAK=1$). This register is writable again as soon as the Initialization Mode is left ($INITRQ=0$ and $INITAK=0$).

Read: find the lowest ordered bit set to “1”, all other bits will be read as “0”

Write: Anytime when not in Initialization Mode

TX2 - TX0 — Transmit Buffer Select

The lowest numbered bit places the respective transmit buffer in the CANTXFG register space (e.g. TX1=1 and TX0=1 selects transmit buffer TX0, TX1=1 and TX0=0 selects transmit buffer TX1)

Read and write accesses to the selected transmit buffer will be blocked, if the corresponding TXEx bit is cleared and the buffer is scheduled for transmission **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**.

1 = The associated message Buffer is selected, if lowest numbered bit.

0 = The associated message buffer is deselected

NOTE: The following gives a short programming example of the usage of the CANTBSEL register:

The application software wants to get the next available transmit buffer. It reads the CANTFLG register and writes this value back into the CANTBSEL register. In this example Tx buffers TX1 and TX2 are available. The value read from CANTFLG is therefore 0b0000_0110. When writing this value back to CANTBSEL the Tx buffer TX1 is selected in the CANTXFG because the lowest numbered bit set to “1” is at bit position 1. Reading back this value out of CANTBSEL results in 0b0000_0010, because only the lowest numbered bit position set to “1” is presented. This mechanism eases the application software the selection of the next available Tx buffer.

LDD CANTFLG; value read is 0b0000_0110

STD CANTBSEL; value written is 0b0000_0110

LDD CANTBSEL; value read is 0b0000_0010

If all transmit message buffers are deselected no accesses are allowed to the CANTXFG registers.

3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)

The CANIDAC register provides for identifier acceptance control as described below.

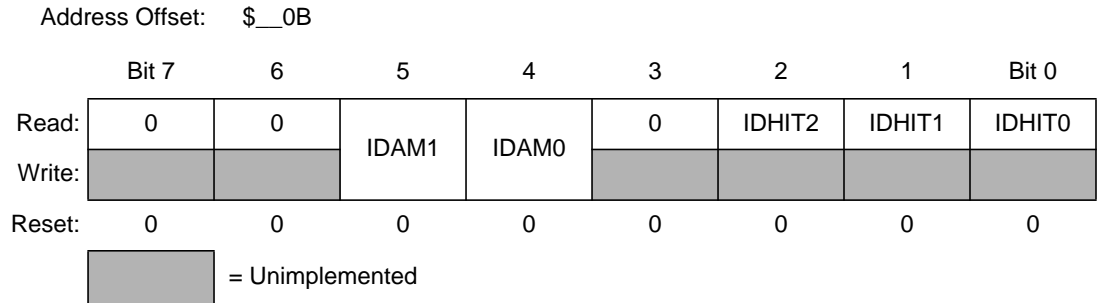


Figure 3-12 MSCAN Identifier Acceptance Control Register (CANIDAC)

Read: Anytime

Write: Anytime in Initialization Mode (INTRQ=1 and INITAK=1), except bits IDHITx which are read-only

IDAM1 - IDAM0 — Identifier Acceptance Mode

The CPU sets these flags to define the identifier acceptance filter organization **4.3 Identifier Acceptance Filter**. **Table 3-7** summarizes the different settings. In Filter Closed mode, no message is accepted such that the foreground buffer is never reloaded.

Table 3-7 Identifier Acceptance Mode Settings

IDAM1	IDAM0	Identifier Acceptance Mode
0	0	Two 32 bit Acceptance Filters
0	1	Four 16 bit Acceptance Filters
1	0	Eight 8 bit Acceptance Filters
1	1	Filter Closed

IDHIT2 - IDHIT0 — Identifier Acceptance Hit Indicator

The MSCAN sets these flags to indicate an identifier acceptance hit **4.3 Identifier Acceptance Filter**. **Table 3-8** summarizes the different settings.

Table 3-8 Identifier Acceptance Hit Indication

IDHIT2	IDHIT1	IDHIT0	Identifier Acceptance Hit
0	0	0	Filter 0 Hit
0	0	1	Filter 1 Hit
0	1	0	Filter 2 Hit
0	1	1	Filter 3 Hit
1	0	0	Filter 4 Hit
1	0	1	Filter 5 Hit
1	1	0	Filter 6 Hit
1	1	1	Filter 7 Hit

The IDHITx indicators are always related to the message in the foreground buffer (RxFG). When a message gets shifted into the foreground buffer of the receiver FIFO the indicators are updated as well.

3.3.1.13 Reserved Registers

These registers are reserved for factory testing of the MSCAN module and are not available in normal system operation modes.

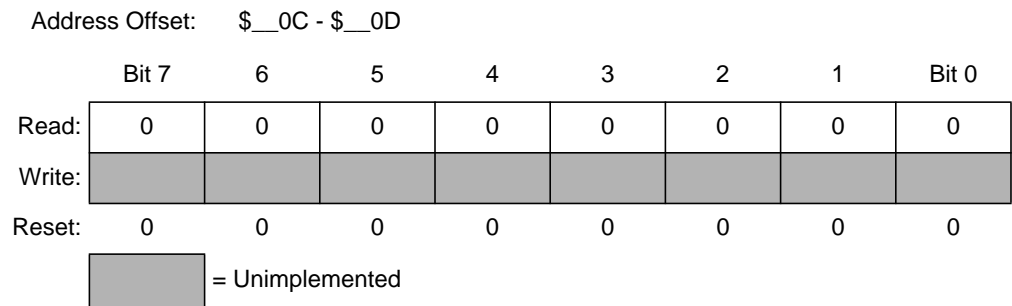


Figure 3-13 Reserved Registers

Read: always read \$00 in normal system operation modes

Write: Unimplemented in normal system operation modes

NOTE: *Writing to these registers when in special modes can alter the MSCAN functionality.*

3.3.1.14 MSCAN Receive Error Counter Register (CANRXERR)

This register reflects the status of the MSCAN receive error counter.

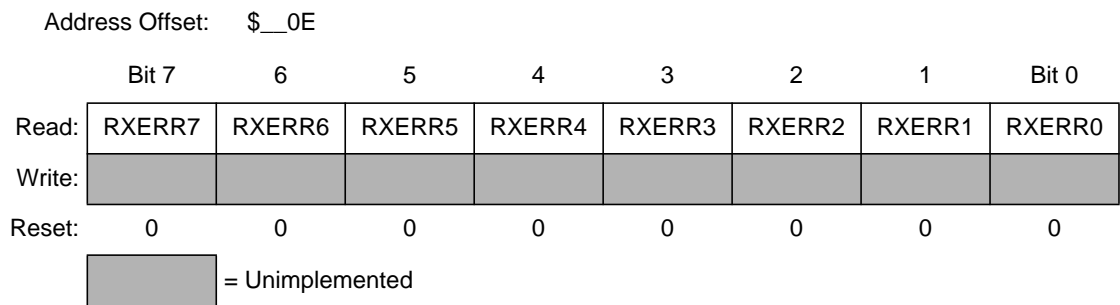


Figure 3-14 MSCAN Receive Error Counter Register (CANRXERR)

Read: only when in Sleep Mode (SLPRQ=1 and SLPK=1) or Initialization Mode (INITRQ=1 and INITAK=1)

Write: Unimplemented

NOTE: Reading this register when in any other mode other than Sleep or Initialization Mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

NOTE: Writing to this register when in special modes can alter the MSCAN functionality.

3.3.1.15 MSCAN Transmit Error Counter Register (CANTXERR)

This register reflects the status of the MSCAN transmit error counter.

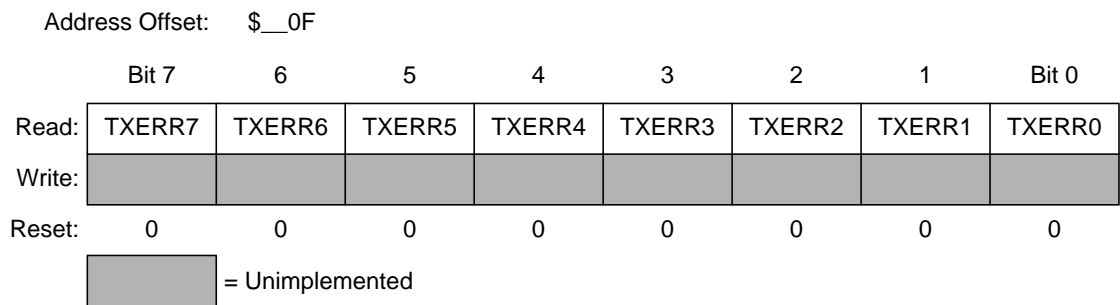


Figure 3-15 MSCAN Transmit Error Counter Register (CANTXERR)

Read: only when in Sleep Mode (SLPRQ=1 and SLPK=1) or Initialization Mode (INITRQ=1 and INITAK=1)

Write: Unimplemented

NOTE: Reading this register when in any other mode other than Sleep or Initialization Mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

NOTE: Writing to this register when in special modes can alter the MSCAN functionality.

3.3.1.16 MSCAN Identifier Acceptance Registers (CANIDAR0-7)

On reception, each message is written into the background receive buffer. The CPU is only signalled to read the message if it passes the criteria in the identifier acceptance and identifier mask registers (accepted); otherwise, the message is overwritten by the next message (dropped).

The acceptance registers of the MSCAN are applied on the IDR0 to IDR3 registers **3.3.2.1 Identifier Registers (IDR0-3)** of incoming messages in a bit by bit manner **4.3 Identifier Acceptance Filter**.

For extended identifiers, all four acceptance and mask registers are applied. For standard identifiers, only the first two (CANIDAR0/1, CANIDMR0/1) are applied.

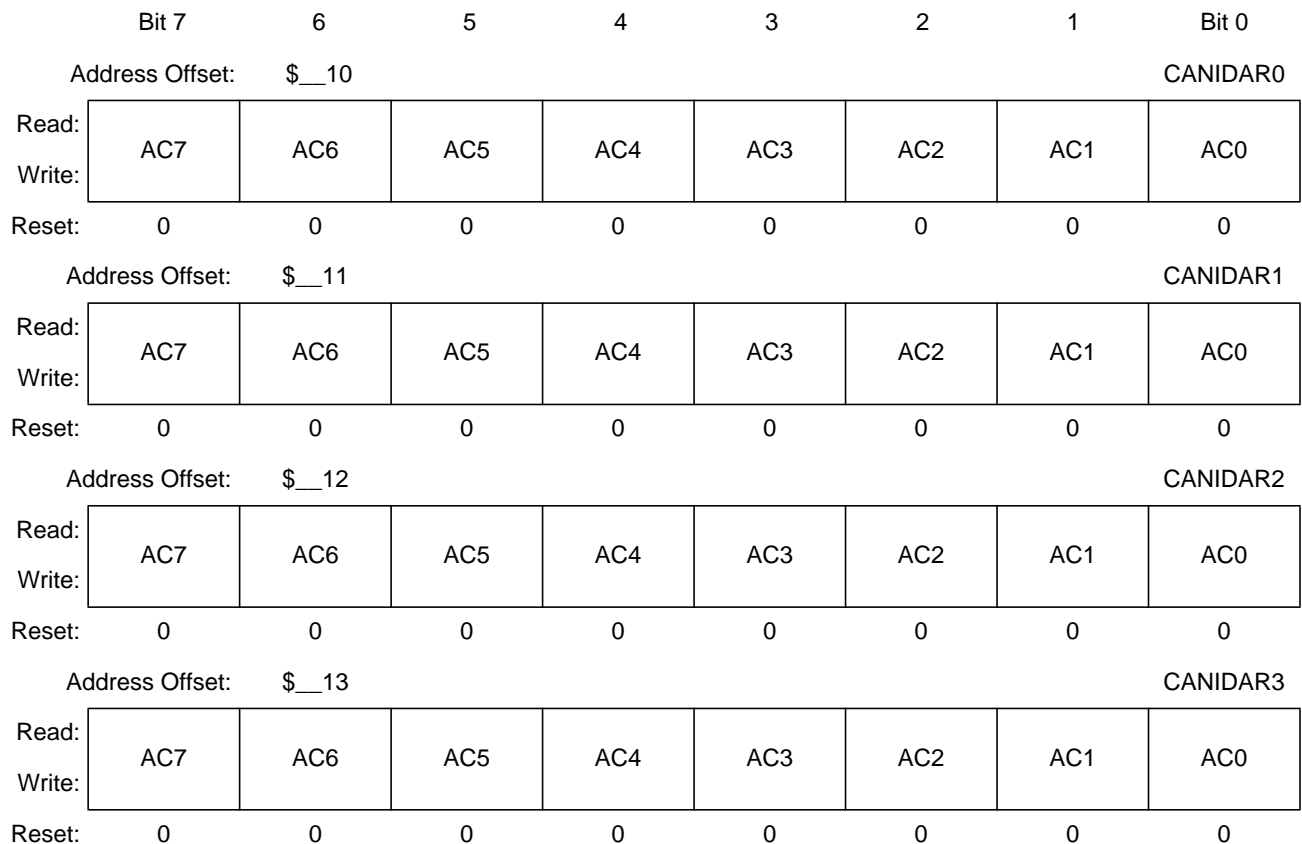


Figure 3-16 MSCAN Identifier Acceptance Registers (1st Bank)



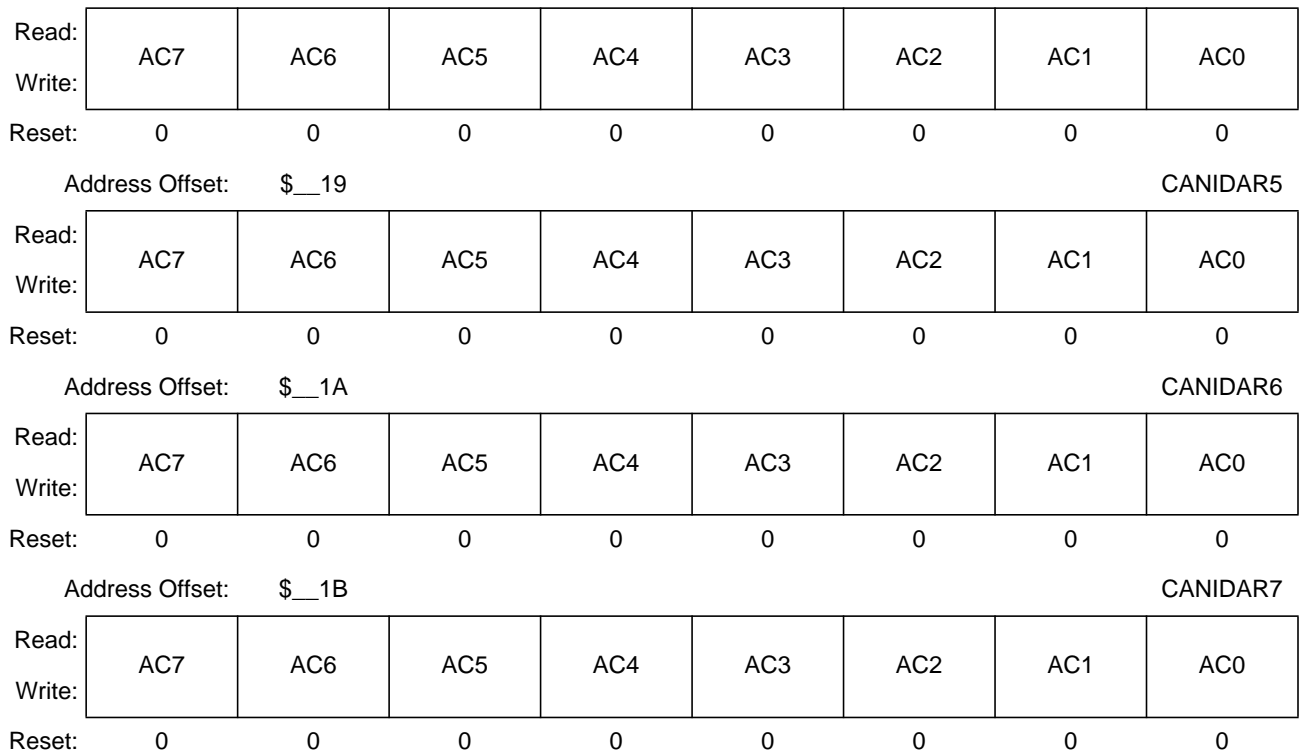


Figure 3-17 MSCAN Identifier Acceptance Registers (2nd Bank)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

AC7 – AC0 — Acceptance Code Bits

AC7 – AC0 comprise a user defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

3.3.1.17 MSCAN Identifier Mask Registers (CANIDMR0-7)

The identifier mask register specifies which of the corresponding bits in the identifier acceptance register are relevant for acceptance filtering. To receive standard identifiers in 32 bit filter mode, it is required to program the last three bits (AM2 - AM0) in the mask registers CANIDMR1 and CANIDMR5 to “don’t care”. To receive standard identifiers in 16 bit filter mode, it is required to program the last three bits (AM2 - AM0) in the mask registers CANIDMR1, CANIDMR3, CANIDMR5 and CANIDMR7 to “don’t care”.



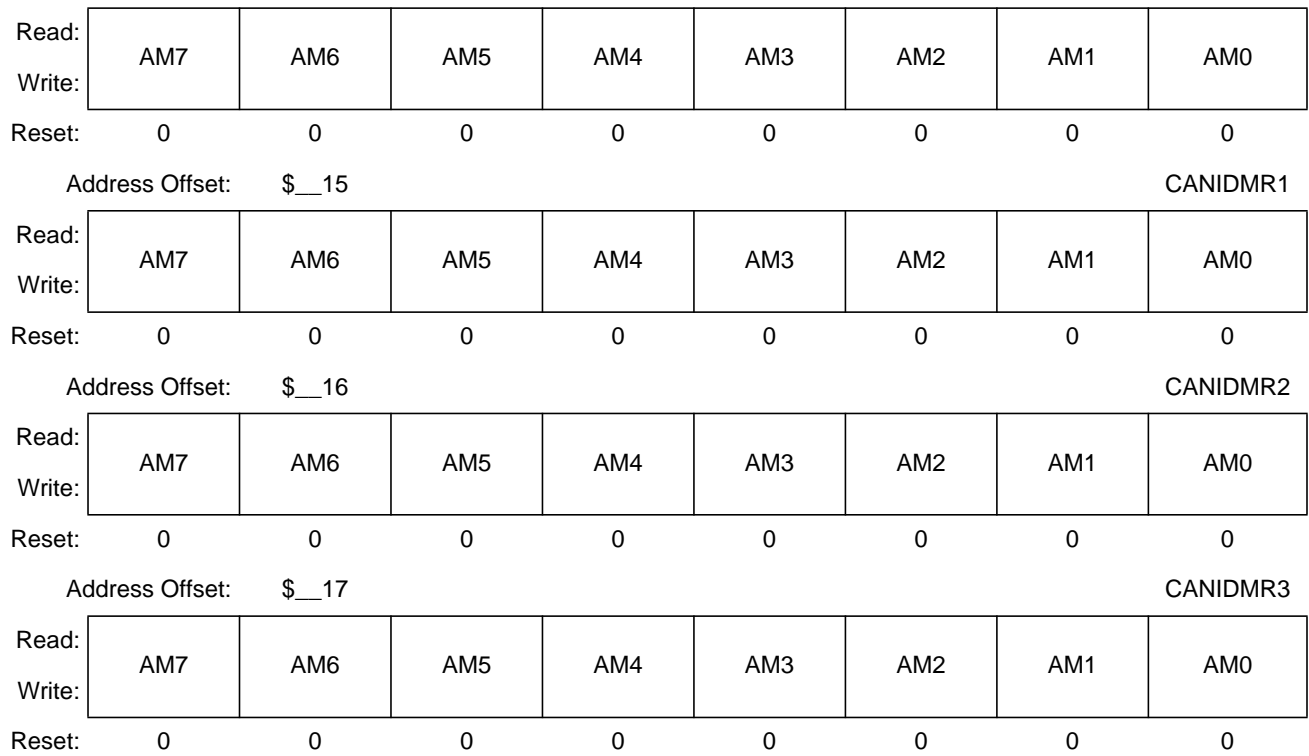


Figure 3-18 MSCAN Identifier Mask Registers (1st Bank)

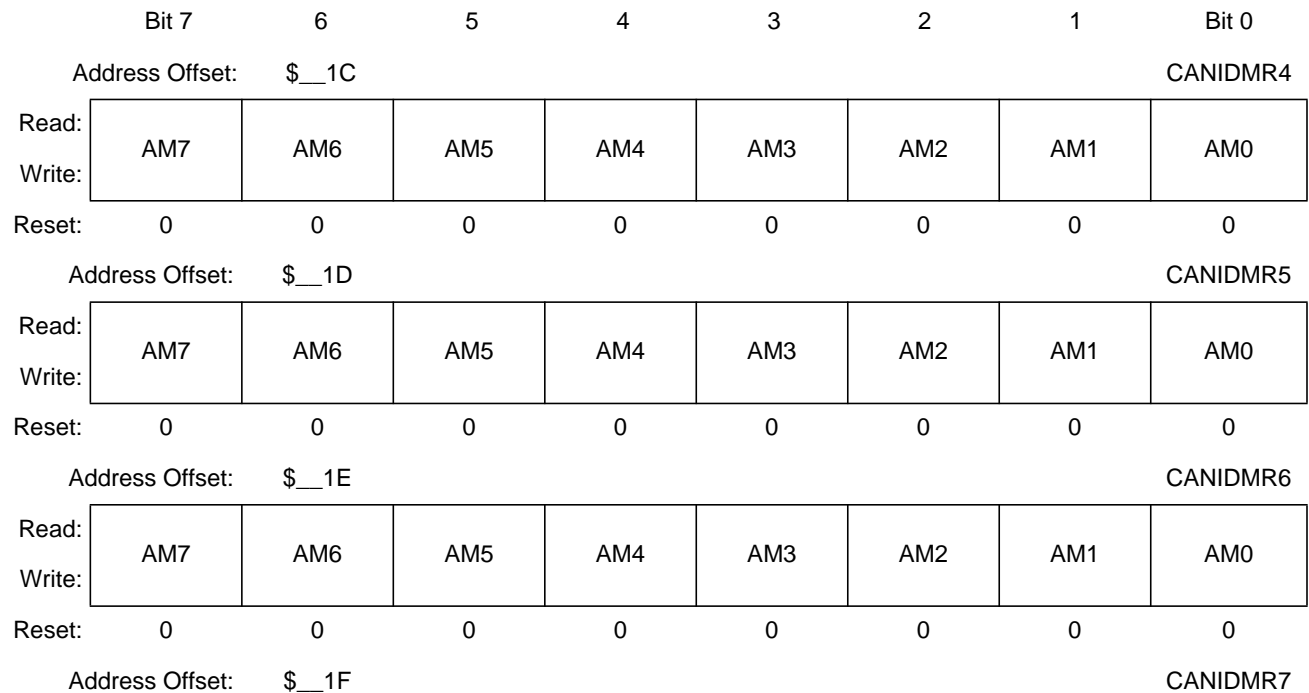


Figure 3-19 MSCAN Identifier Mask Registers (2nd Bank)

Read:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 3-19 MSCAN Identifier Mask Registers (2nd Bank)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

AM7 – AM0 — Acceptance Mask Bits

If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted.

1 = Ignore corresponding acceptance code register bit.

0 = Match corresponding acceptance code register and identifier bits.

3.3.2 Programmer’s Model of Message Storage

The following section details the organization of the receive and transmit message buffers and the associated control registers.

For reasons of programmer interface simplification, the receive and transmit message buffers have the same outline. Each message buffer allocates 16 bytes in the memory map containing a 13 byte data structure.

An additional Transmit Buffer Priority Register (TBPR) is defined for the transmit buffers. Within the last two bytes of this memory map the MSCAN stores a special 16-bit time stamp, which is sampled from an internal timer after successful transmission or reception of a message. This feature is only available for transmit and receiver buffers, if the TIME bit is set (**3.3.1.1 MSCAN Control 0 Register (CANCTL0)**).

The Time Stamp register is written by the MSCAN. The CPU can only read these registers.

Addr	Register Name
\$_x0	Identifier Register 0
\$_x1	Identifier Register 1
\$_x2	Identifier Register 2
\$_x3	Identifier Register 3
\$_x4	Data Segment Register 0
\$_x5	Data Segment Register 1
\$_x6	Data Segment Register 2
\$_x7	Data Segment Register 3
\$_x8	Data Segment Register 4
\$_x9	Data Segment Register 5
\$_xA	Data Segment Register 6
\$_xB	Data Segment Register 7
\$_xC	Data Length Register
\$_xD	Transmit Buffer Priority Register ¹
\$_xE	Time Stamp Register (High Byte) ²
\$_xF	Time Stamp Register (Low Byte) ³

Table 3-9 Message Buffer Organization

NOTES:

- 1. Not Applicable for Receive Buffers
- 2. Read-Only for CPU
- 3. Read-Only for CPU

Figure 3-20 shows the common 13 byte data structure of receive and transmit buffers for extended identifiers. The mapping of standard identifiers into the IDR registers is shown in **Figure 3-21**.

All bits of the receive and transmit buffers are ‘x’ out of reset because of RAM based implementation¹. All reserved or unused bits of the receive and transmit buffers are always read ‘x’.

Register name		Bit 7	6	5	4	3	2	1	Bit 0	ADDR
IDR0	Read:	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	\$_x0
	Write:									
IDR1	Read:	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15	\$_x1
	Write:									
IDR2	Read:	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	\$_x2
	Write:									
IDR3	Read:	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR	\$_x3
	Write:									

 = Unused¹

Figure 3-20 Receive / Transmit Message Buffer Extended Identifier

NOTES:

- 1. Exception: The Transmit Priority Registers are “0” out of reset

Register name		Bit 7	6	5	4	3	2	1	Bit 0	ADDR
DSR0	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x4
	Write:									
DSR1	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x5
	Write:									
DSR2	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x6
	Write:									
DSR3	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x7
	Write:									
DSR4	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x8
	Write:									
DSR5	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__x9
	Write:									
DSR6	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__xA
	Write:									
DSR7	Read:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	\$__xB
	Write:									
DLR	Read:					DLC3	DLC2	DLC1	DLC0	\$__xC
	Write:									

= Unused¹

Figure 3-20 Receive / Transmit Message Buffer Extended Identifier

NOTES:

1. Unused bits are always read 'x'

Read: Anytime for transmit buffers when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**); only when RXF flag is set for receive buffers (see **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)**).

Write: Anytime for transmit buffers when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**); unimplemented for receive buffers

Reset: \$xx because of RAM based implementation

Register name		Bit 7	6	5	4	3	2	1	Bit 0	ADDR
IDR0	Read:	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	\$__x0
	Write:									
IDR1	Read:	ID2	ID1	ID0	RTR	IDE (=0)				\$__x1
	Write:									
IDR2	Read:									\$__x2
	Write:									
IDR3	Read:									\$__x3
	Write:									

= Unused¹

Figure 3-21 Standard Identifier Mapping

NOTES:

1. Unused bits are always read 'x'

3.3.2.1 Identifier Registers (IDR0-3)

The identifier registers for an extended format identifier consist of a total of 32 bits; ID28 - ID0, SRR, IDE, and RTR bits. The identifier registers for a standard format identifier consist of a total of 13 bits; ID10 - ID0, RTR, and IDE bits.

ID28 - ID0 — Extended format identifier

The identifiers consist of 29 bits (ID28 - ID0) for the extended format. ID28 is the most significant bit and is transmitted first on the bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

ID10 - ID0 — Standard format identifier

The identifiers consist of 11 bits (ID10 - ID0) for the standard format. ID10 is the most significant bit and is transmitted first on the bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

SRR — Substitute Remote Request

This fixed recessive bit is used only in extended format. It must be set to 1 by the user for transmission buffers and is stored as received on the CAN bus for receive buffers.

IDE — ID Extended

This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send.

1 = Extended format (29 bit)

0 = Standard format (11 bit)

RTR — Remote Transmission Request

This flag reflects the status of the Remote Transmission Request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent.

1 = Remote frame

0 = Data frame

3.3.2.2 Data Segment Registers (DSR0-7)

The eight data segment registers, each with bits DB7-DB0, contain the data to be transmitted or received. The number of bytes to be transmitted or received is determined by the data length code in the corresponding DLR register.

DB7 - DB0 — Data Bits 7-0

3.3.2.3 Data Length Register (DLR)

This register keeps the data length field of the CAN frame.

DLC3 - DLC0 — Data Length Code bits

The data length code contains the number of bytes (data byte count) of the respective message. During the transmission of a remote frame, the data length code is transmitted as programmed while the number of transmitted data bytes is always 0. The data byte count ranges from 0 to 8 for a data frame.

Table 3-10 shows the effect of setting the DLC bits.

Table 3-10 Data length codes

Data length code				Data byte count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

3.3.2.4 Transmit Buffer Priority Register (TBPR)

This register defines the local priority of the associated message buffer. The local priority is used for the internal prioritization process of the MSCAN and is defined to be highest for the smallest binary number. The MSCAN implements the following internal prioritization mechanisms:

- All transmission buffers with a cleared TXEx flag participate in the prioritization immediately before the SOF (Start of Frame) is sent.

- The transmission buffer with the lowest local priority field wins the prioritization.

In cases of more than one buffer having the same lowest priority, the message buffer with the lower index number wins.

Address Offset: \$xxxD

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PRI07	PRI06	PRI05	PRI04	PRI03	PRI02	PRI01	PRI00
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 3-22 Transmit Buffer Priority Register (TBPR)

Read: Anytime when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**)

Write: Anytime when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**)

3.3.2.5 Time Stamp Register (TSRH, TSRL)

If the TIME bit is enabled, the MSCAN will write a special time stamp to the respective registers in the active transmit or receive buffer as soon as a message has been acknowledged on the CAN bus (**3.3.1.1 MSCAN Control 0 Register (CANCTL0)**). The time stamp is written on the bit sample point for the recessive bit of the ACK delimiter in the CAN frame. In case of a transmission, the CPU can only read the time stamp after the respective transmit buffer has been flagged empty.

The timer value, which is used for stamping, is taken from a free running internal CAN bit clock. A timer overrun is not indicated by the MSCAN. The timer is reset (all bits set to “0”) during Initialization Mode. The CPU can only read the Time Stamp registers.

Address Offset: \$xxxE

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
Write:								
Reset:	X	X	X	X	X	X	X	X

Figure 3-23 Time Stamp Register (TSRH - High Byte)

Address Offset: \$xxxF

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
Write:								
Reset:	X	X	X	X	X	X	X	X

Figure 3-24 Time Stamp Register (TSRL - Low Byte)

Read: Anytime when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**)

Write: Unimplemented

Section 4 Functional Description

4.1 General

This section provides a complete functional description of the MSCAN. It describes each of the features and modes listed in the introduction.

4.2 Message Storage

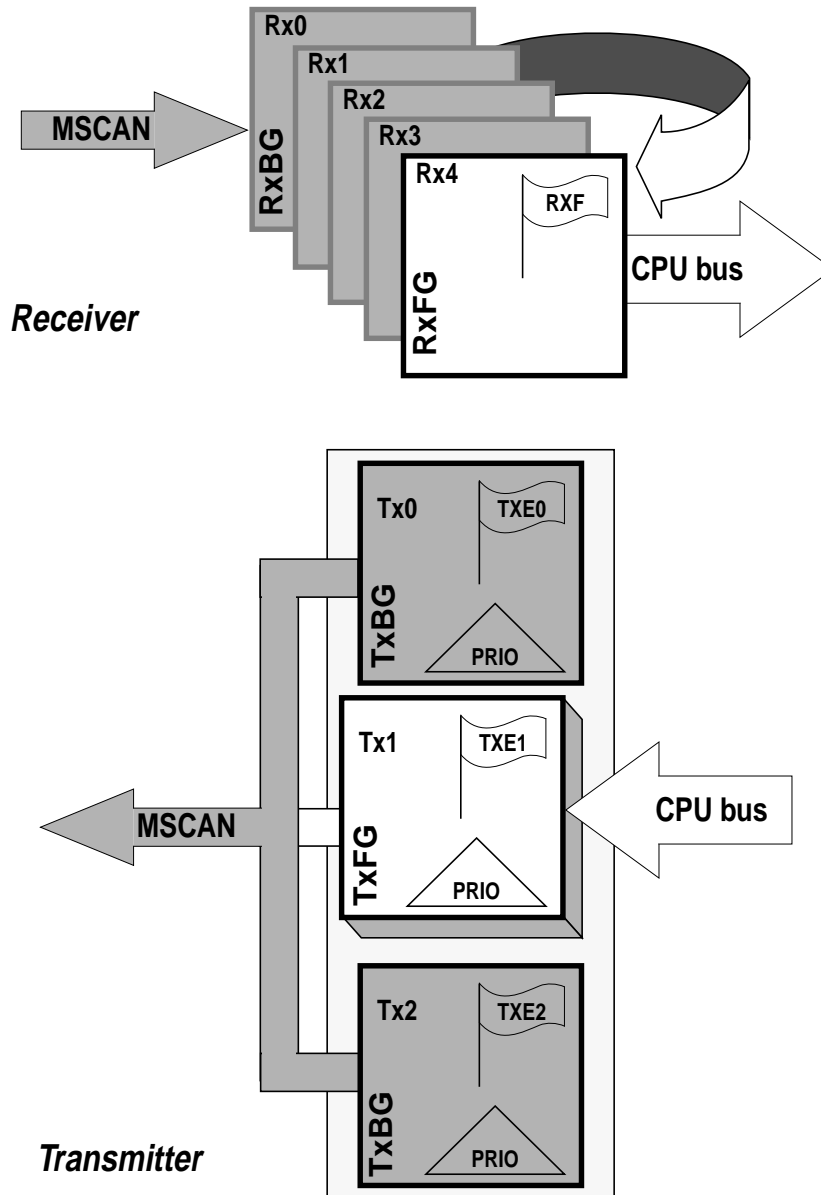


Figure 4-1 User Model for Message Buffer Organization

MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

4.2.1 Message Transmit Background

Modern application layer software is built upon two fundamental assumptions:

- Any CAN node is able to send out a stream of scheduled messages without releasing the bus between the two messages. Such nodes arbitrate for the bus immediately after sending the previous message and only release the bus in case of lost arbitration.
- The internal message queue within any CAN node is organized such that the highest priority message is sent out first, if more than one message is ready to be sent.

The above behavior cannot be achieved with a single transmit buffer. That buffer must be reloaded right after the previous message is sent. This loading process lasts a finite amount of time and has to be completed within the Inter-Frame Sequence (IFS)¹ to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the CPU react with short latencies to the transmit interrupt.

A double buffer scheme de-couples the reloading of the transmit buffer from the actual message sending and, as such, reduces the reactivity requirements on the CPU. Problems can arise if the sending of a message is finished while the CPU re-loads the second buffer. No buffer would then be ready for transmission and the bus would be released.

At least three transmit buffers are required to meet the first of the above requirements under all circumstances. The MSCAN has three transmit buffers.

The second requirement calls for some sort of internal prioritization which the MSCAN implements with the “local priority” concept described in **4.2.2 Transmit Structures**.

4.2.2 Transmit Structures

The MSCAN has a triple transmit buffer scheme which allows multiple messages to be set up in advance and achieve an optimized real-time performance. The three buffers are arranged as shown in **Figure 4-1 User Model for Message Buffer Organization**.

All three buffers have a 13 byte data structure similar to the outline of the receive buffers **3.3.2 Programmer’s Model of Message Storage**. An additional **3.3.2.4 Transmit Buffer Priority Register (TBPR)** contains an 8-bit “Local Priority” field (PRIO) (see **3.3.2.4 Transmit Buffer Priority Register (TBPR)**). The remaining two bytes are used for time stamping of a message, if required (see **3.3.2.5 Time Stamp Register (TSRH, TSRL)**).

To transmit a message, the CPU has to identify an available transmit buffer which is indicated by a set Transmitter Buffer Empty (TXEx) flag **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**. If a transmit buffer is available, the CPU has to set a pointer to this buffer by writing to the CANTBSEL register (**3.3.1.11 MSCAN Transmit Buffer Selection (CANTBSEL)**). This makes the respective buffer

NOTES:

1. Reference the Bosch CAN 2.0A/B protocol specification dated September 1991.

accessible within the CANTXFG address space **3.3.2 Programmer's Model of Message Storage**. The algorithmic feature associated with the CANTBSEL register simplifies the transmit buffer selection. In addition this scheme makes the handler software simpler as only one address area is applicable for the transmit process. In addition the required address space is minimized.

The CPU then stores the identifier, the control bits and the data content into one of the transmit buffers. Finally, the buffer is flagged as ready for transmission by clearing the associated TXE flag.

The MSCAN then schedules the message for transmission and signals the successful transmission of the buffer by setting the associated TXE flag. A transmit interrupt **4.9.1 Transmit Interrupt** is generated¹ when TXEx is set and can be used to drive the application software to re-load the buffer.

In case more than one buffer is scheduled for transmission when the CAN bus becomes available for arbitration, the MSCAN uses the “local priority” setting of the three buffers to determine the prioritization. For this purpose, every transmit buffer has an 8-bit local priority field (PRIO). The application software programs this field when the message is set up. The local priority reflects the priority of this particular message relative to the set of messages being transmitted from this node. The lowest binary value of the PRIO field is defined to be the highest priority. The internal scheduling process takes place whenever the MSCAN arbitrates for the bus. This is also the case after the occurrence of a transmission error.

When a high priority message is scheduled by the application software, it may become necessary to abort a lower priority message in one of the three transmit buffers. As messages that are already in transmission cannot be aborted, the user has to request the abort by setting the corresponding Abort Request bit (ABTRQ) **3.3.1.9 MSCAN Transmitter Message Abort Control (CANTARQ)**. The MSCAN then grants the request, if possible, by: 1) setting the corresponding Abort Acknowledge flag (ABTAK) in the CANTAACK register, 2) setting the associated TXE flag to release the buffer, and 3) generating a transmit interrupt. The transmit interrupt handler software can tell from the setting of the ABTAK flag whether the message was aborted (ABTAK=1) or sent (ABTAK=0).

4.2.3 Receive Structures

The received messages are stored in a five stage input FIFO. The five message buffers are alternately mapped into a single memory area **Figure 4-1 User Model for Message Buffer Organization**. While the background receive buffer (RxBG) is exclusively associated with the MSCAN, the foreground receive buffer (RxFG) is addressable by the CPU **Figure 4-1 User Model for Message Buffer Organization**. This scheme simplifies the handler software as only one address area is applicable for the receive process.

All receive buffers have a size of 15 bytes to store the CAN control bits, the identifier (standard or extended), the data contents and a time stamp, if enabled (for details **3.3.2 Programmer's Model of Message Storage**)².

The Receiver Full flag (RXF) **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** signals the status of the foreground receive buffer. When the buffer contains a correctly received message with a matching identifier, this flag is set.

NOTES:

1. The transmit interrupt occurs only if not masked. A polling scheme can be applied on TXEx also.
2. Reference the Bosch CAN 2.0A/B protocol specification dated September 1991 for details.

On reception, each message is checked to see if it passes the filter (**4.3 Identifier Acceptance Filter**) and in parallel, is written into the active RxBG. After successful reception of a valid message the MSCAN shifts the content of RxBG into the receiver FIFO¹, sets the RXF flag, and generates a receive interrupt **4.9.2 Receive Interrupt** to the CPU². The user's receive handler has to read the received message from the RxFG and then reset the RXF flag to acknowledge the interrupt and to release the foreground buffer. A new message, which can follow immediately after the IFS field of the CAN frame, is received into the next available RxBG. If the MSCAN receives an invalid message in its RxBG (wrong identifier, transmission errors etc.) the actual contents of the buffer will be over-written by the next message. The buffer will then not be shifted into the FIFO.

When the MSCAN module is transmitting, the MSCAN receives its own transmitted messages into the background receive buffer, RxBG, but does not shift it into the receiver FIFO, generate a receive interrupt, or acknowledge its own messages on the CAN bus. The exception to this rule is in loop back mode **3.3.1.2 MSCAN Control 1 Register (CANCTL1)** where the MSCAN treats its own messages exactly like all other incoming messages. The MSCAN receives its own transmitted messages in the event that it loses arbitration³. If arbitration is lost, the MSCAN must be prepared to become a receiver.

An overrun condition occurs when all receive message buffers in the FIFO are filled with correctly received messages with accepted identifiers and another message is correctly received from the bus with an accepted identifier. The latter message is discarded and an error interrupt with overrun indication is generated if enabled **4.9.4 Error Interrupt**. The MSCAN is still able to transmit messages while the receiver FIFO being filled, but all incoming messages are discarded. As soon as a receive buffer in the FIFO is available again, new valid messages will be accepted.

4.3 Identifier Acceptance Filter

The MSCAN Identifier Acceptance Registers (**3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)**) define the acceptable patterns of the standard or extended identifier (ID10 - ID0 or ID28 - ID0). Any of these bits can be marked 'don't care' in the MSCAN Identifier Mask Registers **3.3.1.17 MSCAN Identifier Mask Registers (CANIDMR0-7)**.

A filter hit is indicated to the application software by a set Receive Buffer Full flag (RXF=1) and three bits in the CANIDAC register **3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)**. These Identifier Hit flags (IDHIT2-0) clearly identify the filter section that caused the acceptance. They simplify the application software's task to identify the cause of the receiver interrupt. In case more than one hit occurs (two or more filters match), the lower hit has priority.

A very flexible programmable generic identifier acceptance filter has been introduced to reduce the CPU interrupt loading. The filter is programmable to operate in four different modes⁴:

- Two identifier acceptance filters, each to be applied to a) the full 29 bits of the extended identifier and to the following bits of the CAN 2.0B frame: Remote Transmission Request (RTR), Identifier Extension (IDE), and Substitute Remote Request (SRR) or b)⁵ the 11 bits of the standard identifier

NOTES:

1. Only if the RXF flag is not set.
2. The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.
3. Reference the Bosch CAN 2.0A/B protocol specification dated September 1991 for details.
4. For a better understanding of references made within the filter mode description, reference the Bosch specification dated September 1991 which details the CAN 2.0A/B protocol.

plus the RTR and IDE bits of the CAN 2.0A/B messages. This mode implements two filters for a full length CAN 2.0B compliant extended identifier. **Figure 4-2** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces a filter 0 hit. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces a filter 1 hit.

- Four identifier acceptance filters, each to be applied to a) the 14 most significant bits of the extended identifier plus the SRR and IDE bits of CAN 2.0B messages or b) the 11 bits of the standard identifier, the RTR and IDE bits of CAN 2.0A/B messages. **Figure 4-3** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces filter 0 and 1 hits. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces filter 2 and 3 hits.
- Eight identifier acceptance filters, each to be applied to the first 8 bits of the identifier. This mode implements eight independent filters for the first 8 bits of a CAN 2.0A/B compliant standard identifier or a CAN 2.0B compliant extended identifier. **Figure 4-4** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces filter 0 to 3 hits. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces filter 4 to 7 hits.
- Closed filter. No CAN message is copied into the foreground buffer RxFG, and the RXF flag is never set.

NOTES:

5. Although this mode can be used for standard identifiers, it is recommended to use the four or eight identifier acceptance filters for standard identifiers

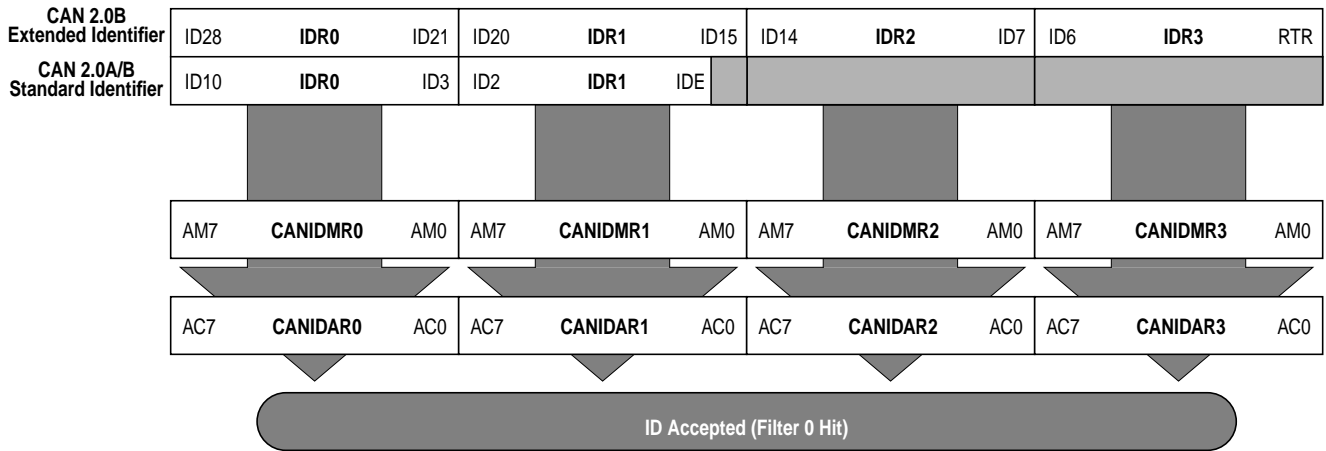


Figure 4-2 32-bit Maskable Identifier Acceptance Filter

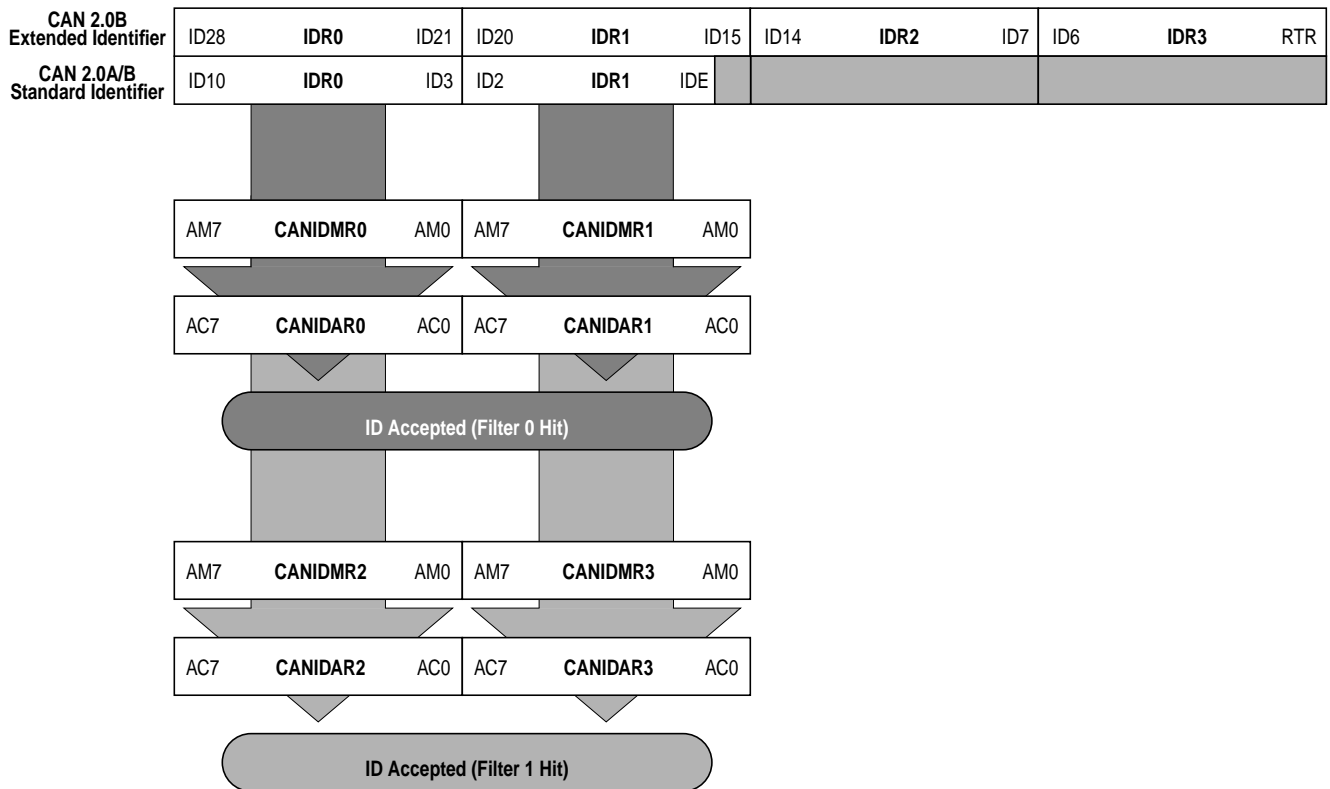


Figure 4-3 16-bit Maskable Identifier Acceptance Filters

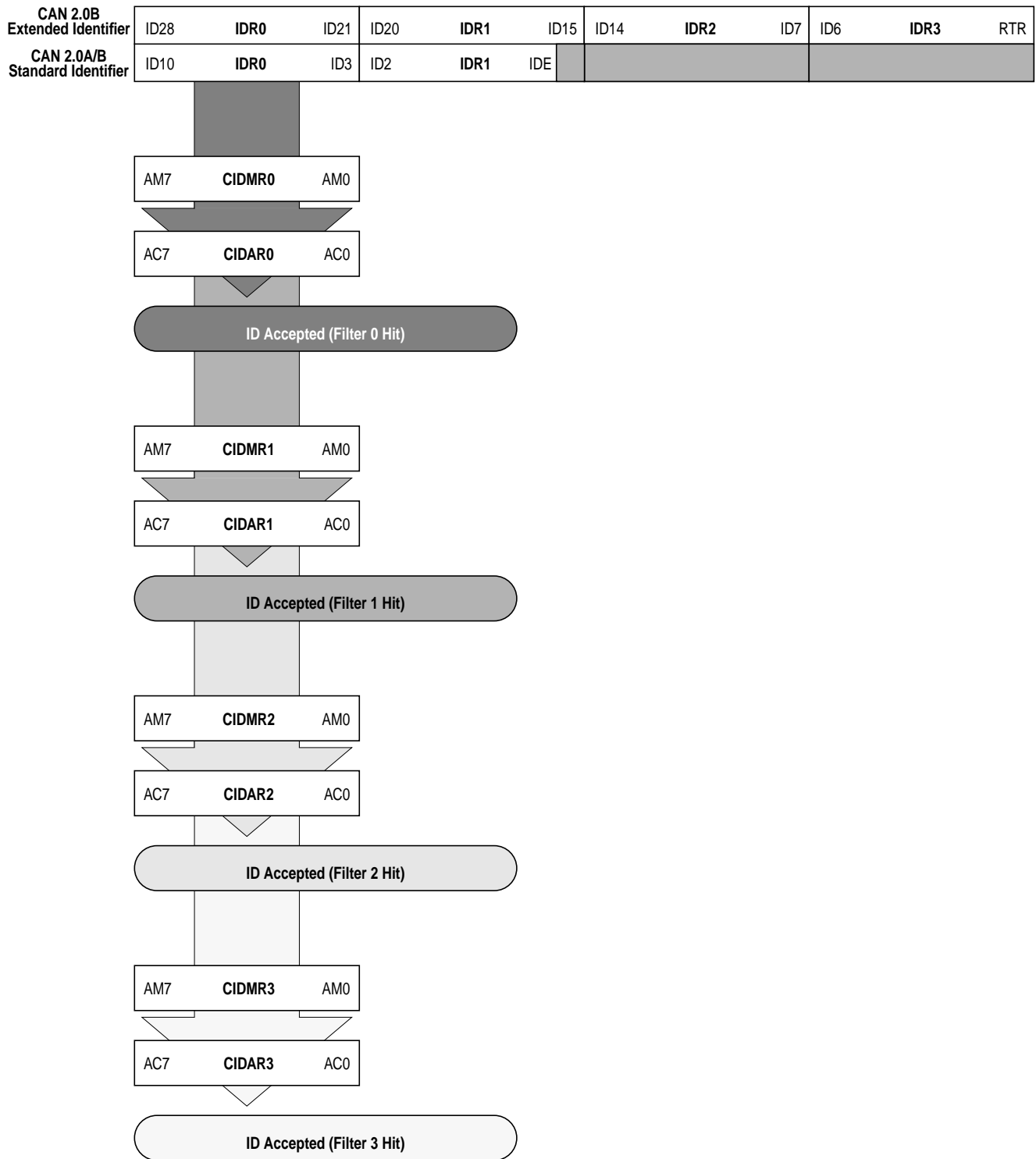


Figure 4-4 8-bit Maskable Identifier Acceptance Filters

4.3.1 Protocol Violation Protection

The MSCAN protects the user from accidentally violating the CAN protocol through programming errors. The protection logic implements the following features:

- The receive and transmit error counters cannot be written or otherwise manipulated.
- All registers which control the configuration of the MSCAN cannot be modified while the MSCAN is on-line. The MSCAN has to be in Initialization Mode. The corresponding INITRQ/INITAK handshake bits in the CANCTL0/CANCTL1 registers **3.3.1.1 MSCAN Control 0 Register (CANCTL0)** serve as a lock to protect the following registers:
 - MSCAN Control 1 Register (CANCTL1)
 - MSCAN Bus Timing Registers 0 and 1 (CANBTR0, CANBTR1)
 - MSCAN Identifier Acceptance Control Register (CANIDAC)
 - MSCAN Identifier Acceptance Registers (CANIDAR0-7)
 - MSCAN Identifier Mask Registers (CANIDMR0-7)
- The TXCAN pin is immediately forced to a recessive state when the MSCAN goes into the Power Down Mode or Initialization Mode (see **4.6.6 MSCAN Power Down Mode** and **4.6.5 MSCAN Initialization Mode**).
- The MSCAN enable bit (CANE) is only writable once in normal system operation modes as further protection against inadvertently disabling the MSCAN.

4.3.2 Clock System

Figure 4-5 shows the structure of the MSCAN clock generation circuitry. With this flexible clocking scheme, the MSCAN is able to handle CAN bus rates ranging from 10 Kbps up to 1 Mbps.

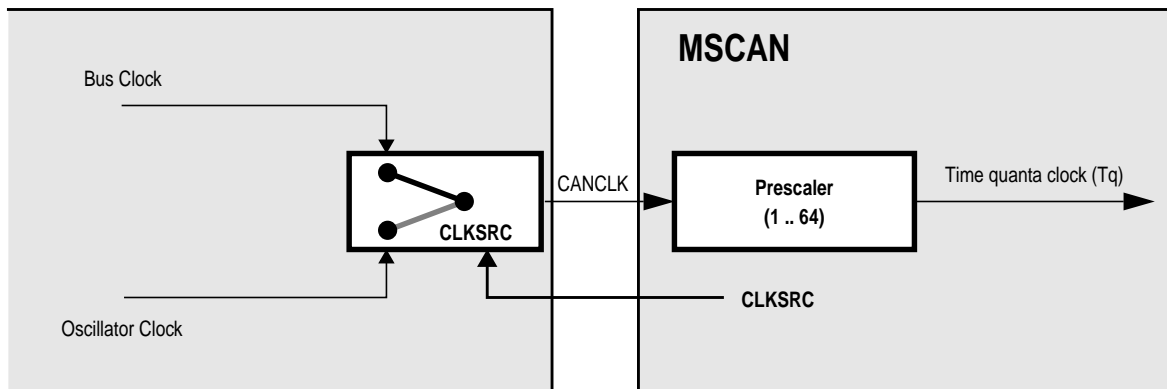


Figure 4-5 MSCAN Clocking Scheme

The clock source bit (CLKSRC) in the CANCTL1 register **3.3.1.2 MSCAN Control 1 Register (CANCTL1)** defines whether the internal CANCLK is connected to the output of a crystal oscillator (Oscillator Clock) or to the Bus Clock.

The clock source has to be chosen such that the tight oscillator tolerance requirements (up to 0.4%) of the CAN protocol are met. Additionally, for high CAN bus rates (1 Mbps), a 45%-55% duty cycle of the clock is required.

If the Bus Clock is generated from a PLL, it is recommended to select the Oscillator Clock rather than the Bus Clock due to jitter considerations, especially at the faster CAN bus rates.

For microcontrollers without a clock and reset generator (CRG), CANCLK is driven from the crystal oscillator (Oscillator Clock).

A programmable prescaler generates the time quanta (Tq) clock from CANCLK. A time quantum is the atomic unit of time handled by the MSCAN.

$$f_{Tq} = \frac{f_{CANCLK}}{(\text{Prescaler value})}$$

A bit time is subdivided into three segments^{1 2} (reference **Figure 4-6**):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time Segment 1: This segment includes the PROP_SEG and the PHASE_SEG1 of the CAN standard. It can be programmed by setting the parameter TSEG1 to consist of 4 to 16 time quanta.
- Time Segment 2: This segment represents the PHASE_SEG2 of the CAN standard. It can be programmed by setting the TSEG2 parameter to be 2 to 8 time quanta long.

$$\text{Bit Rate} = \frac{f_{Tq}}{(\text{number of Time Quanta})}$$

NOTES:

1. For further explanation of the under-lying concepts please refer to ISO/DIS 11519-1, Section 10.3.
2. Reference the Bosch CAN 2.0A/B protocol specification dated September 1991 for bit timing.

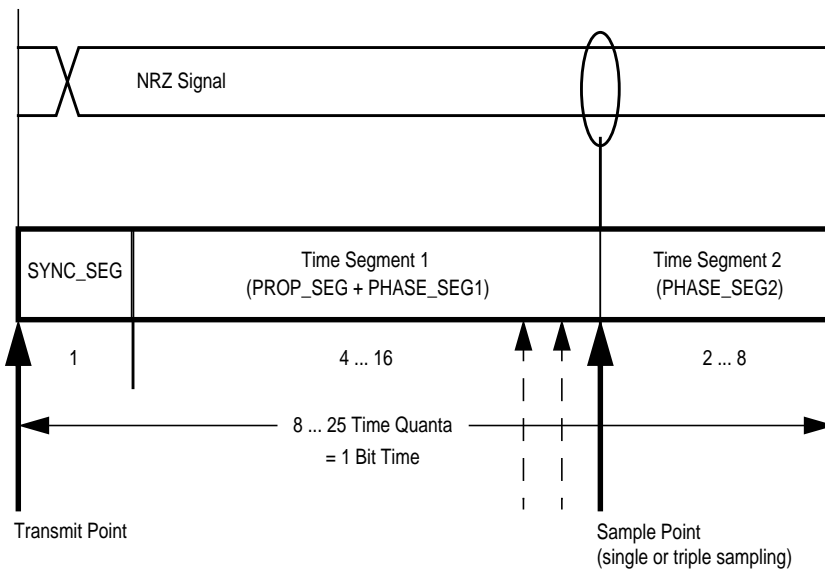


Figure 4-6 Segments within the Bit Time

Table 4-1 Time Segment Syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node in receive mode samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The Synchronization Jump Width¹ can be programmed in a range of 1 to 4 time quanta by setting the SJW parameter.

The above parameters are set by programming the MSCAN Bus Timing Registers (CANBTR0, CANBTR1) (see 3.3.1.3 MSCAN Bus Timing Register 0 (CANBTR0) and 3.3.1.4 MSCAN Bus Timing Register 1 (CANBTR1)).

Table 4-2 gives an overview of the CAN compliant segment settings and the related parameter values.

NOTE: It is the user’s responsibility to ensure the bit time settings are in compliance with the CAN standard.

NOTES:

1. Reference the Bosch CAN 2.0A/B protocol specification dated September 1991 for bit timing.

Time Segment 1	TSEG1	Time Segment 2	TSEG2	Synchronization Jump Width	SJW
5 .. 10	4 .. 9	2	1	1 .. 2	0 .. 1
4 .. 11	3 .. 10	3	2	1 .. 3	0 .. 2
5 .. 12	4 .. 11	4	3	1 .. 4	0 .. 3
6 .. 13	5 .. 12	5	4	1 .. 4	0 .. 3
7 .. 14	6 .. 13	6	5	1 .. 4	0 .. 3
8 .. 15	7 .. 14	7	6	1 .. 4	0 .. 3
9 .. 16	8 .. 15	8	7	1 .. 4	0 .. 3

Table 4-2 CAN Standard Compliant Bit Time Segment Settings

4.4 Timer Link

The MSCAN generates an internal time stamp whenever a valid frame is received or transmitted and the TIME bit is enabled. Because the CAN specification defines a frame to be valid if no errors occur before the End of Frame (EOF) field is transmitted successfully, the actual value of an internal timer is written at EOF to the appropriate time stamp position within the transmit buffer. For receive frames the time stamp is written to the receive buffer.

4.5 Modes of Operation

4.5.1 Normal Modes

The MSCAN module behaves as described within this specification in all normal system operation modes.

4.5.2 Special Modes

The MSCAN module behaves as described within this specification in all special system operation modes.

4.5.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like normal system operation modes as described within this specification.

4.5.4 Listen-Only Mode

In an optional bus monitoring mode (Listen-Only), the CAN node is able to receive valid data frames and valid remote frames, but it sends only “recessive” bits on the CAN bus. In addition it cannot start a transmission. If the MAC sub-layer is required to send a “dominant” bit (ACK bit, overload flag, active

error flag), the bit is rerouted internally so that the MAC sub-layer monitors this “dominant” bit, although the CAN bus may remain in recessive state externally.

4.5.5 Security Modes

The MSCAN module has no security features.

4.6 Low Power Options

If the MSCAN is disabled (CANE=0), the MSCAN clocks are stopped for power savings.

If the MSCAN is enabled (CANE=1), the MSCAN has two additional modes with reduced power consumption, compared to Normal Mode: Sleep and Power Down Mode. In Sleep Mode power consumption is reduced by stopping all clocks except those to access the registers from the CPU side. In Power Down Mode, all clocks are stopped and no power is consumed.

Table 4-3 summarizes the combinations of MSCAN and CPU modes. A particular combination of modes is entered by the given settings on the CSWAI and SLPRQ/SLPAK bits.

For all modes, an MSCAN Wake-Up interrupt can only occur if the MSCAN is in Sleep Mode (SLPRQ=1 and SLPAK=1), wake-up functionality is enabled (WUPE=1) and the Wake-Up interrupt is enabled (WUPIE=1).

Table 4-3 CPU vs. MSCAN Operating Modes

CPU Mode	MSCAN Mode			
	Normal	Reduced Power Consumption		
		Sleep	Power Down	Disabled (CANE=0)
RUN	CSWAI = X ¹ SLPRQ = 0 SLPAK = 0	CSWAI = X SLPRQ = 1 SLPAK = 1		CSWAI = X SLPRQ = X SLPAK = X
WAIT	CSWAI = 0 SLPRQ = 0 SLPAK = 0	CSWAI = 0 SLPRQ = 1 SLPAK = 1	CSWAI = 1 SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X
STOP			CSWAI = X SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X

NOTES:

1. 'X' means don't care.

4.6.1 CPU Run Mode

As can be seen in **Table 4-3 CPU vs. MSCAN Operating Modes**, only MSCAN Sleep Mode is available as low power option, when CPU is in Run Mode.

4.6.2 CPU Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode. If the CSWAI bit is set, then additional power can be saved in Power Down Mode since the CPU clocks are stopped. After leaving this Power Down Mode the MSCAN restarts its internal controllers and enters Normal Mode again.

While the CPU is in Wait Mode, the MSCAN can be operated in Normal Mode and generate interrupts (registers can be accessed via background debug mode). The MSCAN can also operate in any of the low power modes depending on the values of the SLPRQ/SLPAK and CSWAI bits as seen in **Table 4-3 CPU vs. MSCAN Operating Modes**.

4.6.3 CPU Stop Mode

The STOP instruction puts the MCU in a low power consumption stand-by mode. In Stop Mode, the MSCAN set in Power Down mode regardless of the value of the SLPRQ/SLPAK and CSWAI bits **Table 4-3**.

4.6.4 MSCAN Sleep Mode

The CPU can request the MSCAN to enter this low power mode by asserting the SLPRQ bit in the CANCTL0 register. The time when the MSCAN enters Sleep Mode depends on a fixed synchronization delay and its current activity:

- If there are one or more message buffers scheduled for transmission (TXEx = 0), the MSCAN will continue to transmit until all transmit message buffers are empty (TXEx = 1, transmitted successfully or aborted) and then goes into Sleep Mode.
- If it is receiving, it continues to receive and goes into Sleep Mode as soon as the CAN bus next becomes idle.
- If it is neither transmitting nor receiving, it immediately goes into Sleep Mode.

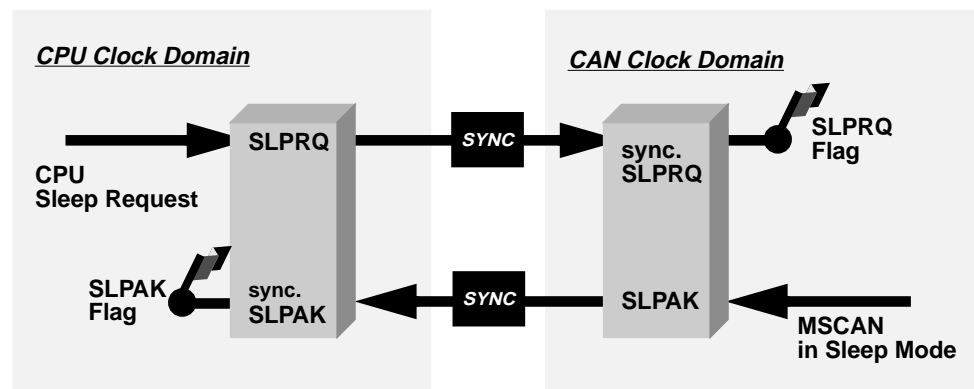


Figure 4-7 Sleep Request / Acknowledge Cycle

NOTE: *The application software must avoid setting up a transmission (by clearing one or more TXEx flag(s)) and immediately request Sleep Mode (by setting SLPRQ). It depends on the exact sequence of operations whether the MSCAN starts transmitting or goes into Sleep Mode directly.*

If Sleep Mode is active, the SLPRQ and SLPK bits are set (**Figure 4-7**). The application software must use SLPK as a handshake indication for the request (SLPRQ) to go into Sleep Mode.

When in Sleep Mode (SLPRQ=1 and SLPK=1), the MSCAN stops its internal clocks. However, clocks to allow register accesses from the CPU side still run.

If the MSCAN is in Bus-Off state, it stops counting the 128*11 consecutive recessive bits due to the stopped clocks. The TXCAN pin remains in a recessive state. If RXF=1, the message can be read and RXF can be cleared. Shifting a new message into the foreground buffer of the receiver FIFO (RxFG) does not take place while in Sleep Mode.

It is possible to access the transmit buffers and to clear the associated TXE flags. No message abort takes place while in Sleep Mode.

If the WUPE bit in CANCLT0 is not asserted, the MSCAN will mask any activity it detects on CAN. The RXCAN pin is therefore held internally in a recessive state. This locks the MSCAN in Sleep Mode (**Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode**).

The MSCAN is only able to leave Sleep Mode (wake up) when

- bus activity occurs and WUPE=1 or
- the CPU clears the SLPRQ bit

NOTE: *The CPU cannot clear the SLPRQ bit before Sleep Mode (SLPRQ=1 and SLPK=1) is active.*

After wake-up, the MSCAN waits for 11 consecutive recessive bits to synchronize to the bus. As a consequence, if the MSCAN is woken-up by a CAN frame, this frame is not received.

The receive message buffers (RxFG and RxBG) contain messages if they were received before Sleep Mode was entered. All pending actions will be executed upon wake-up; copying of RxBG into RxFG, message aborts and message transmissions. If the MSCAN is still in Bus-Off state after Sleep Mode was left, it continues counting the 128*11 consecutive recessive bits.

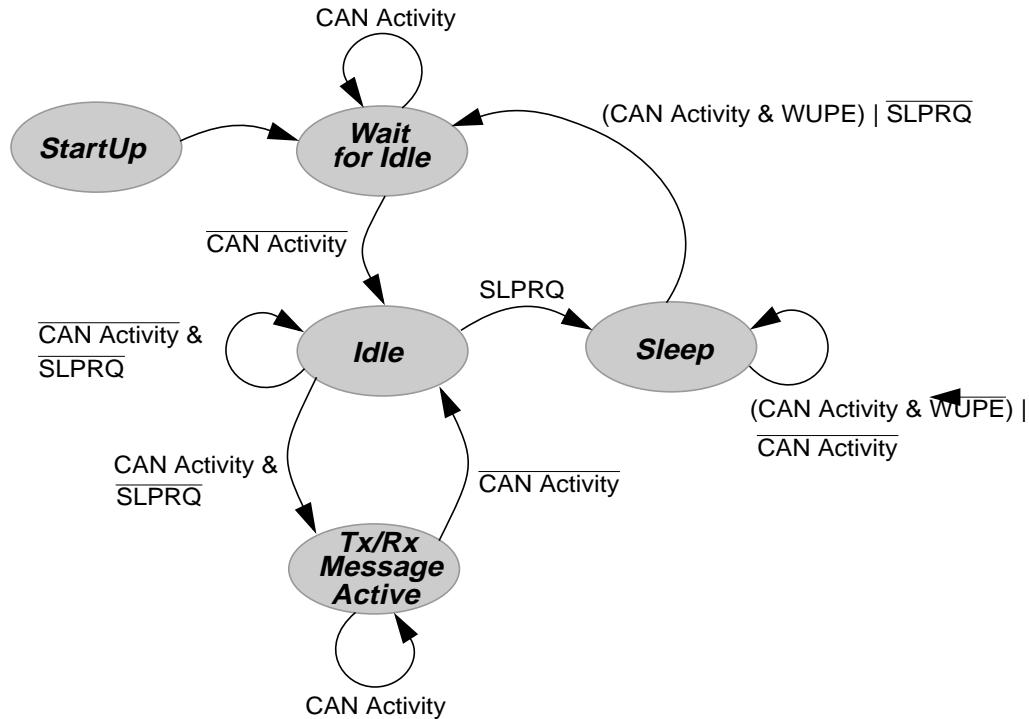


Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode

4.6.5 MSCAN Initialization Mode

In Initialization Mode, any ongoing transmission or reception is immediately aborted and synchronization to the bus is lost potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations, the MSCAN immediately drives the TXCAN pin into a recessive state.

NOTE: *The user is responsible for ensuring that the MSCAN is not active when Initialization Mode is entered. The recommended procedure is to bring the MSCAN into Sleep Mode (SLPRQ=1 and SLPK=1) before setting the INTRQ bit in the CANCTL0 register. Otherwise the abort of an ongoing message can cause an error condition and can have an impact on the other bus devices.*

In Initialization Mode, the MSCAN is stopped. However, interface registers can still be accessed. This mode is used to reset the CANCTL0, CANRFLG, CARRIER, CANTFLG, CANTIER, CANTARQ, CANTAACK, CANTBSEL registers to their default values. In addition it enables the configuration of the CANBTR0, CANBTR1 bit timing registers, CANIDAC and the CANIDAR, CANIDMR message filters. **3.3.1.1 MSCAN Control 0 Register (CANCTL0)** for a detailed description of the Initialization Mode.

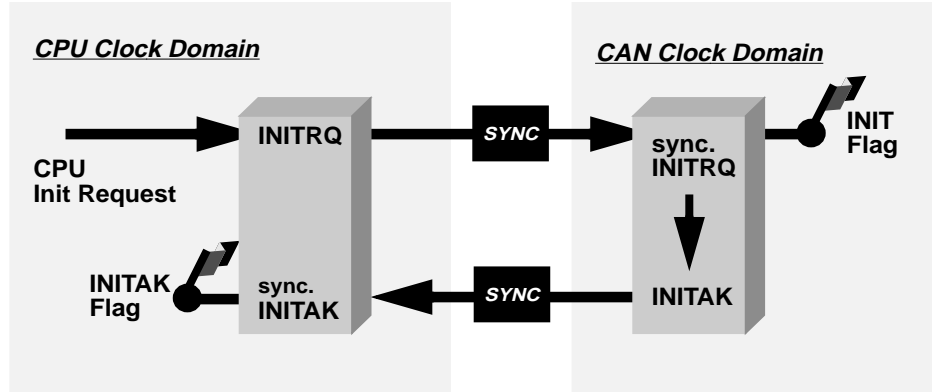


Figure 4-9 Initialization Request/Acknowledge Cycle

Due to independent clock domains within the MSCAN the INITRQ has to be synchronized to all domains by using a special handshake mechanism. This handshake causes additional synchronization delay (**Figure 4-9 Initialization Request/Acknowledge Cycle**).

If there is no message transfer ongoing on the CAN bus, the minimum delay will be two additional bus clocks and three additional CAN clocks. When all parts of the MSCAN are in Initialization Mode the INITAK flag is set. The application software must use INITAK as a handshake indication for the request (INITRQ) to go into Initialization Mode.

NOTE: *The CPU cannot clear the INITRQ bit before Initialization Mode (INITRQ=1 and INITAK=1) is active.*

4.6.6 MSCAN Power Down Mode

The MSCAN is in Power Down Mode (**Table 4-3**) when

- the CPU is in Stop Mode or
- the CPU is in Wait Mode and the CSWAI bit is set.

When entering the Power Down Mode, the MSCAN immediately stops all ongoing transmissions and receptions, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations to the above rule, the MSCAN immediately drives the TXCAN pin into a recessive state.

NOTE: *The user is responsible for ensuring that the MSCAN is not active when Power Down Mode is entered. The recommended procedure is to bring the MSCAN into Sleep Mode before the STOP or WAI instruction (if CSWAI is set) is executed. Otherwise the abort of an ongoing message can cause an error condition and can have an impact on the other bus devices.*

In Power Down Mode, all clocks are stopped and no registers can be accessed. If the MSCAN was not in Sleep Mode before Power Down Mode became active, the module would perform an internal recovery cycle after powering up. This causes some fixed delay before the module enters Normal Mode again.

4.6.7 Programmable Wake-Up Function

The MSCAN can be programmed to wake up the MSCAN as soon as bus activity is detected (see control bit WUPE in **3.3.1.1 MSCAN Control 0 Register (CANCTL0)**). The sensitivity to existing bus action can be modified by applying a low-pass filter function to the RXCAN input line while in Sleep Mode (see control bit WUPM in **3.3.1.2 MSCAN Control 1 Register (CANCTL1)**).

This feature can be used to protect the MSCAN from wake-up due to short glitches on the CAN bus lines. Such glitches can result e.g. from electromagnetic interference within noisy environments.

4.7 Reset Initialization

The reset state of each individual bit is listed within the Register Description section **3.3 Register Descriptions** which details all the registers and their bit-fields.

4.8 General

This section describes all interrupts originated by the MSCAN. It documents the enable bits and generated flags, if applicable. Each interrupt is listed and described separately. The description explains what causes the interrupt, the registers affected, and how the interrupt request is provided to the core.

Table 4-4 CRG Interrupt Vectors

Vector Address	Interrupt Source	CCR Mask	Local Enable	HPRIO Value to Elevate
1	Wake-Up Interrupt (WUPIF)	1 bit	CANRIER (WUPIE)	2
3	Error Interrupts Interrupt (CSCIF, OVRIF)	1 bit	CANRIER (CSCIE, OVRIE)	4
5	Receive Interrupt (RXF)	1 bit	CANRIER (RXFIE)	6
7	Transmit Interrupts (TXE2 - TXE0)	1 bit	CANTIER (TXEIE2 - TXEIE0)	8

NOTES:

1. Wake-Up interrupt vector address is specific to MCU, refer to MCU specification.
2. Wake-Up interrupt HPRIO value is specific to MCU, refer to MCU specification.
3. Error interrupt vector address is specific to MCU, refer to MCU specification.
4. Error interrupt HPRIO value is specific to MCU, refer to MCU specification.
5. Receive interrupt vector address is specific to MCU, refer to MCU specification.
6. Receive interrupt HPRIO value is specific to MCU, refer to MCU specification.
7. Transmit interrupt vector address is specific to MCU, refer to MCU specification.
8. Transmit interrupt HPRIO value is specific to MCU, refer to MCU specification.

4.9 Description of Interrupt Operation

The MSCAN supports four interrupt vectors, any of which can be individually masked (for details see sections **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)** to **3.3.1.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)**):

4.9.1 Transmit Interrupt

At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. The TXEx flag of the empty message buffer is set.

4.9.2 Receive Interrupt

A message is successfully received and shifted into the foreground buffer (RxFG) of the receiver FIFO. This interrupt is generated immediately after receiving the EOF symbol. The RXF flag is set. If there are multiple messages in the receiver FIFO, the RXF flag is set as soon as the next message is shifted to the foreground buffer.

4.9.3 Wake-Up Interrupt

Activity on the CAN bus occurred during MSCAN internal Sleep Mode and WUPE **3.3.1.1 MSCAN Control 0 Register (CANCTL0)** enabled.

4.9.4 Error Interrupt

An overrun of the receiver FIFO, error, warning or Bus-Off condition occurred. The **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** indicates one of the following conditions:

- **Overrun**
An overrun condition of the receiver FIFO as described in **4.2.3 Receive Structures** occurred.
- **CAN Status Change**
The actual value of the Transmit and Receive Error Counters control the bus state of the MSCAN. As soon as the error counters skip into a critical range (Tx/Rx-Warning, Tx/Rx-Error, Bus-Off) the MSCAN flags an error condition. The status change, which caused the error condition, is indicated by the TSTAT and RSTAT flags (see section **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** and **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)**).

4.10 Interrupt Acknowledge

Interrupts are directly associated with one or more status flags in either the **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** or the **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**. Interrupts are pending as long as one of the corresponding flags is set. The flags in the above registers must be reset within the interrupt handler to handshake the interrupt. The flags are reset by writing a “1” to the corresponding bit position. A flag cannot be cleared if the respective condition still prevails.

NOTE: *It must be guaranteed that the CPU only clears the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.*

4.11 Recovery from STOP or WAIT

The MSCAN can recover from STOP or WAIT via the Wake-Up interrupt. This interrupt can only occur if the MSCAN was in Sleep Mode (SLPRQ=1 and SLPK=1) before entering Power Down Mode, the wake-up option is enabled (WUPE=1) and the Wake-Up interrupt is enabled (WUPIE=1).

Section 5 Initialization/Application Information

5.1 MSCAN initialization

The procedure to initially start up the MSCAN module out of reset is as follows:

1. Assert CANE
2. Write to the configuration registers in Initialization Mode
3. Clear INTRQ to leave Initialization Mode and enter Normal Mode

If the configuration of registers which are writable in Initialization Mode only needs to be changed when the MSCAN module is in Normal Mode:

1. Make sure that the MSCAN transmission queue gets empty and bring the module into Sleep Mode by asserting SLPRQ and awaiting SLPK
2. Enter Initialization Mode: Assert INTRQ and await INITAK
3. Write to the configuration registers in Initialization Mode
4. Clear INTRQ to leave Initialization Mode and continue in Normal Mode

Index

–A–

Abort Acknowledge 30
 Abort Request 30
 ABTAK2 - ABTAK0 30
 ABTRQ2 - ABTRQ0 30
 AC7 – AC0 36
 Acceptance Code Bits 36
 Acceptance Mask Bits 38
 address offset 15
 AM7 – AM0 38

–B–

base address 15
 Baud Rate Prescaler 22
 Block Diagram 11
 BOSCH specification 11
 BRP 22
 bus monitoring mode 55
 bus rates 52
 Bus-Off 12, 19, 25, 26, 27, 58, 62

–C–

CAN protocol 11
 CAN Status Change 62
 CAN Status Change Interrupt Enable 27
 CAN Status Change Interrupt Flag 25
 CAN Stops in Wait Mode 18
 CAN system 13
 CANBTR0 19, 21
 CANBTR1 21, 22
 CANCTL0 17, 19
 CANCTL1 19, 21
 CANE 20
 CANIDAC 21, 32
 CANIDAR0-7 21, 35
 CANIDMR0-7 21
 CANRFLG 19, 24
 CANNIER 19, 26
 CANRXERR 34
 CANTAACK 19, 30
 CANTARQ 19, 29
 CANTBSEL 19, 31
 CANTFLG 19, 28
 CANTIER 19, 29
 CANTXERR 34
 CLKSRC 20
 clock 52

clock source 20
 CRC 26
 crystal 53
 CSCIE 27
 CSCIF 25
 CSWAI 18

–D–

Data Length Code bits 42
 data segment register 42
 DB7 - DB0 42
 DLC3 - DLC0 42
 DLR 42

–E–

emulation modes 55
 End of Frame 55
 EOF 55
 error counter 19, 20, 25, 52
 Error Passive 12
 Extended format identifier 41
 extended identifiers 39

–F–

features 12
 FIFO 12, 26, 33, 47, 48, 58, 62

–H–

handshake 19
 hard reset 19

–I–

ID Extended 41
 ID10 - ID0 41
 ID28 - ID0 41
 IDAM1 - IDAM0 32
 IDE 41
 Identifier Acceptance Mode 32
 identifier register 41
 IDHIT2 - IDHIT0 32
 IDR0-3 41
 IFS 46
 INITAK 17, 19, 21
 Initialization Mode 19, 21, 59
 Initialization Mode Acknowledge 21
 Initialization Mode Request 19
 INTRQ 17, 19
 interrupt enable 26
 Interrupt Operation 62

-L-

LISTEN 20
 Listen Only Mode 20
 Listen-Only 55
 local priority 46
 Loop Back Self Test Mode 20
 LOOPB 20

-M-

Memory Map 15
 Motorola Scalable Controller Area Network 11
 MSCAN 11
 MSCAN Clock Source 20
 MSCAN Enable 20
 MSCAN12 11

-N-

normal modes 55

-O-

Overrun 62
 overrun 26
 Overrun Interrupt Flag 26
 OVRIF 26

-P-

PHASE_SEG1 53
 PHASE_SEG2 53
 PLL 53
 Power Down Mode 60
 power saving 18, 56
 prescaler 53
 PROP_SEG 53

-R-

REC 25
 Receive Buffer Full Flag 26
 Receive Error Counter 25, 34
 Received Frame Flag 17
 Receiver Active Status 17
 Receiver Input Pin 13
 Receiver Status Bits 25
 register map 15
 Remote Transmission Request 41
 Reserved Registers 33
 reset 61
 RSTAT1, RSTAT0 25
 RSTATE1, RSTATE0 27
 RTR 41
 run mode 17, 56

RXACT 17
 RXCAN 13
 RXFRM 17

-S-

SAMP 23
 sample point 23
 Sampling 23
 security 56
 SJW1, SJW0 21
 Sleep Mode 18, 21, 57, 58, 59, 60, 61, 62, 63
 Sleep Mode Acknowledge 21
 Sleep Mode Request 18
 SLPRQ 17, 18
 SOF 42
 special modes 55
 SRR 41
 Standard format identify 41
 standard identifiers 39
 Start of Frame 42
 STOP 18, 63
 Stop Mode 57
 Substitute Remote Request 41
 SYNC_SEG 53
 SYNCH 18
 Synchronization Jump Width 21, 54
 Synchronized Status 18

-T-

TBPR 38, 42
 TEC 25
 TIME 18, 43
 Time Segment 1 23, 53
 Time Segment 2 23, 53
 time stamp 55
 Time Stamp register 38
 Timer Enable 18
 transceiver 13
 Transmit Buffer Priority Register 38
 Transmit Buffer Select 31
 Transmit Error Counter 25, 34
 Transmitter Output Pin 13
 TSEG1 53
 TSEG13 – TSEG10 23
 TSEG2 53
 TSEG22 – TSEG20 23
 TSRH, TSRL 43
 TSTAT1, TSTAT0 25
 TXCAN 13

-W-

WAIT 18, 63

Wait Mode 57
wake-up 27
Wake-Up Enable 18
Wake-Up Function 61
Wake-up Interrupt Enable 27
Wake-up Interrupt Flag 25
Warning 12
warning condition 62
WUPE 17, 18
WUPIE 18
WUPIF 25
WUPM 20

Block Guide End Sheet

**FINAL PAGE OF
70
PAGES**

CRG

Block User Guide

V04.05

Original Release Date: 29 Feb. 2000
Revised: 2 August 2002

Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
1.0	02/16/00	02/16/00		Initial Release
2.0	11/09/00	10/18/00		Initial SRS2.0 compliant release
2.01	03/03/01	03/03/01		Added RSBCK bit.
2.02	03/22/01			Make the content SRS compliant (usable as a customer document)
3.00	10 May 2001	10 May 2001		Modified according to new Pierce Oscillator feature and different PLLSEL bit write conditions
3.01	13 July 01	13 July 01		Minor corrections
3.02	30 July 01	30 July 01		Enhanced Block diagram (VREG, POR), Corrected register figure CRGINT to read and write
V03.03	2 Aug. 01	2 Aug. 01		Enhanced Clock Quality Check Diagram. Corrections in Reset section
V03.04	27 Aug. 01	27 Aug. 01		Improved description of exiting Wait and Pseudo Stop Mode. Enhanced SCME Bit description.
V03.05	5 Sept. 01	5 Sept. 01		Corrected C _{DC} position in Colpitts Oscillator Connections diagram.
V03.06	1 Oct. 01	1 Oct. 01		Motorola internal: spec tagging
V04.00	10 Oct. 01	10 Oct. 01		Added Low Voltage Reset feature
V04.01	21 Nov. 01	21 Nov. 01		Added Interrupts to Block Diagram. Corrected LVRF flag description.
V04.02	22 Nov. 01	22 Nov. 01		Re-Corrected LVRF flag description.
V04.03	11 Mar 02	11 Mar 02		Removed document number from all pages except cover page Replaced f_{VCOMIN} by f_{SCM} . Added Bus Clock formulas close to PLLCLK formulas. Spelling improvements
V04.04	03 May 02	03 May 02		Corrected in COPCTL register description: RSBCK is write once EXTAL signal description: mentioning pull-down resistor which is active in full stop mode
V04.05	2 Aug.02	2 Aug. 02		Removed oscillator specific information as separate Oscillator Block Guide is available now.

Table of Contents

Section 1 Introduction

1.1	Overview	9
1.2	Features	9
1.3	Modes of Operation	9
1.4	Block Diagram	10

Section 2 Signal Description

2.1	Overview	13
2.2	Detailed Signal Descriptions	13
2.2.1	VDDPLL, VSSPLL	13
2.2.2	XFC	13
2.2.3	RESET	13

Section 3 Memory Map and Registers

3.1	Overview	15
3.2	Module Memory Map	15
3.3	Register Descriptions	15
3.3.1	CRG Synthesizer Register (SYNR)	15
3.3.2	CRG Reference Divider Register (REFDV)	16
3.3.3	Reserved Register (CTFLG)	17
3.3.4	CRG Flags Register (CRGFLG)	17
3.3.5	CRG Interrupt Enable Register (CRGINT)	19
3.3.6	CRG Clock Select Register (CLKSEL)	19
3.3.7	CRG PLL Control Register (PLLCTL)	21
3.3.8	CRG RTI Control Register (RTICTL)	23
3.3.9	CRG COP Control Register (COPCTL)	24
3.3.10	Reserved Register (FORBYP)	25
3.3.11	Reserved Register (CTCTL)	26
3.3.12	CRG COP Timer Arm/Reset Register (ARMCOP)	26

Section 4 Functional Description

4.1	Functional Blocks	29
4.1.1	Phase Locked Loop (PLL)	29

4.1.2	System Clocks Generator	32
4.1.3	Clock Monitor (CM)	33
4.1.4	Clock Quality Checker	33
4.1.5	Computer Operating Properly Watchdog (COP)	35
4.1.6	Real Time Interrupt (RTI)	36
4.2	Operation Modes	37
4.2.1	Normal Mode	37
4.2.2	Self Clock Mode	37
4.3	Low Power Options	37
4.3.1	Run Mode	37
4.3.2	Wait Mode	37
4.3.3	CPU Stop Mode	42

Section 5 Resets

5.1	General	47
5.2	Description of Reset Operation	47
5.2.1	Clock Monitor Reset	48
5.2.2	Computer Operating Properly Watchdog (COP) Reset	49
5.2.3	Power On Reset, Low Voltage Reset	49

Section 6 Interrupts

6.1	General	51
6.2	Description of Interrupt Operation	51
6.2.1	Real Time Interrupt	51
6.2.2	PLL Lock Interrupt	51
6.2.3	Self Clock Mode Interrupt	51

List of Figures

Figure 1-1	Block diagram of CRG	11
Figure 2-1	PLL Loop Filter Connections	13
Figure 3-1	CRG Synthesizer Register (SYNR)	16
Figure 3-2	CRG Reference Divider Register (REFDV)	16
Figure 3-3	Reserved Register (CTFLG)	17
Figure 3-4	CRG Flags Register (CRGFLG)	17
Figure 3-5	CRG Interrupt Enable Register (CRGINT)	19
Figure 3-6	CRG Clock Select Register (CLKSEL)	19
Figure 3-7	CRG PLL Control Register (PLLCTL)	21
Figure 3-8	CRG RTI Control Register (RTICTL)	23
Figure 3-9	CRG COP Control Register (COPCTL)	24
Figure 3-10	Reserved Register (FORBYP)	26
Figure 3-11	Reserved Register (CTCTL)	26
Figure 3-12	ARMCOP Register Diagram	27
Figure 4-1	PLL Functional Diagram	29
Figure 4-2	System Clocks Generator	32
Figure 4-3	Core Clock and Bus Clock relationship	33
Figure 4-4	Check Window Example	34
Figure 4-5	Sequence for Clock Quality Check	34
Figure 4-6	Clock Chain for COP	35
Figure 4-7	Clock Chain for RTI	36
Figure 4-8	Wait Mode Entry/Exit Sequence	39
Figure 4-9	Stop Mode Entry/Exit Sequence	43
Figure 5-1	RESET Timing	48
Figure 5-2	RESET pin tied to VDD (by a pull-up resistor)	49
Figure 5-3	RESET pin held low externally	50

List of Tables

Table 3-1	CRG Memory Map	15
Table 3-2	RTI Frequency Divide Rates	23
Table 3-3	COP Watchdog Rates	25
Table 4-1	MCU configuration during Wait Mode	38
Table 4-2	Outcome of Clock Loss in Wait Mode	40
Table 4-3	Outcome of Clock Loss in Pseudo-Stop Mode	44
Table 5-1	Reset Summary	47
Table 5-2	Reset Vector Selection	47
Table 6-1	CRG Interrupt Vectors	51

Section 1 Introduction

1.1 Overview

This specification describes the function of the Clocks and Reset Generator (CRG).

1.2 Features

The main features of this block are:

- Phase Locked Loop (PLL) frequency multiplier
 - Reference divider
 - Automatic bandwidth control mode for low-jitter operation
 - Automatic frequency lock detector
 - CPU interrupt on entry or exit from locked condition
 - Self Clock Mode in absence of reference clock
- System Clock Generator
 - Clock Quality Check
 - Clock switch for either Oscillator or PLL based system clocks
 - User selectable disabling of clocks during Wait Mode for reduced power consumption.
- Computer Operating Properly (COP) watchdog timer with time-out clear window.
- System Reset generation from the following possible sources:
 - Power on reset
 - Low voltage reset
 - **Refer to device specification for availability of this feature.**
 - COP reset
 - Loss of clock reset
 - External pin reset
- Real-Time Interrupt (RTI)

1.3 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the CRG.

- Run Mode

All functional parts of the CRG are running during normal Run Mode. If RTI or COP functionality is required the individual bits of the associated rate select registers (COPCTL, RTICTL) have to be set to a non zero value.

- Wait Mode

This mode allows to disable the system and core clocks depending on the configuration of the individual bits in the CLKSEL register.

- Stop Mode

Depending on the setting of the PSTP bit Stop Mode can be differentiated between Full Stop Mode (PSTP=0) and Pseudo Stop Mode (PSTP=1).

- Full Stop Mode

The oscillator is disabled and thus all system and core clocks are stopped. The COP and the RTI remain frozen.

- Pseudo Stop Mode

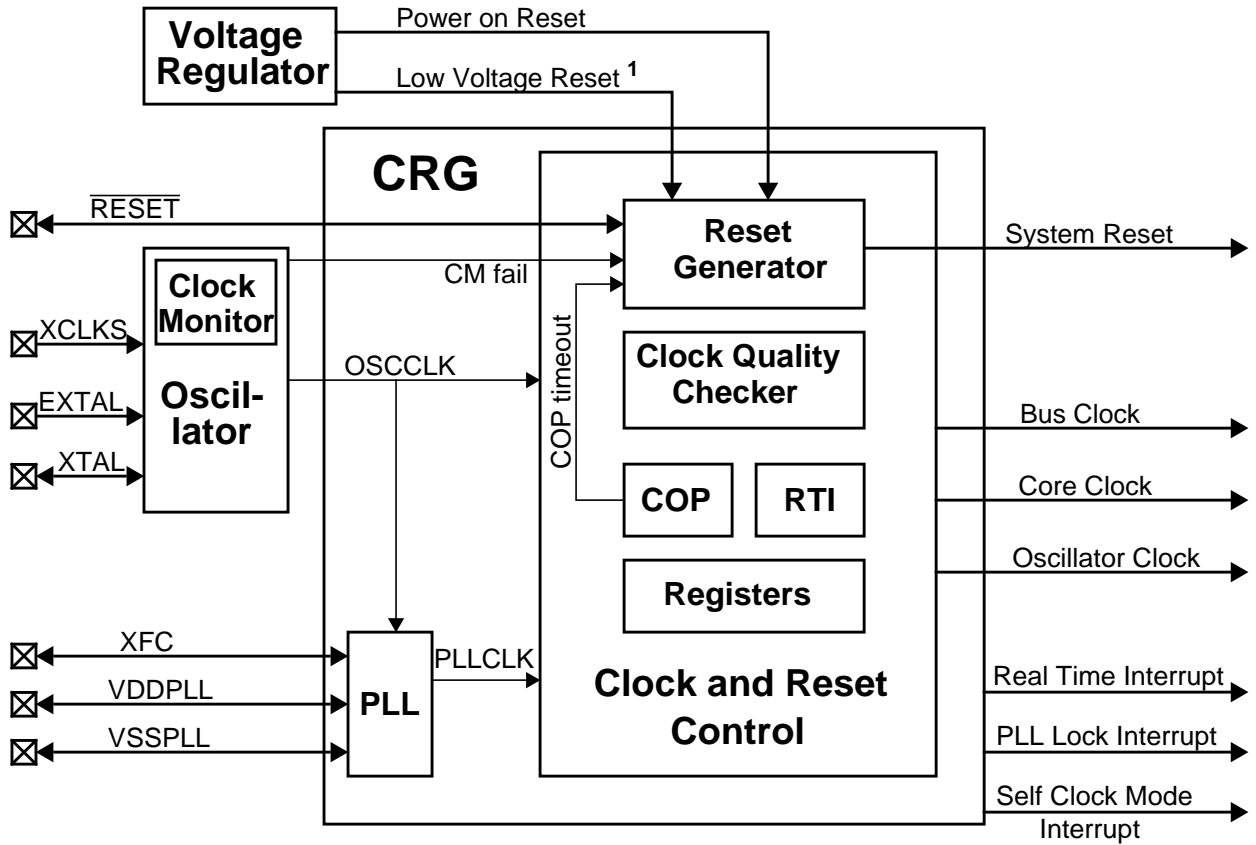
The oscillator continues to run and most of the system and core clocks are stopped. If the respective enable bits are set the COP and RTI will continue to run, else they remain frozen.

- Self Clock Mode

Self Clock Mode will be entered if the Clock Monitor Enable Bit (CME) and the Self Clock Mode Enable Bit (SCME) are both asserted and the clock monitor in the oscillator block detects a loss of clock. As soon as Self Clock Mode is entered the CRG starts to perform a clock quality check. Self Clock Mode remains active until the clock quality check indicates that the required quality of the incoming clock signal is met (frequency and amplitude). Self Clock Mode should be used for safety purposes only. It provides reduced functionality to the MCU in case a loss of clock is causing severe system conditions.

1.4 Block Diagram

Figure 1-1 shows a block diagram of the CRG.



1) Refer to device specification for availability of the low voltage reset feature.

Figure 1-1 Block diagram of CRG

Section 2 Signal Description

2.1 Overview

This section lists and describes the signals that connect off chip.

2.2 Detailed Signal Descriptions

2.2.1 VDDPLL, VSSPLL

These pins provides operating voltage (VDDPLL) and ground (VSSPLL) for the PLL circuitry. This allows the supply voltage to the PLL to be independently bypassed. Even if PLL usage is not required VDDPLL and VSSPLL must be connected to properly.

2.2.2 XFC

A passive external loop filter must be placed on the XFC pin. The filter is a second-order, low-pass filter to eliminate the VCO input ripple. The value of the external filter network and the reference frequency determines the speed of the corrections and the stability of the PLL. **Refer to device specification for calculation of PLL Loop Filter (XFC) components.** If PLL usage is not required the XFC pin must be tied to VDDPLL.

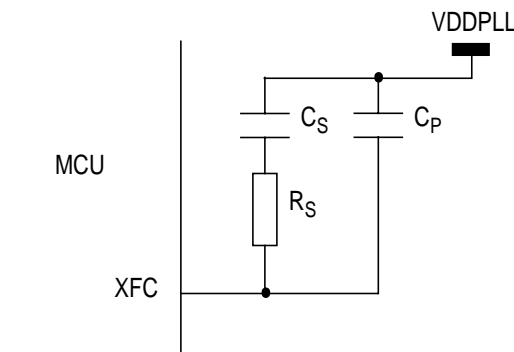


Figure 2-1 PLL Loop Filter Connections

2.2.3 $\overline{\text{RESET}}$

$\overline{\text{RESET}}$ is an active low bidirectional reset pin. As an input it initializes the MCU asynchronously to a known start-up state. As an open-drain output it indicates that an system reset (internal to MCU) has been triggered.

Section 3 Memory Map and Registers

3.1 Overview

This section provides a detailed description of all registers accessible in the CRG.

3.2 Module Memory Map

Table 3-1 gives an overview on all CRG registers.

Table 3-1 CRG Memory Map

Address Offset	Use	Access
\$_00	CRG Synthesizer Register (SYNR)	R/W
\$_01	CRG Reference Divider Register (REFDV)	R/W
\$_02	CRG Test Flags Register (CTFLG) ¹	R/W
\$_03	CRG Flags Register (CRGFLG)	R/W
\$_04	CRG Interrupt Enable Register (CRGINT)	R/W
\$_05	CRG Clock Select Register (CLKSEL)	R/W
\$_06	CRG PLL Control Register (PLLCTL)	R/W
\$_07	CRG RTI Control Register (RTICTL)	R/W
\$_08	CRG COP Control Register (COPCTL)	R/W
\$_09	CRG Force and Bypass Test Register (FORBYP) ²	R/W
\$_0A	CRG Test Control Register (CTCTL) ³	R/W
\$_0B	CRG COP Arm/Timer Reset (ARMCOP)	R/W

NOTES:

1. CTFLG is intended for factory test purposes only.
2. FORBYP is intended for factory test purposes only.
3. CTCTL is intended for factory test purposes only.

NOTE: *Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.*

3.3 Register Descriptions

This section describes in address order all the CRG registers and their individual bits.

3.3.1 CRG Synthesizer Register (SYNR)

The SYNR register controls the multiplication factor of the PLL. If the PLL is on, the count in the loop divider (SYNR) register effectively multiplies up the PLL clock (PLLCLK) from the reference frequency by $2 \times (\text{SYNR} + 1)$. PLLCLK will not be below the minimum VCO frequency (f_{SCM}).

$$PLLCLK = 2 \times OSCCLK \times \frac{(SYNR + 1)}{(REFDV + 1)}$$

NOTE: If PLL is selected (PLLSEL=1), Bus Clock = PLLCLK / 2
 Bus Clock must not exceed the maximum operating system frequency.

Address Offset: \$_00

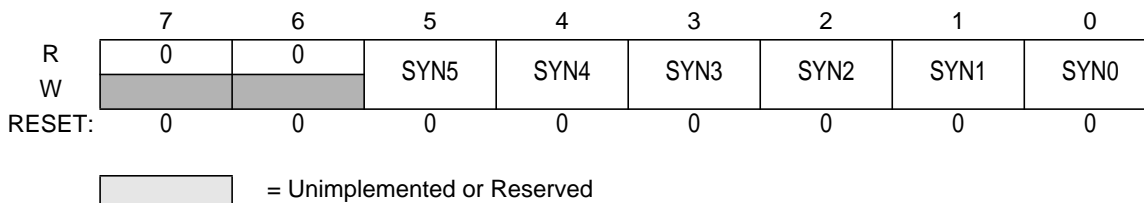


Figure 3-1 CRG Synthesizer Register (SYNR)

Read: anytime

Write: anytime except if PLLSEL = 1

NOTE: Write to this register initializes the lock detector bit and the track detector bit.

3.3.2 CRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the PLL multiplier steps. The count in the reference divider divides OSCCLK frequency by REFDV+1.

Address Offset: \$_01

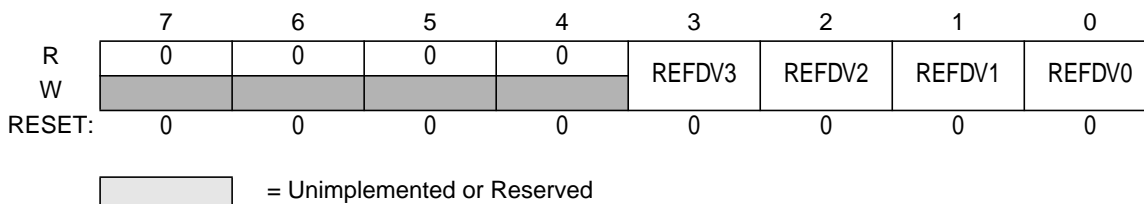


Figure 3-2 CRG Reference Divider Register (REFDV)

Read: anytime

Write: anytime except when PLLSEL = 1

NOTE: Write to this register initializes the lock detector bit and the track detector bit.

3.3.3 Reserved Register (CTFLG)

This register is reserved for factory testing of the CRG module and is not available in normal modes.

Address Offset: $\$_02$

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-3 Reserved Register (CTFLG)

Read: always reads \$00 in normal modes

Write: unimplemented in normal modes

NOTE: Writing to this register when in special mode can alter the CRG functionality.

3.3.4 CRG Flags Register (CRGFLG)

This register provides CRG status bits and flags.

Address Offset: $\$_03$

	7	6	5	4	3	2	1	0
R	RTIF	PORF	LVRF	LOCKIF	LOCK	TRACK	SCMIF	SCM
W								
RESET:	0	1	2	0	0	0	0	0

= Unimplemented or Reserved

NOTES:

1. PORF is set to 1 when a power on reset occurs. Unaffected by system reset.
2. LVRF is set to 1 when a low voltage reset occurs. Unaffected by system reset.

Figure 3-4 CRG Flags Register (CRGFLG)

Read: anytime

Write: refer to each bit for individual write conditions

RTIF — Real Time Interrupt Flag

RTIF is set to 1 at the end of the RTI period. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (RTIE=1), RTIF causes an interrupt request.

1 = RTI time-out has occurred.

0 = RTI time-out has not yet occurred.

PORF — Power on Reset Flag

PORF is set to 1 when a power on reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect.

- 1 = Power on reset has occurred.
- 0 = Power on reset has not occurred.

LVRF — Low Voltage Reset Flag

If low voltage reset feature is not available (see device specification) LVRF always reads 0.

LVRF is set to 1 when a low voltage reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect.

- 1 = Low voltage reset has occurred.
- 0 = Low voltage reset has not occurred.

LOCKIF — PLL Lock Interrupt Flag

LOCKIF is set to 1 when LOCK status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (LOCKIE=1), LOCKIF causes an interrupt request.

- 1 = LOCK bit has changed.
- 0 = No change in LOCK bit.

LOCK — Lock Status Bit

LOCK reflects the current state of PLL lock condition. This bit is cleared in Self Clock Mode. Writes have no effect.

- 1 = PLL VCO is within the desired tolerance of the target frequency.
- 0 = PLL VCO is not within the desired tolerance of the target frequency.

TRACK — Track Status Bit

TRACK reflects the current state of PLL track condition. This bit is cleared in Self Clock Mode. Writes have no effect.

- 1 = Tracking mode status.
- 0 = Acquisition mode status.

SCMIF — Self Clock Mode Interrupt Flag

SCMIF is set to 1 when SCM status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (SCMIE=1), SCMIF causes an interrupt request.

- 1 = SCM bit has changed.
- 0 = No change in SCM bit.

SCM — Self Clock Mode Status Bit

SCM reflects the current clocking mode. Writes have no effect.

- 1 = MCU is operating in Self Clock Mode with OSCCLK in an unknown state. All clocks are derived from PLLCLK running at its minimum frequency f_{SCM} .
- 0 = MCU is operating normally with OSCCLK available.

3.3.5 CRG Interrupt Enable Register (CRGINT)

This register enables CRG interrupt requests.

Address Offset: \$_04

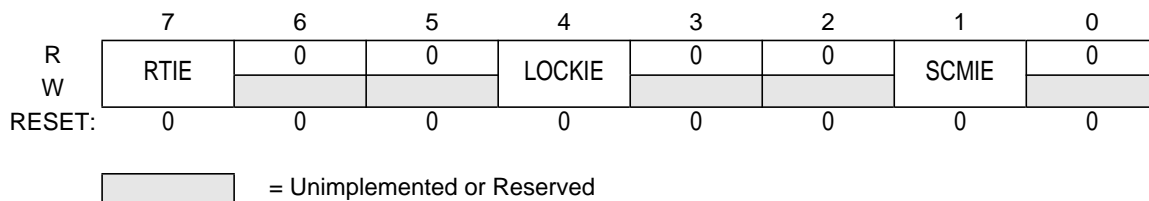


Figure 3-5 CRG Interrupt Enable Register (CRGINT)

Read: anytime

Write: anytime

RTIE — Real Time Interrupt Enable Bit.

1 = Interrupt will be requested whenever RTIF is set.

0 = Interrupt requests from RTI are disabled.

LOCKIE — Lock Interrupt Enable Bit

1 = Interrupt will be requested whenever LOCKIF is set.

0 = LOCK interrupt requests are disabled.

SCMIE — Self Clock Mode Interrupt Enable Bit

1 = Interrupt will be requested whenever SCMIF is set.

0 = SCM interrupt requests are disabled.

3.3.6 CRG Clock Select Register (CLKSEL)

This register controls CRG clock selection. Refer to **Figure 4-2 System Clocks Generator** for more details on the effect of each bit.

Address Offset: \$_05

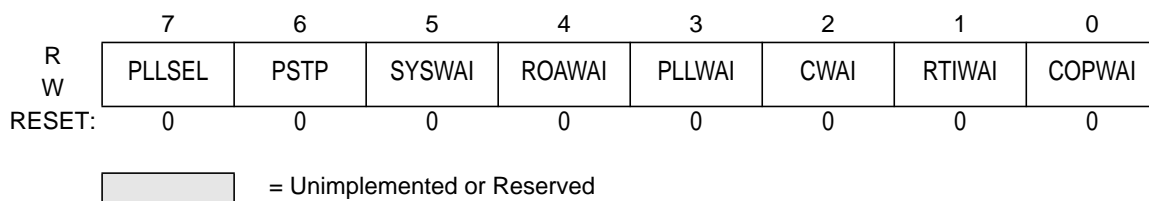


Figure 3-6 CRG Clock Select Register (CLKSEL)

Read: anytime

Write: refer to each bit for individual write conditions

PLLSEL — PLL Select Bit

Write anytime. Writing a one when LOCK=0 and AUTO=1, or TRACK=0 and AUTO=0 has no effect. This prevents the selection of an unstable PLLCLK as SYSCLK. PLLSEL bit is cleared when the MCU enters Self Clock Mode, Stop Mode or Wait Mode with PLLWAI bit set.

1 = System clocks are derived from PLLCLK (Bus Clock = PLLCLK / 2).

0 = System clocks are derived from OSCCLK (Bus Clock = OSCCLK / 2).

PSTP — Pseudo Stop Bit

Write: anytime

This bit controls the functionality of the oscillator during Stop Mode.

1 = Oscillator continues to run in Stop Mode (Pseudo Stop). The oscillator amplitude is reduced.

Refer to oscillator block description for availability of a reduced oscillator amplitude.

0 = Oscillator is disabled in Stop Mode.

NOTE: *Pseudo-STOP allows for faster STOP recovery and reduces the mechanical stress and aging of the resonator in case of frequent STOP conditions at the expense of a slightly increased power consumption.*

Lower oscillator amplitude exhibits lower power consumption but could have adverse effects during any Electro-Magnetic Susceptibility (EMS) tests.

SYSWAI — System clocks stop in Wait Mode Bit

Write: anytime

1 = In Wait Mode the system clocks stop.

0 = In Wait Mode the system clocks continue to run.

NOTE: *RTI and COP are not affected by SYSWAI bit.*

ROAWAI — Reduced Oscillator Amplitude in Wait Mode Bit.

Refer to oscillator block description for availability of a reduced oscillator amplitude. If no such feature exists in the oscillator block then setting this bit to one will not have any effect on power consumption!

Write: anytime

1 = Reduced oscillator amplitude in Wait Mode.

0 = Normal oscillator amplitude in Wait Mode.

NOTE: *Lower oscillator amplitude exhibits lower power consumption but could have adverse effects during any Electro-Magnetic Susceptibility (EMS) tests.*

PLLWAI — PLL stops in Wait Mode Bit

Write: anytime

If PLLWAI is set, the CRG will clear the PLLSEL bit before entering Wait Mode. The PLLON bit remains set during Wait Mode but the PLL is powered down. Upon exiting Wait Mode, the PLLSEL bit has to be set manually if PLL clock is required.

While the PLLWAI bit is set the AUTO bit is set to 1 in order to allow the PLL to automatically lock on the selected target frequency after exiting Wait Mode.

1 = PLL stops in Wait Mode.

0 = PLL keeps running in Wait Mode.

CWAI — Core stops in Wait Mode Bit

Write: anytime

1 = Core clock stops in Wait Mode.

0 = Core clock keeps running in Wait Mode.

RTIWAI — RTI stops in Wait Mode Bit

Write: anytime

1 = RTI stops and initializes the RTI dividers whenever the part goes into Wait Mode.

0 = RTI keeps running in Wait Mode.

COPWAI — COP stops in Wait Mode Bit

Normal modes: Write once

Special modes: Write anytime

1 = COP stops and initializes the COP dividers whenever the part goes into Wait Mode.

0 = COP keeps running in Wait Mode.

3.3.7 CRG PLL Control Register (PLLCTL)

This register controls the PLL functionality.

Address Offset: $\$_06$

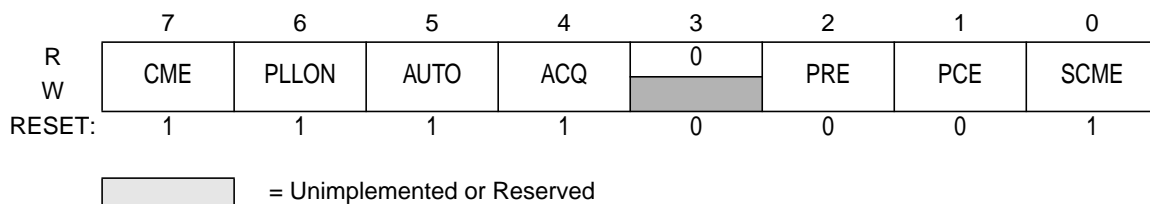


Figure 3-7 CRG PLL Control Register (PLLCTL)

Read: anytime

Write: refer to each bit for individual write conditions

CME — Clock Monitor Enable Bit

CME enables the clock monitor. Write anytime except when SCM = 1.

1 = Clock monitor is enabled. Slow or stopped clocks will cause a clock monitor reset sequence or Self Clock Mode.

0 = Clock monitor is disabled.

NOTE: *Operating with CME=0 will not detect any loss of clock. In case of poor clock quality this could cause unpredictable operation of the MCU!*

In Stop Mode (PSTP=0) the clock monitor is disabled independently of the CME bit setting and any loss of clock will not be detected.

PLLON — Phase Lock Loop On Bit

PLLON turns on the PLL circuitry. In Self Clock Mode, the PLL is turned on, but the PLLON bit reads the last latched value. Write anytime except when PLLSEL = 1.

- 1 = PLL is turned on. If AUTO bit is set, the PLL will lock automatically.
- 0 = PLL is turned off.

AUTO — Automatic Bandwidth Control Bit

AUTO selects either the high bandwidth (acquisition) mode or the low bandwidth (tracking) mode depending on how close to the desired frequency the VCO is running. Write anytime except when PLLWAI=1, because PLLWAI sets the AUTO bit to 1.

- 1 = Automatic Mode Control is enabled and ACQ bit has no effect.
- 0 = Automatic Mode Control is disabled and the PLL is under software control, using ACQ bit.

ACQ — Acquisition Bit

Write anytime. If AUTO=1 this bit has no effect.

- 1 = High bandwidth filter is selected.
- 0 = Low bandwidth filter is selected.

PRE — RTI Enable during Pseudo Stop Bit

PRE enables the RTI during Pseudo Stop Mode. Write anytime.

- 1 = RTI continues running during Pseudo Stop Mode.
- 0 = RTI stops running during Pseudo Stop Mode.

NOTE: *If the PRE bit is cleared the RTI dividers will go static while Pseudo-Stop Mode is active. The RTI dividers will not initialize like in Wait Mode with RTIWAI bit set.*

PCE — COP Enable during Pseudo Stop Bit

PCE enables the COP during Pseudo Stop Mode. Write anytime.

- 1 = COP continues running during Pseudo Stop Mode
- 0 = COP stops running during Pseudo Stop Mode

NOTE: *If the PCE bit is cleared the COP dividers will go static while Pseudo-Stop Mode is active. The COP dividers will not initialize like in Wait Mode with COPWAI bit set.*

SCME — Self Clock Mode Enable Bit

Normal modes: Write once

Special modes: Write anytime

SCME can not be cleared while operating in Self Clock Mode (SCM=1).

- 0 = Detection of crystal clock failure causes clock monitor reset (see **5.2.1 Clock Monitor Reset**).

1 = Detection of crystal clock failure forces the MCU in Self Clock Mode (see 4.2.2 Self Clock Mode).

3.3.8 CRG RTI Control Register (RTICTL)

This register selects the timeout period for the Real Time Interrupt.

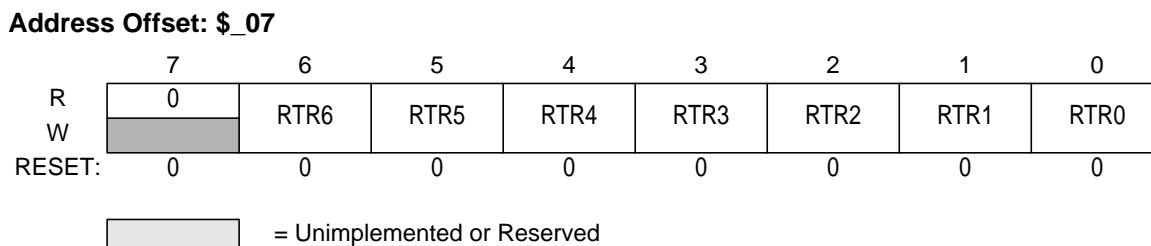


Figure 3-8 CRG RTI Control Register (RTICTL)

Read: anytime

Write: anytime

NOTE: A write to this register initializes the RTI counter.

RTR[6:4] — Real Time Interrupt Prescale Rate Select Bits

These bits select the prescale rate for the RTI. See **Table 3-2**.

RTR[3:0] — Real Time Interrupt Modulus Counter Select Bits

These bits select the modulus counter target value to provide additional granularity. **Table 3-2** shows all possible divide values selectable by the RTICTL register. The source clock for the RTI is OSCCLK.

Table 3-2 RTI Frequency Divide Rates

RTR[3:0]	RTR[6:4] =							
	000 (OFF)	001 (2 ¹⁰)	010 (2 ¹¹)	011 (2 ¹²)	100 (2 ¹³)	101 (2 ¹⁴)	110 (2 ¹⁵)	111 (2 ¹⁶)
0000 (÷1)	OFF*	2 ¹⁰	2 ¹¹	2 ¹²	2 ¹³	2 ¹⁴	2 ¹⁵	2 ¹⁶
0001 (÷2)	OFF*	2x2 ¹⁰	2x2 ¹¹	2x2 ¹²	2x2 ¹³	2x2 ¹⁴	2x2 ¹⁵	2x2 ¹⁶
0010 (÷3)	OFF*	3x2 ¹⁰	3x2 ¹¹	3x2 ¹²	3x2 ¹³	3x2 ¹⁴	3x2 ¹⁵	3x2 ¹⁶
0011 (÷4)	OFF*	4x2 ¹⁰	4x2 ¹¹	4x2 ¹²	4x2 ¹³	4x2 ¹⁴	4x2 ¹⁵	4x2 ¹⁶
0100 (÷5)	OFF*	5x2 ¹⁰	5x2 ¹¹	5x2 ¹²	5x2 ¹³	5x2 ¹⁴	5x2 ¹⁵	5x2 ¹⁶
0101 (÷6)	OFF*	6x2 ¹⁰	6x2 ¹¹	6x2 ¹²	6x2 ¹³	6x2 ¹⁴	6x2 ¹⁵	6x2 ¹⁶

Table 3-2 RTI Frequency Divide Rates

RTR[3:0]	RTR[6:4] =							
0110 (÷7)	OFF*	7x2 ¹⁰	7x2 ¹¹	7x2 ¹²	7x2 ¹³	7x2 ¹⁴	7x2 ¹⁵	7x2 ¹⁶
0111 (÷8)	OFF*	8x2 ¹⁰	8x2 ¹¹	8x2 ¹²	8x2 ¹³	8x2 ¹⁴	8x2 ¹⁵	8x2 ¹⁶
1000 (÷9)	OFF*	9x2 ¹⁰	9x2 ¹¹	9x2 ¹²	9x2 ¹³	9x2 ¹⁴	9x2 ¹⁵	9x2 ¹⁶
1001 (÷10)	OFF*	10x2 ¹⁰	10x2 ¹¹	10x2 ¹²	10x2 ¹³	10x2 ¹⁴	10x2 ¹⁵	10x2 ¹⁶
1010 (÷11)	OFF*	11x2 ¹⁰	11x2 ¹¹	11x2 ¹²	11x2 ¹³	11x2 ¹⁴	11x2 ¹⁵	11x2 ¹⁶
1011 (÷12)	OFF*	12x2 ¹⁰	12x2 ¹¹	12x2 ¹²	12x2 ¹³	12x2 ¹⁴	12x2 ¹⁵	12x2 ¹⁶
1100 (÷13)	OFF*	13x2 ¹⁰	13x2 ¹¹	13x2 ¹²	13x2 ¹³	13x2 ¹⁴	13x2 ¹⁵	13x2 ¹⁶
1101 (÷14)	OFF*	14x2 ¹⁰	14x2 ¹¹	14x2 ¹²	14x2 ¹³	14x2 ¹⁴	14x2 ¹⁵	14x2 ¹⁶
1110 (÷15)	OFF*	15x2 ¹⁰	15x2 ¹¹	15x2 ¹²	15x2 ¹³	15x2 ¹⁴	15x2 ¹⁵	15x2 ¹⁶
1111 (÷16)	OFF*	16x2 ¹⁰	16x2 ¹¹	16x2 ¹²	16x2 ¹³	16x2 ¹⁴	16x2 ¹⁵	16x2 ¹⁶

* Denotes the default value out of reset. This value should be used to disable the RTI to ensure future backwards compatibility.

3.3.9 CRG COP Control Register (COPCTL)

This register controls the COP (Computer Operating Properly) watchdog.

Address Offset: \$_08

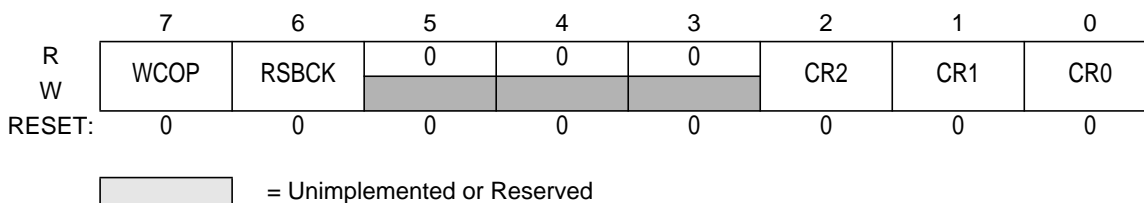


Figure 3-9 CRG COP Control Register (COPCTL)

Read: anytime

Write: WCOP, CR2, CR1, CR0: once in user mode, anytime in special mode

Write: RSBCK: once

WCOP — Window COP Mode Bit

When set, a write to the ARMCOP register must occur in the last 25% of the selected period. A write during the first 75% of the selected period will reset the part. As long as all writes occur during this window, \$55 can be written as often as desired. Once \$AA is written after the \$55, the time-out logic restarts and the user must wait until the next window before writing to ARMCOP. **Table 3-3** shows the exact duration of this window for the seven available COP rates.

- 1 = Window COP operation
- 0 = Normal COP operation

RSBCK — COP and RTI stop in Active BDM mode Bit

- 1 = Stops the COP and RTI counters whenever the part is in Active BDM mode.
- 0 = Allows the COP and RTI to keep running in Active BDM mode.

CR[2:0] — COP Watchdog Timer Rate select

These bits select the COP time-out rate (see **Table 3-3**). The COP time-out period is OSCCLK period divided by CR[2:0] value. Writing a nonzero value to CR[2:0] enables the COP counter and starts the time-out period. A COP counter time-out causes a system reset. This can be avoided by periodically (before time-out) reinitializing the COP counter via the ARMCOP register.

Table 3-3 COP Watchdog Rates¹

CR2	CR1	CR0	OSCCLK cycles to time-out
0	0	0	COP disabled
0	0	1	2^{14}
0	1	0	2^{16}
0	1	1	2^{18}
1	0	0	2^{20}
1	0	1	2^{22}
1	1	0	2^{23}
1	1	1	2^{24}

NOTES:

1. OSCCLK cycles are referenced from the previous COP time-out reset (writing \$55/\$AA to the ARMCOP register)

3.3.10 Reserved Register (FORBYP)

NOTE: *This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special modes can alter the CRG's functionality.*

Address Offset: \$_09

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
RESET:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-10 Reserved Register (FORBYP)

Read: always read \$00 except in special modes

Write: only in special modes

3.3.11 Reserved Register (CTCTL)

NOTE: *This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special test modes can alter the CRG’s functionality.*

Address Offset: \$_0A

	7	6	5	4	3	2	1	0
R	1	0	0	0	0	0	0	0
W								
RESET:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-11 Reserved Register (CTCTL)

Read: always read \$80 except in special modes

Write: only in special modes

3.3.12 CRG COP Timer Arm/Reset Register (ARMCOP)

This register is used to restart the COP time-out period.

Address Offset: \$_0B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RESET:	0	0	0	0	0	0	0	0

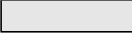
 = Unimplemented or Reserved

Figure 3-12 ARMCOP Register Diagram

Read: always reads \$00

Write: anytime

When the COP is disabled (CR[2:0] = “000”) writing to this register has no effect.

When the COP is enabled by setting CR[2:0] nonzero, the following applies:

Writing any value other than \$55 or \$AA causes a COP reset. To restart the COP time-out period you must write \$55 followed by a write of \$AA. Other instructions may be executed between these writes but the sequence (\$55, \$AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of \$55 writes or sequences of \$AA writes are allowed. When the WCOP bit is set, \$55 and \$AA writes must be done in the last 25% of the selected time-out period; writing any value in the first 75% of the selected period will cause a COP reset.

Section 4 Functional Description

4.1 Functional Blocks

4.1.1 Phase Locked Loop (PLL)

The PLL is used to run the MCU from a different time base than the incoming OSCCLK. For increased flexibility, OSCCLK can be divided in a range of 1 to 16 to generate the reference frequency. This offers a finer multiplication granularity. The PLL can multiply this reference clock by a multiple of 2, 4, 6,... 126,128 based on the SYN register.

$$PLLCLK = 2 \times OSCCLK \times \frac{[SYNR + 1]}{[REFDV + 1]}$$

CAUTION: Although it is possible to set the two dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU. If (PLLSEL=1), Bus Clock = PLLCLK / 2

The PLL is a frequency generator that operates in either acquisition mode or tracking mode, depending on the difference between the output frequency and the target frequency. The PLL can change between acquisition and tracking modes either automatically or manually.

The VCO has a minimum operating frequency, which corresponds to the self clock mode frequency f_{SCM} .

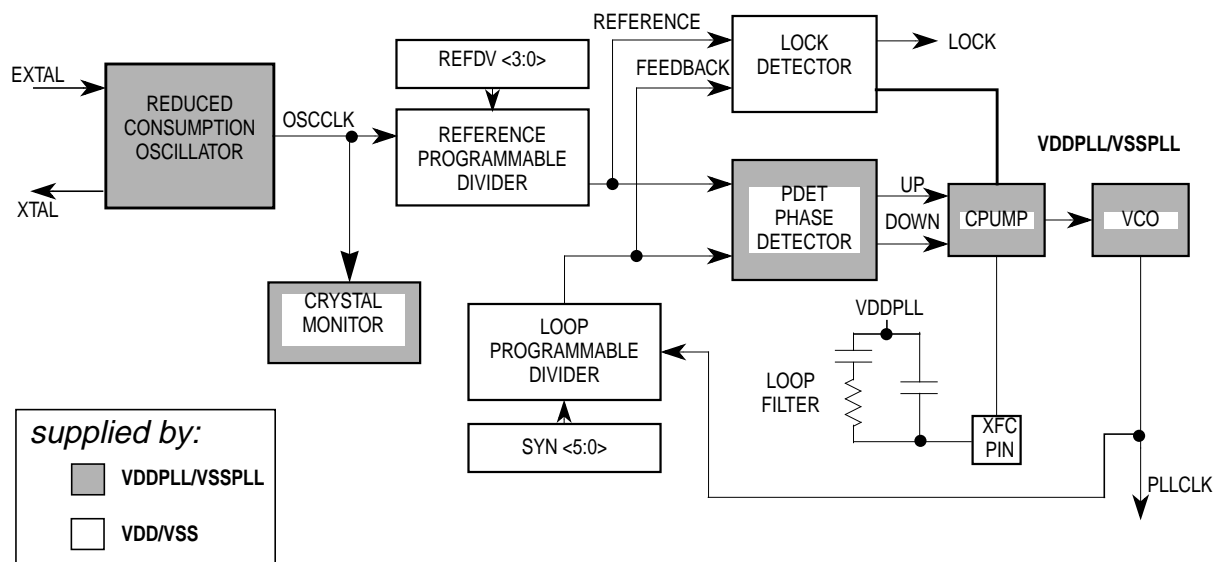


Figure 4-1 PLL Functional Diagram

4.1.1.1 PLL Operation

The oscillator output clock signal (OSCCLK) is fed through the reference programmable divider and is divided in a range of 1 to 16 (REFDV+1) to output the REFERENCE clock. The VCO output clock, (PLLCLK) is fed back through the programmable loop divider and is divided in a range of 2 to 128 in increments of $[2 \times (\text{SYNR} + 1)]$ to output the FEEDBACK clock. See **Figure 4-1**.

The phase detector then compares the FEEDBACK clock, with the REFERENCE clock. Correction pulses are generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the external filter capacitor connected to XFC pin, based on the width and direction of the correction pulse. The filter can make fast or slow corrections depending on its mode, as described in the next subsection. The values of the external filter network and the reference frequency determine the speed of the corrections and the stability of the PLL.

4.1.1.2 Acquisition and Tracking Modes

The lock detector compares the frequencies of the FEEDBACK clock, and the REFERENCE clock. Therefore, the speed of the lock detector is directly proportional to the final reference frequency. The circuit determines the mode of the PLL and the lock condition based on this comparison.

The PLL filter can be manually or automatically configured into one of two possible operating modes:

- Acquisition mode

In acquisition mode, the filter can make large frequency corrections to the VCO. This mode is used at PLL start-up or when the PLL has suffered a severe noise hit and the VCO frequency is far off the desired frequency. When in acquisition mode, the TRACK status bit is cleared in the CRGFLG register.

- Tracking mode

In tracking mode, the filter makes only small corrections to the frequency of the VCO. PLL jitter is much lower in tracking mode, but the response to noise is also slower. The PLL enters tracking mode when the VCO frequency is nearly correct and the TRACK bit is set in the CRGFLG register.

The PLL can change the bandwidth or operational mode of the loop filter manually or automatically.

In automatic bandwidth control mode (AUTO = 1), the lock detector automatically switches between acquisition and tracking modes. Automatic bandwidth control mode also is used to determine when the PLL clock (PLLCLK) is safe to use as the source for the system and core clocks. If PLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and then check the LOCK bit. If CPU interrupts are disabled, software can poll the LOCK bit continuously (during PLL start-up, usually) or at periodic intervals. In either case, only when the LOCK bit is set, is the PLLCLK clock safe to use as the source for the system and core clocks. If the PLL is selected as the source for the system and core clocks and the LOCK bit is clear, the PLL has suffered a severe noise hit and the software must take appropriate action, depending on the application.

The following conditions apply when the PLL is in automatic bandwidth control mode (AUTO=1):

- The TRACK bit is a read-only indicator of the mode of the filter.

- The TRACK bit is set when the VCO frequency is within a certain tolerance, Δ_{trk} , and is clear when the VCO frequency is out of a certain tolerance, Δ_{unt} .
- The LOCK bit is a read-only indicator of the locked state of the PLL.
- The LOCK bit is set when the VCO frequency is within a certain tolerance, Δ_{Lock} , and is cleared when the VCO frequency is out of a certain tolerance, Δ_{unl} .
- CPU interrupts can occur if enabled ($\text{LOCKIE} = 1$) when the lock condition changes, toggling the LOCK bit.

The PLL can also operate in manual mode ($\text{AUTO} = 0$). Manual mode is used by systems that do not require an indicator of the lock condition for proper operation. Such systems typically operate well below the maximum system frequency (f_{sys}) and require fast start-up. The following conditions apply when in manual mode:

- ACQ is a writable control bit that controls the mode of the filter. Before turning on the PLL in manual mode, the ACQ bit should be asserted to configure the filter in acquisition mode.
- After turning on the PLL by setting the PLLON bit software must wait a given time (t_{acq}) before entering tracking mode ($\text{ACQ} = 0$).
- After entering tracking mode software must wait a given time (t_{al}) before selecting the PLLCLK as the source for system and core clocks ($\text{PLLSEL} = 1$).

4.1.2 System Clocks Generator

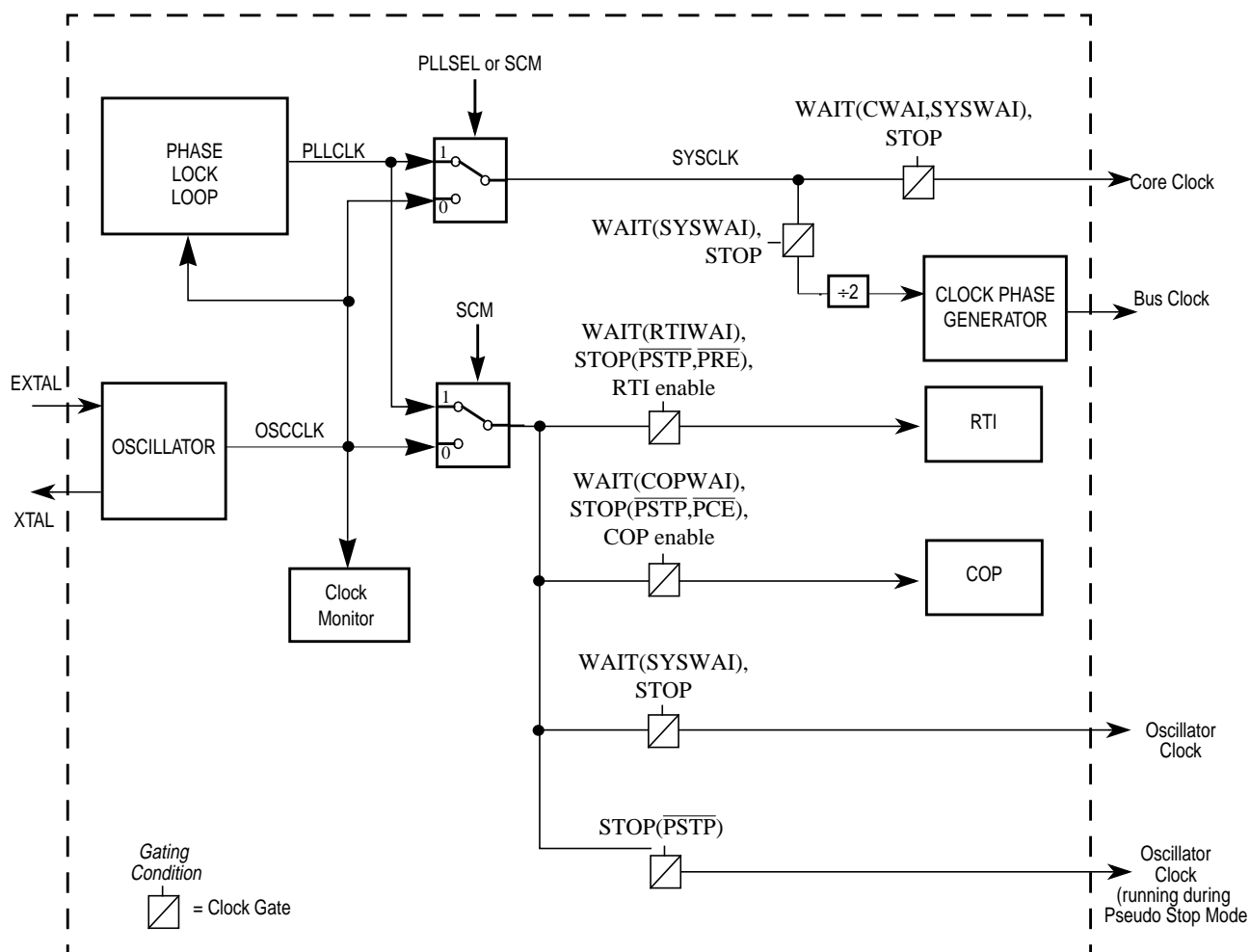


Figure 4-2 System Clocks Generator

The clock generator creates the clocks used in the MCU (see **Figure 4-2**). The gating condition placed on top of the individual clock gates indicates the dependencies of different modes (STOP, WAIT) and the setting of the respective configuration bits.

The peripheral modules use the Bus Clock. Some peripheral modules also use the Oscillator Clock. The memory blocks use the Bus Clock. If the MCU enters Self Clock Mode (see **4.2.2 Self Clock Mode**) Oscillator clock source is switched to PLLCLK running at its minimum frequency f_{SCM} . The Bus Clock is used to generate the clock visible at the ECLK pin. The Core Clock signal is the clock for the CPU. The Core Clock is twice the Bus Clock as shown in **Figure 4-3**. But note that a CPU cycle corresponds to one Bus Clock.

PLL clock mode is selected with PLLSEL bit in the CLKSEL register. When selected, the PLL output clock drives SYSCLK for the main system including the CPU and peripherals. The PLL cannot be turned off by clearing the PLLON bit, if the PLL clock is selected. When PLLSEL is changed, it takes a maximum

of 4 OSCCLK plus 4 PLLCLK cycles to make the transition. During the transition, all clocks freeze and CPU activity ceases.

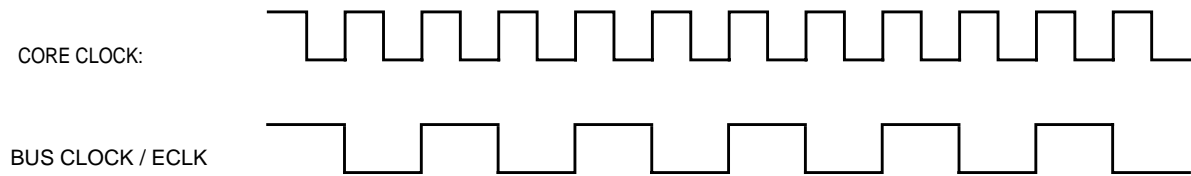


Figure 4-3 Core Clock and Bus Clock relationship

4.1.3 Clock Monitor (CM)

If no OSCCLK edges are detected within a certain time, the clock monitor within the oscillator block generates a clock monitor fail event. The CRG then asserts self clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated by the oscillator block. The clock monitor function is enabled/disabled by the CME control bit.

4.1.4 Clock Quality Checker

The clock monitor performs a coarse check on the incoming clock signal. The clock quality checker provides a more accurate check in addition to the clock monitor.

A clock quality check is triggered by any of the following events:

- Power on reset (*POR*)
- Low voltage reset (*LVR*)
- Wake-up from Full Stop Mode (*exit full stop*)
- Clock Monitor fail indication (*CM fail*)

A time window of 50000 VCO clock cycles¹ is called *check window*.

A number greater equal than 4096 rising OSCCLK edges within a *check window* is called *osc ok*. Note that *osc ok* immediately terminates the current *check window*. See **Figure 4-4** as an example.

NOTES:

1. VCO clock cycles are generated by the PLL when running at minimum frequency f_{SCM} .

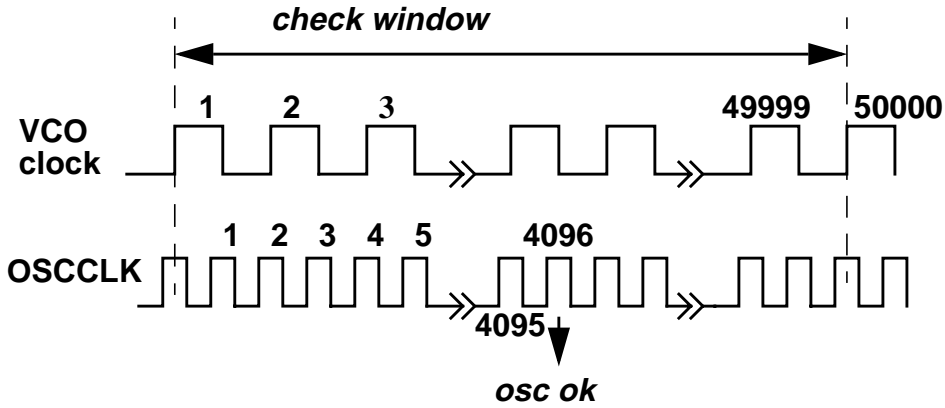


Figure 4-4 Check Window Example

The Sequence for clock quality check is shown in Figure 4-5.

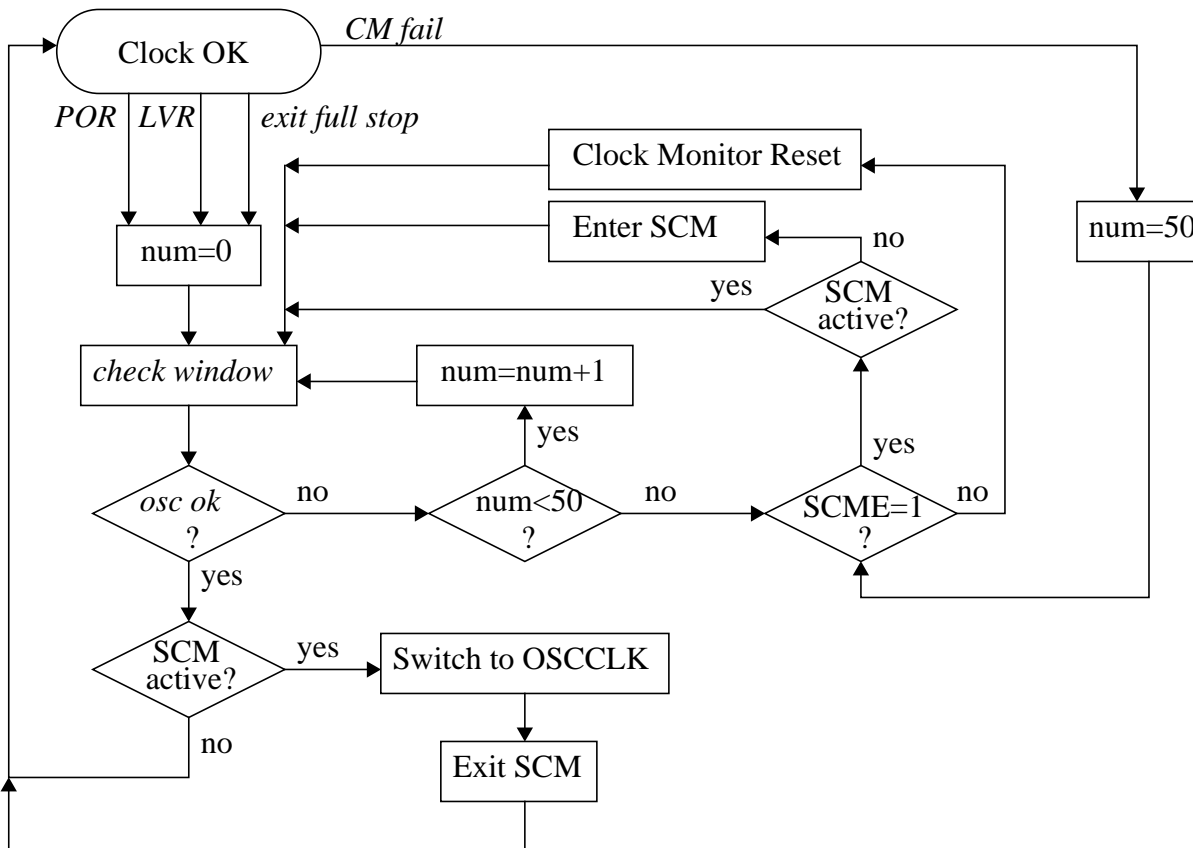


Figure 4-5 Sequence for Clock Quality Check

NOTE: Remember that in parallel to additional actions caused by Self Clock Mode or Clock Monitor Reset¹ handling the clock quality checker **continues** to check the OSCCLK signal.

NOTE: The Clock Quality Checker enables the PLL and the voltage regulator (VREG) anytime a clock check has to be performed. An ongoing clock quality check could also cause a running PLL (f_{SCM}) and an active VREG during Pseudo-Stop Mode or Wait Mode.

4.1.5 Computer Operating Properly Watchdog (COP)

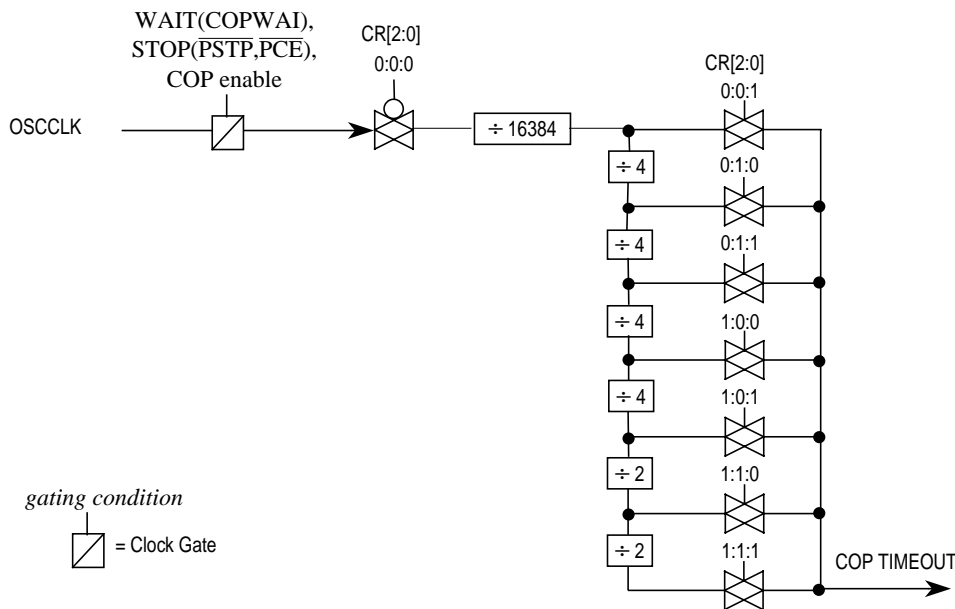


Figure 4-6 Clock Chain for COP

The COP (free running watchdog timer) enables the user to check that a program is running and sequencing properly. The COP is disabled out of reset. When the COP is being used, software is responsible for keeping the COP from timing out. If the COP times out it is an indication that the software is no longer being executed in the intended sequence; thus a system reset is initiated (see **5.2.2 Computer Operating Properly Watchdog (COP) Reset**). The COP runs with a gated OSCCLK (see **Figure 4-6 Clock Chain for COP**). Three control bits in the COPCTL register allow selection of seven COP time-out periods.

When COP is enabled, the program must write \$55 and \$AA (in this order) to the ARMCOP register during the selected time-out period. Once this is done, the COP time-out period is restarted. If the program

NOTES:

1. A Clock Monitor Reset will always set the SCME bit to logical '1'

fails to do this and the COP times out, the part will reset. Also, if any value other than \$55 or \$AA is written, the part is immediately reset.

Windowed COP operation is enabled by setting WCOP in the COPCTL register. In this mode, writes to the ARMCOP register to clear the COP timer must occur in the last 25% of the selected time-out period. A premature write will immediately reset the part.

If PCE bit is set, the COP will continue to run in Pseudo-Stop Mode.

4.1.6 Real Time Interrupt (RTI)

The RTI can be used to generate a hardware interrupt at a fixed periodic rate. If enabled (by setting RTIE=1), this interrupt will occur at the rate selected by the RTICTL register. The RTI runs with a gated OSCCLK (see **Figure 4-7 Clock Chain for RTI**). At the end of the RTI time-out period the RTIF flag is set to one and a new RTI time-out period starts immediately.

A write to the RTICTL register restarts the RTI time-out period.

If the PRE bit is set, the RTI will continue to run in Pseudo-Stop Mode.

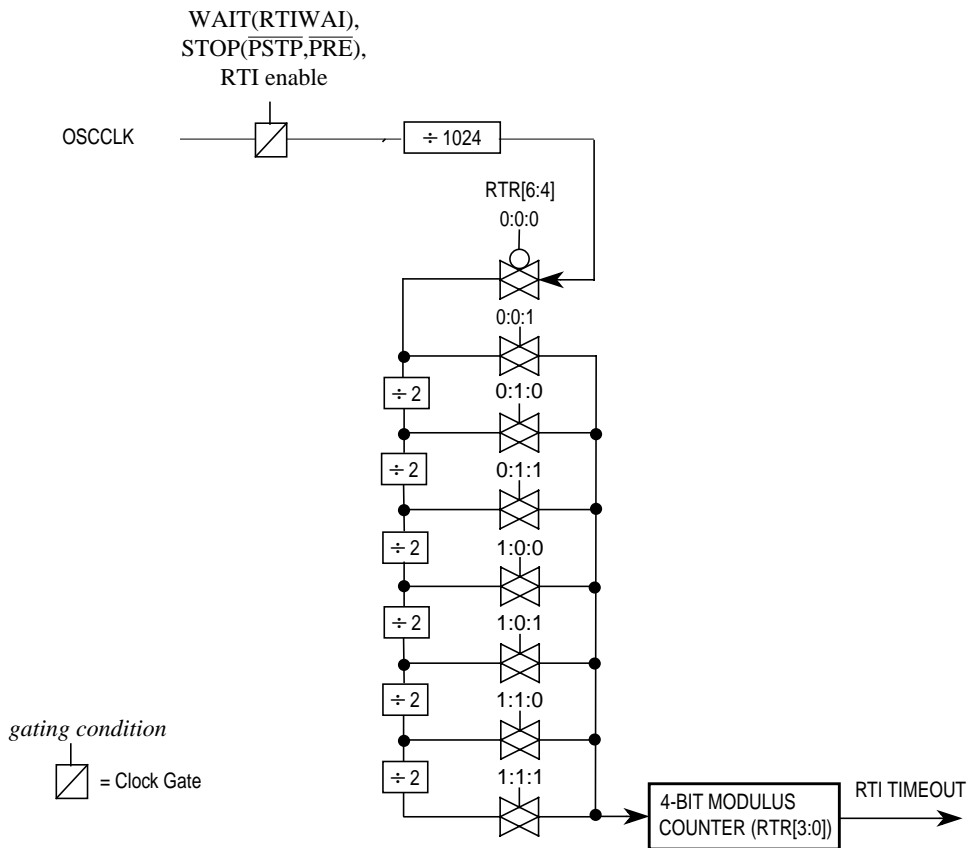


Figure 4-7 Clock Chain for RTI

4.2 Operation Modes

4.2.1 Normal Mode

The CRG block behaves as described within this specification in all normal modes.

4.2.2 Self Clock Mode

The VCO has a minimum operating frequency, f_{SCM} . If the external clock frequency is not available due to a failure or due to long crystal start-up time, the Bus Clock and the Core Clock are derived from the VCO running at minimum operating frequency; this mode of operation is called Self Clock Mode. This requires CME=1 and SCME=1. If the MCU was clocked by the PLL clock prior to entering Self Clock Mode, the PLLSEL bit will be cleared. If the external clock signal has stabilized again, the CRG will automatically select OSCCLK to be the system clock and return to normal mode. See **4.1.4 Clock Quality Checker** for more information on entering and leaving Self Clock Mode.

NOTE: *In order to detect a potential clock loss the CME bit should be always enabled (CME=1)!*

If CME bit is disabled and the MCU is configured to run on PLL clock (PLLCLK), a loss of external clock (OSCCLK) will not be detected and will cause the system clock to drift towards the VCO's minimum frequency f_{SCM} . As soon as the external clock is available again the system clock ramps up to its PLL target frequency. If the MCU is running on external clock any loss of clock will cause the system to go static.

4.3 Low Power Options

This section summarizes the low power options available in the CRG.

4.3.1 Run Mode

The RTI can be stopped by setting the associated rate select bits to zero.

The COP can be stopped by setting the associated rate select bits to zero.

4.3.2 Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode depending on setting of the individual bits in the CLKSEL register. All individual Wait Mode configuration bits can be superposed. This provides enhanced granularity in reducing the level of power consumption during Wait

Mode. **Table 4-1** lists the individual configuration bits and the parts of the MCU that are affected in Wait Mode.

Table 4-1 MCU configuration during Wait Mode

	PLLWAI	CWAI	SYSWAI	RTIWAI	COPWAI	ROAWAI
PLL	stopped	-	-	-	-	-
Core	-	stopped	stopped	-	-	-
System	-	-	stopped	-	-	-
RTI	-	-	-	stopped	-	-
COP	-	-	-	-	stopped	-
Oscillator	-	-	-	-	-	reduced ¹

NOTES:

1. Refer to oscillator block description for availability of a reduced oscillator amplitude!

After executing the WAI instruction the core requests the CRG to switch MCU into Wait Mode. The CRG then checks whether the PLLWAI, CWAI and SYSWAI bits are asserted (see **Figure 4-8 Wait Mode Entry/Exit Sequence**). Depending on the configuration the CRG switches the system and core clocks to OSCCLK by clearing the PLLSEL bit, disables the PLL, disables the core clocks and finally disables the remaining system clocks. As soon as all clocks are switched off Wait Mode is active.

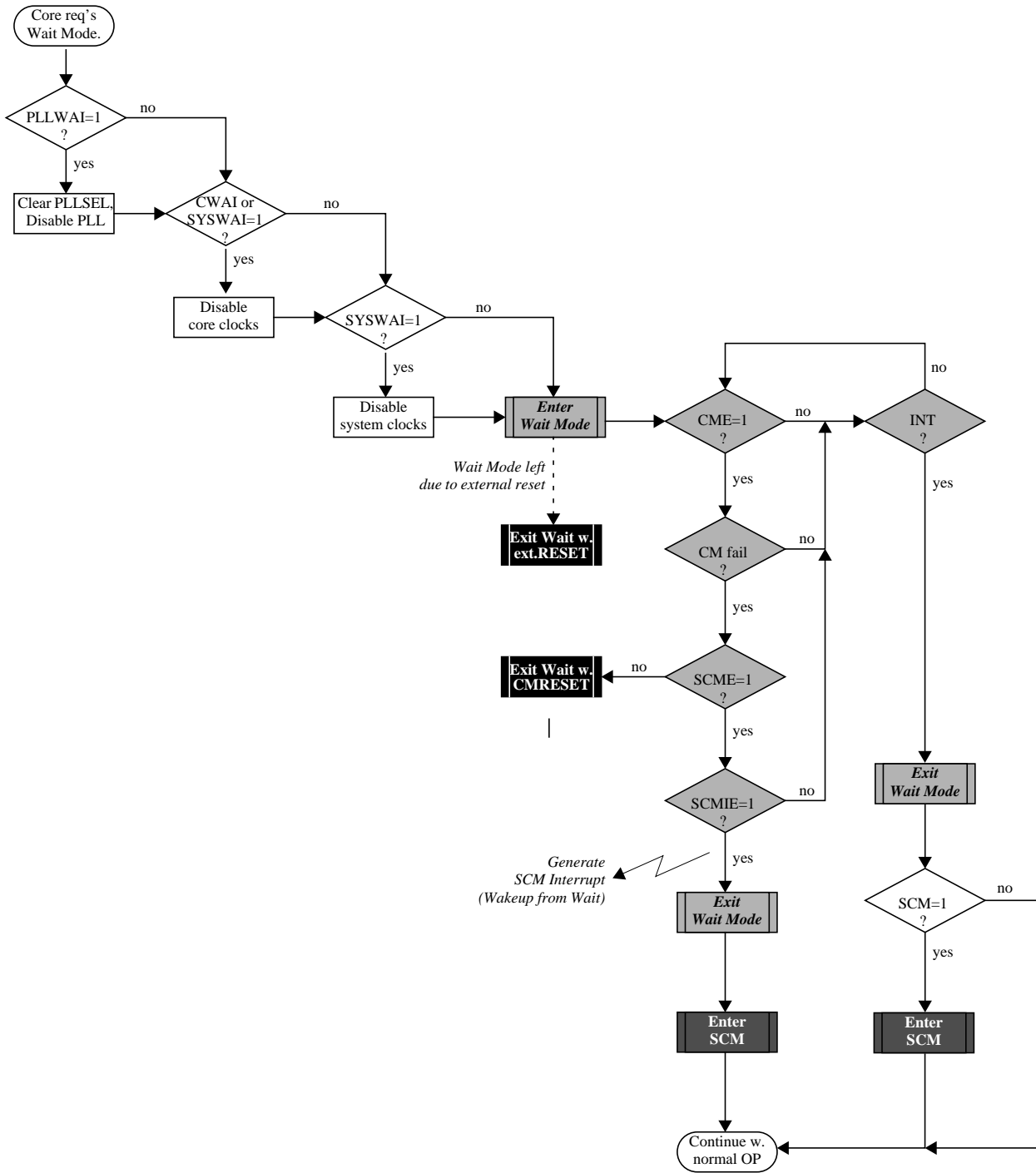


Figure 4-8 Wait Mode Entry/Exit Sequence

There are five different scenarios for the CRG to restart the MCU from Wait Mode:

- External Reset

- Clock Monitor Reset
- COP Reset
- Self Clock Mode Interrupt
- Real Time Interrupt (RTI)

If the MCU gets an external reset during Wait Mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Wait Mode is left and the MCU is in Run Mode again.

If the clock monitor is enabled (CME=1) the MCU is able to leave Wait-Mode when loss of oscillator/external clock is detected by a clock monitor fail. If the SCME bit is not asserted the CRG generates a clock monitor fail reset (CMRESET). The CRG’s behavior for CMRESET is the same compared to external reset, but another reset vector is fetched after completion of the reset sequence. If the SCME bit is asserted the CRG generates a SCM interrupt if enabled (SCMIE=1). After generating the interrupt the CRG enters Self-Clock Mode and starts the clock quality checker (see **4.1.4 Clock Quality Checker**). Then the MCU continues with normal operation. If the SCM interrupt is blocked by SCMIE=0, the SCMIF flag will be asserted and clock quality checks will be performed but the MCU will not wake-up from Wait-Mode.

If any other interrupt source (e.g. RTI) triggers exit from Wait Mode the MCU immediately continues with normal operation. If the PLL has been powered-down during Wait-Mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving Wait-Mode. The software must manually set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

If Wait Mode is entered from Self-Clock Mode the CRG will continue to check the clock quality until clock check is successful. The PLL and voltage regulator (VREG) will remain enabled.

Table 4-2 summarizes the outcome of a clock loss while in Wait Mode.

Table 4-2 Outcome of Clock Loss in Wait Mode

CME	SCME	SCMIE	CRG Actions
0	X	X	Clock failure --> No action, clock loss not detected.
1	0	X	Clock failure --> CRG performs Clock Monitor Reset immediately

Table 4-2 Outcome of Clock Loss in Wait Mode

CME	SCME	SCMIE	CRG Actions
1	1	0	<p>Clock failure --></p> <p>Scenario 1: OSCCLK recovers prior to exiting Wait Mode.</p> <ul style="list-style-type: none"> – MCU remains in Wait Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag. <p><i>Some time later OSCCLK recovers.</i></p> <ul style="list-style-type: none"> – CM no longer indicates a failure, – 4096 OSCCLK cycles later Clock Quality Check indicates clock o.k., – SCM deactivated, – PLL disabled depending on PLLWAI, – VREG remains enabled (<i>never gets disabled in Wait Mode</i>). – MCU remains in Wait Mode. <p><i>Some time later either a wakeup interrupt occurs (no SCM interrupt)</i></p> <ul style="list-style-type: none"> – Exit Wait Mode using OSCCLK as system clock (SYSCLK), – Continue normal operation. <p><i>or an External Reset is applied.</i></p> <ul style="list-style-type: none"> – Exit Wait Mode using OSCCLK as system clock, – Start reset sequence. <p>Scenario 2: OSCCLK does not recover prior to exiting Wait Mode.</p> <ul style="list-style-type: none"> – MCU remains in Wait Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag, – Keep performing Clock Quality Checks (could continue infinitely) while in Wait Mode. <p><i>Some time later either a wakeup interrupt occurs (no SCM interrupt)</i></p> <ul style="list-style-type: none"> – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again. <p><i>or an External RESET is applied.</i></p> <ul style="list-style-type: none"> – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Start reset sequence, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again.

Table 4-2 Outcome of Clock Loss in Wait Mode

CME	SCME	SCMIE	CRG Actions
1	1	1	<p>Clock failure --></p> <ul style="list-style-type: none"> – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – SCMIF set. <p><i>SCMIF generates Self Clock Mode wakeup interrupt.</i></p> <ul style="list-style-type: none"> – Exit Wait Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform a additional Clock Quality Checks until OSCCLK is o.k. again.

4.3.3 CPU Stop Mode

All clocks are stopped in STOP mode, dependent of the setting of the PCE, PRE and PSTP bit. The oscillator is disabled in STOP mode unless the PSTP bit is set. All counters and dividers remain frozen but do not initialize. If the PRE or PCE bits are set, the RTI or COP continues to run in Pseudo-Stop Mode. In addition to disabling system and core clocks the CRG requests other functional units of the MCU (e.g. voltage-regulator) to enter their individual powersaving modes (if available). This is the main difference between Pseudo-Stop Mode and Wait Mode.

After executing the STOP instruction the core requests the CRG to switch the MCU into Stop Mode. If the PLLSEL bit is still set when entering Stop-Mode, the CRG will switch the system and core clocks to OSCCLK by clearing the PLLSEL bit. Then the CRG disables the PLL, disables the core clock and finally disables the remaining system clocks. As soon as all clocks are switched off Stop-Mode is active.

If Pseudo-Stop Mode (PSTP=1) is entered from Self-Clock Mode the CRG will continue to check the clock quality until clock check is successful. The PLL and the voltage regulator (VREG) will remain enabled. If Full-Stop Mode (PSTP=0) is entered from Self-Clock Mode an ongoing clock quality check will be stopped. A complete timeout window check will be started when Stop Mode is left again.

Wake-up from Stop-Mode also depends on the setting of the PSTP bit.

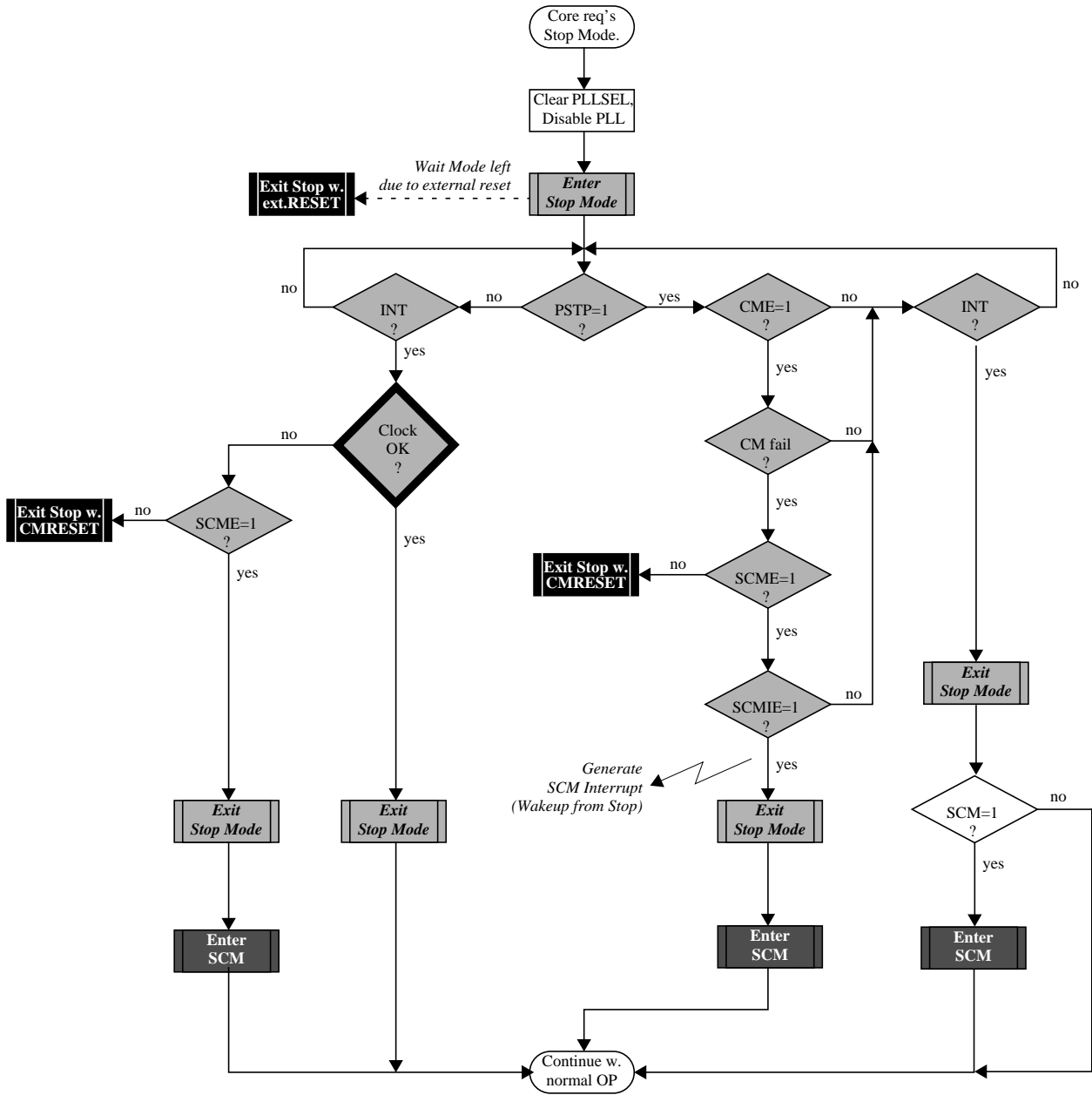


Figure 4-9 Stop Mode Entry/Exit Sequence

4.3.3.1 Wake-up from Pseudo-Stop (PSTD=1)

Wake-up from Pseudo-Stop is the same as wake-up from Wait-Mode. There are also three different scenarios for the CRG to restart the MCU from Pseudo-Stop Mode:

- External Reset
- Clock Monitor Fail
- Wake-up Interrupt

If the MCU gets an external reset during Pseudo-Stop Mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Pseudo-Stop Mode is left and the MCU is in Run Mode again.

If the clock monitor is enabled (CME=1) the MCU is able to leave Pseudo-Stop Mode when loss of oscillator/external clock is detected by a clock monitor fail. If the SCME bit is not asserted the CRG generates a clock monitor fail reset (CMRESET). The CRG’s behavior for CMRESET is the same compared to external reset, but another reset vector is fetched after completion of the reset sequence. If the SCME bit is asserted the CRG generates a SCM interrupt if enabled (SCMIE=1). After generating the interrupt the CRG enters Self-Clock Mode and starts the clock quality checker (see **4.1.4 Clock Quality Checker**). Then the MCU continues with normal operation. If the SCM interrupt is blocked by SCMIE=0, the SCMIF flag will be asserted but the CRG will not wake-up from Pseudo-Stop Mode.

If any other interrupt source (e.g. RTI) triggers exit from Pseudo-Stop Mode the MCU immediately continues with normal operation. Because the PLL has been powered-down during Stop-Mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving Stop-Mode. The software must set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

Table 4-3 summarizes the outcome of a clock loss while in Pseudo-Stop Mode.

Table 4-3 Outcome of Clock Loss in Pseudo-Stop Mode

CME	SCME	SCMIE	CRG Actions
0	X	X	Clock failure --> No action, clock loss not detected.
1	0	X	Clock failure --> CRG performs Clock Monitor Reset immediately

Table 4-3 Outcome of Clock Loss in Pseudo-Stop Mode

CME	SCME	SCMIE	CRG Actions
1	1	0	<p>Clock Monitor failure --></p> <p>Scenario 1: OSCCLK recovers prior to exiting Pseudo-Stop Mode.</p> <ul style="list-style-type: none"> – MCU remains in Pseudo-Stop Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag. <p><i>Some time later OSCCLK recovers.</i></p> <ul style="list-style-type: none"> – CM no longer indicates a failure, – 4096 OSCCLK cycles later Clock Quality Check indicates clock o.k., – SCM deactivated, – PLL disabled, – VREG disabled. – MCU remains in Pseudo-Stop Mode. <p><i>Some time later either a wakeup interrupt occurs (no SCM interrupt)</i></p> <ul style="list-style-type: none"> – Exit Pseudo-Stop Mode using OSCCLK as system clock (SYSCLK), – Continue normal operation. <p><i>or an External Reset is applied.</i></p> <ul style="list-style-type: none"> – Exit Pseudo-Stop Mode using OSCCLK as system clock, – Start reset sequence. <p>Scenario 2: OSCCLK does not recover prior to exiting Pseudo-Stop Mode.</p> <ul style="list-style-type: none"> – MCU remains in Pseudo-Stop Mode, – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – Set SCMIF interrupt flag, – Keep performing Clock Quality Checks (could continue infinitely) while in Pseudo-Stop Mode. <p><i>Some time later either a wakeup interrupt occurs (no SCM interrupt)</i></p> <ul style="list-style-type: none"> – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again. <p><i>or an External RESET is applied.</i></p> <ul style="list-style-type: none"> – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock – Start reset sequence, – Continue to perform additional Clock Quality Checks until OSCCLK is o.k. again.

Table 4-3 Outcome of Clock Loss in Pseudo-Stop Mode

CME	SCME	SCMIE	CRG Actions
1	1	1	<p>Clock failure --></p> <ul style="list-style-type: none"> – VREG enabled, – PLL enabled, – SCM activated, – Start Clock Quality Check, – SCMIF set. <p><i>SCMIF generates Self Clock Mode wakeup interrupt.</i></p> <ul style="list-style-type: none"> – Exit Pseudo-Stop Mode in SCM using PLL clock (f_{SCM}) as system clock, – Continue to perform a additional Clock Quality Checks until OSCCLK is o.k. again.

4.3.3.2 Wake-up from Full Stop (PSTP=0)

The MCU requires an external interrupt or an external reset in order to wake-up from Stop-Mode.

If the MCU gets an external reset during Full Stop Mode active, the CRG asynchronously restores all configuration bits in the register space to its default settings and will perform a maximum of 50 clock *check_windows*(see **4.1.4 Clock Quality Checker**). After completing the clock quality check the CRG starts the reset generator. After completing the reset sequence processing begins by fetching the normal reset vector. Full Stop-Mode is left and the MCU is in Run Mode again.

If the MCU is woken-up by an interrupt, the CRG will also perform a maximum of 50 clock *check_windows* (see **4.1.4 Clock Quality Checker**). If the clock quality check is successful, the CRG will release all system and core clocks and will continue with normal operation. If all clock checks within the Timeout-Window are failing, the CRG will switch to Self-Clock Mode or generate a clock monitor reset (CMRESET) depending on the setting of the SCME bit.

Because the PLL has been powered-down during Stop-Mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving Stop-Mode. The software must manually set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

NOTE: *In Full Stop Mode the clock monitor is disabled and any loss of clock will not be detected.*

Section 5 Resets

5.1 General

This section describes how to reset the CRG and how the CRG itself controls the reset of the MCU. It explains all special reset requirements. Since the reset generator for the MCU is part of the CRG this section also describes all automatic actions that occur during or as a result of individual reset conditions. The reset values of registers and signals are provided in **Section 3 Memory Map and Registers**. All reset sources are listed in **Table 5-1**. Refer to MCU specification for related vector addresses and priorities.

Table 5-1 Reset Summary

Reset Source	Local Enable
Power on Reset	None
Low Voltage Reset	None
External Reset	None
Clock Monitor Reset	PLLCTL (CME=1, SCME=0)
COP Watchdog Reset	COPCTL (CR[2:0] nonzero)

5.2 Description of Reset Operation

The reset sequence is initiated by any of the following events:

- Low level is detected at the $\overline{\text{RESET}}$ pin (External Reset).
- Power on is detected.
- Low voltage is detected.
- COP watchdog times out.
- Clock monitor failure is detected and Self-Clock Mode was disabled (SCME=0).

Upon detection of any reset event, an internal circuit drives the $\overline{\text{RESET}}$ pin low for 128 SYSCLK cycles (see **Figure 5-1**). Since entry into reset is asynchronous it does not require a running SYSCLK. However, the internal reset circuit of the CRG cannot sequence out of current reset condition without a running SYSCLK. The number of 128 SYSCLK cycles might be increased by $n=3$ to 6 additional SYSCLK cycles depending on the internal synchronization latency. After $128+n$ SYSCLK cycles the $\overline{\text{RESET}}$ pin is released. The reset generator of the CRG waits for additional 64 SYSCLK cycles and then samples the RESET pin to determine the originating source. **Table 5-2** shows which vector will be fetched.

Table 5-2 Reset Vector Selection

sampled $\overline{\text{RESET}}$ pin (64 cycles after release)	Clock Monitor Reset pending	COP Reset pending	Vector fetch
1	0	0	POR / LVR / External Reset
1	1	X	Clock Monitor Reset
1	0	1	COP Reset

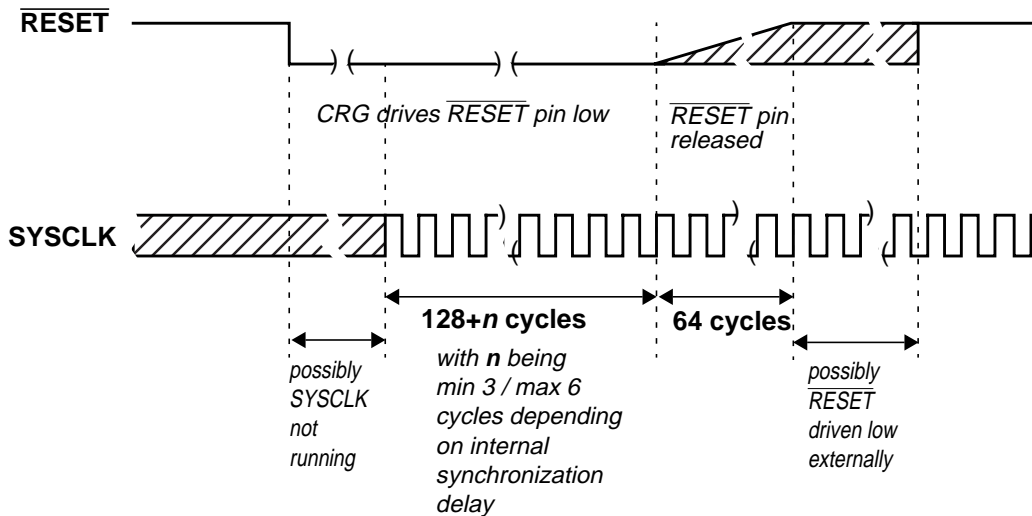
Table 5-2 Reset Vector Selection

sampled $\overline{\text{RESET}}$ pin (64 cycles after release)	Clock Monitor Reset pending	COP Reset pending	Vector fetch
0	X	X	POR / LVR / External Reset with rise of $\overline{\text{RESET}}$ pin

NOTE: External circuitry connected to the $\overline{\text{RESET}}$ pin should not include a large capacitance that would interfere with the ability of this signal to rise to a valid logic one within 64 SYSCLK cycles after the low drive is released.

The internal reset of the MCU remains asserted while the reset generator completes the 192 SYSCLK long reset sequence. The reset generator circuitry always makes sure the internal reset is deasserted synchronously after completion of the 192 SYSCLK cycles. In case the $\overline{\text{RESET}}$ pin is externally driven low for more than these 192 SYSCLK cycles (External Reset), the internal reset remains asserted too.

Figure 5-1 RESET Timing



5.2.1 Clock Monitor Reset

The CRG generates a Clock Monitor Reset in case all of the following conditions are true:

- Clock monitor is enabled (CME=1)
- Loss of clock is detected
- Self-Clock Mode is disabled (SCME=0).

The reset event asynchronously forces the configuration registers to their default settings (see **Section 3 Memory Map and Registers**). In detail the CME and the SCME are reset to logical ‘1’ (which doesn’t change the state of the CME bit, because it has already been set). As a consequence the CRG

immediately enters Self Clock Mode and starts its internal reset sequence. In parallel the clock quality check starts. As soon as clock quality check indicates a valid Oscillator Clock the CRG switches to OSCCLK and leaves Self Clock Mode. Since the clock quality checker is running in parallel to the reset generator, the CRG may leave Self Clock Mode while still completing the internal reset sequence. When the reset sequence is finished the CRG checks the internally latched state of the clock monitor fail circuit. If a clock monitor fail is indicated processing begins by fetching the Clock Monitor Reset vector.

5.2.2 Computer Operating Properly Watchdog (COP) Reset

When COP is enabled, the CRG expects sequential write of \$55 and \$AA (in this order) to the ARMCOP register during the selected time-out period. Once this is done, the COP time-out period restarts. If the program fails to do this the CRG will generate a reset. Also, if any value other than \$55 or \$AA is written, the CRG immediately generates a reset. In case windowed COP operation is enabled writes (\$55 or \$AA) to the ARMCOP register must occur in the last 25% of the selected time-out period. A premature write the CRG will immediately generate a reset.

As soon as the reset sequence is completed the reset generator checks the reset condition. If no clock monitor failure is indicated and the latched state of the COP timeout is true, processing begins by fetching the COP vector.

5.2.3 Power On Reset, Low Voltage Reset

The on-chip voltage regulator detects when VDD to the MCU has reached a certain level and asserts power on reset or low voltage reset or both. As soon as a power on reset or low voltage reset is triggered the CRG performs a quality check on the incoming clock signal. As soon as clock quality check indicates a valid Oscillator Clock signal the reset sequence starts using the Oscillator clock. If after 50 check windows the clock quality check indicated a non-valid Oscillator Clock the reset sequence starts using Self-Clock Mode.

Figure 5-2 and **Figure 5-3** show the power-up sequence for cases when the $\overline{\text{RESET}}$ pin is tied to VDD and when the $\overline{\text{RESET}}$ pin is held low.

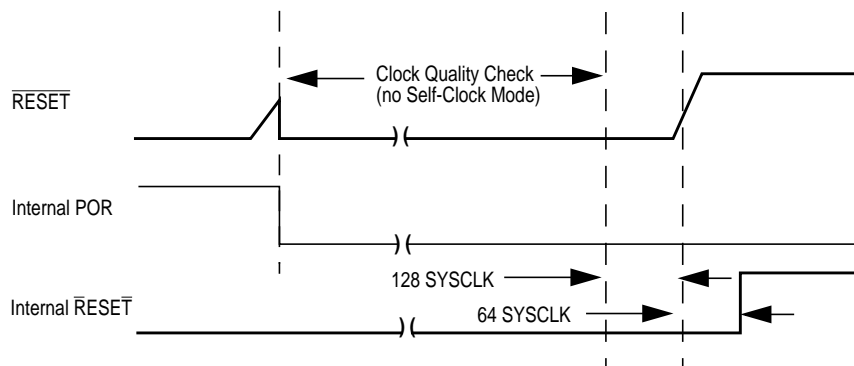


Figure 5-2 $\overline{\text{RESET}}$ pin tied to VDD (by a pull-up resistor)

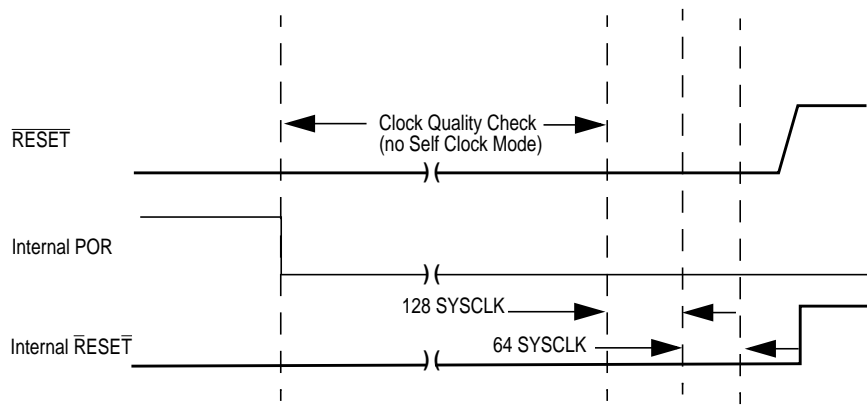


Figure 5-3 $\overline{\text{RESET}}$ pin held low externally

Section 6 Interrupts

6.1 General

The interrupts/reset vectors requested by the CRG are listed in **Table 6-1**. Refer to MCU specification for related vector addresses and priorities.

Table 6-1 CRG Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Real time interrupt	I bit	CRGINT (RTIE)
LOCK interrupt	I bit	CRGINT (LOCKIE)
SCM interrupt	I bit	CRGINT (SCMIE)

6.2 Description of Interrupt Operation

6.2.1 Real Time Interrupt

The CRG generates a real time interrupt when the selected interrupt time period elapses. RTI interrupts are locally disabled by setting the RTIE bit to zero. The real time interrupt flag (RTIF) is set to 1 when a timeout occurs, and is cleared to 0 by writing a 1 to the RTIF bit.

The RTI continues to run during Pseudo Stop Mode if the PRE bit is set to 1. This feature can be used for periodic wakeup from Pseudo Stop if the RTI interrupt is enabled.

6.2.2 PLL Lock Interrupt

The CRG generates a PLL Lock interrupt when the LOCK condition of the PLL has changed, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to zero. The PLL Lock interrupt flag (LOCKIF) is set to 1 when the LOCK condition has changed, and is cleared to 0 by writing a 1 to the LOCKIF bit.

6.2.3 Self Clock Mode Interrupt

The CRG generates a Self Clock Mode interrupt when the SCM condition of the system has changed, either entered or exited Self Clock Mode. SCM conditions can only change if the Self Clock Mode enable bit (SCME) is set to 1. SCM conditions are caused by a failing clock quality check after power on reset (POR) or low voltage reset (LVR) or recovery from Full Stop Mode (PSTP=0) or Clock Monitor failure. For details on the clock quality check refer to **4.1.4 Clock Quality Checker**. If the clock monitor is enabled (CME=1) a loss of external clock will also cause a SCM condition (SCME=1).

SCM interrupts are locally disabled by setting the SCMIE bit to zero. The SCM interrupt flag (SCMIF) is set to 1 when the SCM condition has changed, and is cleared to 0 by writing a 1 to the SCMIF bit.

**FINAL PAGE OF
54
PAGES**

ECT_16B8C

Block User Guide

V01.06

Original Release Date: 2-Sep-1999
Revised: 30-Apr-2010

FreeScale Semiconducotr Inc.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
0.1	2-Sep-99	2-Sep-99		Original draft. Distributed only within Freescale QS9000 Verified.
0.2	24-Sep-99	24-Sep-99		<ul style="list-style-type: none"> • Changed the specs as per MSRS format. • Modified ECT16b8c Block diagram. • Modified IP Bus signal names and their description. • Modified ECT output signal names. • Deleted bits 3-0 of TSCR1 register in Register Map(Sheet 1 of 2). • Modified register addresses in the description of TSFRZ, WAIT, NORMAL mode(Modes of Operation). • In Figure 1-6 changed text font to Halvetica. • Renamed TMSK1 and TMSK2 register as TIE and TSCR2 also renamed TSCR as TSCR1. • Modified TFLG2 bit setting sentence. • Added explanation about the abbreviation(M clock, PACLK) used. • Removed duplication of lines at the end of register description of PACN3/PACN2. • Corrected the reset value of MCCNT from \$FF to \$FFFF in the description of register MCCTL. • Corrected Table format for delay counter select and Modulus counter Prescalar Select. • Corrected all the cross-references used in section 3 of the document. • Deleted and added some module specific signals. • Changed all interrupts from active LOW to active HIGH. • Added description about successful output compare and forced output compare taking place simultaneously and their effect on flag. • Added abbreviation section. • In Fig 1-3 changed host data bus to IPbus.
0.3	25-Oct-99	25-Oct-99		<ul style="list-style-type: none"> • Changed block name in Section 2. • Removed signals ipp_ect_ic_ibe and ipp_ect_ic_offval from port list.
0.4	11-Nov-99	11-Nov-99		<ul style="list-style-type: none"> • Removed signal ipb_read & ipb_write from portlist. ipb_rwb added

Version Number	Revision Date	Effective Date	Author	Description of Changes
0.5	1-Dec-99	1-Dec-99		<ul style="list-style-type: none"> Incorporated feedback received from Joachim Kruecken on 30-Nov-99
01.00	10-May-01	11-May-01		<ul style="list-style-type: none"> formal changes for SRS compliance (cover, master pages, paragraph formats, register templates) updated cross-references marked TIMTST register description with 'non_cust' tag
01.01	19-July-01			<ul style="list-style-type: none"> Document names have been added Names and Variable definitions have been hidden
01.02	10-Jan-02	10-Jan-02		<ul style="list-style-type: none"> Note added in Section 3.3.18 for PACN1/PACN0
01.03	18-Jul-02	18-Jul-02		<ul style="list-style-type: none"> Changed the register bit information of MCCTL on bits ICLAT and FLMC
01.04	11-Nov-02	11-Nov-02	Solomon Chinnam	<ul style="list-style-type: none"> Eliminated all references of "n" to "x" .e.g. IOSn to IOSx
01.06	28-Apr-10	28-Apr-10	Ame Wang	<ul style="list-style-type: none"> in3.3.8 add Note2 and Table 3-3 in3.3.11,TCRE descripction part,add Note2 and Figure 3-12

Table of Contents

Section 1 Introduction

1.1	Overview	13
1.2	Features	13
1.3	Modes of Operation	13
1.4	Block Diagram	14

Section 2 Signal Description

2.1	Overview	15
2.2	Detailed Signal Descriptions	15
2.2.1	IOC7 - Input capture and Output compare channel 7	15
2.2.2	IOC6 - Input capture and Output compare channel 6	15
2.2.3	IOC5 - Input capture and Output compare channel 5	15
2.2.4	IOC4 - Input capture and Output compare channel 4	15
2.2.5	IOC3 - Input capture and Output compare channel 3	15
2.2.6	IOC2 - Input capture and Output compare channel 2	15
2.2.7	IOC1 - Input capture and Output compare channel 1	15
2.2.8	IOC0 - Input capture and Output compare channel 0	15

Section 3 Memory Map and Registers

3.1	Overview	17
3.2	Module Memory Map	17
3.3	Register Descriptions	19
3.3.1	TIOS — Timer Input Capture/Output Compare Select Register	19
3.3.2	CFORC — Timer Compare Force Register	20
3.3.3	OC7M — Output Compare 7 Mask Register	20
3.3.4	OC7D — Output Compare 7 Data Register	21
3.3.5	TCNT — Timer Count Register	21
3.3.6	TSCR1 — Timer System Control Register 1	22
3.3.7	TTOV — Timer Toggle On Overflow Register 1	23
3.3.8	TCTL1/TCTL2 — Timer Control Register 1/Timer Control Register 2	23
3.3.9	TCTL3/TCTL4 — Timer Control Register 3/Timer Control Register 4	25
3.3.10	TIE — Timer Interrupt Enable Register	25
3.3.11	TSCR2 — Timer System Control Register 2	26

3.3.12	TFLG1 — Main Timer Interrupt Flag 1	27
3.3.13	TFLG2 — Main Timer Interrupt Flag 2	28
3.3.14	Timer Input Capture/Output Compare Registers 0-7	29
3.3.15	PACTL — 16-Bit Pulse Accumulator A Control Register	30
3.3.16	PAFLG — Pulse Accumulator A Flag Register	32
3.3.17	PACN3, PACN2 — Pulse Accumulators Count Registers	33
3.3.18	PACN1, PACN0 — Pulse Accumulators Count Registers	34
3.3.19	MCCTL — 16-Bit Modulus Down-Counter Control Register	34
3.3.20	MCFLG — 16-Bit Modulus Down-Counter FLAG Register	36
3.3.21	ICPAR — Input Control Pulse Accumulators Register	37
3.3.22	DLYCT — Delay Counter Control Register	37
3.3.23	ICOVW — Input Control Overwrite Register	38
3.3.24	ICSYS — Input Control System Control Register	38
3.3.25	TIMTST — Timer Test Register	40
3.3.26	PBCTL — 16-Bit Pulse Accumulator B Control Register	40
3.3.27	PBFLG — Pulse Accumulator B Flag Register	41
3.3.28	PA3H–PA0H — 8-Bit Pulse Accumulators Holding Registers	42
3.3.29	MCCNT — Modulus Down-Counter Count Register	43
3.3.30	Timer Input Capture Holding Registers 0-3	44

Section 4 Functional Description

4.1	General	47
4.2	Enhanced Capture Timer Modes of Operation	52
4.2.1	IC Channels	52
4.2.1.1	Non-Buffered IC Channels	53
4.2.1.2	Buffered IC Channels	53
4.2.2	Pulse Accumulators	53
4.2.2.1	Pulse Accumulator latch mode	54
4.2.2.2	Pulse Accumulator queue mode	54
4.2.3	Modulus Down-Counter	54

Section 5 Reset

5.1	General	55
-----	-------------------	----

Section 6 Interrupts

6.1	General	57
-----	-------------------	----

6.2	Description of Interrupt Operation	57
6.2.1	Channel [7:0] Interrupt	57
6.2.2	Modulus Counter Interrupt	57
6.2.3	Pulse Accumulator B Overflow Interrupt)	57
6.2.4	Pulse Accumulator A Input Interrupt	57
6.2.5	Pulse Accumulator A Overflow Interrupt	58
6.2.6	Timer Overflow Interrupt	58

List of Figures

Figure 1-1	Timer Block Diagram	14
Figure 3-1	Timer Input Capture/Output Compare Register (TIOS)	19
Figure 3-2	Timer Compare Force Register (CFORC)	20
Figure 3-3	Output Compare 7 Mask Register (OC7M)	20
Figure 3-4	Output Compare 7 Data Register (OC7D)	21
Figure 3-5	Timer Count Register (TCNT)	21
Figure 3-6	Timer System Control Register 1 (TSCR1)	22
Figure 3-7	Timer Toggle On Overflow Register 1 (TTOV)	23
Figure 3-8	Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)23	
Figure 3-9	Timer Control Register 3/Timer Control Register 4 (TCTL3/TCTL4)25	
Figure 3-10	Timer Interrupt Enable Register (TIE)	25
Figure 3-11	Timer System Control Register 2 (TSCR2)	26
Figure 3-12	The TCNT cycle diagram under TCRE=1 condition	27
Figure 3-13	Main Timer Interrupt Flag 1 (TFLG1)	27
Figure 3-14	Main Timer Interrupt Flag 2 (TFLG2)	28
Figure 3-15	Timer Input Capture/Output Compare Registers 0-7	30
Figure 3-16	16-Bit Pulse Accumulator Control Register (PACTL)	30
Figure 3-17	Pulse Accumulator A Flag Register (PAFLG)	32
Figure 3-18	Pulse Accumulators Count Register 3 (PACN3)	33
Figure 3-19	Pulse Accumulators Count Register 2 (PACN2)	33
Figure 3-20	Pulse Accumulators Count Register 1 (PACN1)	34
Figure 3-21	Pulse Accumulators Count Register 0 (PACN0)	34
Figure 3-22	16-Bit Modulus Down-Counter Control Register (MCCTL)	34
Figure 3-23	16-Bit Modulus Down-Counter FLAG Register (MCFLG)	36
Figure 3-24	Input Control Pulse Accumulators Register (ICPAR)	37
Figure 3-25	Delay Counter Control Register (DLYCT)	37
Figure 3-26	Input Control Overwrite Register (ICOVW)	38
Figure 3-27	Input Control System Register (ICSYS)	38
Figure 3-28	Timer Test Register (TIMTST)	40
Figure 3-29	16-Bit Pulse Accumulator B Control Register (PBCTL)	40
Figure 3-30	Pulse Accumulator B Flag Register (PBFLG)	41
Figure 3-31	8-Bit Pulse Accumulators Holding Register 3 (PA3H)	42
Figure 3-32	8-Bit Pulse Accumulators Holding Register 2 (PA2H)	42

Figure 3-33	8-Bit Pulse Accumulators Holding Register 1 (PA1H)	42
Figure 3-34	8-Bit Pulse Accumulators Holding Register 0 (PA0H)	42
Figure 3-35	Modulus Down-Counter Count Register (MCCNT)	43
Figure 3-36	Timer Input Capture Holding Register 0 (TC0H)	44
Figure 3-37	Timer Input Capture Holding Register 1 (TC1H)	44
Figure 3-38	Timer Input Capture Holding Register 2 (TC2H)	44
Figure 3-39	Timer Input Capture Holding Register 3 (TC3H)	44
Figure 4-1	Detailed Timer Block Diagram in Latch mode	48
Figure 4-2	Detailed Timer Block Diagram in Queue mode	49
Figure 4-3	8-Bit Pulse Accumulators Block Diagram	50
Figure 4-4	16-Bit Pulse Accumulators Block Diagram	51
Figure 4-5	Block Diagram for Port7 with Output compare/Pulse Accumulator A52	

Table 3-1	Module Memory Map	17
Table 3-2	Compare Result Output Action	24
Table 3-3	The OC7 and OCx event priority	24
Table 3-4	Edge Detector Circuit Configuration	25
Table 3-5	Prescaler Selection	27
Table 3-6	Pin Action	31
Table 3-7	Clock Selection	31
Table 3-8	Modulus Counter Prescaler Select	36
Table 3-9	Delay Counter Select	38
Table 6-1	ECT Interrupts	57

Section 1 Introduction

1.1 Overview

The HCS12 Enhanced Capture Timer module has the features of the HCS12 Standard Timer module enhanced by additional features in order to enlarge the field of applications, in particular for automotive ABS applications.

This design specification describes the standard timer as well as the additional features.

The basic timer consists of a 16-bit, software-programmable counter driven by a prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from microseconds to many seconds.

A full access for the counter registers or the input capture/output compare registers should take place in one clock cycle. Accessing high byte and low byte separately for all of these registers may not yield the same result as accessing them in one word.

1.2 Features

- 16-Bit Buffer Register for four Input Capture (IC) channels.
- Four 8-Bit Pulse Accumulators with 8-bit buffer registers associated with the four buffered IC channels. Configurable also as two 16-Bit Pulse Accumulators.
- 16-Bit Modulus Down-Counter with 4-bit Prescaler.
- Four user selectable Delay Counters for input noise immunity increase.

1.3 Modes of Operation

STOP: Timer and modulus counter are off since clocks are stopped.

FREEZE: Timer and modulus counter keep on running, unless TSFRZ in TSCR(\$06) is set to one.

WAIT: Counters keep on running, unless TSWAI in TSCR (\$06) is set to one.

NORMAL: Timer and modulus counter keep on running, unless TEN in TSCR(\$06) respectively MCEN in MCCTL (\$26) are cleared.

1.4 Block Diagram

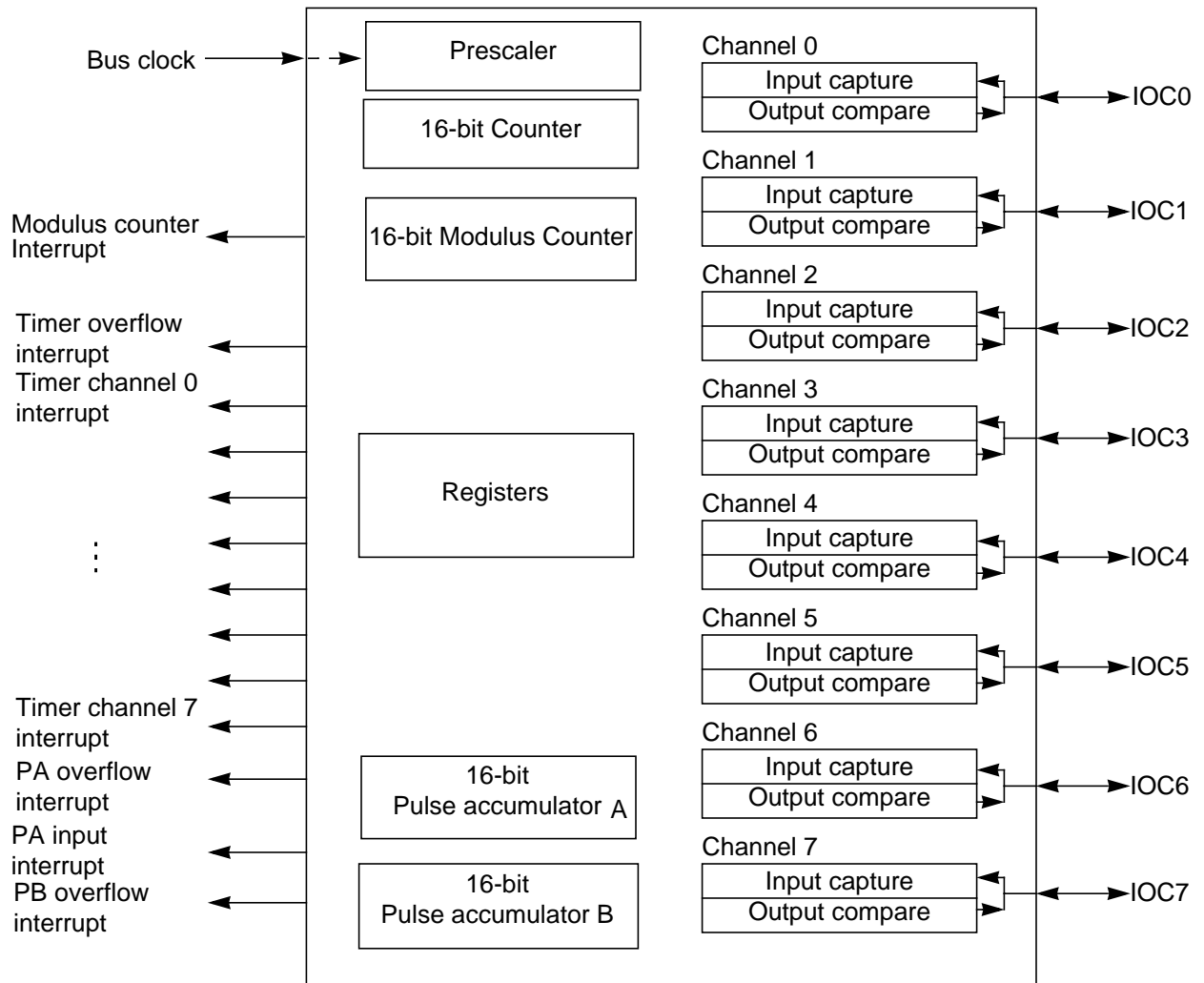


Figure 1-1 Timer Block Diagram

Section 2 Signal Description

2.1 Overview

The ECT_16B8C module has a total 8 external pins.

2.2 Detailed Signal Descriptions

2.2.1 IOC7 - Input capture and Output compare channel 7

This pin serves as input capture or output compare for channel 7.

2.2.2 IOC6 - Input capture and Output compare channel 6

This pin serves as input capture or output compare for channel 6.

2.2.3 IOC5 - Input capture and Output compare channel 5

This pin serves as input capture or output compare for channel 7.

2.2.4 IOC4 - Input capture and Output compare channel 4

This pin serves as input capture or output compare for channel 4.

2.2.5 IOC3 - Input capture and Output compare channel 3

This pin serves as input capture or output compare for channel 3.

2.2.6 IOC2 - Input capture and Output compare channel 2

This pin serves as input capture or output compare for channel 2.

2.2.7 IOC1 - Input capture and Output compare channel 1

This pin serves as input capture or output compare for channel 1.

2.2.8 IOC0 - Input capture and Output compare channel 0

This pin serves as input capture or output compare for channel 0.

NOTE: *For the description of interrupts see Section 6 Interrupts.*

Section 3 Memory Map and Registers

3.1 Overview

This section provides a detailed description of all memory and registers.

3.2 Module Memory Map

The memory map for the ECT module is given below in **Table 3-1**. The Address listed for each register is the address offset. The total address for each register is the sum of the base address for the ECT module and the address offset for each register.

Table 3-1 Module Memory Map

Offset	Use	Access
\$_00	Timer Input Capture/Output Compare Select (TIOS)	Read/Write
\$_01	Timer Compare Force Register (CFORC)	Read/Write ¹
\$_02	Output Compare 7 Mask Register (OC7M)	Read/Write
\$_03	Output Compare 7 Data Register (OC7D)	Read/Write
\$_04	Timer Count Register High (TCNT)	Read/Write ²
\$_05	Timer Count Register Low (TCNT)	Read/Write ²
\$_06	Timer System Control Register1 (TSCR1)	Read/Write
\$_07	Timer Toggle Overflow Register (TTOV)	Read/Write
\$_08	Timer Control Register1 (TCTL1)	Read/Write
\$_09	Timer Control Register2 (TCTL2)	Read/Write
\$_0A	Timer Control Register3 (TCTL3)	Read/Write
\$_0B	Timer Control Register4 (TCTL4)	Read/Write
\$_0C	Timer Interrupt Enable Register (TIE)	Read/Write
\$_0D	Timer System Control Register2 (TSCR2)	Read/Write
\$_0E	Main Timer Interrupt Flag1 (TFLG1)	Read/Write
\$_0F	Main Timer Interrupt Flag2 (TFLG2)	Read/Write
\$_10	Timer Input Capture/Output Compare Register0 High (TC0)	Read/Write ³
\$_11	Timer Input Capture/Output Compare Register0 Low (TC0)	Read/Write ³
\$_12	Timer Input Capture/Output Compare Register1 High (TC1)	Read/Write ³
\$_13	Timer Input Capture/Output Compare Register1 Low (TC1)	Read/Write ³
\$_14	Timer Input Capture/Output Compare Register2 High (TC2)	Read/Write ³
\$_15	Timer Input Capture/Output Compare Register2 Low (TC2)	Read/Write ³
\$_16	Timer Input Capture/Output Compare Register3 High (TC3)	Read/Write ³

Table 3-1 Module Memory Map

\$_17	Timer Input Capture/Output Compare Register3 Low (TC3)	Read/Write ³
\$_18	Timer Input Capture/Output Compare Register4 High (TC4)	Read/Write ³
\$_19	Timer Input Capture/Output Compare Register4 Low (TC4)	Read/Write ³
\$_1A	Timer Input Capture/Output Compare Register5 High (TC5)	Read/Write ³
\$_1B	Timer Input Capture/Output Compare Register5 Low (TC5)	Read/Write ³
\$_1C	Timer Input Capture/Output Compare Register6 High (TC6)	Read/Write ³
\$_1D	Timer Input Capture/Output Compare Register6 Low (TC6)	Read/Write ³
\$_1E	Timer Input Capture/Output Compare Register7 High (TC7)	Read/Write ³
\$_1F	Timer Input Capture/Output Compare Register7 Low (TC7)	Read/Write ³
\$_20	16-Bit Pulse Accumulator A Control Register (PACTL)	Read/Write
\$_21	Pulse Accumulator A Flag Register (PAFLG)	Read/Write
\$_22	Pulse Accumulator Count Register3 (PACN3)	Read/Write
\$_23	Pulse Accumulator Count Register2 (PACN2)	Read/Write
\$_24	Pulse Accumulator Count Register1 (PACN1)	Read/Write
\$_25	Pulse Accumulator Count Register0 (PACN0)	Read/Write
\$_26	16-Bit Modulus Down Counter Register (MCCTL)	Read/Write
\$_27	16-Bit Modulus Down Counter Flag Register (MCFLG)	Read/Write
\$_28	Input Control Pulse Accumulator Register (ICPAR)	Read/Write
\$_29	Delay Counter Control Register (DLYCT)	Read/Write
\$_2A	Input Control Overwrite Register (ICOVW)	Read/Write
\$_2B	Input Control System Control Register (ICSYS)	Read/Write ⁴
\$_2C	Reserved	--
\$_2D	Timer Test Register (TIMTST)	Read/Write ²
\$_2E	Reserved	--
\$_2F	Reserved	--
\$_30	16-Bit Pulse Accumulator B Control Register (PBCTL)	Read/Write
\$_31	16-Bit Pulse Accumulator B Flag Register (PBFLG)	Read/Write
\$_32	8-Bit Pulse Accumulator Holding Register3 (PA3H)	Read/Write ⁵
\$_33	8-Bit Pulse Accumulator Holding Register2 (PA2H)	Read/Write ⁵
\$_34	8-Bit Pulse Accumulator Holding Register1 (PA1H)	Read/Write ⁵
\$_35	8-Bit Pulse Accumulator Holding Register0 (PA0H)	Read/Write ⁵
\$_36	Modulus Down-Counter Count Register High (MCCNT)	Read/Write

Table 3-1 Module Memory Map

\$_37	Modulus Down-Counter Count Register Low (MCCNT)	Read/Write
\$_38	Timer Input Capture Holding Register0 High (TC0H)	Read/Write ⁵
\$_39	Timer Input Capture Holding Register0 Low (TC0L)	Read/Write ⁵
\$_3A	Timer Input Capture Holding Register1 High (TC1H)	Read/Write ⁵
\$_3B	Timer Input Capture Holding Register1 Low (TC1L)	Read/Write ⁵
\$_3C	Timer Input Capture Holding Register2 High (TC2H)	Read/Write ⁵
\$_3D	Timer Input Capture Holding Register2 Low (TC2L)	Read/Write ⁵
\$_3E	Timer Input Capture Holding Register3 High (TC3H)	Read/Write ⁵
\$_3F	Timer Input Capture Holding Register3 Low (TC3L)	Read/Write ⁵

1. Always read \$00.
2. Only writable in special modes (test_mode = 1).
3. Write to these registers have no meaning or effect during input capture.
4. May be written once (test_mode = 0) but writes are always permitted when test_mode = 0
5. Write has no effect.

3.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

3.3.1 TIOS — Timer Input Capture/Output Compare Select Register

Register offset: \$_00

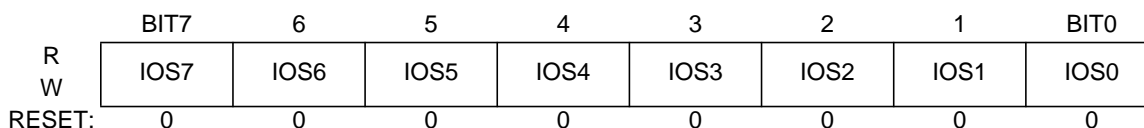


Figure 3-1 Timer Input Capture/Output Compare Register (TIOS)

Read or write anytime.

IOS[7:0] — Input Capture or Output Compare Channel Configuration

0 = The corresponding channel acts as an input capture

1 = The corresponding channel acts as an output compare.

3.3.2 CFORC — Timer Compare Force Register

Register offset: \$_01

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	0	0	0	0
W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
RESET:	0	0	0	0	0	0	0	0

Figure 3-2 Timer Compare Force Register (CFORC)

Read anytime but will always return \$00 (1 state is transient). Write anytime.

FOC[7:0] — Force Output Compare Action for Channel 7-0

A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set.

NOTE: *A successful channel 7 output compare overrides any channel 6:0 compares. If forced output compare on any channel occurs at the same time as the successful output compare then forced output compare action will take precedence and interrupt flag won't get set.*

3.3.3 OC7M — Output Compare 7 Mask Register

Register offset: \$_02

	BIT7	6	5	4	3	2	1	BIT0
R								
W	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
RESET:	0	0	0	0	0	0	0	0

Figure 3-3 Output Compare 7 Mask Register (OC7M)

Read or write anytime.

Setting the OC7Mx (x ranges from 0 to 6) will set the corresponding port to be an output port when the corresponding TIOSx (x ranges from 0 to 6) bit is set to be an output compare.

NOTE: *A successful channel 7 output compare overrides any channel 6:0 compares. For each OC7M bit that is set, the output compare action reflects the corresponding OC7D bit.*

3.3.4 OC7D — Output Compare 7 Data Register

Register offset: $\$_03$

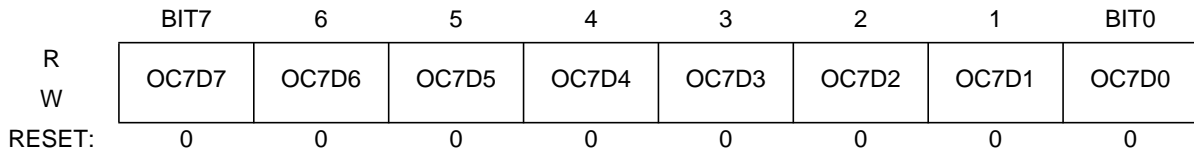


Figure 3-4 Output Compare 7 Data Register (OC7D)

Read or write anytime.

A channel 7 output compare can cause bits in the output compare 7 data register to transfer to the timer port data register depending on the output compare 7 mask register.

3.3.5 TCNT — Timer Count Register

Register offset: $\$_04$ - $\$_05$

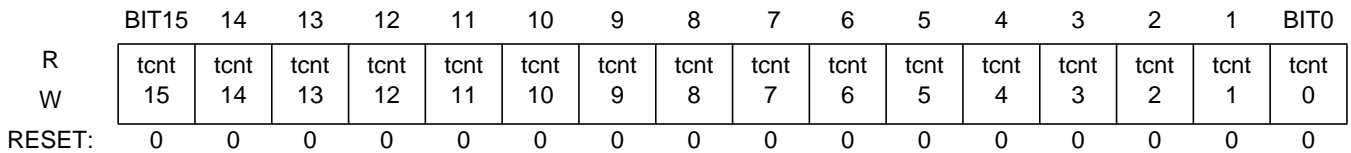


Figure 3-5 Timer Count Register (TCNT)

The 16-bit main timer is an up counter.

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

Read anytime.

Write has no meaning or effect in the normal mode; only writable in special modes ($test_mode = 1$).

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

3.3.6 TSCR1 — Timer System Control Register 1

Register offset: \$_06

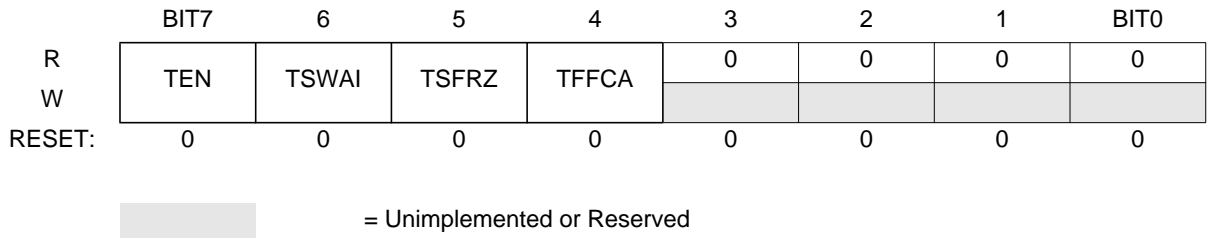


Figure 3-6 Timer System Control Register 1 (TSCR1)

Read or write anytime.

TEN — Timer Enable

- 0 = Disables the main timer, including the counter. Can be used for reducing power consumption.
- 1 = Allows the timer to function normally.

If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator since the ÷64 is generated by the timer prescaler.

TSWAI — Timer Module Stops While in Wait

- 0 = Allows the timer module to continue running during wait.
- 1 = Disables the timer module when the MCU is in the wait mode. Timer interrupts cannot be used to get the MCU out of wait.

TSWAI also affects pulse accumulators and modulus down counters.

TSFRZ — Timer and Modulus Counter Stop While in Freeze Mode

- 0 = Allows the timer and modulus counter to continue running while in freeze mode.
- 1 = Disables the timer and modulus counter whenever the MCU is in freeze mode. This is useful for emulation.

TSFRZ does not stop the pulse accumulator.

TFFCA — Timer Fast Flag Clear All

- 0 = Allows the timer flag clearing to function normally.
- 1 = For TFLG1 (\$0E), a read from an input capture or a write to the output compare channel (\$10–\$1F) causes the corresponding channel flag, CnF, to be cleared. For TFLG2 (\$0F), any access to the TCNT register (\$04, \$05) clears the TOF flag. Any access to the PACN3 and PACN2 registers (\$22, \$23) clears the PAOVF and PAIF flags in the PAFLG register (\$21). Any access to the PACN1 and PACN0 registers (\$24, \$25) clears the PBOVF flag in the PBFLG register (\$31). This has the advantage of eliminating software overhead in a separate clear sequence. Extra care is required to avoid accidental flag clearing due to unintended accesses.

3.3.7 TTOV — Timer Toggle On Overflow Register 1

Register offset: $\$_07$

	BIT7	6	5	4	3	2	1	BIT0
R	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
W								
RESET:	0	0	0	0	0	0	0	0

Figure 3-7 Timer Toggle On Overflow Register 1 (TTOV)

Read or write anytime.

TOV_x — Toggle On Overflow Bits

TOV_x toggles output compare pin on overflow. This feature only takes effect when in output compare mode. When set, it takes precedence over forced output compare but not channel 7 override events.

0 = Toggle output compare pin on overflow feature disabled

1 = Toggle output compare pin on overflow feature enabled

3.3.8 TCTL1/TCTL2 — Timer Control Register 1/Timer Control Register 2

Register offset: $\$_08$

	BIT7	6	5	4	3	2	1	BIT0
R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
W								
RESET	0	0	0	0	0	0	0	0

Register offset: $\$_09$

	BIT7	6	5	4	3	2	1	BIT0
R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
W								
RESET	0	0	0	0	0	0	0	0

Figure 3-8 Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)

Read or write anytime.

OM_x — Output Mode

OL_x — Output Level

These eight pairs of control bits are encoded to specify the output action to be taken as a result of a successful OC_x compare. When either OM_x or OL_x is one, the pin associated with OC_x becomes an output tied to OC_x.

NOTE: To enable output action by *OMx* and *OLx* bits on timer port, the corresponding bit in *OC7M* should be cleared.

Table 3-2 Compare Result Output Action

OMx	OLx	Action
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

Note1: To operate the 16-bit pulse accumulators A and B (PACA and PACB) independently of input capture or output compare 7 and 0 respectively the user must set the corresponding bits $IOSx = 1$, $OMx = 0$ and $OLx = 0$.

Note2: To enable output action using the *OM7* and *OL7* bits on the timer port, the corresponding bit *OC7M7* in the *OC7M* register must also be cleared. The settings for these bits can be seen in **Table 3-3**

Table 3-3 The OC7 and OCx event priority

OC7M7=0				OC7M7=1			
OC7Mx=1		OC7Mx=0		OC7Mx=1		OC7Mx=0	
TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx
IOCx=OC7Dx IOC7=OM7/O L7	IOCx=OC7Dx +OMx/OLx IOC7=OM7/O L7	IOCx=OMx/OLx IOC7=OM7/OL7		IOCx=OC7Dx IOC7=OC7D7	IOCx=OC7Dx +OMx/OLx IOC7=OC7D7	IOCx=OMx/OLx IOC7=OC7D7	

Note: in **Table 3-3** the *IOS7* and *IOSx* should be set to 1

IOSx is the register *TIOS* bit *x*,

OC7Mx is the register *OC7M* bit *x*,

TCx is timer Input Capture/Output Compare register,

IOCx is channel *x*,

OMx/OLx is the register *TCTL1/TCTL2*,

OC7Dx is the register *OC7D* bit *x*.

$IOCx = OC7Dx + OMx/OLx$, means that both *OC7* event and *OCx* event will change channel *x* value.

3.3.9 TCTL3/TCTL4 — Timer Control Register 3/Timer Control Register 4

Register offset: $\$_{0A}$

	BIT7	6	5	4	3	2	1	BIT0
R	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
W								
RESET:	0	0	0	0	0	0	0	0

Register offset: $\$_{0B}$

	BIT7	6	5	4	3	2	1	BIT0
R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
W								
RESET:	0	0	0	0	0	0	0	0

Figure 3-9 Timer Control Register 3/Timer Control Register 4 (TCTL3/TCTL4)

Read or write anytime.

EDGxB, EDGxA — Input Capture Edge Control

These eight pairs of control bits configure the input capture edge detector circuits.

The four pairs of control bits of TCTL4 also configure the 8 bit pulse accumulators PAC0 - 3.

For 16 - bit pulse accumulator PACB, EDGE0B & EDGE0A, control bits of TCTL4 will decide the active edge.

Table 3-4 Edge Detector Circuit Configuration

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

3.3.10 TIE — Timer Interrupt Enable Register

Register offset: $\$_{0C}$

	BIT7	6	5	4	3	2	1	BIT0
R	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
W								
RESET:	0	0	0	0	0	0	0	0

Figure 3-10 Timer Interrupt Enable Register (TIE)

Read or write anytime.

The bits in TIE correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause an interrupt.

C7I–C0I — Input Capture/Output Compare “x” Interrupt Enable

3.3.11 TSCR2 — Timer System Control Register 2

Register offset: \$_0D

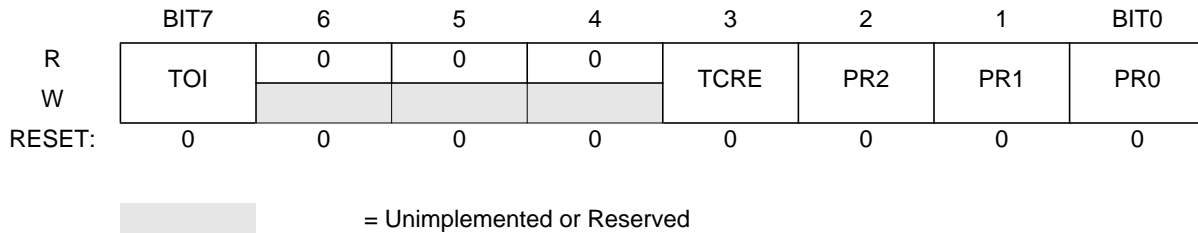


Figure 3-11 Timer System Control Register 2 (TSCR2)

Read or write anytime.

TOI — Timer Overflow Interrupt Enable

0 = Interrupt inhibited

1 = Hardware interrupt requested when TOF flag set

TCRE — Timer Counter Reset Enable

This bit allows the timer counter to be reset by a successful output compare 7 event. This mode of operation is similar to an up-counting modulus counter.

0 = Counter reset inhibited and counter free runs

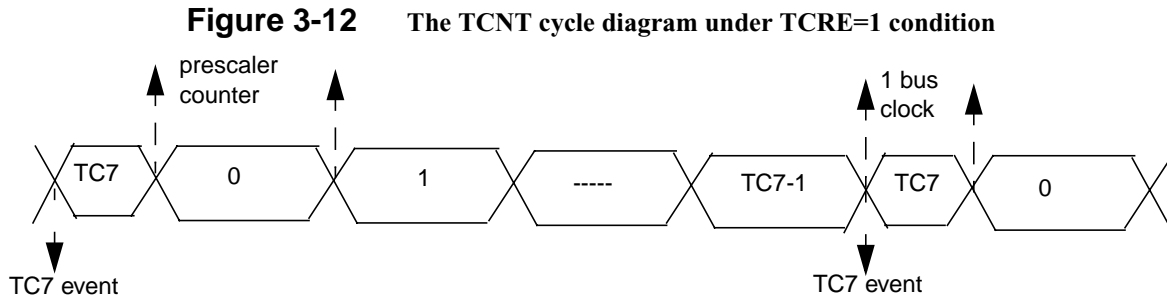
1 = Counter reset by a successful output compare 7

Note1: If TC7 = \$0000 and TCRE = 1, TCNT will stay at \$0000 continuously. If TC7 = \$FFFF and TCRE = 1, TOF will never be set when TCNT is reset from \$FFFF to \$0000.

Note2: TCRE=1 and TC7!=0, the TCNT cycle period will be TC7 x "prescaler counter width" + "1 Bus Clock".

When TCRE is set and TC7 is not equal to 0, TCNT will cycle from 0 to TC7. When TCNT reaches TC7 value, it will last only one bus cycle then reset to 0. for a more detail explanation please refer to **Figure 3-12**

Note: in **Figure 3-12**, if PR[2:0] is equal to 0, one prescaler counter equal to one bus clock



PR2, PR1, PR0 — Timer Prescaler Select

These three bits specify the number of ÷2 stages that are to be inserted between the bus clock and the main timer counter.

Table 3-5 Prescaler Selection

PR2	PR1	PR0	Prescale Factor
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

3.3.12 TFLG1 — Main Timer Interrupt Flag 1

Register offset: \$_0E

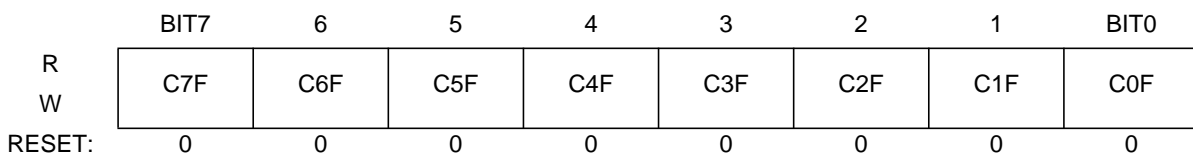


Figure 3-13 Main Timer Interrupt Flag 1 (TFLG1)

TFLG1 indicates when interrupt conditions have occurred. To clear a bit in the flag register, write a one to the bit.

Use of the TFMOD bit in the ICSYS register (\$2B) in conjunction with the use of the ICOVW register (\$2A) allows a timer interrupt to be generated after capturing two values in the capture and holding registers instead of generating an interrupt for every capture.

Read anytime. Write used in the clearing mechanism (set bits cause corresponding bits to be cleared). Writing a zero will not affect current status of the bit.

When TFFCA bit in TSCR register is set, a read from an input capture or a write into an output compare channel (\$10–\$1F) will cause the corresponding channel flag CxF to be cleared.

C7F–C0F — Input Capture/Output Compare Channel “x” Flag.

C0F can also be set by 16 - bit Pulse Accumulator B (PACB). C3F - C0F can also be set by 8 - bit pulse accumulators PAC3 - PAC0.

3.3.13 TFLG2 — Main Timer Interrupt Flag 2

Register offset: \$_0F

	BIT7	6	5	4	3	2	1	BIT0
R	TOF	0	0	0	0	0	0	0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-14 Main Timer Interrupt Flag 2 (TFLG2)

TFLG2 indicates when interrupt conditions have occurred. To clear a bit in the flag register, write the bit to one.

Read anytime. Write used in clearing mechanism (set bits cause corresponding bits to be cleared).

Any access to TCNT will clear TFLG2 register if the TFFCA bit in TSCR register is set.

TOF — Timer Overflow Flag

Set when 16-bit free-running timer overflows from \$FFFF to \$0000. This bit is cleared automatically by a write to the TFLG2 register with bit 7 set. (See also TCRE control bit explanation.)

3.3.14 Timer Input Capture/Output Compare Registers 0-7

TC0 — Timer Input Capture/Output Compare Register 0 **Register offset: \$ _10–\$ _11**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0	tc0
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC1 — Timer Input Capture/Output Compare Register 1 **Register offset: \$ _12–\$ _13**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1	tc1
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC2 — Timer Input Capture/Output Compare Register 2 **Register offset: \$ _14–\$ _15**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2	tc2
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC3 — Timer Input Capture/Output Compare Register 3 **Register offset: \$ _16–\$ _17**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3	tc3
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC4 — Timer Input Capture/Output Compare Register 4 **Register offset: \$ _18–\$ _19**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4	tc4
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC5 — Timer Input Capture/Output Compare Register 5 **Register offset: \$ _1A–\$ _1B**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5	tc5
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC6 — Timer Input Capture/Output Compare Register 6 **Register offset: \$ _1C–\$ _1D**

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6	tc6
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TC7 — Timer Input Capture/Output Compare Register 7 Register offset: \$_1E–\$_1F

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7	tc7
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 3-15 Timer Input Capture/Output Compare Registers 0-7

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

Read anytime. Write anytime for output compare function. Writes to these registers have no meaning or effect during input capture. All timer input capture/output compare registers are reset to \$0000.

3.3.15 PACTL — 16-Bit Pulse Accumulator A Control Register

Register offset: \$_20

	BIT7	6	5	4	3	2	1	BIT0
R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
W								
RESET:	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 3-16 16-Bit Pulse Accumulator Control Register (PACTL)

16-Bit Pulse Accumulator A (PACA) is formed by cascading the 8-bit pulse accumulators PAC3 and PAC2.

When PAEN is set, the PACA is enabled. The PACA shares the input pin with IC7.

Read: any time

Write: any time

PAEN — Pulse Accumulator A System Enable

0 = 16-Bit Pulse Accumulator A system disabled. 8-bit PAC3 and PAC2 can be enabled when their related enable bits in ICPAR (\$28) are set.

Pulse Accumulator Input Edge Flag (PAIF) function is disabled.

1 = 16-Bit Pulse Accumulator A system enabled. The two 8-bit pulse accumulators PAC3 and PAC2 are cascaded to form the PACA 16-bit pulse accumulator. When PACA is enabled, the PACN3 and PACN2 registers contents are respectively the high and low byte of the PACA.

PA3EN and PA2EN control bits in ICPAR (\$28) have no effect.

Pulse Accumulator Input Edge Flag (PAIF) function is enabled.

PAEN is independent from TEN. With timer disabled, the pulse accumulator can still function unless pulse accumulator is disabled.

PAMOD — Pulse Accumulator Mode

This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1).

- 0 = event counter mode
- 1 = gated time accumulation mode

PEDGE — Pulse Accumulator Edge Control

This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1).

For PAMOD bit = 0 (event counter mode).

- 0 = falling edges on PT7 pin cause the count to be incremented
- 1 = rising edges on PT7 pin cause the count to be incremented

For PAMOD bit = 1 (gated time accumulation mode).

- 0 = PT7 input pin high enables bus clock divided by 64 to Pulse Accumulator and the trailing falling edge on PT7 sets the PAIF flag.
- 1 = PT7 input pin low enables bus clock divided by 64 to Pulse Accumulator and the trailing rising edge on PT7 sets the PAIF flag

Table 3-6 Pin Action

PAMOD	PEDGE	Pin Action
0	0	Falling edge
0	1	Rising edge
1	0	Div. by 64 clock enabled with pin high level
1	1	Div. by 64 clock enabled with pin low level

If the timer is not active (TEN = 0 in TSCR), there is no divide-by-64 since the ÷64 clock is generated by the timer prescaler.

CLK1, CLK0 — Clock Select Bits

Table 3-7 Clock Selection

CLK1	CLK0	Clock Source
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

For the description of PACLK please refer **Figure 4-4**.

If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written.

PAOVI — Pulse Accumulator A Overflow Interrupt enable

- 0 = interrupt inhibited
- 1 = interrupt requested if PAOVF is set

PAI — Pulse Accumulator Input Interrupt enable

- 0 = interrupt inhibited
- 1 = interrupt requested if PAIF is set

3.3.16 PAFLG — Pulse Accumulator A Flag Register

Register offset: \$_21

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	0	0	PAOVF	PAIF
W								
RESET:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-17 Pulse Accumulator A Flag Register (PAFLG)

Read or write anytime. When the TFFCA bit in the TSCR register is set, any access to the PACNT register will clear all the flags in the PAFLG register.

PAOVF — Pulse Accumulator A Overflow Flag

Set when the 16-bit pulse accumulator A overflows from \$FFFF to \$0000, or when 8-bit pulse accumulator 3 (PAC3) overflows from \$FF to \$00.

When PACMX = 1, PAOVF bit can also be set if 8-bit pulse accumulator 3 (PAC3) reaches \$FF followed by an active edge on PT3.

This bit is cleared automatically by a write to the PAFLG register with bit 1 set.

PAIF — Pulse Accumulator Input edge Flag

Set when the selected edge is detected at the PT7 input pin. In event mode the event edge triggers PAIF and in gated time accumulation mode the trailing edge of the gate signal at the PT7 input pin triggers PAIF.

This bit is cleared by a write to the PAFLG register with bit 0 set.

Any access to the PACN3, PACN2 registers will clear all the flags in this register when TFFCA bit in register TSCR(\$06) is set.

3.3.17 PACN3, PACN2 — Pulse Accumulators Count Registers

Register offset: \$_22

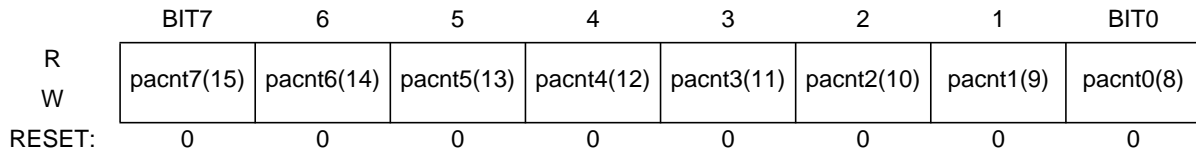


Figure 3-18 Pulse Accumulators Count Register 3 (PACN3)

Register offset: \$_23

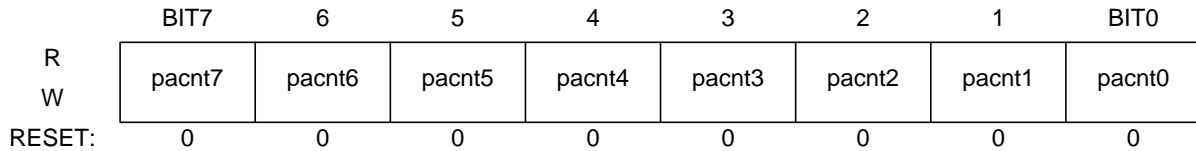


Figure 3-19 Pulse Accumulators Count Register 2 (PACN2)

Read or write any time.

The two 8-bit pulse accumulators PAC3 and PAC2 are cascaded to form the PACA 16-bit pulse accumulator. When PACA is enabled (PAEN=1 in PACTL, \$20) the PACN3 and PACN2 registers contents are respectively the high and low byte of the PACA.

When PACN3 overflows from \$FF to \$00, the Interrupt flag PAOVF in PAFLG (\$21) is set.

Full count register access should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

NOTE : When clocking pulse and write to the registers occurs simultaneously, write takes priority and the register is not incremented.

3.3.18 PACN1, PACN0 — Pulse Accumulators Count Registers

Register offset: \$_24

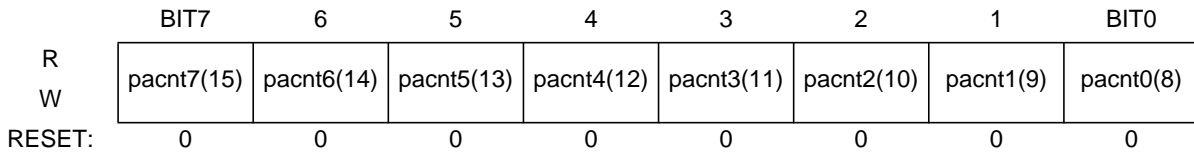


Figure 3-20 Pulse Accumulators Count Register 1 (PACN1)

Register offset: \$_25

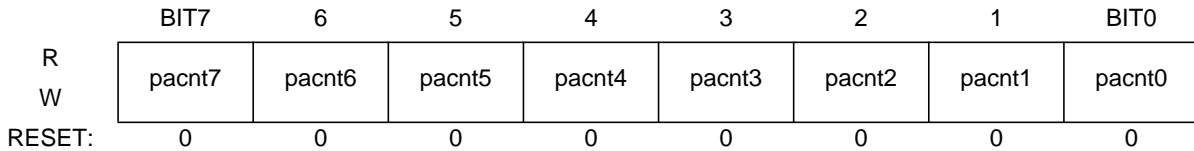


Figure 3-21 Pulse Accumulators Count Register 0 (PACN0)

Read or write any time.

The two 8-bit pulse accumulators PAC1 and PAC0 are cascaded to form the PACB 16-bit pulse accumulator. When PACB is enabled, (PBEN=1 in PBCTL, \$30) the PACN1 and PACN0 registers contents are respectively the high and low byte of the PACB.

When PACN1 overflows from \$FF to \$00, the Interrupt flag PBOVF in PBFLG (\$31) is set.

Full count register access should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word

NOTE : When clocking pulse and write to the registers occurs simultaneously , write takes priority and the register is not incremented.

3.3.19 MCCTL — 16-Bit Modulus Down-Counter Control Register

Register offset: \$_26

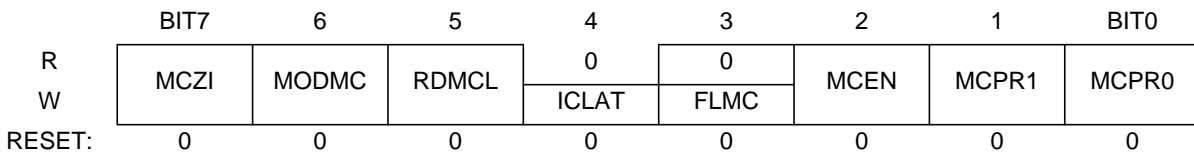


Figure 3-22 16-Bit Modulus Down-Counter Control Register (MCCTL)

Read or write any time.

MCZI — Modulus Counter Underflow Interrupt Enable

0 = Modulus counter interrupt is disabled.

1 = Modulus counter interrupt is enabled.

MODMC — Modulus Mode Enable

0 = The counter counts once from the value written to it and will stop at \$0000.

1 = Modulus mode is enabled. When the counter reaches \$0000, the counter is loaded with the latest value written to the modulus count register.

NOTE: *For proper operation, the MCEN bit should be cleared before modifying the MODMC bit in order to reset the modulus counter to \$FFFF.*

RDMCL — Read Modulus Down-Counter Load

0 = Reads of the modulus count register will return the present value of the count register.

1 = Reads of the modulus count register will return the contents of the load register.

ICLAT — Input Capture Force Latch Action

When input capture latch mode is enabled (LATQ and BUFEN bit in ICSYS (\$2B) are set, a write one to this bit immediately forces the contents of the input capture registers TC0 to TC3 and their corresponding 8-bit pulse accumulators to be latched into the associated holding registers. The pulse accumulators will be automatically cleared when the latch action occurs.

Writing zero to this bit has no effect. Read of this bit will return always zero.

FLMC — Force Load Register into the Modulus Counter Count Register

This bit is active only when the modulus down-counter is enabled (MCEN=1).

A write one into this bit loads the load register into the modulus counter count register. This also resets the modulus counter prescaler.

Write zero to this bit has no effect.

When MODMC=0, counter starts counting and stops at \$0000.

Read of this bit will return always zero.

MCEN — Modulus Down-Counter Enable

0 = Modulus counter disabled.

1 = Modulus counter is enabled.

When MCEN=0, the counter is preset to \$FFFF. This will prevent an early interrupt flag when the modulus down-counter is enabled.

MCPR1, MCPR0 — Modulus Counter Prescaler select

These two bits specify the division rate of the modulus counter prescaler.

The newly selected prescaler division rate will not be effective until a load of the load register into the modulus counter count register occurs.

Table 3-8 Modulus Counter Prescaler Select

M CPR1	M CPR0	Prescaler division rate
0	0	1
0	1	4
1	0	8
1	1	16

3.3.20 MCFLG — 16-Bit Modulus Down-Counter FLAG Register

Register offset: \$_27

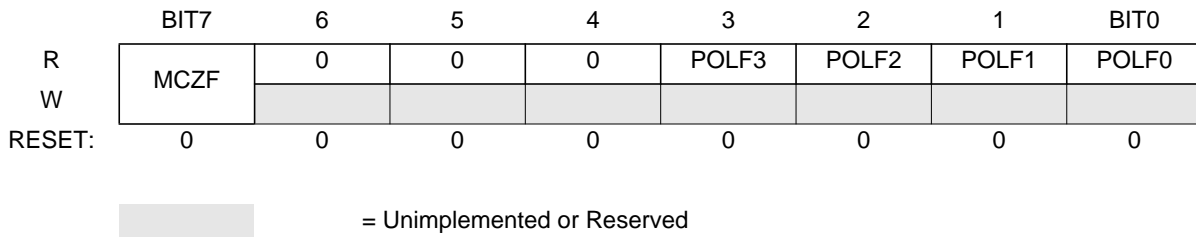


Figure 3-23 16-Bit Modulus Down-Counter FLAG Register (MCFLG)

Read: any time

Write: Only for clearing bit 7

MCZF — Modulus Counter Underflow Flag

The flag is set when the modulus down-counter reaches \$0000.

A write one to this bit clears the flag. Write zero has no effect.

Any access to the MCCNT register will clear the MCZF flag in this register when TFFCA bit in register TSCR(\$06) is set.

POLF3 – POLF0 — First Input Capture Polarity Status

This are read only bits. Write to these bits has no effect.

Each status bit gives the polarity of the first edge which has caused an input capture to occur after capture latch has been read.

Each POLFx corresponds to a timer PORTx input.

0 = The first input capture has been caused by a falling edge.

1 = The first input capture has been caused by a rising edge.

3.3.21 ICPAR — Input Control Pulse Accumulators Register

Register offset: \$_28

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	PA3EN	PA2EN	PA1EN	PA0EN
W								
RESET:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-24 Input Control Pulse Accumulators Register (ICPAR)

The 8-bit pulse accumulators PAC3 and PAC2 can be enabled only if PAEN in PATCL (\$20) is cleared. If PAEN is set, PA3EN and PA2EN have no effect.

The 8-bit pulse accumulators PAC1 and PAC0 can be enabled only if PBEN in PBTCL (\$30) is cleared. If PBEN is set, PA1EN and PA0EN have no effect.

Read or write any time.

PAXEN — 8-Bit Pulse Accumulator ‘x’ Enable

0 = 8-Bit Pulse Accumulator is disabled.

1 = 8-Bit Pulse Accumulator is enabled.

3.3.22 DLYCT — Delay Counter Control Register

Register offset: \$_29

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	0	0	DLY1	DLY0
W								
RESET:	0	0	0	0	0	0	0	0

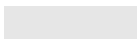
 = Unimplemented or Reserved

Figure 3-25 Delay Counter Control Register (DLYCT)

Read or write any time.

If enabled, after detection of a valid edge on input capture pin, the delay counter counts the pre-selected number of bus clock cycles, then it will generate a pulse on its output. The pulse is generated only if the level of input signal, after the preset delay, is the opposite of the level before the transition. This will avoid reaction to narrow input pulses.

After counting, the counter will be cleared automatically.

Delay between two active edges of the input signal period should be longer than the selected counter delay.

Table 3-9 Delay Counter Select

DLY1	DLY0	Delay
0	0	Disabled (bypassed)
0	1	256 bus clock cycles
1	0	512 bus clock cycles
1	1	1024 bus clock cycles

3.3.23 ICOVW — Input Control Overwrite Register

Register offset: \$_2A

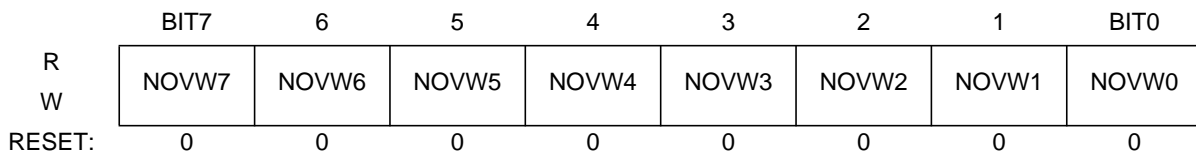


Figure 3-26 Input Control Overwrite Register (ICOVW)

Read or write any time.

An IC register is empty when it has been read or latched into the holding register.

A holding register is empty when it has been read.

NOVW_x — No Input Capture Overwrite

- 0 = The contents of the related capture register or holding register can be overwritten when a new input capture or latch occurs.
- 1 = The related capture register or holding register cannot be written by an event unless they are empty (see **4.2.1 IC Channels**). This will prevent the captured value to be overwritten until it is read or latched in the holding register.

3.3.24 ICSYS — Input Control System Control Register

Register offset: \$_2B

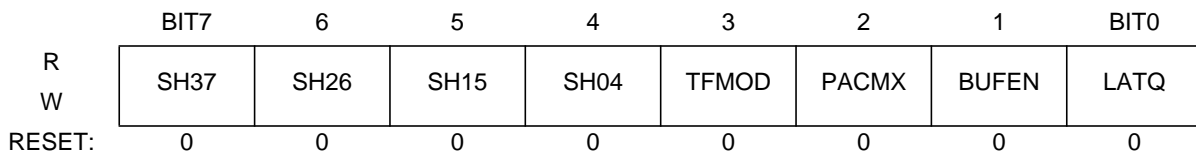


Figure 3-27 Input Control System Register (ICSYS)

Read: any time

Write: Can be written once (test_mode =0). Writes are always permitted when test_mode =1.

SH_{xy} — Share Input action of Input Capture Channels x and y

0 = Normal operation

1 = The channel input 'x' causes the same action on the channel 'y'. The port pin 'x' and the corresponding edge detector is used to be active on the channel 'y'.

TFMOD — Timer Flag-setting Mode

Use of the TFMOD bit in the ICSYS register (\$2B) in conjunction with the use of the ICOVW register (\$2A) allows a timer interrupt to be generated after capturing two values in the capture and holding registers instead of generating an interrupt for every capture.

By setting TFMOD in queue mode, when NOVW bit is set and the corresponding capture and holding registers are emptied, an input capture event will first update the related input capture register with the main timer contents. At the next event the TCx data is transferred to the TCxH register, The TCx is updated and the CxF interrupt flag is set.

In all other input capture cases the interrupt flag is set by a valid external event on PTx.

0 = The timer flags C3F–C0F in TFLG1 (\$0E) are set when a valid input capture transition on the corresponding port pin occurs.

1 = If in queue mode (BUFEN=1 and LATQ=0), the timer flags C3F–C0F in TFLG1 (\$0E) are set only when a latch on the corresponding holding register occurs.

If the queue mode is not engaged, the timer flags C3F–C0F are set the same way as for TFMOD=0.

PACMX — 8-Bit Pulse Accumulators Maximum Count

0 = Normal operation. When the 8-bit pulse accumulator has reached the value \$FF, with the next active edge, it will be incremented to \$00.

1 = When the 8-bit pulse accumulator has reached the value \$FF, it will not be incremented further. The value \$FF indicates a count of 255 or more.

BUFEN — IC Buffer Enable

0 = Input Capture and pulse accumulator holding registers are disabled.

1 = Input Capture and pulse accumulator holding registers are enabled. The latching mode is defined by LATQ control bit.

Write one into ICLAT bit in MCCTL (\$26), when LATQ is set will produce latching of input capture and pulse accumulators registers into their holding registers.

LATQ — Input Control Latch or Queue Mode Enable

The BUFEN control bit should be set in order to enable the IC and pulse accumulators holding registers. Otherwise LATQ latching modes are disabled.

Write one into ICLAT bit in MCCTL (\$26), when LATQ and BUFEN are set will produce latching of input capture and pulse accumulators registers into their holding registers.

0 = Queue Mode of Input Capture is enabled.

The main timer value is memorized in the IC register by a valid input pin transition.

With a new occurrence of a capture, the value of the IC register will be transferred to its holding register and the IC register memorizes the new timer value.

1 = Latch Mode is enabled. Latching function occurs when modulus down-counter reaches zero or a zero is written into the count register MCCNT (see **4.2.1.2 Buffered IC Channels**).

With a latching event the contents of IC registers and 8-bit pulse accumulators are transferred to their holding registers. 8-bit pulse accumulators are cleared.

3.3.25 TIMTST — Timer Test Register

Register offset: \$_2D

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	0	0	TCBYP	0
W								
RESET:	0	0	0	0	0	0	0	0

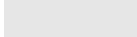
 = Unimplemented or Reserved

Figure 3-28 Timer Test Register (TIMTST)

Read: any time

Write: only in special mode (test_mode = 1).

TCBYP — Main Timer Divider Chain Bypass

0 = Normal operation

1 = For testing only. The 16-bit free-running timer counter is divided into two 8-bit halves and the prescaler is bypassed. The clock drives both halves directly.

When the high byte of timer counter TCNT (\$04) overflows from \$FF to \$00, the TOF flag in TFLG2 (\$0F) will be set.

3.3.26 PBCTL — 16-Bit Pulse Accumulator B Control Register

Register offset: \$_30

	BIT7	6	5	4	3	2	1	BIT0
R	0	PBEN	0	0	0	0	PBOVI	0
W								
RESET:	0	0	0	0	0	0	0	0

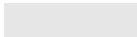
 = Unimplemented or Reserved

Figure 3-29 16-Bit Pulse Accumulator B Control Register (PBCTL)

Read or write any time.

16-Bit Pulse Accumulator B (PACB) is formed by cascading the 8-bit pulse accumulators PAC1 and PAC0.

When PBEN is set, the PACB is enabled. The PACB shares the input pin with IC0.

PBEN — Pulse Accumulator B System Enable

0 = 16-bit Pulse Accumulator system disabled. 8-bit PAC1 and PAC0 can be enabled when their related enable bits in ICPAR (\$28) are set.

1 = Pulse Accumulator B system enabled. The two 8-bit pulse accumulators PAC1 and PAC0 are cascaded to form the PACB 16-bit pulse accumulator. When PACB is enabled, the PACN1 and PACN0 registers contents are respectively the high and low byte of the PACB. PA1EN and PA0EN control bits in ICPAR (\$28) have no effect.

PBEN is independent from TEN. With timer disabled, the pulse accumulator can still function unless pulse accumulator is disabled.

PBOVI — Pulse Accumulator B Overflow Interrupt enable

0 = interrupt inhibited

1 = interrupt requested if PBOVF is set

3.3.27 PBFLG — Pulse Accumulator B Flag Register

Register offset: \$_31

	BIT7	6	5	4	3	2	1	BIT0
R	0	0	0	0	0	0	PBOVF	0
W								
RESET:	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 3-30 Pulse Accumulator B Flag Register (PBFLG)

Read or write any time.

PBOVF — Pulse Accumulator B Overflow Flag

This bit is set when the 16-bit pulse accumulator B overflows from \$FFFF to \$0000, or when 8-bit pulse accumulator 1 (PAC1) overflows from \$FF to \$00.

This bit is cleared by a write to the PBFLG register with bit 1 set.

Any access to the PACN1 and PACN0 registers will clear the PBOVF flag in this register when TFFCA bit in register TSCR(\$06) is set.

When PACMX = 1, PBOVF bit can also be set if 8-bit pulse accumulator 1 (PAC1) reaches \$FF and followed an active edge comes on PT1.

3.3.28 PA3H–PA0H — 8-Bit Pulse Accumulators Holding Registers

Register offset: \$_32

	BIT7	6	5	4	3	2	1	BIT0
R	PA3H7	PA3H6	PA3H5	PA3H4	PA3H3	PA3H2	PA3H1	PA3H0
W								
RESET:	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-31 8-Bit Pulse Accumulators Holding Register 3 (PA3H)

Register offset: \$_33

	BIT7	6	5	4	3	2	1	BIT0
R	PA2H7	PA2H6	PA2H5	PA2H4	PA2H3	PA2H2	PA2H1	PA2H0
W								
RESET:	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 3-32 8-Bit Pulse Accumulators Holding Register 2 (PA2H)

Register offset: \$_34

	BIT7	6	5	4	3	2	1	BIT0
R	PA1H7	PA1H6	PA1H5	PA1H4	PA1H3	PA1H2	PA1H1	PA1H0
W								
RESET:	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 3-33 8-Bit Pulse Accumulators Holding Register 1 (PA1H)

Register offset: \$_35

	BIT7	6	5	4	3	2	1	BIT0
R	PA0H7	PA0H6	PA0H5	PA0H4	PA0H3	PA0H2	PA0H1	PA0H0
W								
RESET:	0	0	0	0	0	0	0	0

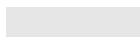
 = Unimplemented or Reserved

Figure 3-34 8-Bit Pulse Accumulators Holding Register 0 (PA0H)

Read: any time
 Write: has no effect.

These registers are used to latch the value of the corresponding pulse accumulator when the related bits in register ICPAR (\$28) are enabled (see **4.2.2 Pulse Accumulators**).

3.3.29 MCCNT — Modulus Down-Counter Count Register

Register address: \$36-\$37

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt	mccnt
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESET	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 3-35 Modulus Down-Counter Count Register (MCCNT)

Read or write any time.

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give different result than accessing them as a word.

If the RDMCL bit in MCCTL register is cleared, reads of the MCCNT register will return the present value of the count register. If the RDMCL bit is set, reads of the MCCNT will return the contents of the load register.

If a \$0000 is written into MCCNT and modulus counter while LATQ and BUFEN in ICSYS (\$2B) register are set, the input capture and pulse accumulator registers will be latched.

With a \$0000 write to the MCCNT, the modulus counter will stay at zero and does not set the MCZF flag in MCFLG register.

If modulus mode is enabled (MODMC=1), a write to this address will update the load register with the value written to it. The count register will not be updated with the new value until the next counter underflow.

The FLMC bit in MCCTL (\$26) can be used to immediately update the count register with the new value if an immediate load is desired.

If modulus mode is not enabled (MODMC=0), a write to this address will clear the prescaler and will immediately update the counter register with the value written to it and down-counts once to \$0000.

3.3.30 Timer Input Capture Holding Registers 0-3

Register offset: \$ _38-\$ _39

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-36 Timer Input Capture Holding Register 0 (TC0H)

Register offset: \$ _3A-\$ _3B

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-37 Timer Input Capture Holding Register 1 (TC1H)

Register offset: \$ _3C-\$ _3D

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-38 Timer Input Capture Holding Register 2 (TC2H)

Register offset: \$ _3E-\$ _3F

	BIT15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT0
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0


 = Unimplemented or Reserved

Figure 3-39 Timer Input Capture Holding Register 3 (TC3H)

Read: any time

Write: has no effect.

These registers are used to latch the value of the input capture registers TC0 – TC3. The corresponding IOSx bits in TIOS (\$00) should be cleared (see **4.2.1 IC Channels**).

Section 4 Functional Description

4.1 General

This section provides a complete functional description of the ECT block, detailing the operation of the design from the end user perspective in a number of subsections.

Refer to the Timer Block Diagrams from **Figure 4-1** to **Figure 4-5** as necessary.

Figure 4-1 Detailed Timer Block Diagram in Latch mode

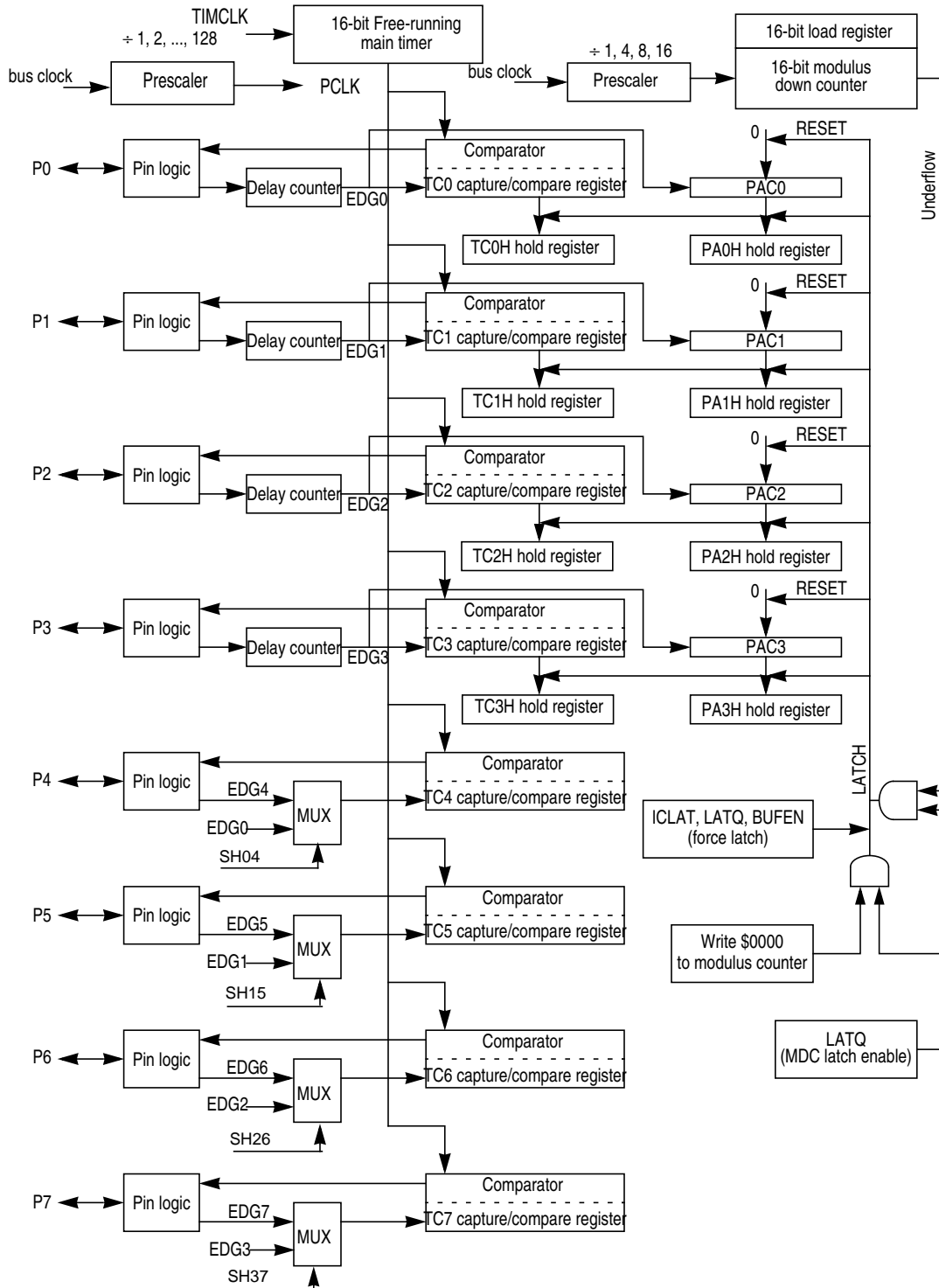


Figure 4-3 8-Bit Pulse Accumulators Block Diagram

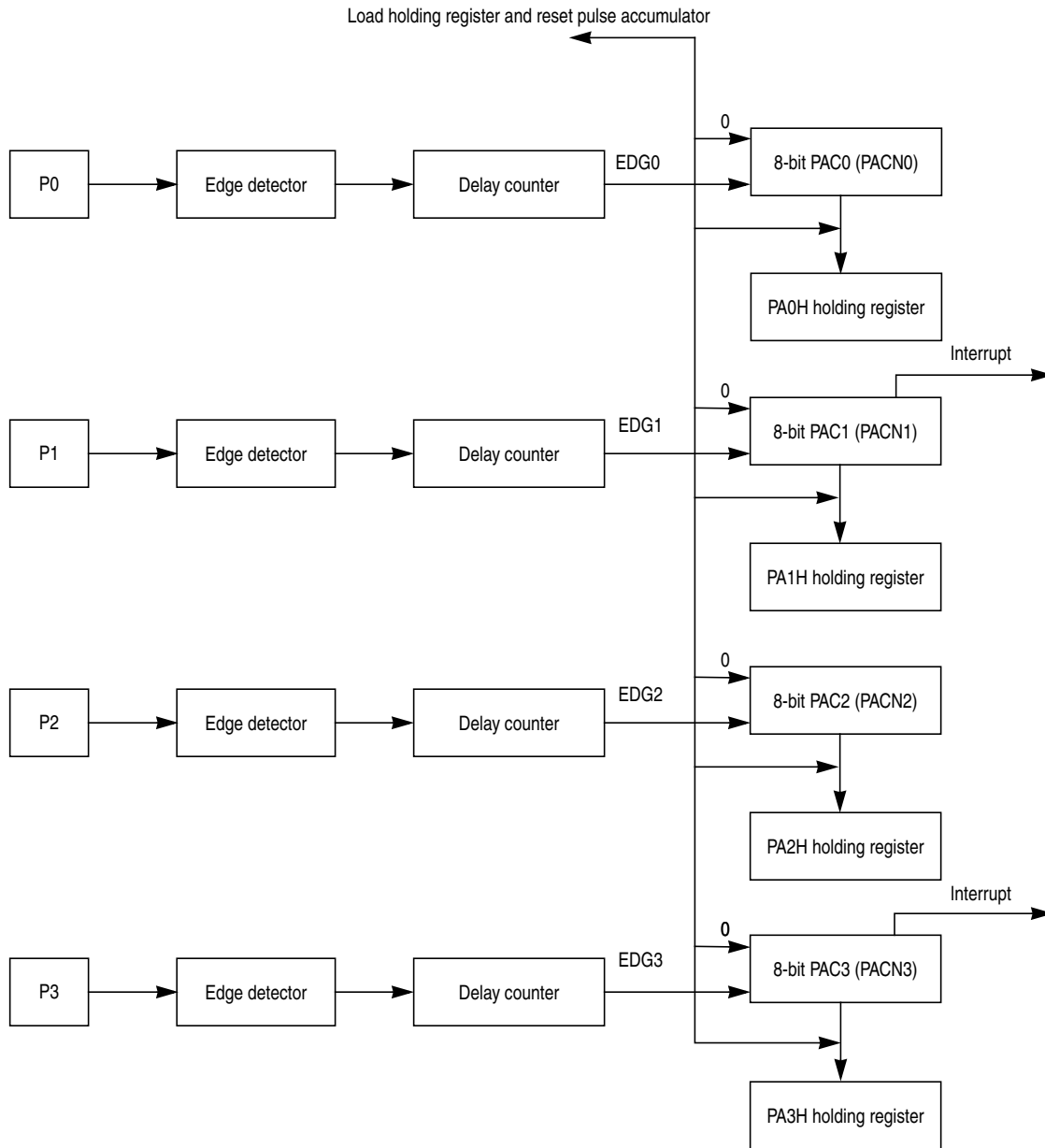
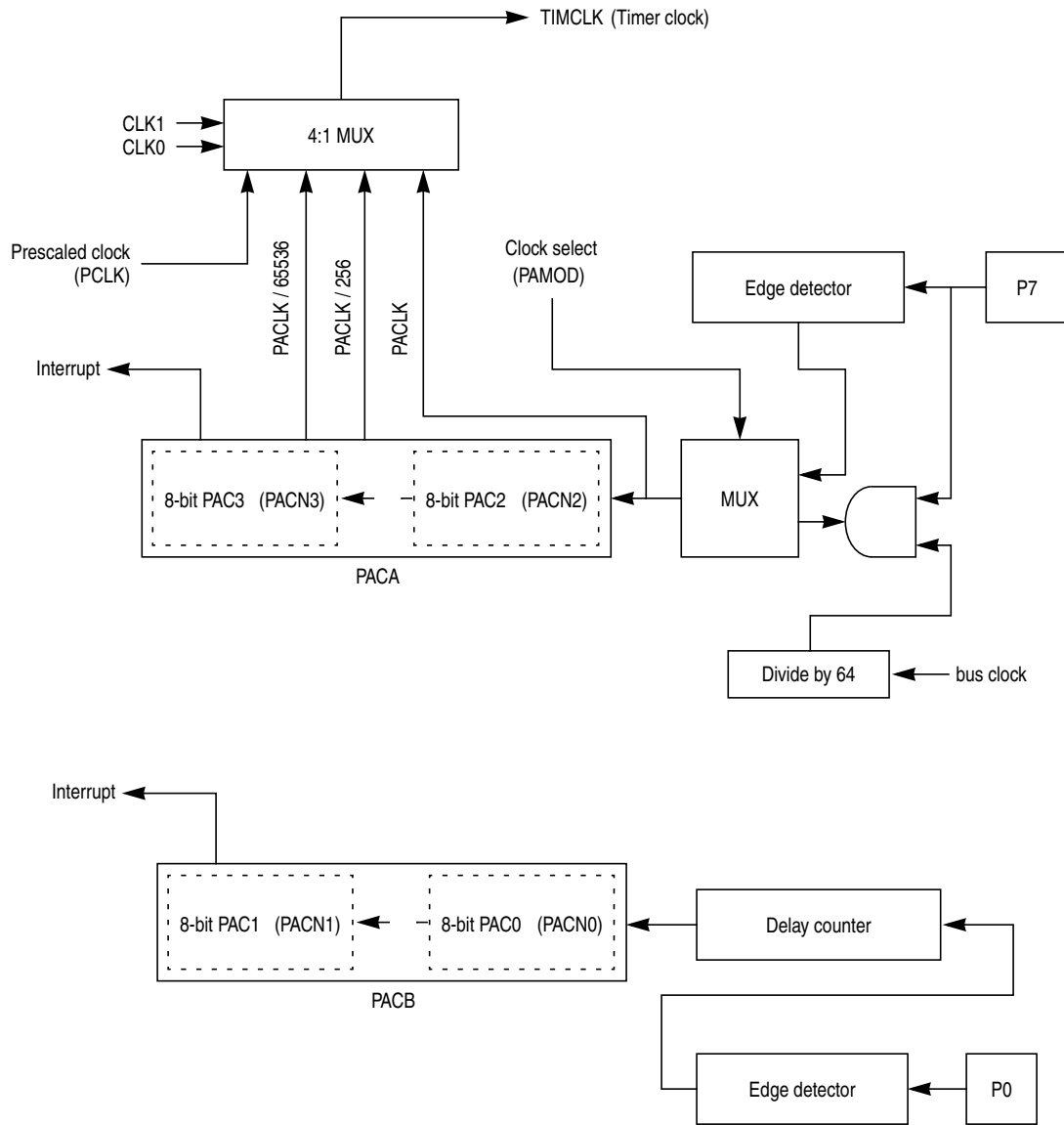


Figure 4-4 16-Bit Pulse Accumulators Block Diagram



4.2.1.1 Non-Buffered IC Channels

The main timer value is memorized in the IC register by a valid input pin transition. If the corresponding NOVW_x bit of the ICOVW register is cleared, with a new occurrence of a capture, the contents of IC register are overwritten by the new value.

If the corresponding NOVW_x bit of the ICOVW register is set, the capture register cannot be written unless it is empty.

This will prevent the captured value to be overwritten until it is read.

4.2.1.2 Buffered IC Channels

There are two modes of operations for the buffered IC channels.

- IC Latch Mode:

When enabled (LATQ=1), the main timer value is memorized in the IC register by a valid input pin transition. See **Figure 4-1**

The value of the buffered IC register is latched to its holding register by the Modulus counter for a given period when the count reaches zero, by a write \$0000 to the modulus counter or by a write to ICLAT in the MCCTL register.

If the corresponding NOVW_x bit of the ICOVW register is cleared, with a new occurrence of a capture, the contents of IC register are overwritten by the new value. In case of latching, the contents of its holding register are overwritten.

If the corresponding NOVW_x bit of the ICOVW register is set, the capture register or its holding register cannot be written by an event unless they are empty (see **4.2.1**). This will prevent the captured value to be overwritten until it is read or latched in the holding register.

- IC queue mode:

When enabled (LATQ=0), the main timer value is memorized in the IC register by a valid input pin transition. See **Figure 4-2**

If the corresponding NOVW_x bit of the ICOVW register is cleared, with a new occurrence of a capture, the value of the IC register will be transferred to its holding register and the IC register memorizes the new timer value.

If the corresponding NOVW_x bit of the ICOVW register is set, the capture register or its holding register cannot be written by an event unless they are empty (see **4.2.1**).

In queue mode, reads of holding register will latch the corresponding pulse accumulator value to its holding register.

4.2.2 Pulse Accumulators

There are four 8-bit pulse accumulators with four 8-bit holding registers associated with the four IC buffered channels. A pulse accumulator counts the number of active edges at the input of its channel.

The user can prevent 8-bit pulse accumulators counting further than \$FF by PACMX control bit in ICSYS (\$2B). In this case a value of \$FF means that 255 counts or more have occurred.

Each pair of pulse accumulators can be used as a 16-bit pulse accumulator. See **Figure 4-4**

There are two modes of operation for the pulse accumulators.

4.2.2.1 Pulse Accumulator latch mode

The value of the pulse accumulator is transferred to its holding register when the modulus down-counter reaches zero, a write \$0000 to the modulus counter or when the force latch control bit ICLAT is written.

At the same time the pulse accumulator is cleared.

4.2.2.2 Pulse Accumulator queue mode

When queue mode is enabled, reads of an input capture holding register will transfer the contents of the associated pulse accumulator to its holding register.

At the same time the pulse accumulator is cleared.

4.2.3 Modulus Down-Counter

The modulus down-counter can be used as a time base to generate a periodic interrupt. It can also be used to latch the values of the IC registers and the pulse accumulators to their holding registers.

The action of latching can be programmed to be periodic or only once.

Section 5 Reset

5.1 General

The reset state of each individual bit is listed within the Register Description section (**Section 3 Memory Map and Registers**) which details the registers and their bit-fields.

Section 6 Interrupts

6.1 General

This section describes interrupts originated by the ECT_16B8C block. The MCU must service the interrupt requests. **Table 6-1** lists the interrupts generated by the ECT to communicate with the MCU.

Table 6-1 ECT Interrupts

Interrupt Source	Description
Timer Channel 7-0	Active high timer channel interrupts 7-0
Modulus counter underflow	Active high modulus counter interrupt
Pulse Accumulator B Overflow	Active high pulse accumulator B interrupt
Pulse Accumulator A Input	Active high pulse accumulator A input interrupt
Pulse Accumulator A Overflow	Pulse accumulator overflow interrupt
Timer Overflow	Timer Overflow interrupt

6.2 Description of Interrupt Operation

The ECT_16B8C only originates interrupt requests. The following is a description of how the module makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are chip dependent.

6.2.1 Channel [7:0] Interrupt

This active high output will be asserted by the module to request a timer channel 7 - 0 interrupt to be serviced by the system controller.

6.2.2 Modulus Counter Interrupt

This active high output will be asserted by the module to request a modulus counter underflow interrupt to be serviced by the system controller.

6.2.3 Pulse Accumulator B Overflow Interrupt)

This active high output will be asserted by the module to request a timer pulse accumulator B overflow interrupt to be serviced by the system controller.

6.2.4 Pulse Accumulator A Input Interrupt

This active high output will be asserted by the module to request a timer pulse accumulator A input interrupt to be serviced by the system controller.

6.2.5 Pulse Accumulator A Overflow Interrupt

This active high output will be asserted by the module to request a timer pulse accumulator A overflow interrupt to be serviced by the system controller.

6.2.6 Timer Overflow Interrupt

This active high output will be asserted by the module to request a timer overflow interrupt to be serviced by the system controller.

User Guide End Sheet


**FINAL PAGE OF
60
PAGES**

ECT_16B8C, V01.06

HCS12 Inter-Integrated Circuit(IIC) Block Guide V02.08

Original Release Date: 08 SEP 1999
Revised: Jun 3, 2004

8/16 Bit Division,TSPG
Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

©Motorola, Inc., 2002

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
0.1	8-Sep-99		Vipin Agrawal, Puneet Goel	Original draft. Distributed only within Motorola
0.2	30-Sep-99		Puneet Goel	Minor corrections as suggested by Joachim Kruecken.
2.0	12-Feb-01		Gautam Kar, Gurdarshan Kalra	Reformatted for SRS v2.0
2.1	2-Mar-2001		Gurdarshan Kalra	Minor corrections as suggested by Jens Winkler
2.2	6-Mar-2001		Gurdarshan Kalra	Minor corrections as suggested by Jens Winkler
2.03	26-Mar-2001		Gurdarshan Kalra Jens Winkler	Minor updates in format
2.04	19-July-2001		Dirk Rowald	Document names have been added, Names and variable definitions have been hidden
2.05	7-Mar-2002		Stephen Zhou	Minor updates in format
2.06	18-Aug-2002		Stephen Zhou	Reformatted for SRS3.0, and add examples for programming general use and some diagrams to make it more user friendly as suggested by Joachim
2.07	11-Apr-2003		Stephen Zhou	Clearly claim support 400kps; Add notes for TCF bit in Section 5.1.3 Correct Section 7 for IBIF is cleared by writing '1'
2.08	3-Jun-2004		Vickers Cai	Correct the wrong divider values for SDA Hold from IBC=\$60 to IBC=\$7F

Table of Contents

Section 1 Introduction

1.1	Overview	11
1.2	Features	11
1.3	Modes of Operation	11
1.4	Block Diagram	12

Section 2 External Signal Description

2.1	Overview	14
2.2	Detailed Signal Descriptions	14
2.2.1	SCL	14
2.2.2	SDA	14

Section 3 Memory Map/Register Definition

3.1	Overview	15
3.2	Module Memory Map	15
3.3	Register Descriptions	15
3.3.1	IIC Address Register	15
3.3.2	IIC Frequency Divider Register	16
3.3.3	IIC Control Register	25
3.3.4	IIC Status Register	27
3.3.5	IIC Data I/O Register	29

Section 4 Functional Description

4.1	General	30
4.2	I-Bus Protocol	30
4.2.1	START Signal	31
4.2.2	Slave Address Transmission	31
4.2.3	Data Transfer	32
4.2.4	STOP Signal	32
4.2.5	Repeated START Signal	32
4.2.6	Arbitration Procedure	32
4.2.7	Clock Synchronization	33
4.2.8	Handshaking	33

4.2.9	Clock Stretching	33
4.3	Modes of Operation	34
4.3.1	Run Mode	34
4.3.2	Wait Mode	34
4.3.3	Stop Mode	34

Section 5 Initialization/Application Information

5.1	IIC Programming Examples	35
5.1.1	Initialization Sequence	35
5.1.2	Generation of START	35
5.1.3	Post-Transfer Software Response	35
5.1.4	Generation of STOP	36
5.1.5	Generation of Repeated START	37
5.1.6	Slave Mode	37
5.1.7	Arbitration Lost	37

Section 6 Resets

6.1	General	40
-----	-------------------	----

Section 7 Interrupts

7.1	General	41
7.2	Interrupt Description	41

List of Figures

Figure 1-1	IIC Block Diagram	12
Figure 3-1	IIC Bus Address Register (IBAD).	15
Figure 3-2	IIC Bus Frequency Divider Register (IBFD).	16
Figure 3-3	SCL divider and SDA hold.	18
Figure 3-4	IIC-Bus Control Register (IBCR)	25
Figure 3-5	IIC Bus Status Register (IBSR)	27
Figure 3-6	IIC Bus Data I/O Register (IBDR)	29
Figure 4-1	IIC-Bus Transmission Signals	30
Figure 4-2	Start and Stop conditions.	31
Figure 4-3	IIC-Bus Clock Synchronization	33
Figure 5-1	Flow-Chart of Typical IIC Interrupt Routine	39

List of Tables

Table 3-1	Module Memory Map	15
Table 3-2	I-Bus Tap and Prescale Values	16
Table 3-3	Multiplier Factor	17
Table 3-4	IIC Divider and Hold Values.	18
Table 7-1	Interrupt Summary	41

Preface

N/A.

Section 1 Introduction

1.1 Overview

The Inter-IC Bus (IIC or I2C) is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange between devices. Being a two-wire device, the IIC Bus minimizes the need for large numbers of connections between devices, and eliminates the need for an address decoder.

This bus is suitable for applications requiring occasional communications over a short distance between a number of devices. It also provides flexibility, allowing additional devices to be connected to the bus for further expansion and system development.

The interface will operate at baud rates of up to 100kbps with maximum capacitive bus loading. With reduced bus slew rate, the device is capable of operating at higher baud rates, up to a maximum of $[\text{MCU}_{\text{bus}}]\text{clock}/20$. The module can operate up to a baud rate of 400kbps provided the IIC bus slew rate is less than 100ns. The maximum communication interconnect length and the number of devices that can be connected to the bus are limited by a maximum bus capacitance of 400pF in all instances.

1.2 Features

The IIC module has the following key features:

- Compatible with I2C Bus standard
- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection

1.3 Modes of Operation

The IIC functions the same in normal, special, and emulation modes. It has two low power modes, wait and stop modes.

- Run Mode

This is the basic mode of operation.

- Wait Mode

IIC operation in wait mode can be configured. Depending on the state of internal bits, the IIC can operate normally when the CPU is in wait mode or the IIC clock generation can be turned off and the IIC module enters a power conservation state during wait mode. In the latter case, any transmission or reception in progress stops at wait mode entry.

- Stop Mode

The IIC is inactive in stop mode for reduced power consumption. The STOP instruction does not affect IIC register states.

1.4 Block Diagram

The block diagram of the IIC module is shown in **Figure 1-1**

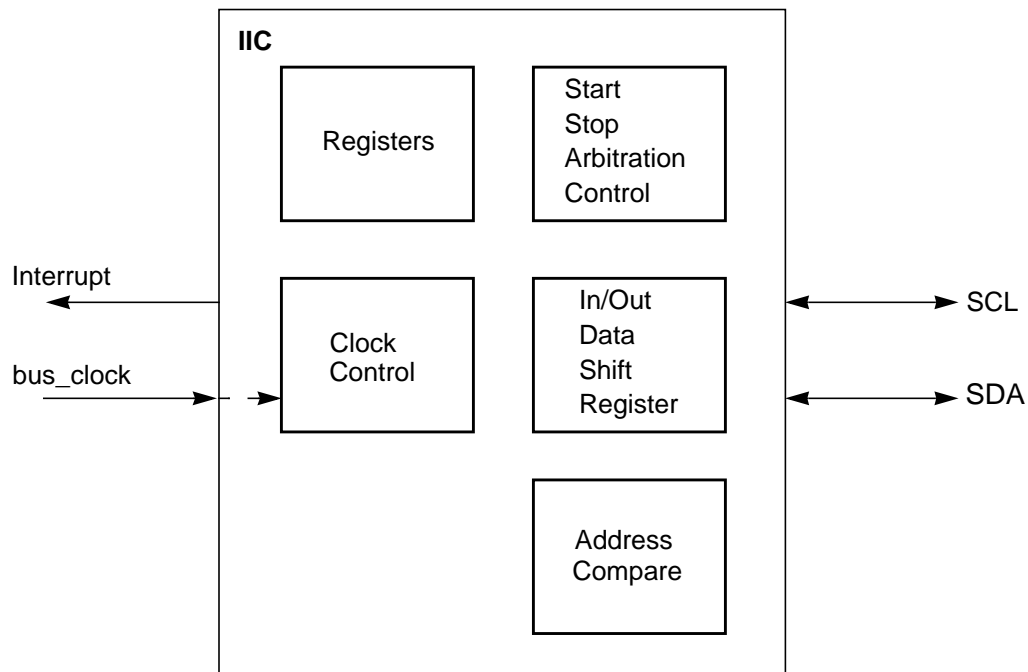
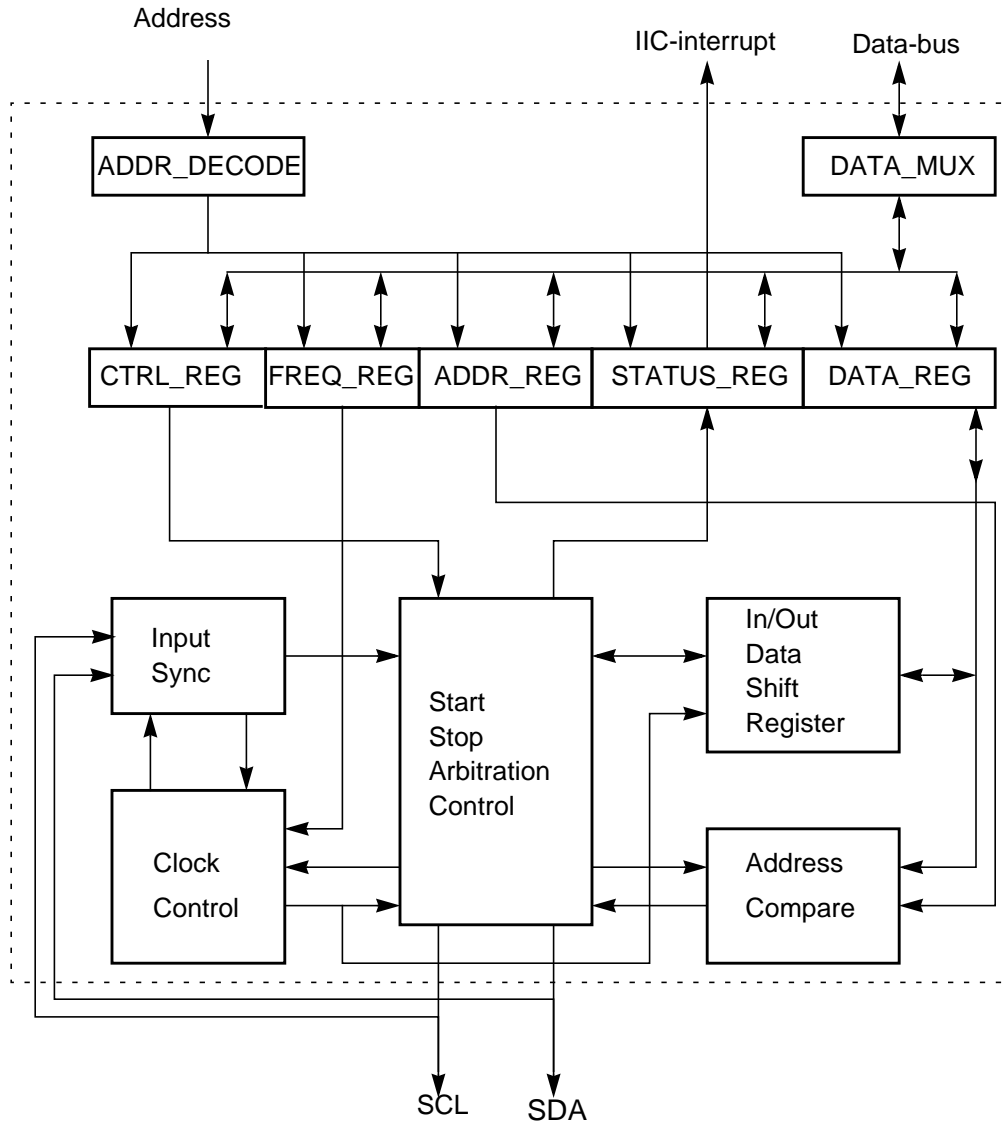


Figure 1-1 IIC Block Diagram



Section 2 External Signal Description

2.1 Overview

The IIC module has a total of 2 external pins.

2.2 Detailed Signal Descriptions

2.2.1 SCL

This is the bidirectional Serial Clock Line (SCL) of the module, compatible to the IIC-Bus specification.

2.2.2 SDA

This is the bidirectional Serial Data line (SDA) of the module, compatible to the IIC-Bus specification.

Section 3 Memory Map/Register Definition

3.1 Overview

This section provides a detailed description of all memory and registers for the IIC module.

3.2 Module Memory Map

The memory map for the IIC module is given below in **Table 3-1**. The Address listed for each register is the address offset. The total address for each register is the sum of the base address for the IIC module and the address offset for each register.

Table 3-1 Module Memory Map

Address	Use	Access
Base Address + \$_0	IIC-Bus Address Register (IBAD)	Read/Write
Base Address + \$_1	IIC-Bus Frequency Divider Register (IBFD)	Read/Write
Base Address + \$_2	IIC-Bus Control Register (IBCR)	Read/Write
Base Address + \$_3	IIC-Bus Status Register (IBSR)	Read/Write
Base Address + \$_4	IIC-Bus Data I/O Register (IBDR)	Read/Write

3.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

3.3.1 IIC Address Register

Register address: Base Address + \$0000)



Figure 3-1 IIC Bus Address Register (IBAD)

Read and write anytime

This register contains the address the IIC Bus will respond to when addressed as a slave; note that it is not the address sent on the bus during the address transfer.

ADR7–ADR1 — Slave Address

Bit 1 to bit 7 contain the specific slave address to be used by the IIC Bus module. The default mode of IIC Bus is slave mode for an address match on the bus.

RESERVED

Bit 0 of the IBAD is reserved for future compatibility. This bit will always read 0.

3.3.2 IIC Frequency Divider Register

Register address: Base address + \$0001

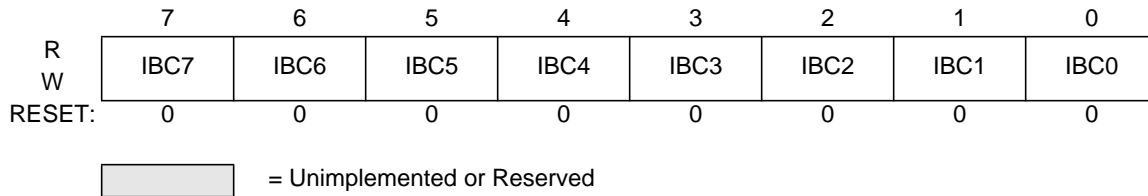


Figure 3-2 IIC Bus Frequency Divider Register (IBFD)

Read and write anytime

IBC7–IBC0 — I-Bus Clock Rate 7–0

This field is used to prescale the clock for bit rate selection. The bit clock generator is implemented as a prescale divider - IBC7-6, prescaled shift register - IBC5-3 select the prescaler divider and IBC2-0 select the shift register tap point. The IBC bits are decoded to give the Tap and Prescale values as shown in **Table 3-2**

Table 3-2 I-Bus Tap and Prescale Values

IBC2-0 (bin)	SCL Tap (clocks)	SDA Tap (clocks)
000	5	1
001	6	1
010	7	2
011	8	2
100	9	3
101	10	3
110	12	4
111	15	4

IBC5-3 (bin)	scl2start (clocks)	scl2stop (clocks)	scl2tap (clocks)	tap2tap (clocks)
000	2	7	4	1

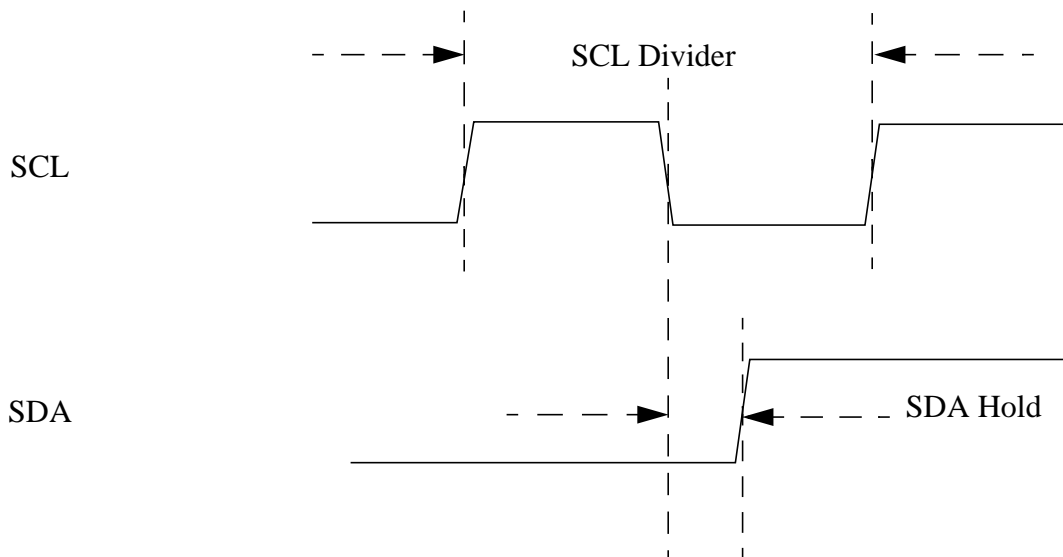
IBC5-3 (bin)	scl2start (clocks)	scl2stop (clocks)	scl2tap (clocks)	tap2tap (clocks)
001	2	7	4	2
010	2	9	6	4
011	6	9	6	8
100	14	17	14	16
101	30	33	30	32
110	62	65	62	64
111	126	129	126	128

Table 3-3 Multiplier Factor

IBC7-6	MUL
00	01
01	02
10	04
11	RESERVED

The number of clocks from the falling edge of SCL to the first tap (Tap[1]) is defined by the values shown in the scl2tap column of **Table 3-2**, all subsequent tap points are separated by $2^{\text{IBC5-3}}$ as shown in the tap2tap column in **Table 3-2**. The SCL Tap is used to generate the SCL period and the SDA Tap is used to determine the delay from the falling edge of SCL to SDA changing, the SDA hold time

IBC7-6 defines the multiplier factor MUL. The values of MUL are shown in the **Table 3-3**



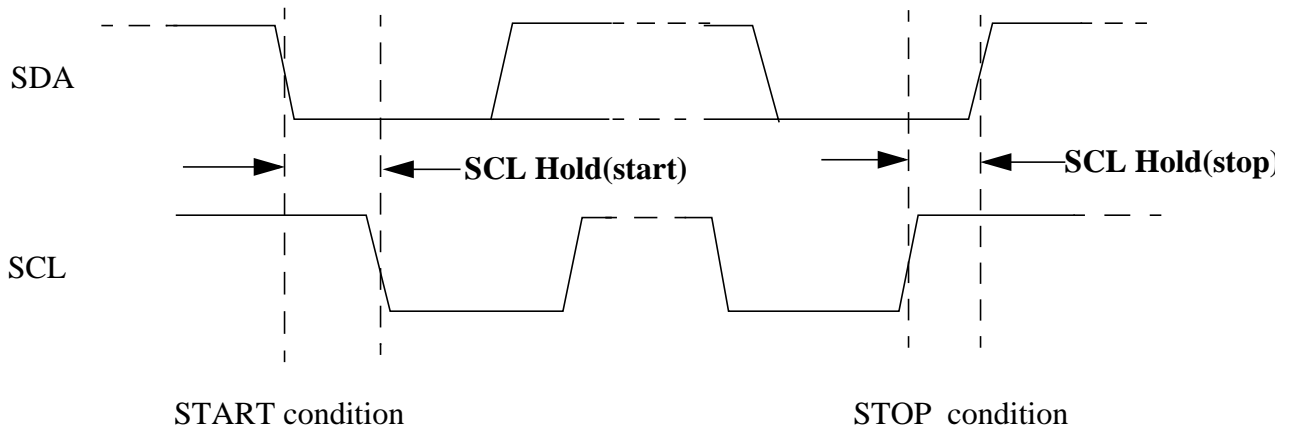


Figure 3-3 SCL divider and SDA hold

The equation used to generate the divider values from the IBFD bits is:

$$\text{SCL Divider} = \text{MUL} \times \{2 \times (\text{scl2tap} + [(\text{SCL_Tap} - 1) \times \text{tap2tap}] + 2)\}$$

The SDA hold delay is equal to the CPU clock period multiplied by the SDA Hold value shown in **Table 3-4**. The equation used to generate the SDA Hold value from the IBFD bits is:

$$\text{SDA Hold} = \text{MUL} \times \{\text{scl2tap} + [(\text{SDA_Tap} - 1) \times \text{tap2tap}] + 3\}$$

The equation for SCL Hold values to generate the start and stop conditions from the IBFD bits is:

$$\text{SCL Hold(start)} = \text{MUL} \times [\text{scl2start} + (\text{SCL_Tap} - 1) \times \text{tap2tap}]$$

$$\text{SCL Hold(stop)} = \text{MUL} \times [\text{scl2stop} + (\text{SCL_Tap} - 1) \times \text{tap2tap}]$$

Table 3-4 IIC Divider and Hold Values

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
MUL=1				
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
MUL=2				
40	40	14	12	22
41	44	14	14	24
42	48	16	16	26
43	52	16	18	28

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
44	56	18	20	30
45	60	18	22	32
46	68	20	26	36
47	80	20	32	42
48	56	14	20	30
49	64	14	24	34
4A	72	18	28	38
4B	80	18	32	42
4C	88	22	36	46
4D	96	22	40	50
4E	112	26	48	58
4F	136	26	60	70
50	96	18	36	50
51	112	18	44	58
52	128	26	52	66
53	144	26	60	74
54	160	34	68	82
55	176	34	76	90
56	208	42	92	106
57	256	42	116	130
58	160	18	76	82
59	192	18	92	98
5A	224	34	108	114
5B	256	34	124	130
5C	288	50	140	146
5D	320	50	156	162
5E	384	66	188	194
5F	480	66	236	242
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
64	576	98	284	290
65	640	98	316	322
66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074
7F	7680	1026	3836	3842
MUL=4				
80	80	28	24	44
81	88	28	28	48
82	96	32	32	52

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
83	104	32	36	56
84	112	36	40	60
85	120	36	44	64
86	136	40	52	72
87	160	40	64	84
88	112	28	40	60
89	128	28	48	68
8A	144	36	56	76
8B	160	36	64	84
8C	176	44	72	92
8D	192	44	80	100
8E	224	52	96	116
8F	272	52	120	140
90	192	36	72	100
91	224	36	88	116
92	256	52	104	132
93	288	52	120	148
94	320	68	136	164
95	352	68	152	180
96	416	84	184	212
97	512	84	232	260
98	320	36	152	164
99	384	36	184	196
9A	448	68	216	228
9B	512	68	248	260
9C	576	100	280	292
9D	640	100	312	324
9E	768	132	376	388
9F	960	132	472	484
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452

IBC[7:0] (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964
A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284
B1	3072	260	1528	1540
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

3.3.3 IIC Control Register

Register address: Base address + \$0002

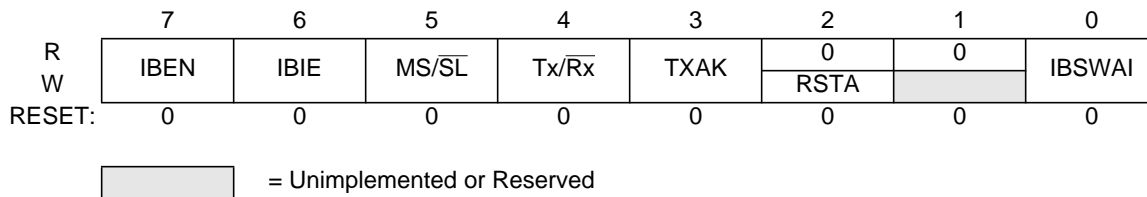


Figure 3-4 IIC-Bus Control Register (IBCR)

Read and write anytime

IBEN — I-Bus Enable

This bit controls the software reset of the entire IIC Bus module.

0 = The module is reset and disabled. This is the power-on reset situation. When low the interface is held in reset but registers can still be accessed

1 = The IIC Bus module is enabled. This bit must be set before any other IBCR bits have any effect

If the IIC Bus module is enabled in the middle of a byte transfer the interface behaves as follows: slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected. Master mode will not be aware that the bus is busy, hence if a start cycle is initiated then the current bus cycle may become corrupt. This would ultimately result in either the current bus master or the IIC Bus module losing arbitration, after which bus operation would return to normal.

IBIE — I-Bus Interrupt Enable

0 = Interrupts from the IIC Bus module are disabled. Note that this does not clear any currently pending interrupt condition

1 = Interrupts from the IIC Bus module are enabled. An IIC Bus interrupt occurs provided the IBIF bit in the status register is also set.

MS/ $\overline{\text{SL}}$ — Master/Slave mode select bit

Upon reset, this bit is cleared. When this bit is changed from 0 to 1, a START signal is generated on the bus, and the master mode is selected. When this bit is changed from 1 to 0, a STOP signal is generated and the operation mode changes from master to slave. A STOP signal should only be generated if the IBIF flag is set. MS/ $\overline{\text{SL}}$ is cleared without generating a STOP signal when the master loses arbitration.

0 = Slave Mode

1 = Master Mode

Tx/ $\overline{\text{Rx}}$ — Transmit/Receive mode select bit

This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the status register. In master mode this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high.

0 = Receive

1 = Transmit

TXAK — Transmit Acknowledge enable

This bit specifies the value driven onto SDA during data acknowledge cycles for both master and slave receivers. The IIC module will always acknowledge address matches, provided it is enabled, regardless of the value of TXAK. Note that values written to this bit are only used when the IIC Bus is a receiver, not a transmitter.

- 0 = An acknowledge signal will be sent out to the bus at the 9th clock bit after receiving one byte data
- 1 = No acknowledge signal response is sent (i.e., acknowledge bit = 1)

RSTA — Repeat Start

Writing a 1 to this bit will generate a repeated START condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration.

- 1 = Generate repeat start cycle

RESERVED

Bit 1 of the IBCR is reserved for future compatibility. This bit will always read 0.

IBSWAI — I-Bus Interface Stop in WAIT mode

- 0 = IIC Bus module clock operates normally
- 1 = Halt IIC Bus module clock generation in WAIT mode

Wait mode is entered via execution of a CPU WAI instruction. In the event that the IBSWAI bit is set, all clocks internal to the IIC will be stopped and any transmission currently in progress will halt. If the CPU were woken up by a source other than the IIC module, then clocks would restart and the IIC would continue where it left off in the previous transmission. It is not possible for the IIC to wake up the CPU when its internal clocks are stopped.

If it were the case that the IBSWAI bit was cleared when the WAI instruction was executed, the IIC internal clocks and interface would remain alive, continuing the operation which was currently underway. It is also possible to configure the IIC such that it will wake up the CPU via an interrupt at the conclusion of the current operation. See the discussion on the IBIF and IBIE bits in the IBSR and IBCR, respectively.

3.3.4 IIC Status Register

Register address: Base address + \$0003

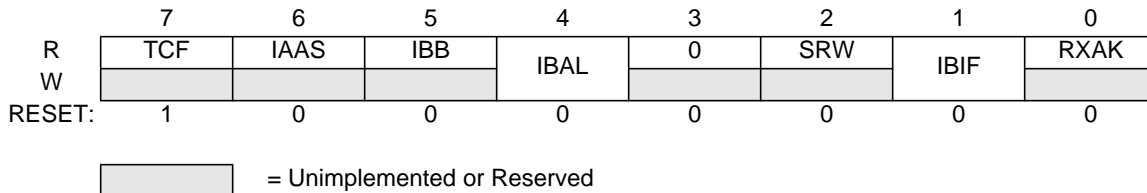


Figure 3-5 IIC Bus Status Register (IBSR)

This status register is read-only with exception of bit 1 (IBIF) and bit 4 (IBAL), which are software clearable

TCF — Data transferring bit

While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the 9th clock of a byte transfer. Note that this bit is only valid during or immediately following a transfer to the IIC module or from the IIC module.

- 0 = Transfer in progress
- 1 = Transfer complete

IAAS — Addressed as a slave bit

When its own specific address (I-Bus Address Register) is matched with the calling address, this bit is set. The CPU is interrupted provided the IBIE is set. Then the CPU needs to check the SRW bit and set its Tx/Rx mode accordingly. Writing to the I-Bus Control Register clears this bit.

- 0 = Not addressed
- 1 = Addressed as a slave

IBB — Bus busy bit

- 0 = This bit indicates the status of the bus. When a START signal is detected, the IBB is set. If a STOP signal is detected, IBB is cleared and the bus enters idle state.
- 1 = Bus is busy

IBAL — Arbitration Lost

The arbitration lost bit (IBAL) is set by hardware when the arbitration procedure is lost. Arbitration is lost in the following circumstances:

1. SDA sampled low when the master drives a high during an address or data transmit cycle.
2. SDA sampled low when the master drives a high during the acknowledge bit of a data receive cycle.
3. A start cycle is attempted when the bus is busy.
4. A repeated start cycle is requested in slave mode.
5. A stop condition is detected when the master did not request it.

This bit must be cleared by software, by writing a one to it. A write of zero has no effect on this bit. |

RESERVED

Bit 3 of IBSR is reserved for future use. A read operation on this bit will return 0.

SRW — Slave Read/Write

When IAAS is set this bit indicates the value of the R/W command bit of the calling address sent from the master

This bit is only valid when the I-Bus is in slave mode, a complete address transfer has occurred with an address match and no other transfers have been initiated.

Checking this bit, the CPU can select slave transmit/receive mode according to the command of the master.

- 0 = Slave receive, master writing to slave
- 1 = Slave transmit, master reading from slave

IBIF — I-Bus Interrupt

The IBIF bit is set when one of the following conditions occurs:

- arbitration lost (IBAL bit set)
- byte transfer complete (TCF bit set)
- addressed as slave (IAAS bit set)

It will cause a processor interrupt request if the IBIE bit is set. This bit must be cleared by software, writing a one to it. A write of zero has no effect on this bit.

RXAK — Received Acknowledge

The value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal has been received after the completion of 8 bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the 9th clock.

- 0 = Acknowledge received
- 1 = No acknowledge received

3.3.5 IIC Data I/O Register

Register address

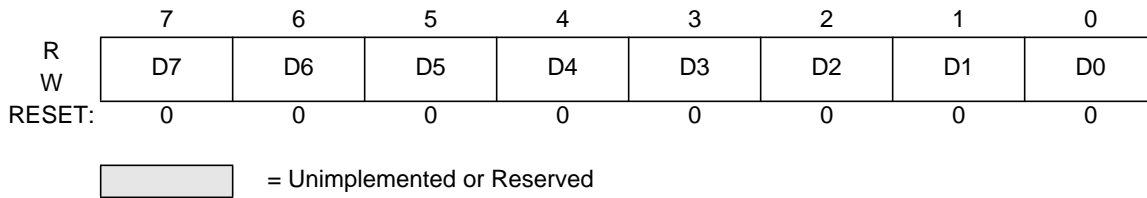


Figure 3-6 IIC Bus Data I/O Register (IBDR)

In master transmit mode, when data is written to the IBDR a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates next byte data receiving. In slave mode, the same functions are available after an address match has occurred. Note that the Tx/Rx bit in the IBCR must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the IIC is configured for master transmit but a master receive is desired, then reading the IBDR will not initiate the receive.

Reading the IBDR will return the last byte received while the IIC is configured in either master receive or slave receive modes. The IBDR does not reflect every byte that is transmitted on the IIC bus, nor can software verify that a byte has been written to the IBDR correctly by reading it back.

In master transmit mode, the first byte of data written to IBDR following assertion of $\overline{MS}/\overline{SL}$ is used for the address transfer and should comprise of the calling address (in position D7-D1) concatenated with the required $\overline{R}/\overline{W}$ bit (in position D0).

Section 4 Functional Description

4.1 General

This section provides a complete functional description of the IIC.

4.2 I-Bus Protocol

The IIC Bus system uses a Serial Data line (SDA) and a Serial Clock Line (SCL) for data transfer. All devices connected to it must have open drain or open collector outputs. Logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors is system dependent.

Normally, a standard communication is composed of four parts: START signal, slave address transmission, data transfer and STOP signal. They are described briefly in the following sections and illustrated in **Figure 4-1**.

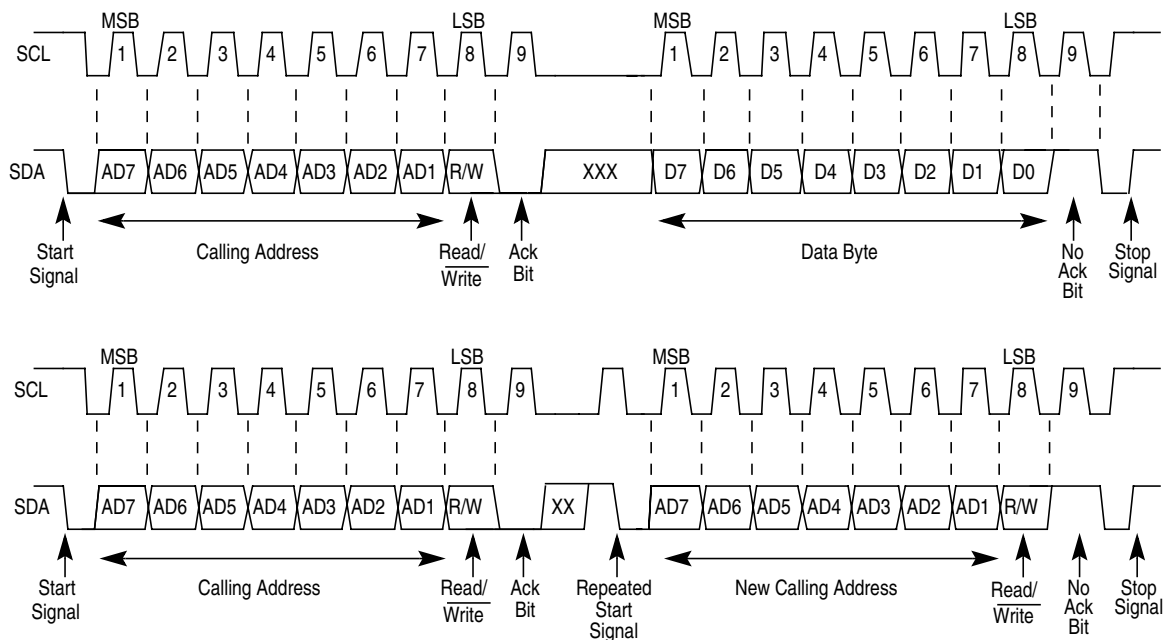


Figure 4-1 IIC-Bus Transmission Signals

4.2.1 START Signal

When the bus is free, i.e. no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START signal. As shown in Figure 4-1, a START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

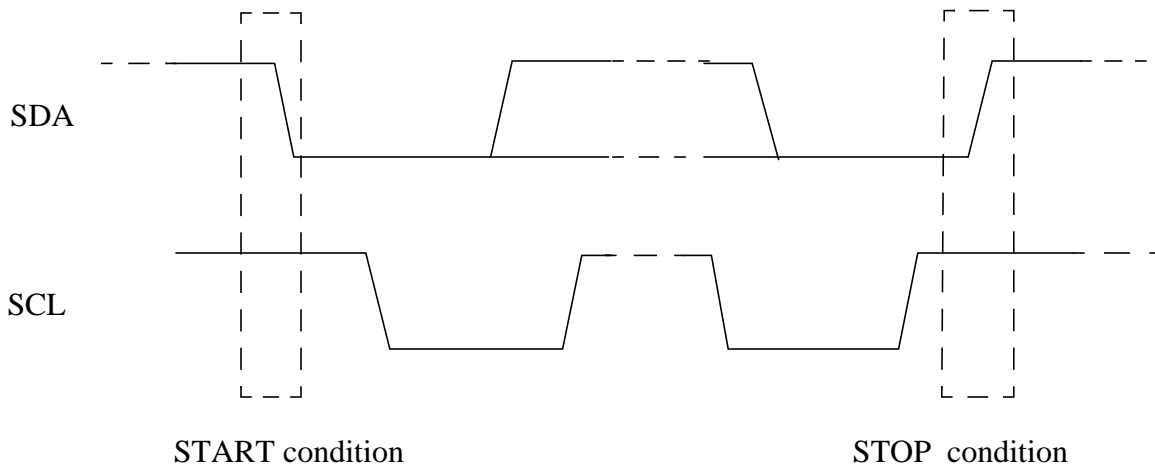


Figure 4-2 Start and Stop conditions

4.2.2 Slave Address Transmission

The first byte of data transfer immediately after the START signal is the slave address transmitted by the master. This is a seven-bit calling address followed by a R/W bit. The R/W bit tells the slave the desired direction of data transfer.

- 1 = Read transfer, the slave transmits data to the master.
- 0 = Write transfer, the master transmits data to the slave.

Only the slave with a calling address that matches the one transmitted by the master will respond by sending back an acknowledge bit. This is done by pulling the SDA low at the 9th clock (see Figure 4-1).

No two slaves in the system may have the same address. If the IIC Bus is master, it must not transmit an address that is equal to its own slave address. The IIC Bus cannot be master and slave at the same time. However, if arbitration is lost during an address cycle the IIC Bus will revert to slave mode and operate correctly even if it is being addressed by another master.

4.2.3 Data Transfer

Once successful slave addressing is achieved, the data transfer can proceed byte-by-byte in a direction specified by the R/W bit sent by the calling master

All transfers that come after an address cycle are referred to as data transfers, even if they carry sub-address information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low and must be held stable while SCL is high as shown in Figure 4-1. There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte has to be followed by an acknowledge bit, which is signalled from the receiving device by pulling the SDA low at the ninth clock. So one complete data byte transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a stop signal to abort the data transfer or a start signal (repeated start) to commence a new calling.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means 'end of data' to the slave, so the slave releases the SDA line for the master to generate STOP or START signal.

4.2.4 STOP Signal

The master can terminate the communication by generating a STOP signal to free the bus. However, the master may generate a START signal followed by a calling command without generating a STOP signal first. This is called repeated START. A STOP signal is defined as a low-to-high transition of SDA while SCL at logical “1” (see **Figure 4-1**).

The master can generate a STOP even if the slave has generated an acknowledge at which point the slave must release the bus.

4.2.5 Repeated START Signal

As shown in **Figure 4-1**, a repeated START signal is a START signal generated without first generating a STOP signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in different mode (transmit/receive mode) without releasing the bus.

4.2.6 Arbitration Procedure

The Inter-IC bus is a true multi-master bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure, a bus master loses arbitration if it transmits logic “1” while another master transmits logic “0”. The losing masters immediately switch over to slave receive mode and stop driving SDA output. In this case the transition from master to slave mode does not generate a STOP condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

4.2.7 Clock Synchronization

Since wire-AND logic is performed on SCL line, a high-to-low transition on SCL line affects all the devices connected on the bus. The devices start counting their low period and once a device's clock has gone low, it holds the SCL line low until the clock high state is reached. However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see **Figure 4-2**). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods. The first device to complete its high period pulls the SCL line low again.

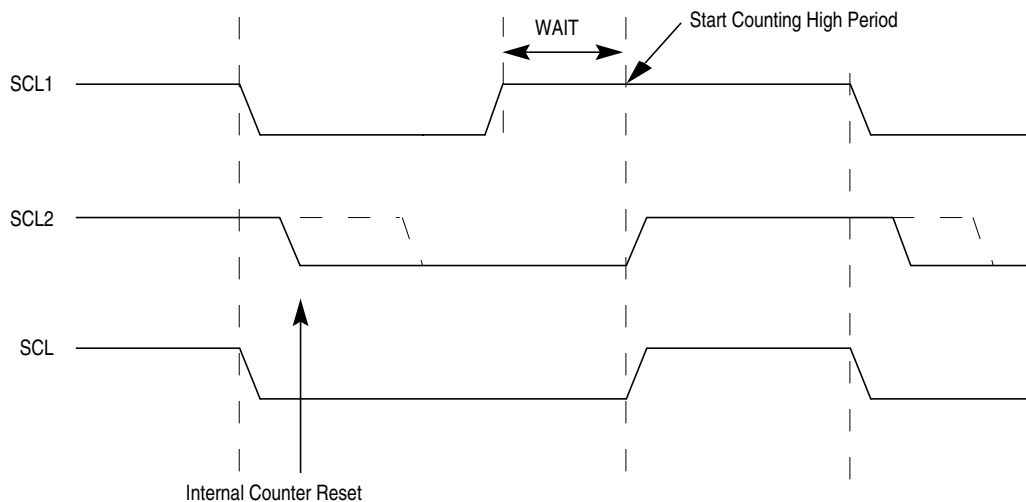


Figure 4-3 IIC-Bus Clock Synchronization

4.2.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (9 bits). In such case, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

4.2.9 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low the slave can drive SCL low for the required period and then release it. If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

4.3 Modes of Operation

The IIC functions the same in normal, special, and emulation modes. It has two low power modes, wait and stop modes

4.3.1 Run Mode

This is the basic mode of operation.

4.3.2 Wait Mode

IIC operation in wait mode can be configured. Depending on the state of internal bits, the IIC can operate normally when the CPU is in wait mode or the IIC clock generation can be turned off and the IIC module enters a power conservation state during wait mode. In the later case, any transmission or reception in progress stops at wait mode entry.

4.3.3 Stop Mode

The IIC is inactive in stop mode for reduced power consumption. The STOP instruction does not affect IIC register states.

Section 5 Initialization/Application Information

5.1 IIC Programming Examples

5.1.1 Initialization Sequence

Reset will put the IIC Bus Control Register to its default status. Before the interface can be used to transfer serial data, an initialization procedure must be carried out, as follows:

1. Update the Frequency Divider Register (IBFD) and select the required division ratio to obtain SCL frequency from system clock.
2. Update the IIC Bus Address Register (IBAD) to define its slave address.
3. Set the IBEN bit of the IIC Bus Control Register (IBCR) to enable the IIC interface system.
4. Modify the bits of the IIC Bus Control Register (IBCR) to select Master/Slave mode, Transmit/Receive mode and interrupt enable or not.

5.1.2 Generation of START

After completion of the initialization procedure, serial data can be transmitted by selecting the 'master transmitter' mode. If the device is connected to a multi-master bus system, the state of the IIC Bus Busy bit (IBB) must be tested to check whether the serial bus is free.

If the bus is free (IBB=0), the start condition and the first byte (the slave address) can be sent. The data written to the data register comprises the slave calling address and the LSB set to indicate the direction of transfer required from the slave.

The bus free time (i.e., the time between a STOP condition and the following START condition) is built into the hardware that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period it may be necessary to wait until the IIC is busy after writing the calling address to the IBDR before proceeding with the following instructions. This is illustrated in the following example.

An example of a program which generates the START signal and transmits the first byte of data (slave address) is shown below:

CHFLAG	BRSET	IBSR,#\$20,*	;WAIT FOR IBB FLAG TO CLEAR
TXSTART	BSET	IBCR,#\$30	;SET TRANSMIT AND MASTER MODE;i.e. GENERATE START CONDITION
	MOVB	CALLING,IBDR	;TRANSMIT THE CALLING ADDRESS, D0=R/W
IBFREE	BRCLR	IBSR,#\$20,*	;WAIT FOR IBB FLAG TO SET

5.1.3 Post-Transfer Software Response

Successful transmission or reception of a byte will set the TCF (data transferring) bit and the IBIF (interrupt flag) bit in the IBSR status register. An interrupt service routine can be called by this action if

the IBIE (interrupt enable) bit in the IBCR control register is set. The IBIF (interrupt flag) bit can be cleared by writing 1 (in the interrupt service routine, if interrupts are used).

The TCF bit will be cleared to indicate data transfer in progress by reading the IBDR data register in receive mode or writing the IBDR in transmit mode. The TCF bit should not be used as a data transfer complete flag as the flag timing is dependent on a number of factors including the IIC bus frequency. This bit may not conclusively provide an indication of a transfer complete situation. It is recommended that transfer complete situations are detected using the IBIF flag

Software may service the IIC I/O in the main program by monitoring the IBIF bit if the interrupt function is disabled. Note that polling should monitor the IBIF bit rather than the TCF bit since their operation is different when arbitration is lost.

Note that when an interrupt occurs at the end of the address cycle the master will always be in transmit mode, i.e. the address is transmitted. If master receive mode is required, indicated by R/W bit in IBDR, then the Tx/Rx bit should be toggled at this stage.

During slave mode address cycles (IAAS=1) the SRW bit in the status register is read to determine the direction of the subsequent transfer and the Tx/Rx bit is programmed accordingly. For slave mode data cycles (IAAS=0) the SRW bit is not valid, the Tx/Rx bit in the control register should be read to determine the direction of the current transfer.

The following is an example of a software response by a 'master transmitter' in the interrupt routine .

ISR	BCLR	IBSR,#\$02	;CLEAR THE IBIF FLAG
	BRCLR	IBCR,#\$20,SLAVE	;BRANCH IF IN SLAVE MODE
	BRCLR	IBCR,#\$10,RECEIVE	;BRANCH IF IN RECEIVE MODE
	BRSET	IBSR,#\$01,END	;IF NO ACK, END OF TRANSMISSION
TRANSMIT	MOVB	DATABUF,IBDR	;TRANSMIT NEXT BYTE OF DATA

5.1.4 Generation of STOP

A data transfer ends with a STOP signal generated by the 'master' device. A master transmitter can simply generate a STOP signal after all the data has been transmitted. The following is an example showing how a stop condition is generated by a master transmitter.

MASTX	TST	TXCNT	;GET VALUE FROM THE TRANSMITTING COUNTER
	BEQ	END	;END IF NO MORE DATA
	BRSET	IBSR,#\$01,END	;END IF NO ACK
	MOVB	DATABUF,IBDR	;TRANSMIT NEXT BYTE OF DATA
	DEC	TXCNT	;DECREASE THE TXCNT
	BRA	EMASTX	;EXIT
END	BCLR	IBCR,#\$20	;GENERATE A STOP CONDITION
EMASTX	RTI		;RETURN FROM INTERRUPT

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data which can be done by setting the transmit acknowledge bit (TXAK)

before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must be generated first. The following is an example showing how a STOP signal is generated by a master receiver.

MASR	DEC	RXCNT	;DECREASE THE RXCNT
	BEQ	ENMASR	;LAST BYTE TO BE READ
	MOVB	RXCNT,D1	;CHECK SECOND LAST BYTE
	DEC	D1	;TO BE READ
	BNE	NXMAR	;NOT LAST OR SECOND LAST
LAMAR	BSET	IBCR,#\$08	;SECOND LAST, DISABLE ACK
			;TRANSMITTING
	BRA	NXMAR	
ENMASR	BCLR	IBCR,#\$20	;LAST ONE, GENERATE 'STOP' SIGNAL
NXMAR	MOVB	IBDR,RXBUF	;READ DATA AND STORE
	RTI		

5.1.5 Generation of Repeated START

At the end of data transfer, if the master still wants to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

RESTART	BSET	IBCR,#\$04	;ANOTHER START (RESTART)
	MOVB	CALLING,IBDR	;TRANSMIT THE CALLING ADDRESS;D0=R/W

5.1.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received . If IAAS is set, software should set the transmit/receive mode select bit (Tx/Rx bit of IBCR) according to the R/W command bit (SRW). Writing to the IBCR clears the IAAS automatically. Note that the only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred, interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer may now be initiated by writing information to IBDR, for slave transmits, or dummy reading from IBDR, in slave receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the IBDR is accessed in the required mode.

In slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. Setting RXAK means an 'end of data' signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

5.1.7 Arbitration Lost

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices which lost arbitration are immediately switched to slave receive mode by the hardware. Their data output to the SDA line is stopped, but SCL is still generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with $IBAL=1$ and $MS/SL=0$. If one master attempts to start transmission while the bus is being engaged by another master, the hardware will inhibit the transmission; switch the MS/SL bit from 1 to 0 without generating *STOP* condition; generate an interrupt to CPU and set the *IBAL* to indicate that the attempt to engage the bus is failed. When considering these cases, the slave service routine should test the *IBAL* first and the software should clear the *IBAL* bit if it is set.

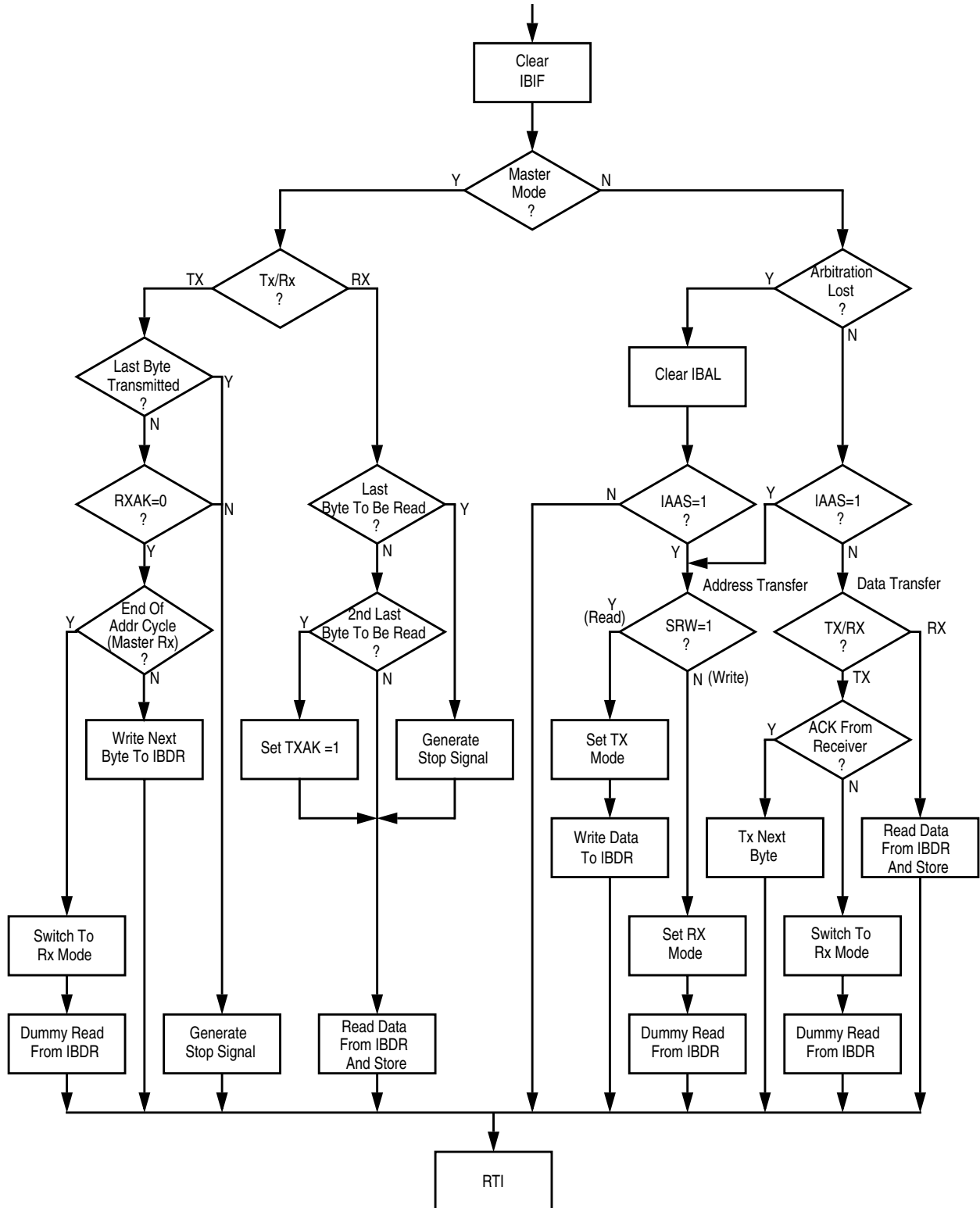


Figure 5-1 Flow-Chart of Typical IIC Interrupt Routine

Section 6 Resets

6.1 General

The reset state of each individual bit is listed within the Register Description section (see **Section 3 Memory Map/Register Definition**) which details the registers and their bit-fields.

Section 7 Interrupts

7.1 General

IIC uses only one interrupt vector.

Table 7-1 Interrupt Summary

Interrupt	Offset	Vector	Priority	Source	Description
IIC Interrupt	-	-	-	IBAL, TCF, IAAS bits in IBSR register	When either of IBAL, TCF or IAAS bits is set may cause an interrupt based on Arbitration lost, Transfer Complete or Address Detect conditions.

7.2 Interrupt Description

Internally there are three types of interrupts in IIC. The interrupt service routine can determine the interrupt type by reading the Status Register.

IIC Interrupt can be generated on

1. Arbitration Lost condition (IBAL bit set)
2. Byte Transfer condition (TCF bit set)
3. Address Detect condition (IAAS bit set)

The IIC interrupt is enabled by the IBIE bit in the IIC Control Register. It must be cleared by writing '1' to the IBIF bit in the interrupt service routine.

Block Guide End Sheet

**FINAL PAGE OF
46
PAGES**



MOTOROLA
intelligence everywhere™

digital dna™

HCS12 Microcontrollers

*Interrupt (INT)
Module V1*

S12INTV1/D
Rev. 1.00
5/2003

MOTOROLA.COM/SEMICONDUCTORS

PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
1.02	5/1/2003	5/1/2003	John Langan	Creation of block user guide from core user guide version 1.5 (Oct. 12, 2001). Changes include: updating format and making end-customer friendly. Original release.

PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED

Table of Contents

PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "**CONTROLLED COPY**" IN RED

List of Figures

Figure 1-1	Interrupt Block Diagram	5
Figure 3-1	Interrupt Register Summary	11
Figure 3-2	Interrupt Test Control Register (ITCR)	11
Figure 3-3	Interrupt TEST Registers (ITEST)	12
Figure 3-4	Highest Priority I Interrupt Register (HPRIO)	13

List of Tables

Table 4-1	Exception Vector Map and Priority	16
-----------	---	----

Section 1 Introduction to Interrupt (INT)

1.1	Overview	8
1.2	Features	8
1.3	Modes of Operation	8
1.3.1	Normal Operation	8
1.3.2	Special Operation	8
1.3.3	Emulation Modes	9
1.4	Low-Power Options	9
1.4.1	Run Mode	9
1.4.2	Wait Mode	9
1.4.3	Stop Mode	9

Section 2 External Signal Description

Section 3 Memory Map/Register Definition

3.1	Interrupt Test Control Register	13
3.2	Interrupt Test Registers	14
3.3	Highest Priority I Interrupt (Optional)	15

Section 4 Functional Description

4.1	Interrupt Exception Requests	17
4.1.1	Interrupt Registers	17
4.1.2	Highest Priority I-Bit Maskable Interrupt	17
4.1.3	Interrupt Priority Decoder	17

4.2 Reset Exception Requests 18

4.3 Exception Priority 18

PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED

Section 1 Introduction to Interrupt (INT)

This section describes the functionality of the Interrupt (INT) sub-block of the S12 Core Platform.

A block diagram of the Interrupt sub-block is shown in [Figure 1-1](#).

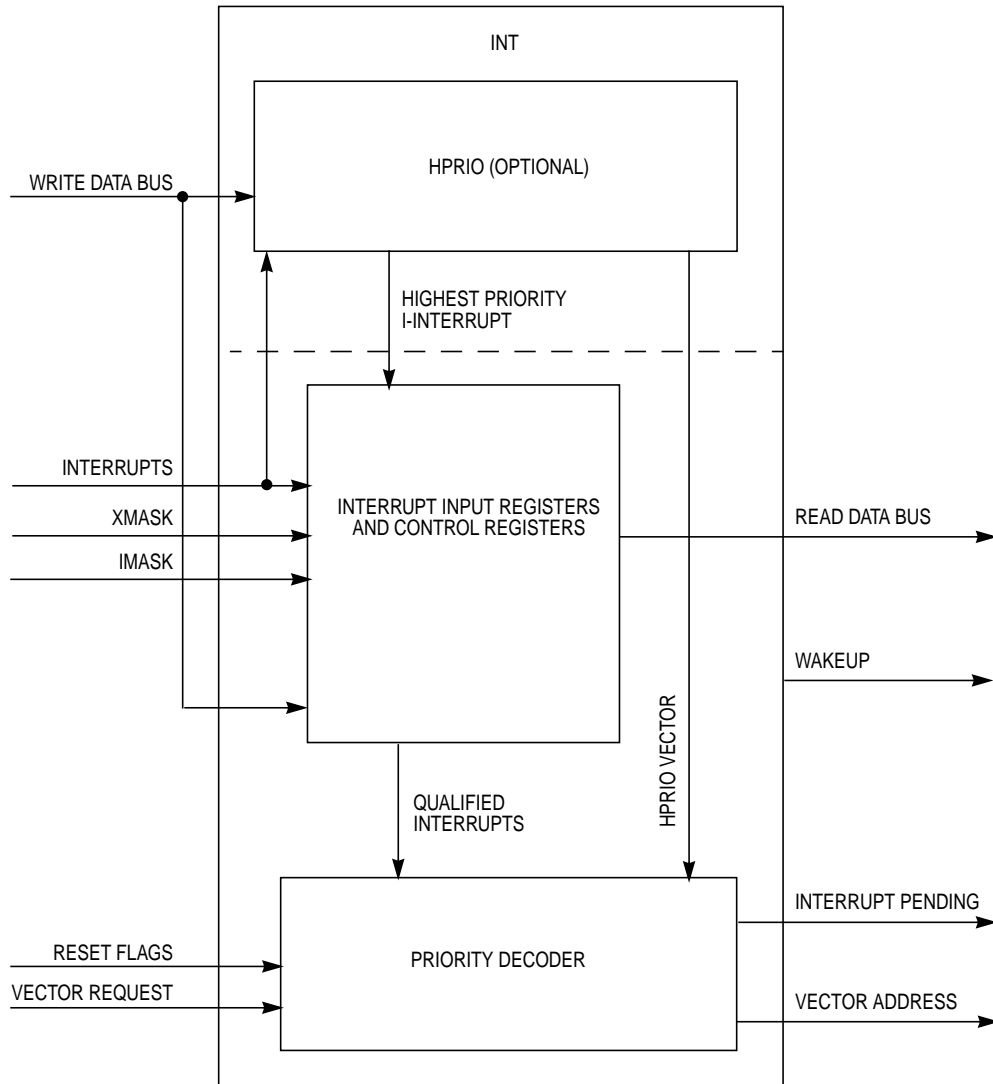


Figure 1-1 Interrupt Block Diagram

1.1 Overview

The Interrupt sub-block decodes the priority of all system exception requests and provides the applicable vector for processing the exception. The INT supports I-bit maskable and X-bit maskable interrupts, a nonmaskable Unimplemented Opcode Trap, a nonmaskable software interrupt (SWI) or Background Debug Mode request, and three system reset vector requests. All interrupt related exception requests are handled by the Interrupt sub-block (INT).

1.2 Features

The INT includes these features:

- Provides two to 122 I-bit maskable interrupt vectors (\$FF00–\$FFF2)
- Provides one X-bit maskable interrupt vector (\$FFF4)
- Provides a nonmaskable software interrupt (SWI) or Background Debug Mode request vector (\$FFF6)
- Provides a nonmaskable Unimplemented Opcode Trap (TRAP) vector (\$FFF8)
- Provides three system reset vectors (\$FFFA–\$FFFE) (Reset, CMR, and COP)
- Determines the appropriate vector and drives it onto the address bus at the appropriate time
- Signals the CPU that interrupts are pending
- Provides control registers which allow testing of interrupts
- Provides additional input signals which prevents requests for servicing I and X interrupts
- Wakes the system from stop or wait mode when an appropriate interrupt occurs or whenever \overline{XIRQ} is active, even if \overline{XIRQ} is masked
- Provides asynchronous path for all I and X interrupts, (\$FF00–\$FFF4)
- (Optional) Selects and stores the highest priority I interrupt based on the value written into the HPRIO register

1.3 Modes of Operation

The functionality of the INT sub-block in various modes of operation is discussed in the subsections that follow.

1.3.1 Normal Operation

The INT operates the same in all normal modes of operation.

1.3.2 Special Operation

Interrupts may be tested in special modes through the use of the interrupt test registers.

1.3.3 Emulation Modes

The INT operates the same in emulation modes as in normal modes.

1.4 Low-Power Options

The INT does not contain any user-controlled options for reducing power consumption. The operation of the INT in low-power modes is discussed in the following subsections.

1.4.1 Run Mode

The INT does not contain any options for reducing power in run mode.

1.4.2 Wait Mode

Clocks to the INT can be shut off during system wait mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any \overline{XIRQ} request.

1.4.3 Stop Mode

Clocks to the INT can be shut off during system stop mode and the asynchronous interrupt path will be used to generate the wake-up signal upon recognition of a valid interrupt or any \overline{XIRQ} request.

Section 2 External Signal Description

Most interfacing with the Interrupt sub-block is done within the Core. However, the Interrupt does receive direct input from the Multiplexed External Bus Interface (MEBI) sub-block of the Core for the $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$ pin data.

PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED

Section 3 Memory Map/Register Definition

A summary of the registers associated with the Interrupt sub-block is shown in [Figure 3-1](#). Detailed descriptions of the registers and associated bits are given in the subsections that follow.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
\$0015	ITCR	Read	0	0	0	WRTINT	ADR3	ADR2	ADR1	ADR0
		Write								
\$0016	ITEST	Read	INTE	INTC	INTA	INT8	INT6	INT4	INT2	INT0
		Write								
\$001F	HPRIO	Read	PSEL7	PSEL6	PSEL5	PSEL4	PSEL3	PSEL2	PSEL1	0
		Write								


 = Unimplemented

Figure 3-1 Interrupt Register Summary

3.1 Interrupt Test Control Register

Register address: Base + \$0015

	7	6	5	4	3	2	1	0
R	0	0	0	WRTINT	ADR3	ADR2	ADR1	ADR0
W								
Reset:	0	0	0	0	1	1	1	1


 = Unimplemented or Reserved

Figure 3-2 Interrupt Test Control Register (ITCR)

Read: see individual bit descriptions

Write: see individual bit descriptions

WRTINT — Write to the Interrupt Test Registers

Read: anytime

Write: only in special modes and with I-bit mask and X-bit mask set.

1 = Disconnect the interrupt inputs from the priority decoder and use the values written into the ITEST registers instead.

0 = Disables writes to the test registers; reads of the test registers will return the state of the interrupt inputs.

NOTE: Any interrupts which are pending at the time that WRTINT is set will remain until they are overwritten.

ADR3–ADR0 — Test Register Select Bits

Read: anytime

Write: anytime

These bits determine which test register is selected on a read or write. The hexadecimal value written here will be the same as the upper nibble of the lower byte of the vector selects. That is, an “F” written into ADR3–ADR0 will select vectors \$FFFE–\$FFF0 while a “7” written to ADR3–ADR0 will select vectors \$FF7E–\$FF70.

3.2 Interrupt Test Registers

Register address: Base + \$0016

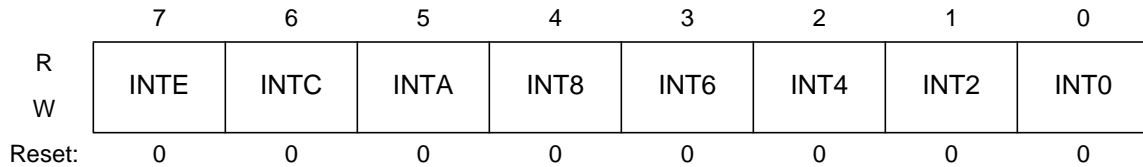


Figure 3-3 Interrupt TEST Registers (ITEST)

Read: Only in special modes. Reads will return either the state of the interrupt inputs of the Interrupt sub-block (WRTINT = 0) or the values written into the TEST registers (WRTINT = 1). Reads will always return zeroes in normal modes.

Write: Only in special modes and with WRTINT = 1 and CCR I mask = 1.

INTE–INT0 — Interrupt TEST Bits

These registers are used in special modes for testing the interrupt logic and priority independent of the system configuration. Each bit is used to force a specific interrupt vector by writing it to a logic one state. Bits are named INTE through INT0 to indicate vectors \$FFxE through \$FFx0. These bits can be written only in special modes and only with the WRTINT bit set (logic one) in the Interrupt Test Control Register (ITCR). In addition, I interrupts must be masked using the I bit in the CCR. In this

state, the interrupt input lines to the Interrupt sub-block will be disconnected and interrupt requests will be generated only by this register. These bits can also be read in special modes to view that an interrupt requested by a system block (such as a peripheral block) has reached the INT module.

There is a test register implemented for every 8 interrupts in the overall system. All of the test registers share the same address and are individually selected using the value stored in the ADR3–ADR0 bits of the Interrupt Test Control Register (ITCR).

NOTE: When ADR3–ADR0 have the value of \$F, only bits 2–0 in the ITEST register will be accessible. That is, vectors higher than \$FFF4 cannot be tested using the test registers and bits 7–3 will always read as a logic zero. If ADR3–ADR0 point to an unimplemented test register, writes will have no effect and reads will always return a logic zero value.

3.3 Highest Priority I Interrupt (Optional)

Register address: Base + \$001F

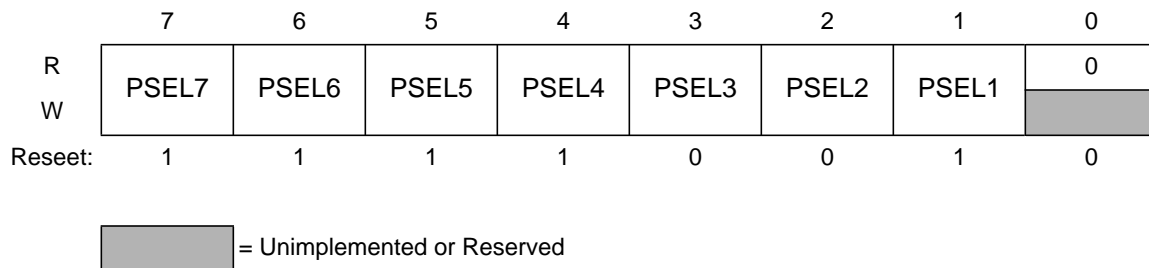


Figure 3-4 Highest Priority I Interrupt Register (HPRIO)

Read: anytime

Write: only if I mask in CCR = 1

PSEL7–PSEL1 — Highest Priority I Interrupt Select Bits

The state of these bits determines which I-bit maskable interrupt will be promoted to highest priority (of the I-bit maskable interrupts). To promote an interrupt, the user writes the least significant byte of the associated interrupt vector address to this register. If an unimplemented vector address or a non I-bit masked vector address (value higher than \$F2) is written, IRQ (\$FFF2) will be the default highest priority interrupt.

Section 4 Functional Description

The Interrupt sub-block processes all exception requests made by the CPU. These exceptions include interrupt vector requests and reset vector requests. Each of these exception types and their overall priority level is discussed in the subsections below.

4.1 Interrupt Exception Requests

As shown in the block diagram in [Figure 1-1](#), the INT contains a register block to provide interrupt status and control, an optional Highest Priority I Interrupt (HPRIO) block, and a priority decoder to evaluate whether pending interrupts are valid and assess their priority.

4.1.1 Interrupt Registers

The INT registers are accessible only in special modes of operation and function as described in [3.1 Interrupt Test Control Register](#) and [3.2 Interrupt Test Registers](#) previously.

4.1.2 Highest Priority I-Bit Maskable Interrupt

When the optional HPRIO block is implemented, the user is allowed to promote a single I-bit maskable interrupt to be the highest priority I interrupt. The HPRIO evaluates all interrupt exception requests and passes the HPRIO vector to the priority decoder if the highest priority I interrupt is active. RTI replaces the promoted interrupt source.

4.1.3 Interrupt Priority Decoder

The priority decoder evaluates all interrupts pending and determines their validity and priority. When the CPU requests an interrupt vector, the decoder will provide the vector for the highest priority interrupt request. Because the vector is not supplied until the CPU requests it, it is possible that a higher priority interrupt request could override the original exception that caused the CPU to request the vector. In this case, the CPU will receive the highest priority vector and the system will process this exception instead of the original request.

NOTE: *Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not get processed.*

If for any reason the interrupt source is unknown (e.g., an interrupt request becomes inactive after the interrupt has been recognized but prior to the vector request), the vector address will default to that of the last valid interrupt that existed during the particular interrupt sequence. If the CPU requests an interrupt vector when there has never been a pending interrupt request, the INT will provide the Software Interrupt (SWI) vector address.

4.2 Reset Exception Requests

The INT supports three system reset exception request types: normal system reset or power-on-reset request, Crystal Monitor reset request, and COP Watchdog reset request. The type of reset exception request must be decoded by the system and the proper request made to the Core. The INT will then provide the service routine address for the type of reset requested.

4.3 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT upon request by the CPU is shown in [Table 4-1](#).

Table 4-1 Exception Vector Map and Priority

Vector Address	Source
\$FFFE-\$FFFF	System reset
\$FFFC-\$FFFD	Crystal monitor reset
\$FFFA-\$FFFB	COP reset
\$FFF8-\$FFF9	Unimplemented opcode trap
\$FFF6-\$FFF7	Software interrupt instruction (SWI) or BDM vector request
\$FFF4-\$FFF5	XIRQ signal
\$FFF2-\$FFF3	IRQ signal
\$FFF0-\$FF00	Device-specific I-bit maskable interrupt sources (priority in descending order)

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

JAPAN:

Motorola Japan Ltd.
SPS, Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.
Silicon Harbour Centre
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

HOME PAGE:

<http://motorola.com/semiconductors>



Information in this document is provided solely to enable system and software implementers to use Motorola products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 consequential or incidental damages. "Typical" parameters that may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

MOTOROLA and the Stylized M Logo are registered in the US Patent and Trademark Office. All other product or service names are the property of their respective owners. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

© Motorola Inc. 2003

S12INTV1/D
Rev. 1.00
5/2003


PIM_9DTB128

Block Guide

V02.05

Original Release Date: 05 FEB 2001
Revised: 02 OCT 2003

Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

©Motorola, Inc., 2001

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
1.0	05 Feb 2001			1st release based on pim_9DP256 rev 2.0
01.01	7 Aug 2001			-Capitalized all pin names to match Marlin DUG -Corrected typo in PPSJ description -added full register names in memory map table
V02.00	01 FEB 2002	01 FEB 2002		Reformatted to SRS 3.0 for HC9S12DT128 (Marlin 2) Corrected mismatches with SoC Guide Added CAN0 reroute to PJ[7:6]
V02.01	19 FEB 2002	19 FEB 2002		Corrected 3.3.3 regarding Byteflight functions on PM[7:4] Updated digital filter descriptions in Fig. 4-2 & Table 4-3 Cleaned up other descriptions including those for Port A, B, E, & K
V02.02	08 MAR 2002	08 MAR 2002		Added document number
V02.03	01 NOV 2002	01 NOV 2002		Non-customer information update Minor change in document layout Added acronyms in Preface
V02.04	03 SEPT 2003	03 SEPT 2003		Clarified external interrupt sources descriptions in section 1.1 Added back GPIO functions to port M in table 2-1
V02.05	02 OCT 2003	02 OCT 2003		Changed to use new document number Updated core references to internal logic & individual block guides

Section 1 Introduction

1.1	Overview	11
1.2	Features	11
1.3	Block Diagram	12

Section 2 External Signal Description

2.1	Overview	13
2.2	Signal properties	13

Section 3 Memory Map/Register Definition

3.1	Overview	18
3.2	Module Memory Map	18
3.3	Register Descriptions	19
3.3.1	Port T Registers	20
3.3.2	Port S Registers	23
3.3.3	Port M Registers	27
3.3.4	Port P Registers	32
3.3.5	Port H Registers	36
3.3.6	Port J Registers	40

Section 4 Functional Description

4.1	General	44
4.1.1	I/O register	44
4.1.2	Input register	45
4.1.3	Data direction register	45
4.1.4	Reduced drive register	45
4.1.5	Pull device enable register	45
4.1.6	Polarity select register	46
4.2	Port T	46
4.3	Port S	46
4.4	Port M	46
4.4.1	Module Routing Register	46
4.5	Port P	47
4.6	Port H	49
4.7	Port J	49
4.8	Port A, B, E, K, and BKGD pin	49

4.9 External Pin Descriptions 49

4.10 Low Power Options 50

4.10.1 Run Mode 50

4.10.2 Wait Mode 50

4.10.3 Stop Mode 50

Section 5 Initialization/Application Information

5.1 General 51

5.2 Reset Initialization 51

Section 6 Interrupts

6.1 General 52

6.2 Interrupt Sources 52

6.3 Recovery from STOP 52

Figure 1-1	PIM_9DTB128 Block Diagram	12
Figure 3-1	Port T I/O Register (PTT)	20
Figure 3-2	Port T Input Register (PTIT)	21
Figure 3-3	Port T Data Direction Register (DDRT)	21
Figure 3-4	Port T Reduced Drive Register (RDRT)	22
Figure 3-5	Port T Pull Device Enable Register (PERT)	22
Figure 3-6	Port T Polarity Select Register (PPST)	23
Figure 3-7	Port S I/O Register (PTS)	23
Figure 3-8	Port S Input Register (PTIS)	24
Figure 3-9	Port S Data Direction Register (DDRS)	24
Figure 3-10	Port S Reduced Drive Register (RDRS)	25
Figure 3-11	Port S Pull Device Enable Register (PERS)	25
Figure 3-12	Port S Polarity Select Register (PPSS)	26
Figure 3-13	Port S Wired-Or Mode Register (WOMS)	26
Figure 3-14	Port M I/O Register (PTM)	27
Figure 3-15	Port M Input Register (PTIM)	28
Figure 3-16	Port M Data Direction Register (DDRM)	28
Figure 3-17	Port M Reduced Drive Register (RDRM)	29
Figure 3-18	Port M Pull Device Enable Register (PERM)	29
Figure 3-19	Port M Polarity Select Register (PPSM)	30
Figure 3-20	Port M Wired-Or Mode Register (WOMM)	30
Figure 3-21	Module Routing Register (MODRR)	31
Figure 3-22	Port P I/O Register (PTP)	32
Figure 3-23	Port P Input Register (PTIP)	32
Figure 3-24	Port P Data Direction Register (DDRP)	33
Figure 3-25	Port P Reduced Drive Register (RDRP)	33
Figure 3-26	Port P Pull Device Enable Register (PERP)	34
Figure 3-27	Port P Polarity Select Register (PPSP)	34
Figure 3-28	Port P Interrupt Enable Register (PIEP)	35
Figure 3-29	Port P Interrupt Flag Register (PIFP)	35
Figure 3-30	Port H I/O Register (PTH)	36
Figure 3-31	Port H Input Register (PTIH)	36
Figure 3-32	Port H Data Direction Register (DDRH)	37
Figure 3-33	Port H Reduced Drive Register (RDRH)	37
Figure 3-34	Port H Pull Device Enable Register (PERH)	38
Figure 3-35	Port H Polarity Select Register (PPSH)	38

Figure 3-36 Port H Interrupt Enable Register (PIEH) 39

Figure 3-37 Port H Interrupt Flag Register (PIFH) 39

Figure 3-38 Port J I/O Register (PTJ) 40

Figure 3-39 Port J Input Register (PTIJ) 40

Figure 3-40 Port J Data Direction Register (DDRJ) 41

Figure 3-41 Port J Reduced Drive Register (RDRJ) 41

Figure 3-42 Port J Pull Device Enable Register (PERJ) 42

Figure 3-43 Port J Polarity Select Register (PPSJ) 42

Figure 3-44 Port J Interrupt Enable Register (PIEJ) 43

Figure 3-45 Port J Interrupt Flag Register (PIFJ) 43

Figure 4-1 Illustration of I/O pin functionality. 45

Figure 4-2 Interrupt Glitch Filter on Port P, H and J (PPS=0) 48

Figure 4-3 Pulse Illustration 48

Table 2-1	Signal Properties	13
Table 3-1	PIM_9DTB128 Memory Map	18
Table 3-2	Pin Configuration Summary	20
Table 3-3	CAN0 Routing	31
Table 3-4	CAN4 Routing	31
Table 3-5	SPI0 Routing	31
Table 3-6	SPI1 Routing	32
Table 4-1	Summary of Functional Priority	44
Table 4-2	Implemented modules on derivatives	47
Table 4-3	Pulse Detection Criteria	48
Table 5-1	Port Reset State Summary	51
Table 6-1	Port Integration Module Interrupt Sources	52

Preface

Terminology

Acronyms and Abbreviations	
BDLC	Byte Level Data Link Controller
CAN	Controller Area Network
IIC	Inter-Integrated Circuit
PIM	Port Integration Module
PTI	Pad Test Interface
PWM	Pulse Width Modulator
SCI	Serial Communication Interface
SPI	Serial Peripheral Interface

Section 1 Introduction

1.1 Overview

The Port Integration Module (PIM) establishes the interface between the peripheral modules and the I/O pins for all ports except AD0 and AD1.

NOTE: *Port A, B, E, and K are related to the internal logic and the multiplexed external bus interface (MEBI). Many of these port/pads logic come from the MEBI module and pass through the PIM module. Brief functional descriptions of these ports are provided for completeness. Refer to HCS12 Multiplexed External Bus Interface Block Guide for details.*

This section covers:

- Port T connected to the timer module
- The serial port S associated with 2 SCI and 1 SPI module
- Port M associated with 3 CAN, 1 Byteflight and 1 BDLC module
- Port P connected to the PWM and 1 SPI module, which also can be used as an external interrupt source
- The standard I/O port H associated with SPI1
- Port J associated with CAN0, 4 and the IIC interface. Ports P, H, and J can also be used as external interrupt sources.

Each I/O pin can be configured by several registers in order to select data direction and drive strength, to enable and select pull-up or pull-down resistors. On certain pins also interrupts can be enabled which result in status flags.

The I/O's of 2 CAN and all 2 SPI modules can be routed from their default location to determined pins.

The implementation of the Port Integration Module is device dependent.

1.2 Features

A standard port has the following minimum features:

- Input/output selection
- 5V output drive with two selectable drive strength
- 5V digital and analog input
- Input with selectable pull-up or pull-down device

Optional features:

- Open drain for wired-or connections

- Interrupt inputs with glitch filtering

1.3 Block Diagram

Figure 1-1 is a block diagram of the PIM_9DTB128.

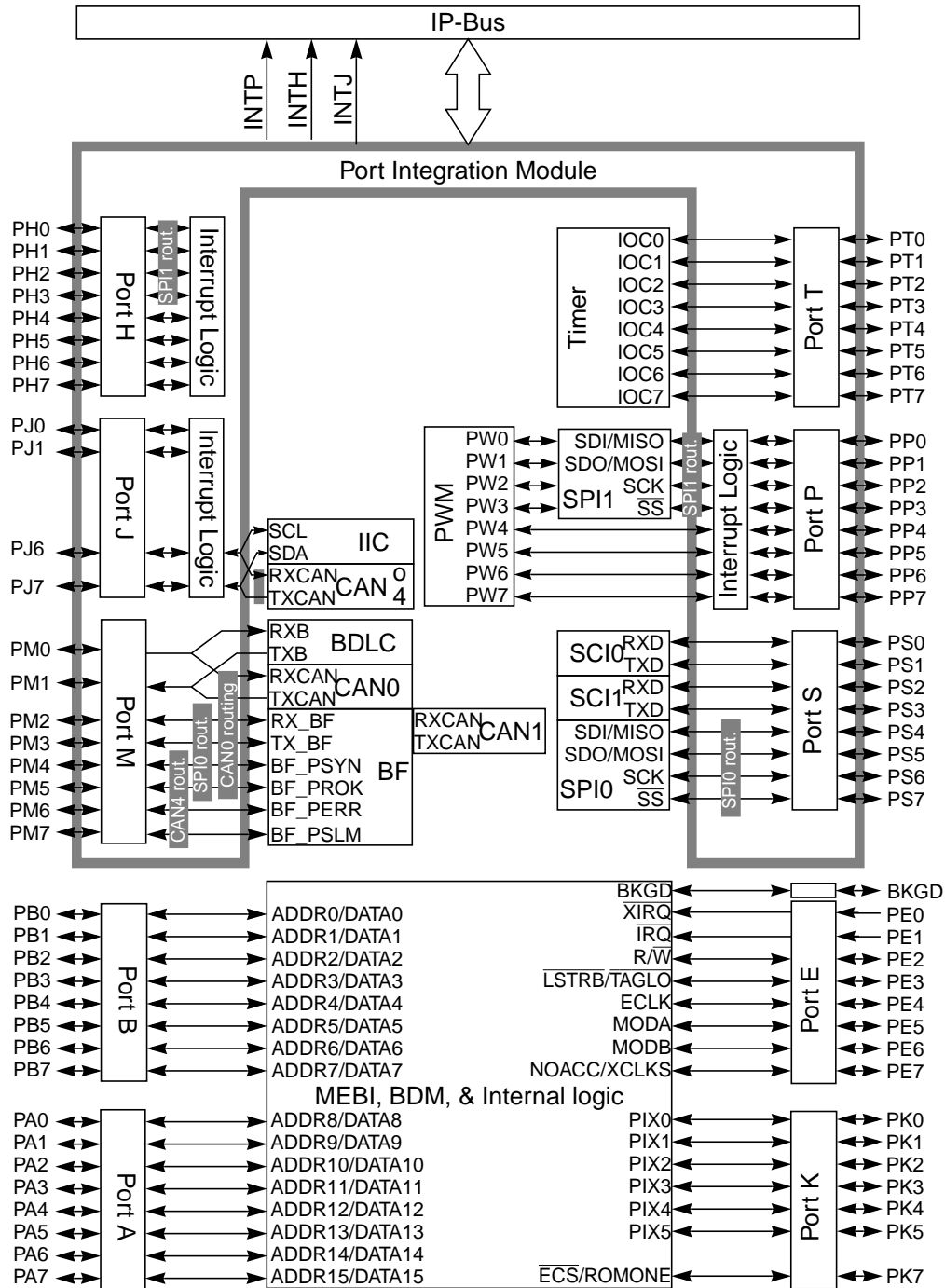


Figure 1-1 PIM_9DTB128 Block Diagram

Section 2 External Signal Description

2.1 Overview

This section lists and describes the signals that do connect off chip.

NOTE: Refer to the *Creation Guide* and the *Integration Guide* documentation of the PIM_9DTB128 for a detailed description of the pad control signals.

2.2 Signal properties

Table 2-1 shows all the pins and their functions that are controlled by the PIM_9DTB128. If there is more than one function associated with a pin, the priority is indicated by the position in the table from top (highest priority) to down (lowest priority).

Table 2-1 Signal Properties

Port	Pin Name	Pin Function & Priority	Description	Reset State	Pull Mode
Port T	PT[7:0]	IOC[7:0]	Enhanced Capture Timer Channel 7 to 0	input	hiz
		GPIO	General-purpose I/O		
Port S	PS7	$\overline{SS0}$	Serial Peripheral Interface 0 slave select output in master mode, input in slave mode or master mode.	input	pull-up
		GPIO	General-purpose I/O		
	PS6	SCK0	Serial Peripheral Interface 0 serial clock pin		
		GPIO	General-purpose I/O		
	PS5	MOSI0	Serial Peripheral Interface 0 master out/slave in pin		
		GPIO	General-purpose I/O		
	PS4	MISO0	Serial Peripheral Interface 0 master in/slave out pin		
		GPIO	General-purpose I/O		
	PS3	TXD1	Serial Communication Interface 1 transmit pin		
		GPIO	General-purpose I/O		
	PS2	RXD1	Serial Communication Interface 1 receive pin		
		GPIO	General-purpose I/O		
	PS1	TXD0	Serial Communication Interface 0 transmit pin		
		GPIO	General-purpose I/O		
	PS0	RXD0	Serial Communication Interface 0 receive pin		
		GPIO	General-purpose I/O		

Port	Pin Name	Pin Function & Priority	Description	Reset State	Pull Mode
Port M	PM7	BF_PSLM	Byteflight Status	Input	hiz
		TXCAN4	MSCAN4 transmit pin		
		GPIO	General-purpose I/O		
	PM6	BF_PERR	Byteflight Error Status		
		RXCAN4	MSCAN4 receive pin		
		GPIO	General-purpose I/O		
	PM5	BF_PROK	Byteflight Status		
		TXCAN0	MSCAN0 transmit pin		
		TXCAN4	MSCAN4 transmit pin		
		SCK0	Serial Peripheral Interface 0 serial clock pin		
		GPIO	General-purpose I/O		
	PM4	BF_PSYN	Byteflight Status		
		RXCAN0	MSCAN0 receive pin		
		RXCAN4	MSCAN4 receive pin		
		MOSI0	Serial Peripheral Interface 0 master out/slave in pin		
	PM3	GPIO	General-purpose I/O		
		TX_BF	Byteflight transmit pin		
		TXCAN1	MSCAN1 transmit pin		
		TXCAN0	MSCAN0 transmit pin		
		$\overline{SS}0^1$	Serial Peripheral Interface 0 slave select output in master mode, input for slave mode or master mode		
	PM2	GPIO	General-purpose I/O		
		RX_BF	Byteflight receive pin		
		RXCAN1	MSCAN1 receive pin		
		RXCAN0	MSCAN0 receive pin		
MISO0 ¹		Serial Peripheral Interface 0 master in/slave out pin			
PM1	GPIO	General-purpose I/O			
	TXCAN0	MSCAN0 transmit pin			
	TXB	BDLC transmit pi			
PM0	GPIO	General-purpose I/O			
	RXCAN0	MSCAN0 receive pin			
	RXB	BDLC receive pin			
		GPIO	General-purpose I/O		

Port	Pin Name	Pin Function & Priority	Description	Reset State	Pull Mode
Port P	PP7	PWM7	Pulse Width Modulator channel 7	input	hiz
		GPIO/KWP7	General-purpose I/O with interrupt		
	PP6	PWM6	Pulse Width Modulator channel 6		
		GPIO/KWP6	General-purpose I/O with interrupt		
	PP5	PWM5	Pulse Width Modulator channel 5		
		GPIO/KWP5	General-purpose I/O with interrupt		
	PP4	PWM4	Pulse Width Modulator channel 4		
		GPIO/KWP4	General-purpose I/O with interrupt		
	PP3	PWM3	Pulse Width Modulator channel 3		
		$\overline{SS1}$	Serial Peripheral Interface 1 slave select output in master mode, input for slave mode or master mode		
	PP2	GPIO/KWP3	General-purpose I/O with interrupt		
		PWM2	Pulse Width Modulator channel 2		
		SCK1	Serial Peripheral Interface 1 serial clock pin		
	PP1	GPIO/KWP2	General-purpose I/O with interrupt		
		PWM1	Pulse Width Modulator channel 1		
MOSI1		Serial Peripheral Interface 1 master out/slave in pin			
PP0	GPIO/KWP1	General-purpose I/O with interrupt			
	PWM0	Pulse Width Modulator channel 0			
	MISO1	Serial Peripheral Interface 1 master in/slave out pin			
Port H	PH7	PWM0	General-purpose I/O with interrupt		
		GPIO/KWP0	General-purpose I/O with interrupt		
	PH6	PWM0	General Purpose I/O and interrupt		
		GPIO/KWP0	General Purpose I/O and interrupt		
	PH5	PWM0	General Purpose I/O and interrupt		
		GPIO/KWP0	General Purpose I/O and interrupt		
	PH4	PWM0	General Purpose I/O and interrupt		
		GPIO/KWP0	General Purpose I/O and interrupt		
	PH3	$\overline{SS1}$	Serial Peripheral Interface 1 slave select output in master mode, input for slave mode or master mode		
		GPIO/KWH3	General Purpose I/O and interrupt		
PH2	SCK1	Serial Peripheral Interface 1 serial clock pin			
	GPIO/KWH2	General Purpose I/O and interrupt			
PH1	MOSI1	Serial Peripheral Interface 1 master out/slave in pin			
	GPIO/KWH1	General Purpose I/O and interrupt			
PH0	MISO1	Serial Peripheral Interface 1 master in/slave out pin			
	GPIO/KWH0	General Purpose I/O and interrupt			

Port	Pin Name	Pin Function & Priority	Description	Reset State	Pull Mode
Port J	PJ7	TXCAN4	MSCAN4 transmit pin	input	pull-up
		SCL	Inter Integrated Circuit serial clock line		
		TXCAN0	MSCAN0 transmit pin		
		GPIO/KWJ7	General-purpose I/O with interrupt		
	PJ6	RXCAN4	MSCAN4 receive pin		
		SDA	Inter Integrated Circuit serial data line		
		RXCAN0	MSCAN0 receive pin		
		GPIO/KWJ6	General-purpose I/O with interrupt		
PJ[1:0]	GPIO/KWJ[1:0]	General Purpose I/O and interrupt			
Port A ²	PA[7:0]	ADDR[15:8]/DATA[15:8]/GPIO	External bus pins share function with general-purpose I/O ports A. In single chip modes, the pins can be used for general-purpose I/O. In expanded modes, the pins are used for the external buses.	See Note 2	See Note 2
Port B ²	PB[7:0]	ADDR[7:0]/DATA[7:0]/GPIO	External bus pins share function with general-purpose I/O port B. In single chip modes, the pins can be used for general-purpose I/O. In expanded modes, the pins are used for the external address and data buses.	See Note 2	See Note 2
Port E ²	PE7	$\overline{\text{XCLKS}}$ /NOACC/GPIO	No Access. Indicates free cycles in expanded mode. Selects also external clock or oscillator during reset. Can be used as general purpose I/O pin.	See Note 2	See Note 2
	PE6	MODB/IPIPE1/GPIO	State of mode select pins during reset determine the initial operating mode of the MCU. After reset, MODB and MODA can be configured as instruction queue tracking signals IPIPE1 and IPIPE0 or as general-purpose I/O pins.		
	PE5	MODA/IPIPE0/GPIO			
	PE4	ECLK/GPIO	E Clock is the output connection for the external bus clock. ECLK is used as a timing reference and for address demultiplexing.		
	PE3	$\overline{\text{LSTRB}}$ /TAGLO/GPIO	Low byte strobe (0 = low byte valid), in all modes this pin can be used as I/O. The low strobe function is the exclusive-NOR of A0 and the internal $\overline{\text{SZ8}}$ signal. (The $\overline{\text{SZ8}}$ internal signal indicates the size 16/8 access.) Pin function TAGLO used in instruction low byte tagging.		
	PE2	R/W/GPIO	Indicates direction of data on expansion bus. Shares function with general-purpose I/O. Read/write in expanded modes.		
	PE1	$\overline{\text{IRQ}}$ /GPI	Maskable interrupt request input provides a means of applying asynchronous interrupt requests to the MCU. Either falling edge-sensitive triggering or level-sensitive triggering is program selectable (INTCR register).		
	PE0	$\overline{\text{XIRQ}}$ /GPI	The $\overline{\text{XIRQ}}$ input provides a means of requesting a nonmaskable interrupt after reset initialization. Because it is level sensitive, it can be connected to a multiple-source wired-OR network.		

Port	Pin Name	Pin Function & Priority	Description	Reset State	Pull Mode
Port K ²	PK7	$\overline{\text{ECS}}$ /ROMONE/GPIO	Emulation Chip select/ROMONE function	See Note 2	See Note 2
	PK[5:0]	XADDR[19:14]/GPIO	Expanded Addresses		
See Notes 3	BKGD ³	MODC/ $\overline{\text{TAGHI}}$ /BKGD	Pseudo_open_drain communication pin for the single-wire background debug mode. At the rising edge on RESET, the state of this pin is latched into the MODC bit to set the mode. When instruction tagging is on, a 0 at the falling edge of E tags the high half of the instruction word being read into the instruction queue.	See Note 3	See Note 3

NOTES:

1. If CAN0 is routed to PM[3:2] the SPI0 can still be used in bidirectional master mode. Refer to SPI Block Guide for details.
2. Refer to HCS12 Multiplexed External Bus Interface Block Guide for details.
3. Refer to HCS12 Background Debug Module Block Guide for details.

Section 3 Memory Map/Register Definition

3.1 Overview

This section provides a detailed description of all registers.

3.2 Module Memory Map

Table 3-1 shows the register map of the Port Integration Module.

Table 3-1 PIM_9DTB128 Memory Map

Address offset	Use	Access
\$00	Port T I/O Register (PTT)	RW
\$01	Port T Input Register (PTIT)	R
\$02	Port T Data Direction Register (DDRT)	RW
\$03	Port T Reduced Drive Register (RDRT)	RW
\$04	Port T Pull Device Enable Register (PERT)	RW
\$05	Port T Polarity Select Register (PPST)	RW
\$06	Reserved	-
\$07	Reserved	-
\$08	Port S I/O Register (PTS)	RW
\$09	Port S Input Register (PTIS)	R
\$0A	Port S Data Direction Register (DDRS)	RW
\$0B	Port S Reduced Drive Register (RDRS)	RW
\$0C	Port S Pull Device Enable Register (PERS)	RW
\$0D	Port S Polarity Select Register (PPSS)	RW
\$0E	Port S Wired-Or Mode Register (WOMS)	RW
\$0F	Reserved	-
\$10	Port M I/O Register (PTM)	RW
\$11	Port M Input Register (PTIM)	R
\$12	Port M Data Direction Register (DDRM)	RW
\$13	Port M Reduced Drive Register (RDRM)	RW
\$14	Port M Pull Device Enable Register (PERM)	RW
\$15	Port M Polarity Select Register (PPSM)	RW
\$16	Port M Wired-Or Mode Register (WOMM)	RW
\$17	Module Routing Register (MODRR)	RW
\$18	Port P I/O Register (PTP)	RW
\$19	Port P Input Register (PTIP)	R
\$1A	Port P Data Direction Register (DDRP)	RW
\$1B	Port P Reduced Drive Register (RDRP)	RW
\$1C	Port P Pull Device Enable Register (PERP)	RW
\$1D	Port P Polarity Select Register (PPSP)	RW
\$1E	Port P Interrupt Enable Register (PIEP)	RW
\$1F	Port P Interrupt Flag Register (PIFP)	RW
\$20	Port H I/O Register (PTH)	RW

\$21	Port H Input Register (PTIH)	R
\$22	Port H Data Direction Register (DDRH)	RW
\$23	Port H Reduced Drive Register (RDRH)	RW
\$24	Port H Pull Device Enable Register (PERH)	RW
\$25	Port H Polarity Select Register (PPSH)	RW
\$26	Port H Interrupt Enable Register (PIEH)	RW
\$27	Port H Interrupt Flag Register (PIFH)	RW
\$28	Port J I/O Register (PTJ)	RW ¹
\$29	Port J Input Register (PTIJ)	R
\$2A	Port J Data Direction Register (DDRJ)	RW ¹
\$2B	Port J Reduced Drive Register (RDRJ)	RW ¹
\$2C	Port J Pull Device Enable Register (PERJ)	RW ¹
\$2D	Port J Polarity Select Register (PPSJ)	RW ¹
\$2E	Port J Interrupt Enable Register (PIEJ)	RW ¹
\$2F	Port J Interrupt Flag Register (PIFJ)	RW ¹
\$30 – \$3F	Reserved	-

NOTES:

1. Write access not applicable for one or more register bits. Please refer to detailed signal description.

NOTE: *Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.*

3.3 Register Descriptions

The following table summarizes the effect on the various configuration bits, data direction (DDR), output level (I/O), reduced drive (RDR), pull enable (PE), pull select (PS) and interrupt enable (IE) for the ports. The configuration bit PS is used for two purposes:

1. Configure the sensitive interrupt edge (rising or falling), if interrupt is enabled.
2. Select either a pull-up or pull-down device if PE is active.

Table 3-2 Pin Configuration Summary

DDR	IO	RDR	PE	PS	IE ¹	Function	Pull Device	Interrupt
0	X	X	0	X	0	Input	Disabled	Disabled
0	X	X	1	0	0	Input	Pull Up	Disabled
0	X	X	1	1	0	Input	Pull Down	Disabled
0	X	X	0	0	1	Input	Disabled	falling edge
0	X	X	0	1	1	Input	Disabled	rising edge
0	X	X	1	0	1	Input	Pull Up	falling edge
0	X	X	1	1	1	Input	Pull Down	rising edge
1	0	0	X	X	0	Output, full drive to 0	Disabled	Disabled
1	1	0	X	X	0	Output, full drive to 1	Disabled	Disabled
1	0	1	X	X	0	Output, reduced drive to 0	Disabled	Disabled
1	1	1	X	X	0	Output, reduced drive to 1	Disabled	Disabled
1	0	0	X	0	1	Output, full drive to 0	Disabled	falling edge
1	1	0	X	1	1	Output, full drive to 1	Disabled	rising edge
1	0	1	X	0	1	Output, reduced drive to 0	Disabled	falling edge
1	1	1	X	1	1	Output, reduced drive to 1	Disabled	rising edge

NOTES:

1. Applicable only on port P, H and J.

NOTE: All bits of all registers in this module are completely synchronous to internal clocks during a register read.

3.3.1 Port T Registers

Address Offset: \$__00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
Write:								
ECT:	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-1 Port T I/O Register (PTT)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

Address Offset: \$__01

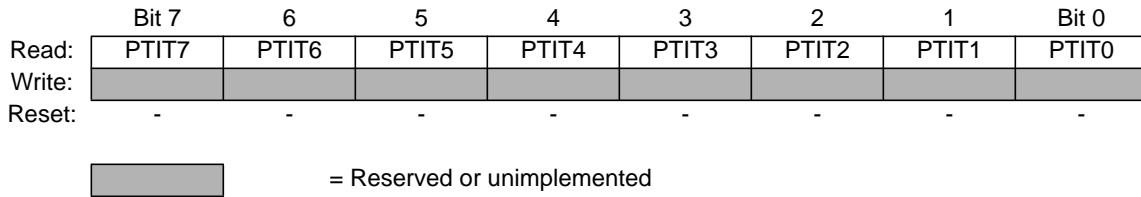


Figure 3-2 Port T Input Register (PTIT)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

Address Offset: \$__02

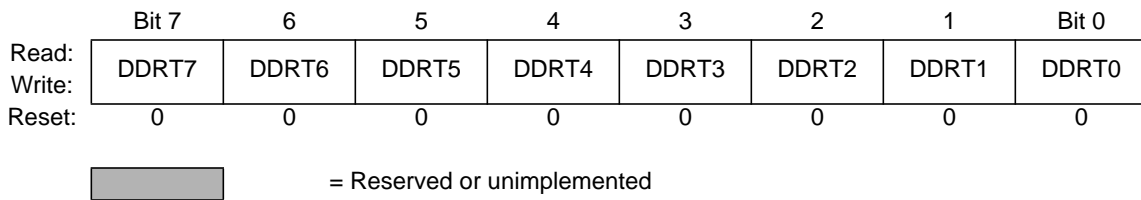


Figure 3-3 Port T Data Direction Register (DDRT)

Read:Anytime.

Write:Anytime.

This register configures each port T pin as either input or output.

The ECT forces the I/O state to be an output for each timer port associated with an enabled output compare. In these cases the data direction bits will not change.

The DDRT bits revert to controlling the I/O direction of a pin when the associated timer output compare is disabled.

The timer input capture always monitors the state of the pin.

DDRT[7:0] — Data Direction Port T

1 = Associated pin is configured as output.

0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTT or PTIT registers, when changing the DDRT register.

Address Offset: \$__03

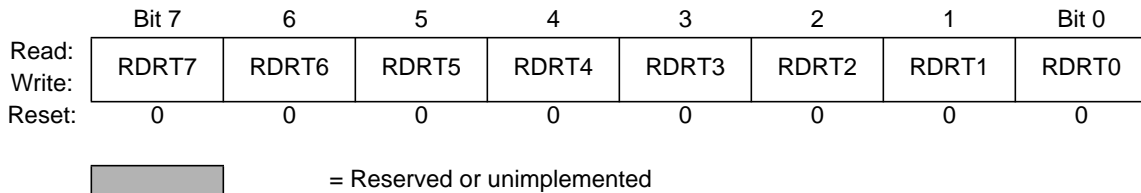


Figure 3-4 Port T Reduced Drive Register (RDRT)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port T output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRT[7:0] — Reduced Drive Port T

1 = Associated pin drives at about 1/6 of the full drive strength.

0 = Full drive strength at output.

Address Offset: \$__04

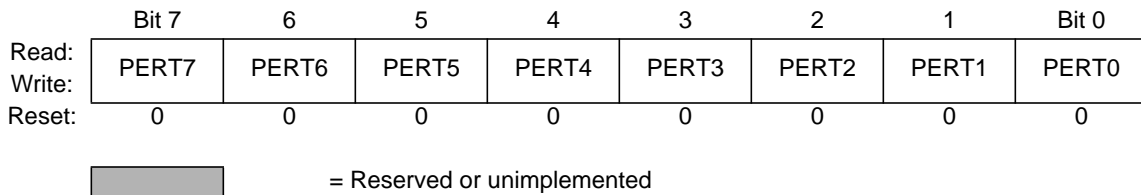


Figure 3-5 Port T Pull Device Enable Register (PERT)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. This bit has no effect if the port is used as output. Out of reset no pull device is enabled.

PERT[7:0] — Pull Device Enable Port T

1 = Either a pull-up or pull-down device is enabled.

0 = Pull-up or pull-down device is disabled.

Address Offset: \$__05

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-6 Port T Polarity Select Register (PPST)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPST[7:0] — Pull Select Port T

1 = A pull-down device is connected to the associated port T pin, if enabled by the associated bit in register PERT and if the port is used as input.

0 = A pull-up device is connected to the associated port T pin, if enabled by the associated bit in register PERT and if the port is used as input.

3.3.2 Port S Registers

Address Offset: \$__08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
Write:								
SPI/SCI	SS0	SCK0	MOSI0	MISO0	TXD1	RXD1	TXD0	RXD0
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-7 Port S I/O Register (PTS)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The SPI pins (PS[7:4]) configuration is determined by several status bits in the SPI module. *See chapter SPI for details.*

The SCI ports associated with transmit pins 3 and 1 are configured as outputs if the transmitter is enabled. The SCI pins associated with receive pins 2 and 0 are configured as inputs if the receiver is enabled. *See chapter SCI for details.*

Address Offset: \$__09

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
Write:								
Reset:	-	-	-	-	-	-	-	-

 = Reserved or unimplemented

Figure 3-8 Port S Input Register (PTIS)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This also can be used to detect overload or short circuit conditions on output pins.

Address Offset:\$__0A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-9 Port S Data Direction Register (DDRS)

Read:Anytime.

Write:Anytime.

This register configures each port S pin as either input or output

If SPI is enabled, the SPI determines the pin direction. *For details see SPI specification.*

If the associated SCI transmit or receive channel is enabled this register has no effect on the pins. The pin is forced to be an output if a SCI transmit channel is enabled, it is forced to be an input if the SCI receive channel is enabled.

The DDRS bits revert to controlling the I/O direction of a pin when the associated channel is disabled.

DDRS[7:0] — Data Direction Port S

1 = Associated pin is configured as output.

0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTS or PTIS registers, when changing the DDRS register.

Address Offset: \$__0B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
Write:								
Reset:	0	0	0	0	0	0	0	0


 = Reserved or unimplemented

Figure 3-10 Port S Reduced Drive Register (RDRS)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port S output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRS[7:0] — Reduced Drive Port S

1 = Associated pin drives at about 1/6 of the full drive strength.

0 = Full drive strength at output.

Address Offset: \$__0C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
Write:								
Reset:	1	1	1	1	1	1	1	1

 = Reserved or unimplemented

Figure 3-11 Port S Pull Device Enable Register (PERS)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input or as output in wired-or (open drain) mode. This bit has no effect if the port is used as push-pull output. Out of reset a pull-up device is enabled.

PERS[7:0] — Pull Device Enable Port S

1 = Either a pull-up or pull-down device is enabled.

0 = Pull-up or pull-down device is disabled.

Address Offset: \$__0D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-12 Port S Polarity Select Register (PPSS)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin.

PPSS[7:0] — Pull Select Port S

- 1 = A pull-down device is connected to the associated port S pin, if enabled by the associated bit in register PERS and if the port is used as input.
- 0 = A pull-up device is connected to the associated port S pin, if enabled by the associated bit in register PERS and if the port is used as input or as wired-or output.

Address Offset: \$__0E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-13 Port S Wired-Or Mode Register (WOMS)

Read:Anytime.

Write:Anytime.

This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. It applies also to the SPI and SCI outputs and allows a multipoint connection of several serial modules. This bit has no influence on pins used as inputs.

WOMS[7:0] — Wired-Or Mode Port S

- 1 = Output buffers operate as open-drain outputs.
- 0 = Output buffers operate as push-pull outputs.

3.3.3 Port M Registers

Address Offset: \$__10

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTM7	PTM6	PTM5	PTM4	PTM3	PTM2	PTM1	PTM0
Write:								
BF:	BF_PSLM	BF_PERR	BF_PROK	BF_PSYN	TX_BF	RX_BF		
CAN:	TXCAN4	RXCAN4	TXCAN0	RXCAN0	TXCAN1	RXCAN1	TXCAN0	RXCAN0
BDLC							TXB	RXB
CAN0					TXCAN0	RXCAN0		
CAN4			TXCAN4	RXCAN4				
SPI0			SCK0	MOSI0	SS0	MISO0		
Reset	0	0	0	0	0	0	0	0


 = Reserved or unimplemented

Figure 3-14 Port M I/O Register (PTM)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

PM[7:6]

The Byteflight function (BF_PSLM and BF_PERR) takes precedence over the CAN4 and the general purpose I/O function if the Byteflight module and the associated output function are enabled. *Refer to Byteflight Block Guide for details.*

The CAN4 function (TXCAN4 and RXCAN4) takes precedence over the general purpose I/O function if the CAN4 module is enabled. *Refer to MSCAN Block Guide for details.*

PM[5:4]

The Byteflight function (BF_PROK and BF_PSYN) takes precedence over the CAN0, CAN4, the SPI0 and the general purpose I/O function if the Byteflight module and the associated output function are enabled. *Refer to Byteflight Block Guide for details.*

The CAN0 function (TXCAN0 and RXCAN0) takes precedence over the CAN4, the SPI0 and the general purpose I/O function if the CAN0 module is enabled. *Refer to MSCAN Block Guide for details.*

The CAN4 function (TXCAN4 and RXCAN4) takes precedence over the SPI0 and general purpose I/O function if the CAN4 module is enabled. *Refer to MSCAN Block Guide for details.*

The SPI0 function (SCK0 and MOSI0) takes precedence of the general purpose I/O function if the SPI0 is enabled. *Refer to SPI Block Guide for details.*

PM[3:2]

The Byteflight function (TX_BF and RX_BF) takes precedence over the CAN1, CAN0, the SPI0 and the general purpose I/O function if the Byteflight module is enabled. Refer to *Byteflight Block Guide for details*.

The CAN1 function (TXCAN1 and RXCAN1) takes precedence over the CAN0, the SPI0 and the general purpose I/O function if the CAN1 module is enabled. Refer to *MSCAN Block Guide for details*.

The SPI0 function ($\overline{SS0}$ and MISO0) takes precedence of the general purpose I/O function if the SPI0 is enabled. Refer to *SPI Block Guide for details*.

PM[1:0]

The CAN0 function (TXCAN0 and RXCAN0) takes precedence over the BDLC and the general purpose I/O function if the CAN0 module is enabled. Refer to *MSCAN Block Guide for details*.

The BDLC function (TXB and RXB) takes precedence over the general purpose I/O function associated if enabled. Refer to *BDLC Block Guide for details*.

Address Offset: \$__11

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIM7	PTIM6	PTIM5	PTIM4	PTIM3	PTIM2	PTIM1	PTIM0
Write:								
Reset:	-	-	-	-	-	-	-	-


 = Reserved or unimplemented

Figure 3-15 Port M Input Register (PTIM)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

Address Offset: \$__12

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRM7	DDRM6	DDRM5	DDRM4	DDRM3	DDRM2	DDRM1	DDRM0
Write:								
Reset:	0	0	0	0	0	0	0	0


 = Reserved or unimplemented

Figure 3-16 Port M Data Direction Register (DDRM)

Read:Anytime.

Write:Anytime.

This register configures each port M pin as either input or output.

The Byteflight/CAN/BDLC forces the I/O state to be an output for each port line associated with an

enabled output (TX_BF, BF_PSYN, BF_PROK, BF_PERR, BF_SLM, TXCAN[4,1:0], TXB). It also forces the I/O state to be an input for each port line associated with an enabled input (RX_BF, RXCAN[4,1:0], RXB). In those cases the data direction bits will not change.

The DDRM bits revert to controlling the I/O direction of a pin when the associated peripheral module is disabled.

DDRM[7:0] — Data Direction Port M

- 1 = Associated pin is configured as output.
- 0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTM or PTIM registers, when changing the DDRM register.

Address Offset: \$__13

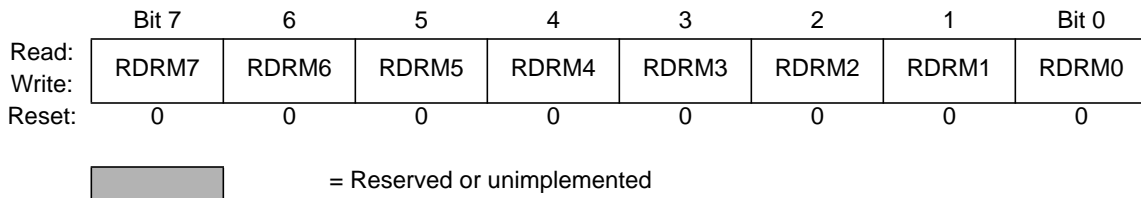


Figure 3-17 Port M Reduced Drive Register (RDRM)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port M output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRM[7:0] — Reduced Drive Port M

- 1 = Associated pin drives at about 1/6 of the full drive strength.
- 0 = Full drive strength at output.

Address Offset: \$__14

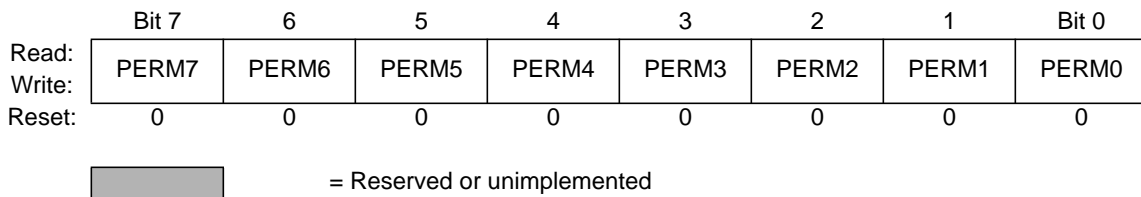


Figure 3-18 Port M Pull Device Enable Register (PERM)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input or wired-or output. This bit has no effect if the port is used as push-pull output. Out of reset no pull device is enabled.

PERM[7:0] — Pull Device Enable Port M

- 1 = Either a pull-up or pull-down device is enabled.
- 0 = Pull-up or pull-down device is disabled.

Address Offset: \$__15

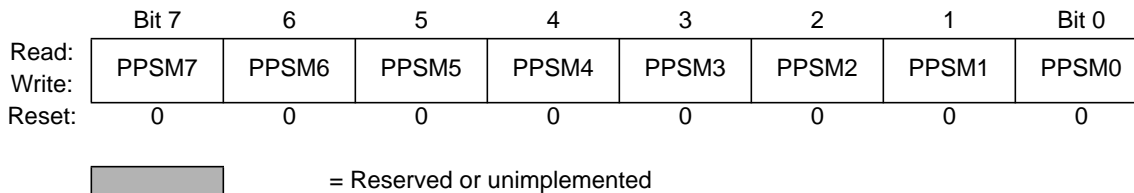


Figure 3-19 Port M Polarity Select Register (PPSM)

Read:Anytime.

Write:Anytime.

This register selects whether a pull-down or a pull-up device is connected to the pin. If Byteflight is active a pull-up device can be activated on the RX_BF input but not a pull-down; if CAN is active a pull-up device can be activated on the RXCAN[4,1:0] inputs, but not a pull-down. If BDLC is active a pull-down device can be activated on the RXB pin but not a pull-up.

PPSM[7:0] — Pull Select Port M

- 1 = A pull-down device is connected to the associated port M pin, if enabled by the associated bit in register PERM and if the port is used as a general purpose or BDLC input but not as RXCAN.
- 0 = A pull-up device is connected to the associated port M pin, if enabled by the associated bit in register PERM and if the port is used as general purpose, Byteflight or RXCAN input but not as BDLC.

Address Offset: \$__16

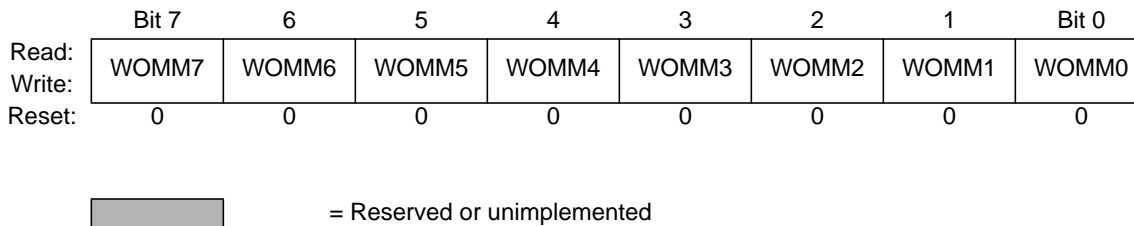


Figure 3-20 Port M Wired-Or Mode Register (WOMM)

Read:Anytime.

Write:Anytime.

This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. It applies also to the Byteflight, CAN and BDLC outputs and allows a multipoint connection of several serial modules. This bit has no influence on pins used as inputs.

WOMM[7:0] — Wired-Or Mode Port M

1 = Output buffers operate as open-drain outputs.

0 = Output buffers operate as push-pull outputs.

Address Offset: \$ _17

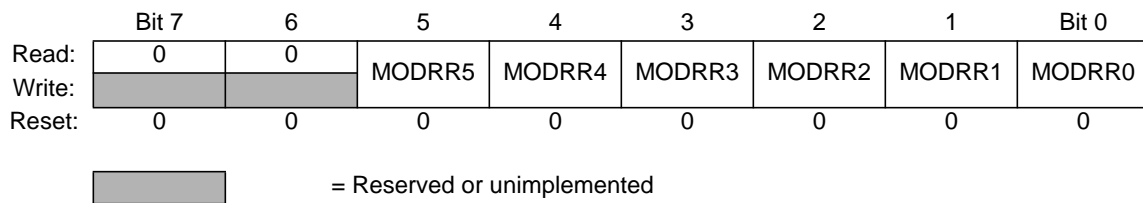


Figure 3-21 Module Routing Register (MODRR)

Read:Anytime.

Write:Anytime.

This register configures the re-routing of CAN0, CAN4, SPI0 and SPI1 on defined port pins.

MODRR[1:0] — CAN0 Routing

Table 3-3 CAN0 Routing

MODRR[1]	MODRR[0]	RXCAN0	TXCAN0
0	0	PM0	PM1
0	1	PM2	PM3
1	0	PM4	PM5
1	1	PJ6	PJ7

MODRR[3:2] — CAN4 Routing

Table 3-4 CAN4 Routing

MODRR[3]	MODRR[2]	RXCAN4	TXCAN4
0	0	PJ6	PJ7
0	1	PM4	PM5
1	0	PM6	PM7
1	1	Reserved	

MODRR[4] — SPI0 Routing

Table 3-5 SPI0 Routing

MODRR[4]	MISO0	MOSI0	SCK0	SS0
0	PS4	PS5	PS6	PS7
1	PM2	PM4	PM5	PM3

MODRR[5] — SPI1 Routing

Table 3-6 SPI1 Routing

MODRR[5]	MISO1	MOSI1	SCK1	SS1
0	PP0	PP1	PP2	PP3
1	PH0	PH1	PH2	PH3

3.3.4 Port P Registers

Address Offset: \$__18

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
Write:								
PWM:	PWM7	PWM6	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
SPI:					SS1	SCK1	MOSI1	MISO1
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-22 Port P I/O Register (PTP)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The PWM function takes precedence over the general purpose I/O function if the associated PWM channel is enabled. While channels 6-0 are output only if the respective channel is enabled, channel 7 can be PWM output or input if the shutdown feature is enabled. *See Chapter PWM.*

The SPI function takes precedence over the general purpose I/O function associated with if enabled. *See Chapter SPI.*

If both PWM and SPI are enabled the PWM functionality takes precedence.

Address Offset: \$__19

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
Write:								
Reset:	-	-	-	-	-	-	-	-

 = Reserved or unimplemented

Figure 3-23 Port P Input Register (PTIP)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can be also used to detect overload or short circuit conditions on output pins.

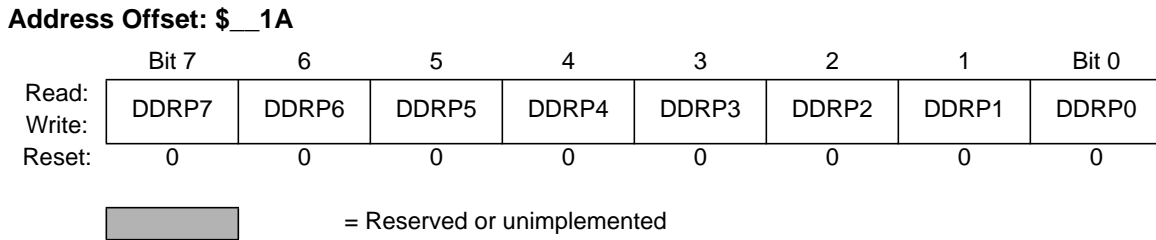


Figure 3-24 Port P Data Direction Register (DDRP)

Read:Anytime.

Write:Anytime.

This register configures each port P pin as either input or output.

If the associated PWM channel or SPI module is enabled this register has no effect on the pins.

The PWM forces the I/O state to be an output for each port line associated with an enabled PWM7-0 channel. Channel 7 can force the pin to input if the shutdown feature is enabled.

If a SPI module is enabled, the SPI determines the pin direction. *For details see SPI specification.*

The DDRM bits revert to controlling the I/O direction of a pin when the associated PWM channel is disabled.

DDRP[7:0] — Data Direction Port P

1 = Associated pin is configured as output.

0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTP or PTIP registers, when changing the DDRP register.

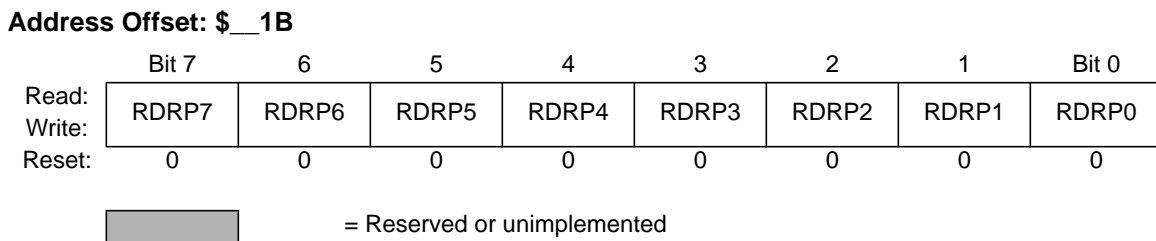


Figure 3-25 Port P Reduced Drive Register (RDRP)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port P output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRP[7:0] — Reduced Drive Port P

- 1 = Associated pin drives at about 1/6 of the full drive strength.
- 0 = Full drive strength at output.

Address Offset: \$__1C

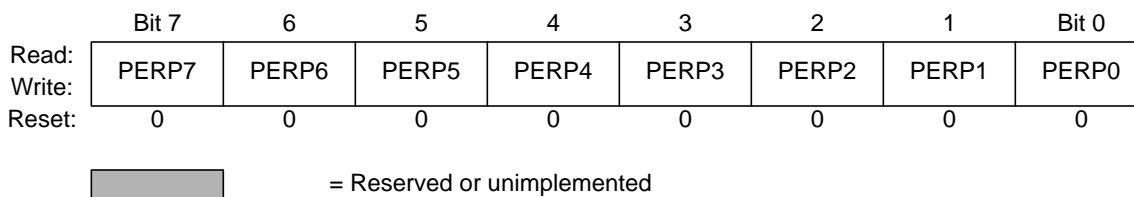


Figure 3-26 Port P Pull Device Enable Register (PERP)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. This bit has no effect if the port is used as output. Out of reset no pull device is enabled.

PERP[7:0] — Pull Device Enable Port P

- 1 = Either a pull-up or pull-down device is enabled.
- 0 = Pull-up or pull-down device is disabled.

Address Offset: \$__1D

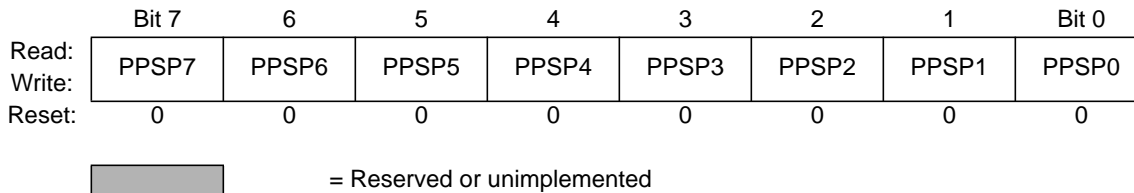


Figure 3-27 Port P Polarity Select Register (PPSP)

Read:Anytime.

Write:Anytime.

This register serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled.

PPSP[7:0] — Polarity Select Port P

- 1 = Rising edge on the associated port P pin sets the associated flag bit in the PIFP register. A pull-down device is connected to the associated port P pin, if enabled by the associated bit in register PERP and if the port is used as input.
- 0 = Falling edge on the associated port P pin sets the associated flag bit in the PIFP register. A pull-up device is connected to the associated port P pin, if enabled by the associated bit in register PERP and if the port is used as input.

Address Offset: \$__1E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIEP7	PIEP6	PIEP5	PIEP4	PIEP3	PIEP2	PIEP1	PIEP0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-28 Port P Interrupt Enable Register (PIEP)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port P.

PIEP[7:0] — Interrupt Enable Port P

1 = Interrupt is enabled.

0 = Interrupt is disabled (interrupt flag masked).

Address Offset: \$__1F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIFP7	PIFP6	PIFP5	PIFP4	PIFP3	PIFP2	PIFP1	PIFP0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-29 Port P Interrupt Flag Register (PIFP)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSP register. To clear this flag, write “1” to the corresponding bit in the PIFP register. Writing a “0” has no effect.

PIFP[7:0] — Interrupt Flags Port P

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).

Writing a “1” clears the associated flag.

0 = No active edge pending.

Writing a “0” has no effect.

3.3.5 Port H Registers

Address Offset: \$__20

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTH7	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
Write:								
SPI1					SS1	SCK1	MOSI1	MISO1
Reset:	0	0	0	0	0	0	0	0


 = Reserved or unimplemented

Figure 3-30 Port H I/O Register (PTH)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

The SPI function takes precedence over the general purpose I/O function associated with if enabled. *Refer to SPI Block Guide for details.*

Address Offset: \$__21

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIH7	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
Write:								
Reset:	-	-	-	-	-	-	-	-


 = Reserved or unimplemented

Figure 3-31 Port H Input Register (PTIH)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

Address Offset: \$ _22

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRH7	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
Write:								
Reset:	0	0	0	0	0	0	0	0


 = Reserved or unimplemented

Figure 3-32 Port H Data Direction Register (DDRH)

Read:Anytime.

Write:Anytime.

This register configures each port H pin as either input or output.

DDRH[7:0] — Data Direction Port H

1 = Associated pin is configured as output.

0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTH or PTIH registers, when changing the DDRH register.

Address Offset: \$ _23

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDRH7	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-33 Port H Reduced Drive Register (RDRH)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port H output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRH[7:0] — Reduced Drive Port H

1 = Associated pin drives at about 1/6 of the full drive strength.

0 = Full drive strength at output.

Address Offset: \$ _24

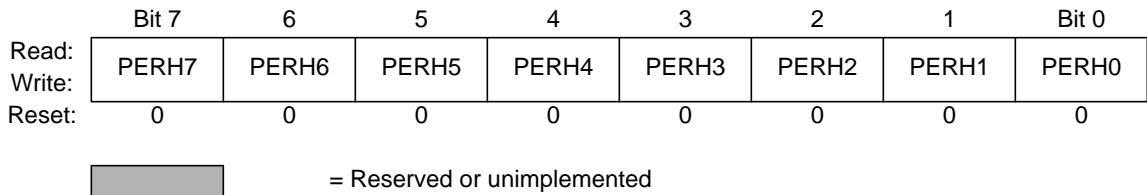


Figure 3-34 Port H Pull Device Enable Register (PERH)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input. This bit has no effect if the port is used as output. Out of reset no pull device is enabled.

PERH[7:0] — Pull Device Enable Port H

- 1 = Either a pull-up or pull-down device is enabled.
- 0 = Pull-up or pull-down device is disabled.

Address Offset: \$ _25



Figure 3-35 Port H Polarity Select Register (PPSH)

Read:Anytime.

Write:Anytime.

This register serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled.

PPSH[7:0] — Polarity Select Port H

- 1 = Rising edge on the associated port H pin sets the associated flag bit in the PIFH register.
A pull-down device is connected to the associated port H pin, if enabled by the associated bit in register PERH and if the port is used as input.
- 0 = Falling edge on the associated port H pin sets the associated flag bit in the PIFH register.
A pull-up device is connected to the associated port H pin, if enabled by the associated bit in register PERH and if the port is used as input.

Address Offset: \$ _26

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIEH7	PIEH6	PIEH5	PIEH4	PIEH3	PIEH2	PIEH1	PIEH0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-36 Port H Interrupt Enable Register (PIEH)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port H.

PIEH[7:0] — Interrupt Enable Port H

1 = Interrupt is enabled.

0 = Interrupt is disabled (interrupt flag masked).

Address Offset: \$ _27

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIFH7	PIFH6	PIFH5	PIFH4	PIFH3	PIFH2	PIFH1	PIFH0
Write:								
Reset:	0	0	0	0	0	0	0	0

 = Reserved or unimplemented

Figure 3-37 Port H Interrupt Flag Register (PIFH)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSH register. To clear this flag, write “1” to the corresponding bit in the PIFH register. Writing a “0” has no effect.

PIFH[7:0] — Interrupt Flags Port H

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).

Writing a “1” clears the associated flag.

0 = No active edge pending.

Writing a “0” has no effect.

3.3.6 Port J Registers

Address Offset: \$ _28

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTJ7	PTJ6	0	0	0	0	PTJ1	PTJ0
Write:								
CAN4:	TXCAN4	RXCAN4						
IIC:	SCL	SDA						
CAN0:	TXCAN0	RXCAN0						
Reset:	0	0	-	-	-	-	0	0


 = Reserved or unimplemented

Figure 3-38 Port J I/O Register (PTJ)

Read:Anytime.

Write:Anytime.

If the data direction bits of the associated I/O pins are set to 1, a read returns the value of the port register, otherwise the value at the pins is read.

PJ[7:6]

The CAN4 function (TXCAN4 and RXCAN4) takes precedence over the IIC, the CAN0 and the general purpose I/O function if the CAN4 module is enabled. *Refer to MSCAN Block Guide for details.*

The IIC function (SCL and SDA) takes precedence over CAN0 and the general purpose I/O function if the IIC is enabled. If the IIC module takes precedence the SDA and SCL outputs are configured as open drain outputs. *Refer to IIC Block Guide for details.*

The CAN0 function (TXCAN0 and RXCAN0) takes precedence over the general purpose I/O function if the CAN0 module is enabled. *Refer to MSCAN Block Guide for details.*

Address Offset: \$ _29

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PTIJ7	PTIJ6	0	0	0	0	PTIJ1	PTIJ0
Write:								
Reset:	-	-	-	-	-	-	-	-


 = Reserved or unimplemented

Figure 3-39 Port J Input Register (PTIJ)

Read:Anytime.

Write:Never, writes to this register have no effect.

This register always reads back the status of the associated pins. This can be used to detect overload or short circuit conditions on output pins.

Address Offset: \$ _2A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	DDRJ7	DDRJ6	0	0	0	0	DDRJ1	DDRJ0
Write:								
Reset:	0	0	-	-	-	-	0	0

= Reserved or unimplemented

Figure 3-40 Port J Data Direction Register (DDRJ)

Read:Anytime.

Write:Anytime.

This register configures each port J pin as either input or output.

The CAN forces the I/O state to be an output on PJ7 (TXCAN4) and an input on pin PJ6 (RXCAN4). The IIC takes control of the I/O if enabled. In these cases the data direction bits will not change. The DDRJ bits revert to controlling the I/O direction of a pin when the associated peripheral module is disabled.

DDRJ[7:6][1:0] — Data Direction Port J

1 = Associated pin is configured as output.

0 = Associated pin is configured as input.

Due to internal synchronization circuits, it can take up to 2 bus cycles until the correct value is read on PTJ or PTIJ registers, when changing the DDRJ register.

Address Offset: \$ _2B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RDRJ7	RDRJ6	0	0	0	0	RDRJ1	RDRJ0
Write:								
Reset:	0	0	-	-	-	-	0	0

= Reserved or unimplemented

Figure 3-41 Port J Reduced Drive Register (RDRJ)

Read:Anytime.

Write:Anytime.

This register configures the drive strength of each port J output pin as either full or reduced. If the port is used as input this bit is ignored.

RDRJ[7:6][1:0] — Reduced Drive Port J

1 = Associated pin drives at about 1/6 of the full drive strength.

0 = Full drive strength at output.

Address Offset: \$ _2C

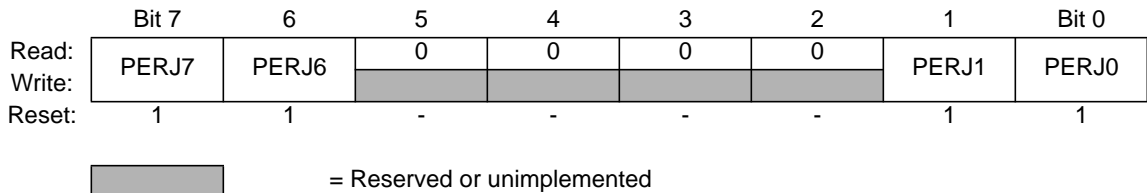


Figure 3-42 Port J Pull Device Enable Register (PERJ)

Read:Anytime.

Write:Anytime.

This register configures whether a pull-up or a pull-down device is activated, if the port is used as input or as wired-or output. This bit has no effect if the port is used as push-pull output. Out of reset a pull-up device is enabled.

PERJ[7:6][1:0] — Pull Device Enable Port J

1 = Either a pull-up or pull-down device is enabled.

0 = Pull-up or pull-down device is disabled.

Address Offset: \$ _2D

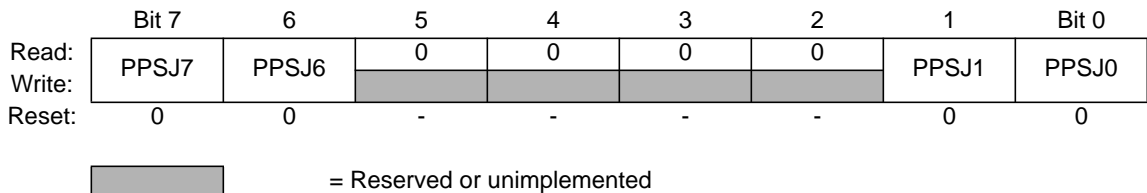


Figure 3-43 Port J Polarity Select Register (PPSJ)

Read:Anytime.

Write:Anytime.

This register serves a dual purpose by selecting the polarity of the active interrupt edge as well as selecting a pull-up or pull-down device if enabled.

PPSJ[7:6][1:0] — Polarity Select Port J

1 = Rising edge on the associated port J pin sets the associated flag bit in the PIFJ register.

A pull-down device is connected to the associated port J pin, if enabled by the associated bit in register PERJ and if the port is used as input.

0 = Falling edge on the associated port J pin sets the associated flag bit in the PIFJ register.

A pull-up device is connected to the associated port J pin, if enabled by the associated bit in register PERJ and if the port is used as general purpose input or as IIC port.

Address Offset: \$ _2E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIEJ7	PIEJ6	0	0	0	0	PIEJ1	PIEJ0
Write:								
Reset:	0	0	-	-	-	-	0	0

 = Reserved or unimplemented

Figure 3-44 Port J Interrupt Enable Register (PIEJ)

Read:Anytime.

Write:Anytime.

This register disables or enables on a per pin basis the edge sensitive external interrupt associated with port J.

PIEJ[7:6][1:0] — Interrupt Enable Port J

1 = Interrupt is enabled.

0 = Interrupt is disabled (interrupt flag masked).

Address Offset: \$ _2F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	PIFJ7	PIFJ6	0	0	0	0	PIFJ1	PIFJ0
Write:								
Reset:	0	0	-	-	-	-	0	0

 = Reserved or unimplemented

Figure 3-45 Port J Interrupt Flag Register (PIFJ)

Read:Anytime.

Write:Anytime.

Each flag is set by an active edge on the associated input pin. This could be a rising or a falling edge based on the state of the PPSJ register. To clear this flag, write “1” to the corresponding bit in the PIFJ register. Writing a “0” has no effect.

PIFJ[7:6][1:0] — Interrupt Flags Port J

1 = Active edge on the associated bit has occurred (an interrupt will occur if the associated enable bit is set).

Writing a “1” clears the associated flag.

0 = No active edge pending.

Writing a “0” has no effect.

Section 4 Functional Description

4.1 General

Each pin can act as general purpose I/O. In addition the pin can act as an output from a peripheral module or an input to a peripheral module. **Table 4-1** summarizes the priority in case of multiple enabled modules trying to control a shared port.

Table 4-1 Summary of Functional Priority

Port	Priority ¹
T	ECT > GPIO
S	SCI, SPI > GPIO
M	Byteflight > CAN0 > BDLC > GPIO Byteflight > CAN1 > (routed) CAN0 > (routed) SPI0 > GPIO Byteflight > (routed) CAN0 > (routed) CAN4 > (routed) SPI0 > GPIO Byteflight > (routed) CAN4 > GPIO
P	PWM > SPI > GPIO
H	SPI > GPIO
J	CAN4 > IIC > (routed) CAN0 > GPIO
A	<i>Refer to HCS12 Multiplexed External Bus Interface Block Guide for details</i>
B	
E	
K	
BKGD pin	<i>Refer to HCS12 Background Debug Mode Block Guide for details</i>

NOTES:

1. Highest priority >... > lowest priority

A set of configuration registers is common to all ports. All registers can be written at any time, however a specific configuration might not become active.

Example:

Selecting a pull-up resistor. This resistor does not become active while the port is used as a push-pull output.

4.1.1 I/O register

This register holds the value driven out to the pin if the port is used as a general purpose I/O.

Writing to this register has only an effect on the pin if the port is used as general purpose output. When reading this address, the value of the pins is returned if the data direction register bits are set to 0.

If the data direction register bits are set to 1, the contents of the I/O register is returned. This is independent of any other configuration (**Figure 4-1**).

4.1.2 Input register

This is a read-only register and always returns the value of the pin (**Figure 4-1**).

4.1.3 Data direction register

This register defines whether the pin is used as an input or an output.

If a peripheral module controls the pin the contents of the data direction register is ignored (**Figure 4-1**).

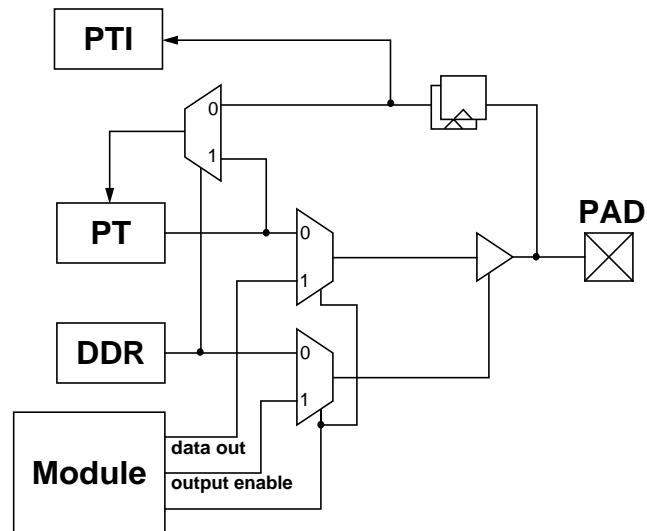


Figure 4-1 Illustration of I/O pin functionality

4.1.4 Reduced drive register

If the port is used as an output the register allows the configuration of the drive strength.

4.1.5 Pull device enable register

This register turns on a pull-up or pull-down device.

It becomes only active if the pin is used as an input or as a wired-or output.

4.1.6 Polarity select register

This register selects either a pull-up or pull-down device if enabled.

It becomes only active if the pin is used as an input. A pull-up device can be activated if the pin is used as a wired-or output.

4.2 Port T

This port is associated with the Enhanced Capture Timer module.

In all modes, port T pins PT[7:0] can be used for either general-purpose I/O, or with the channels of the Enhanced Capture Timer.

During reset, port T pins are configured as high-impedance inputs.

4.3 Port S

This port is associated with the serial SCI and SPI modules.

In all modes, port S pins PS[7:0] can be used either for general-purpose I/O, or with the SCI and SPI subsystems.

During reset, port S pins are configured as inputs with pull-up.

The SPI pins can be re-routed. Refer to section 4.4.1.

4.4 Port M

This port is associated with the Byteflight, BDLC and 3 CAN modules.

In all modes, port M pins PM[7:0] can be used for either general purpose I/O, or with the CAN and BDLC subsystems.

By default, pins PM0 and PM1 are shared between the CAN0 and the BDLC module. If CAN0 is enabled the pins become CAN transmit and receive pins. If BLDC is enabled and CAN0 is disabled, pins become active BDLC transmit and receive pins. Pins PM2-7 are shared amongst Byteflight, CAN0, 1 and 4, and SPI0.

During reset, port M pins are configured as high-impedance inputs.

The CAN pins can be re-routed. Refer to section 4.4.1.

4.4.1 Module Routing Register

This register allows to re-route the CAN0, CAN4, SPI0 and SPI1 pins to predefined pins.

NOTE: *The purpose of the Module Routing Register is to provide maximum flexibility for future derivatives of the MC9S12DT128 with a lower number of MSCAN and SPI modules.*

Table 4-2 Implemented modules on derivatives

Number of modules	MSCAN modules			SPI modules	
	CAN0	CAN1	CAN4	SPI0	SPI1
3	X	X	X	X	X
2	X	-	X	X	X
1	X	-	-	X	-

The Byteflight module has highest priority if enabled. The CAN0 transmit and receive pin can be routed to PM[3:2] or PM[5:4] if Byteflight and CAN1 are disabled, respectively. PM[5:4] or PM[7:6] can be taken by CAN4, if Byteflight is disabled. CAN0 has priority over CAN4 if both modules are trying to access PM[5:4] at the same time.

The SPI0 pins can be routed to PM[5:2] if no other module uses these pins. If the SPI0 module is routed on PM[5:4] and used in bidirectional master mode with disabled \overline{SS} output, PM[3:2] are free to be used with Byteflight, CAN or GPIO.

The SPI1 pins can be routed to PH[3:0].

4.5 Port P

This port is associated with the PWM and one SPI modules.

In all modes, port P pins PP[7:0] can be used for either general purpose I/O, or with the PWM and SPI subsystems.

The pins are shared between the PWM channels and the SPI1 module. If the PWM is enabled the pins become PWM output channels with the exception of pin 7 which can be PWM input or output. If SPI1 is enabled and PWM is disabled, the respective pin configuration is determined by several status bits in the SPI module.

During reset, port P pins are configured as high-impedance inputs.

The SPI pins can be re-routed. Refer to section **4.4.1**.

Port P offers 8 I/O pins with edge triggered interrupt capability in wired-or fashion. The interrupt enable as well as the sensitivity to rising or falling edges can be individually configured on per pin basis. All 8 bits/pins share the same interrupt vector. Interrupts can be used with the pins configured as inputs or outputs.

An interrupt is generated when a bit in the port interrupt flag register and its corresponding port interrupt enable bit are both set. This external interrupt feature is capable to wake up the CPU when it is in STOP or WAIT mode.

A digital filter on each pin prevents pulses (**Figure 4-3**) shorter than a specified time from generating an interrupt. The minimum time varies over process conditions, temperature and voltage (**Figure 4-2** and **Table 4-3**).

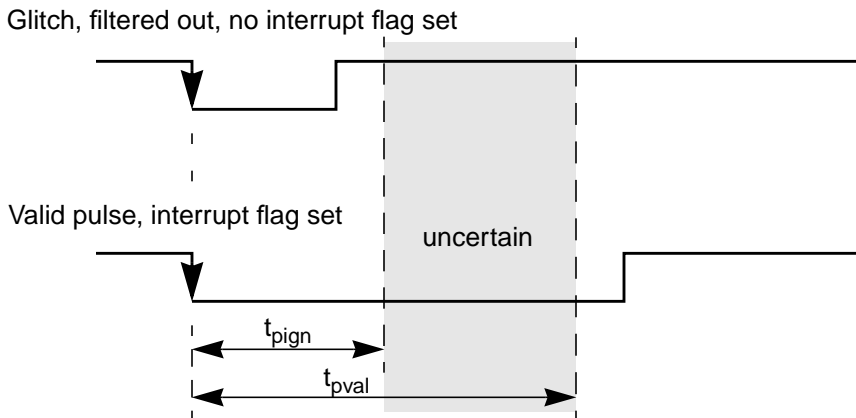


Figure 4-2 Interrupt Glitch Filter on Port P, H and J (PPS=0)

Table 4-3 Pulse Detection Criteria

Pulse	Mode		
	STOP		STOP ¹
		Unit	
Ignored	$t_{pulse} \leq 3$	bus clocks	$t_{pulse} \leq t_{pign}$
Uncertain	$3 < t_{pulse} < 4$	bus clocks	$t_{pign} < t_{pulse} < t_{pval}$
Valid	$t_{pulse} \geq 4$	bus clocks	$t_{pulse} \geq t_{pval}$

NOTES:

1. These values include the spread of the oscillator frequency over temperature, voltage and process.

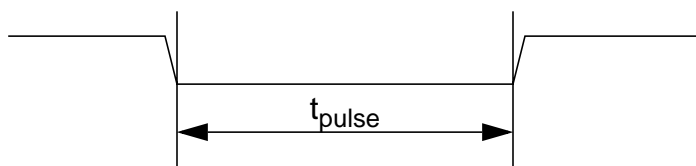


Figure 4-3 Pulse Illustration

A valid edge on an input is detected if 4 consecutive samples of a passive level are followed by 4 consecutive samples of an active level directly or indirectly.

The filters are continuously clocked by the bus clock in RUN and WAIT mode. In STOP mode the clock is generated by a single RC oscillator in the Port Integration Module. To maximize current saving the RC oscillator runs only if the following condition is true on any pin:

Sample count ≤ 4 and port interrupt enabled (PIE=1) and port interrupt flag not set (PIF=0).

4.6 Port H

This port is associated with the SPI1.

Port H pins PH[3:0] can be used for either general purpose I/O, or with the SPI subsystem.

During reset, port H pins are configured as high-impedance inputs.

Port H pins can be used with the routed SPI1 module. Refer to section 4.4.1.

Port H offers 8 I/O ports with the same interrupt features as port P.

Port H pins can be used with the routed SPI1 module. Refer to section 4.4.1.

4.7 Port J

This port is associated with the CAN4, CAN0 and the IIC.

Port J pins PJ[7:6] and PJ[1:0] can be used for either general purpose I/O, or with the CAN and IIC subsystems.

During reset, port J pins are configured as inputs with pull-up.

If IIC takes precedence the pins become IIC open-drain output pins.

The CAN4 pins can be re-routed. Refer to section 4.4.1.

Port J pins can be used with the routed CAN0 modules. Refer to section 4.4.1.

Port J offers 4 I/O ports with the same interrupt features as port P.

4.8 Port A, B, E, K, and BKGD pin

Most port and pin logic is located in the MEBI and BDM modules. *Refer HCS12 Multiplexed External Bus Interface and Background Debug Module Block Guides for details*

4.9 External Pin Descriptions

All ports start up as general purpose inputs on reset.

4.10 Low Power Options

4.10.1 Run Mode

No low power options exist for this module in run mode.

4.10.2 Wait Mode

No low power options exist for this module in wait mode.

4.10.3 Stop Mode

All clocks are stopped. There are however asynchronous paths to generate interrupts from STOP on port P, H and J.

Section 5 Initialization/Application Information

5.1 General

The reset values of all registers are given in the Register Description in section 3.3.

5.2 Reset Initialization

All registers including the data registers get set/reset asynchronously. **Table 5-1** summarizes the port properties after reset initialization.

Table 5-1 Port Reset State Summary

Port	Reset States				
	Data Direction	Pull Mode	Red. Drive	Wired-Or Mode	Interrupt
T	input	hiz	disabled	n/a	n/a
S	input	pull-up	disabled	disabled	n/a
M	input	hiz	disabled	disabled	n/a
P	input	hiz	disabled	n/a	disabled
H	input	hiz	disabled	n/a	disabled
J	input	pull-up	disabled	n/a	disabled
A	<i>Refer to HCS12 Multiplexed External Bus Interface Block Guide for details</i>				
B					
E					
K					
BKGD pin	<i>Refer to HCS12 Background Debug Module Block Guide for details</i>				

Section 6 Interrupts

6.1 General

Ports P, H and J generate a separate edge sensitive interrupt if enabled.

6.2 Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Port P	PIFP[7:0]	PIEP[7:0]	I Bit
Port H	PIFH[7:0]	PIEH[7:0]	I Bit
Port J	PIFJ[7:6,1:0]	PIEJ[7:6,1:0]	I Bit

Table 6-1 Port Integration Module Interrupt Sources

NOTE: *Vector addresses and their relative interrupt priority are determined at the MCU level.*

6.3 Recovery from STOP

The PIM_9DTB128 can generate wake-up interrupts from STOP on ports P, H and J. For other sources of external interrupts refer to the respective Block Guides.

Block Guide End Sheet

**FINAL PAGE OF
56
PAGES**

PWM_8B8C

Block User Guide

V01.17

Original Release Date: 12 MAR 1998
Revised: 1 Aug 2004

Motorola Inc.

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
00.00		3-12-98		First pass release
00.01		3-15-98		<ul style="list-style-type: none"> - Updates of Section 1 based on Nancy Thomas peer review and internal spec review. - Added initial information into Section 2.
00.02		4-7-98		<ul style="list-style-type: none"> - Updates of Section 1 and Section 2 per MSIL review. - Updated cover page per latest spec template review.
01.00		4-15-98		<ul style="list-style-type: none"> - Updated per Rev. 3.0 TSCS Module Spec Template. - Changed the Port and DDR register names to match the latest HCS12 naming convention. - Added the reset state under each counter, period, and duty register in the Register Description sections. - Added Design for Testability sub-section in Section 2 to describe scan implementation. - Updated Module I/O signal names in Section 2 per the latest HCS12 signal naming convention. - Frozen PWM spec sent to Delco.
01.01		4-30-98		<ul style="list-style-type: none"> - Added Document Number (12MRE31052W) to cover page of spec per QS9000 requirements. - Changed PWMEN register to PWME and also changed the bit names from PWENx to PWMEEx to follow the enable naming convention. - Changed PWMEN register to PWMCAE and also changed the bit names from CENx to CAEx to avoid having the same register name as the PWMC module. - Section 2 Module I/O list changed to have only 1 input buffer enable signal (pwm_ibe_t4) for the entire port. The reset signal name was also changed to vsc_reset_t4 per the latest bus definition document. - Added further clarification on DISCRW test bit in Section 2. If set, duty and period registers are not loaded with the buffer value.
01.02		5-14-98		<ul style="list-style-type: none"> - Updated per Rev. 3.1, 3.2 and 3.3 TSCS Module Spec Template. - Removed Table 1-1 PWM Register Address Summary. Added the Address Offset along side the registers in Figure 1-2 PWM Register Map. - Added WARNING regarding writing to the test registers in special modes. - Added footnote regarding the counter value in the Period=0 boundary case. - Removed "weasel" words--may and should.

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.03		5-27-98		<p>Summary of changes:</p> <ul style="list-style-type: none"> - Added clarification on how the counter counts in left and center aligned output modes. - Added further clarification on the Period=0 boundary case. Added that the counter=0. - Added further clarification on what occurs on writes to the counter--output is changed according to the polarity bit. - Added Caution regarding the first PWM cycle after the channel is enabled can be irregular. - Replaced bit 'RDP' with 'RDPPWM' and bit 'PUPP' with 'PUPPWME' to match port control bit naming conventions. - Added 'iam8bit' signal in Table 2-1. - Added Table 2-2, Engineering Electrical Specs. - Added statement in Section 2 regarding DISCRW bit in PWMTST register. When bit is set, the output is not changed according to the polarity bit. - Added statement in Section 2 regarding DISCRM bit in PWMTST register. When bit is set, the duty and period registers do not get loaded with the buffer value. - Corrected left and center aligned max PWM output frequencies in Table A-2. - Created Table A-3 for the PWM Period/Duty Resolution Characteristics. - Miscellaneous clean up.
01.04		7-1-98		<ul style="list-style-type: none"> * Changed reset state of PWMPERx and PWMDTYx registers to FF. * In section 2: changed some bus interface signal names: vsc_wait_t2 changed to vsc_wait_t3 vsc_bdmact_t2 changed to vsc_bdmact_t4 vsc_smod_t2 changed to vsc_smod_t4 * In section 2: Added a note that in concatenated, left aligned, DISCRW=1 writing 16 bit (high-byte-data, low-byte-data) to the counter causes the high byte of the counter to start counting from (high-byte-data) and the low byte of the counter to start counting from (low-byte-data + 1).

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.05		1-6-99		<p>Includes spec tagging in conditional text. There are 3 conditional text tags: Tested- Functional Test (Blue), STATEMENT- Statement (Green), Test Outside Submodule (red). Summary of changes: * Section 1: - Added the following sentence in section 1.6.2.4: When the channel is disabled, writing "0" to the period register will cause the counter to reset on the next selected clock. - Added a caution in section 1.7.15.1 about reading from port register after changing pin to input. - In section 1.9 hanged "Reset state: The prescale free running counter begins to increment" to: "All the channels are disabled and all the counters don't count". * Section 2: - Section 2.2.3 Design for Test: Scan is not implemented on the PWM module. During scan mode (vsc_scanmod = 1) the module is not selected (vsc_pwmssel_t3 = 0) and the module's internal clocks stop. - Table 2-1 PWM Module I/O Signals: Removed scan_e, changed pwm_purst_plug to pwm_puerst_plug, changed vsc_wait_t3 to vsc_wait_t2, changed pwm_outdata_t4[7:0] to pwm_do_t4[7:0], changed pad_indata[7:0] to pwmp_ind[7:0], removed iam8bit, added vsc_en2drv, changed vsc_bdmact_t4 to vsc_bdmact_t2, changed vsc_smod_t4 to vsc_smod_t2. - Added section 2.3.3 and table 2-2 Port Pin Connections. - Section 2.4 table 2-3: Changed Vdd Value to 3 to 3.6v (it was 2.7 to 3.6). Changed System Clock Value to dc to 16MHz (it was 20). - Section 2.4: added Figure 2-7: PWM Timing Diagram. It's like A.6 in the appendix, but includes more details. * Appendix A: Table A-1: System Clock dc to 16MHz (it was 20). Table A-2: E-clock 16MHz (it was 20). A.6 added PWM Timing Diagram.</p>
01.06	09-02-99			<p>* In section 1, Added section 1.7.15 shutdown register(PWMSDN), changed the sec # for the subsequent sections for the registers. * Added emergency shutdown feature in the feature list (sec 1.3) * Added the PWMSDN in the register map (sec 1.5) * Removed \$ _24 from resvd. reg list in sec 1.7.17 * Modified sec 1.12, Interrupt Op., to support the intr. for emergency shutdown, to The PWM module has only one interrupt which is generated at the time of emergency shutdown, if the corresponding enable bit (PWMSDN[6]) is set. * Section 2.2.3: Design for Test: The PWM module will be fully scannable as per the project DFT guidelines * removed GPIO note from section 1.2, 1.3, 1.6.2.1, 1.6.2.7, 1.7.1 * renamed PSBCK to PFRZ in PWMCTL, and removed RDPPWM & PUPPWME bits * changed the interface signals for IP bus. * removed electrical spec details.</p>

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.07		10-25-99		updated the specs after feedback from Munich (removed redundant port signals: ibe, offval, obe[6:0])
01.08		11-25-99		added Global Clock signal to the I/O list. the same is used for reg. writes wherever possible.
01.09		12-08-99		Restart(from shutdown) functionality clarified
01.10		01-17-2000		Tagging Done for Barracuda
01.11		01-25-2000		Tagging for wait mode and freeze mode.
01.12		10-09-2000		format converted for SRS 2.0 compliance.
01.13		19-01-2001		updated section 6.1 and 3.3.1.15 for the shutdown feature changes.
01.14		05-04-2001		Made SRS 2.0 Compliant
01.15		07-19-2001		Document names have been added, Names and Variable definitions have been hidden
01.16		03-14-2002		Syntax corrections, document formal updates
01.17		08-01-2004		Added clarification of PWMIF operation in STOP and WAIT mode. Added notes on minimum pulse width of emergency shutdown signal.

Table 0-1 Revision History

Table of Contents

Section 1 Introduction

1.1	Overview	13
1.2	Features	13
1.3	Modes of Operation	13
1.4	Block Diagram	13

Section 2 PWM8b8cSignal Description

2.1	Overview	15
2.2	Detailed Signal Descriptions	15
2.2.1	PWM7 — PWM8b8c Channel 7	15
2.2.2	PWM6 — PWM8b8c Channel 6	15
2.2.3	PWM5 — PWM8b8c Channel 5	15
2.2.4	PWM4 — PWM8b8c Channel 4	15
2.2.5	PWM3 — PWM8b8c Channel 3	15
2.2.6	PWM2 — PWM8b8c Channel 2	15
2.2.7	PWM1 — PWM8b8c Channel 1	15
2.2.8	PWM0 — PWM8b8c Channel 0	15

Section 3 Memory Map and Register Definition

3.1	Overview	17
3.2	Module Memory Map	17
3.3	Register Descriptions	18
3.3.1	PWM Enable Register (PWME)	18
3.3.2	PWM Polarity Register (PWMPOL)	20
3.3.3	PWM Clock Select Register (PWMCLK)	21
3.3.4	PWM Prescale Clock Select Register (PWMPRCLK)	23
3.3.5	PWM Center Align Enable Register (PWMCAE)	24
3.3.6	PWM Control Register (PWMCTL)	25
3.3.7	Reserved Register (PWMTST)	27
3.3.8	Reserved Register (PWMPRSC)	27
3.3.9	PWM Scale A Register (PWMSCLA)	28
3.3.10	PWM Scale B Register (PWMSCLB)	28
3.3.11	Reserved Registers (PWMSCNTx)	29

3.3.12	PWM Channel Counter Registers (PWMCNTx)	29
3.3.13	PWM Channel Period Registers (PWMPERx)	30
3.3.14	PWM Channel Duty Registers (PWMDTYx)	31
3.3.15	PWM Shutdown Register (PWMSDN)	33

Section 4 Functional Description

4.1	PWM Clock Select	35
4.1.1	Prescale	37
4.1.2	Clock Scale	37
4.1.3	Clock Select	38
4.2	PWM Channel Timers	38
4.2.1	PWM Enable	39
4.2.2	PWM Polarity	39
4.2.3	PWM Period and Duty	40
4.2.4	PWM Timer Counters	40
4.2.5	Left Aligned Outputs	41
4.2.6	Center Aligned Outputs	43
4.2.7	PWM 16-Bit Functions	44
4.2.8	PWM Boundary Cases	47

Section 5 Resets

5.1	General	49
-----	-------------------	----

Section 6 Interrupts

6.1	Interrupt Operation	51
-----	-------------------------------	----

List of Figures

Figure 1-1	PWM_8B8C Block Diagram.	14
Figure 3-1	PWM Enable Register (PWME).	19
Figure 3-2	PWM Polarity Register (PWMPOL).	20
Figure 3-3	PWM Clock Select Register (PWMCLK).	22
Figure 3-4	PWM Prescale Clock Select Register (PWMPRCLK).	23
Figure 3-5	PWM Center Align Enable Register (PWMCAE).	24
Figure 3-6	PWM Control Register (PWMCTL).	25
Figure 3-7	Reserved Register (PWMTST).	27
Figure 3-8	Reserved Register (PWMPRSC).	27
Figure 3-9	PWM Scale A Register (PWMSCLA).	28
Figure 3-10	PWM Scale B Register (PWMSCLB).	29
Figure 3-11	Reserved Registers (PWMSCNTx).	29
Figure 3-12	PWM Channel Counter Registers (PWMCNTx).	30
Figure 3-13	PWM Channel Period Registers (PWMPERx).	31
Figure 3-14	PWM Channel Duty Registers (PWMDTYx).	32
Figure 3-15	PWM Shutdown Register (PWMSDN).	33
Figure 4-1	PWM Clock Select Block Diagram	36
Figure 4-2	PWM Timer Channel Block Diagram	39
Figure 4-3	PWM Left Aligned Output Waveform	42
Figure 4-4	PWM Left Aligned Output Example Waveform.	42
Figure 4-5	PWM Center Aligned Output Waveform.	43
Figure 4-6	PWM Center Aligned Output Example Waveform	44
Figure 4-7	PWM 16-Bit Mode.	45

List of Tables

Table 0-1	Revision History	5
Table 3-1	PWM_8B8C Memory Map	17
Table 3-2	Clock B Prescaler Selects	23
Table 3-3	Clock A Prescaler Selects	24
Table 4-1	PWM Timer Counter Conditions	41
Table 4-2	16-bit Concatenation Mode Summary	46
Table 4-3	PWM Boundary Cases.	47

Section 1 Introduction

1.1 Overview

The PWM_8B8C definition is based on the HC12 PWM definitions. This PWM_8B8C contains the basic features from the HC11 with some of the enhancements incorporated on the HC12., that is center aligned output mode and four available clock sources. The PWM_8B8C module has eight channels with independent control of left and center aligned outputs on each channel.

Each of the eight channels has a programmable period and duty cycle as well as a dedicated counter. A flexible clock select scheme allows a total of four different clock sources to be used with the counters. Each of the modulators can create independent continuous waveforms with software-selectable duty rates from 0% to 100%. The PWM outputs can be programmed as left aligned outputs or center aligned outputs.

1.2 Features

The block includes these distinctive features:

- Eight independent PWM channels with programmable period and duty cycle.
- Dedicated counter for each PWM channel.
- Programmable PWM enable/disable for each channel.
- Software selection of PWM duty pulse polarity for each channel.
- Period and duty cycle are double buffered. Change takes effect when the end of the effective period is reached (PWM counter reaches zero) or when the channel is disabled.
- Programmable center or left aligned outputs on individual channels.
- Eight 8-bit channel or four 16-bit channel PWM resolution.
- Four clock sources (A, B, SA and SB) provide for a wide range of frequencies.
- Programmable Clock Select Logic.
- Emergency shutdown.

1.3 Modes of Operation

There is a software programmable option for low power consumption in Wait mode that disables the input clock to the prescaler.

In freeze mode there is a software programmable option to disable the input clock to the prescaler. This is useful for emulation.

1.4 Block Diagram

Figure 1-1 shows the block diagram for the 8-bit 8-channel PWM_8B8C block.

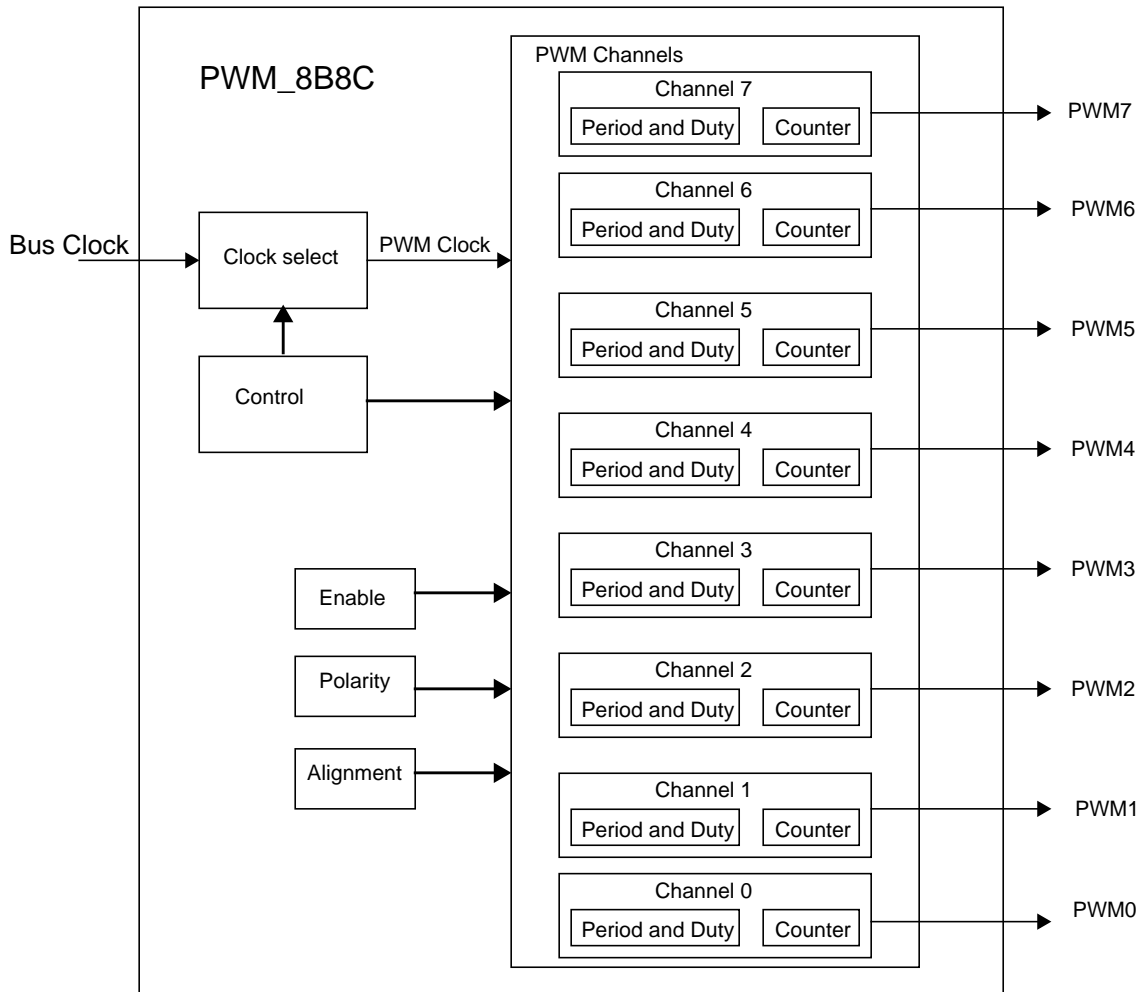


Figure 1-1 PWM_8B8C Block Diagram

Section 2 PWM8b8cSignal Description

2.1 Overview

The PWM_8B8C module has a total of 8 external pins.

2.2 Detailed Signal Descriptions

2.2.1 PWM7 — PWM8b8c Channel 7

This pin serves as waveform output of PWM channel 7 and as an input for the emergency shutdown feature.

2.2.2 PWM6 — PWM8b8c Channel 6

This pin serves as waveform output of PWM channel 6.

2.2.3 PWM5 — PWM8b8c Channel 5

This pin serves as waveform output of PWM channel 5.

2.2.4 PWM4 — PWM8b8c Channel 4

This pin serves as waveform output of PWM channel 4.

2.2.5 PWM3 — PWM8b8c Channel 3

This pin serves as waveform output of PWM channel 3.

2.2.6 PWM2 — PWM8b8c Channel 2

This pin serves as waveform output of PWM channel 2.

2.2.7 PWM1 — PWM8b8c Channel 1

This pin serves as waveform output of PWM channel 1.

2.2.8 PWM0 — PWM8b8c Channel 0

This pin serves as waveform output of PWM channel 0.

Section 3 Memory Map and Register Definition

3.1 Overview

This section describes in detail all the registers and register bits in the PWM_8B8C module.

The special-purpose registers and register bit functions that would not normally be made available to device end users, such as factory test control registers and reserved registers are clearly identified by means of shading the appropriate portions of address maps and register diagrams. Notes explaining the reasons for restricting access to the registers and functions are also explained in the individual register descriptions.

3.2 Module Memory Map

This section describes the content of the registers in the PWM_8B8C module. The base address of the PWM_8B8C module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset. The figure below shows the registers associated with the PWM and their relative offset from the base address. The register detail description follows the order they appear in the register map.

Reserved bits within a register will always read as 0 and the write will be unimplemented. Unimplemented functions are indicated by shading the bit.

Table 3-1 shows the memory map for the PWM_8B8C module

Table 3-1 PWM_8B8C Memory Map

Address	Use	Access
\$_00	PWM Enable Register (PWME)	R/W
\$_01	PWM Polarity Register (PWMPOL)	R/W
\$_02	PWM Clock Select Register (PWMCLK)	R/W
\$_03	PWM Prescale Clock Select Register (PWMPRCLK)	R/W
\$_04	PWM Center Align Enable Register (PWMCAE)	R/W
\$_05	PWM Control Register (PWMCTL)	R/W
\$_06	PWM Test Register (PWMTST) ¹	R/W
\$_07	PWM Prescale Counter Register (PWMPRSC) ²	R/W
\$_08	PWM Scale A Register (PWMSCLA)	R/W
\$_09	PWM Scale B Register (PWMSCLB)	R/W
\$_0A	PWM Scale A Counter Register (PWMSCNTA) ³	R/W
\$_0B	PWM Scale B Counter Register (PWMSCNTB) ⁴	R/W
\$_0C	PWM Channel 0 Counter Register (PWMCNT0)	R/W
\$_0D	PWM Channel 1 Counter Register (PWMCNT1)	R/W
\$_0E	PWM Channel 2 Counter Register (PWMCNT2)	R/W
\$_0F	PWM Channel 3 Counter Register (PWMCNT3)	R/W
\$_10	PWM Channel 4 Counter Register (PWMCNT4)	R/W
\$_11	PWM Channel 5 Counter Register (PWMCNT5)	R/W
\$_12	PWM Channel 6 Counter Register (PWMCNT6)	R/W

Table 3-1 PWM_8B8C Memory Map

\$_13	PWM Channel 7 Counter Register (PWMCNT7)	R/W
\$_14	PWM Channel 0 Period Register (PWMPER0)	R/W
\$_15	PWM Channel 1 Period Register (PWMPER1)	R/W
\$_16	PWM Channel 2 Period Register (PWMPER2)	R/W
\$_17	PWM Channel 3 Period Register (PWMPER3)	R/W
\$_18	PWM Channel 4 Period Register (PWMPER4)	R/W
\$_19	PWM Channel 5 Period Register (PWMPER5)	R/W
\$_1A	PWM Channel 6 Period Register (PWMPER6)	R/W
\$_1B	PWM Channel 7 Period Register (PWMPER7)	R/W
\$_1C	PWM Channel 0 Duty Register (PWMDTY0)	R/W
\$_1D	PWM Channel 1 Duty Register (PWMDTY1)	R/W
\$_1E	PWM Channel 2 Duty Register (PWMDTY2)	R/W
\$_1F	PWM Channel 3 Duty Register (PWMDTY3)	R/W
\$_20	PWM Channel 4 Duty Register (PWMDTY4)	R/W
\$_21	PWM Channel 5 Duty Register (PWMDTY5)	R/W
\$_22	PWM Channel 6 Duty Register (PWMDTY6)	R/W
\$_23	PWM Channel 7 Duty Register (PWMDTY7)	R/W
\$_24	PWM Shutdown Register (PWMSDN)	R/W
\$_25	Reserved	R
\$_26	Reserved	R
\$_27	Reserved	R

NOTES:

1. PWMTST is intended for factory test purposes only.
2. PWMPRSC is intended for factory test purposes only.
3. PWMSCNTA is intended for factory test purposes only.
4. PWMSCNTB is intended for factory test purposes only.

NOTE: Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

3.3 Register Descriptions

This section describes in detail all the registers and register bits in the PWM_8B8C module.

3.3.1 PWM Enable Register (PWME)

Each PWM channel has an enable bit (PWME_x) to start its waveform output. When any of the PWME_x bits are set (PWME_x=1), the associated PWM output is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWME_x and the clock source.

NOTE: The first PWM cycle after enabling the channel can be irregular.

An exception to this is when channels are concatenated. Once concatenated mode is enabled (CON_{xx} bits set in PWMCTL register) then enabling/disabling the corresponding 16-bit PWM channel is controlled by

the low order PWME_x bit. In this case, the high order bytes PWME_x bits have no effect and their corresponding PWM output lines are disabled.

While in run mode, if all eight PWM channels are disabled (PWME7-0=0), the prescaler counter shuts off for power savings.

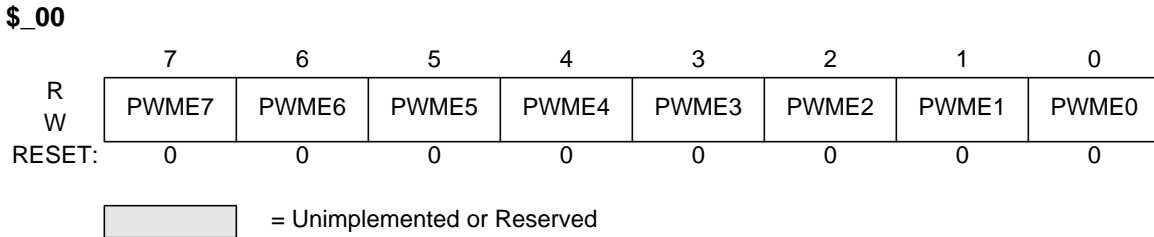


Figure 3-1 PWM Enable Register (PWME)

Read: anytime

Write: anytime

PWME7 — Pulse Width Channel 7 Enable

- 1 = Pulse Width channel 7 is enabled. The pulse modulated signal becomes available at PWM output bit7 when its clock source begins its next cycle.
- 0 = Pulse Width channel 7 is disabled.

PWME6 — Pulse Width Channel 6 Enable

- 1 = Pulse Width channel 6 is enabled. The pulse modulated signal becomes available at PWM output bit6 when its clock source begins its next cycle. If CON67=1, then bit has no effect and PWM output line6 is disabled.
- 0 = Pulse Width channel 6 is disabled.

PWME5 — Pulse Width Channel 5 Enable

- 1 = Pulse Width channel 5 is enabled. The pulse modulated signal becomes available at PWM output bit 5 when its clock source begins its next cycle.
- 0 = Pulse Width channel 5 is disabled.

PWME4 — Pulse Width Channel 4 Enable

- 1 = Pulse Width channel 4 is enabled. The pulse modulated signal becomes available at PWM, output bit 4 when its clock source begins its next cycle. If CON45=1, then bit has no effect and PWM output bit4 is disabled.
- 0 = Pulse Width channel 4 is disabled.

PWME3 — Pulse Width Channel 3 Enable

- 1 = Pulse Width channel 3 is enabled. The pulse modulated signal becomes available at PWM, output bit 3 when its clock source begins its next cycle.
- 0 = Pulse Width channel 3 is disabled.

PWME2 — Pulse Width Channel 2 Enable

- 1 = Pulse Width channel 2 is enabled. The pulse modulated signal becomes available at PWM, output bit 2 when its clock source begins its next cycle. If CON23=1, then bit has no effect and PWM output bit2 is disabled.
- 0 = Pulse Width channel 2 is disabled.

PWME1 — Pulse Width Channel 1 Enable

- 1 = Pulse Width channel 1 is enabled. The pulse modulated signal becomes available at PWM, output bit 1 when its clock source begins its next cycle.
- 0 = Pulse Width channel 1 is disabled.

PWME0 — Pulse Width Channel 0 Enable

- 1 = Pulse Width channel 0 is enabled. The pulse modulated signal becomes available at PWM, output bit 0 when its clock source begins its next cycle. If CON01=1, then bit has no effect and PWM output line0 is disabled.
- 0 = Pulse Width channel 0 is disabled.

3.3.2 PWM Polarity Register (PWMPOL)

The starting polarity of each PWM channel waveform is determined by the associated PPOLx bit in the PWMPOL register. If the polarity bit is one, the PWM channel output is high at the beginning of the cycle and then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

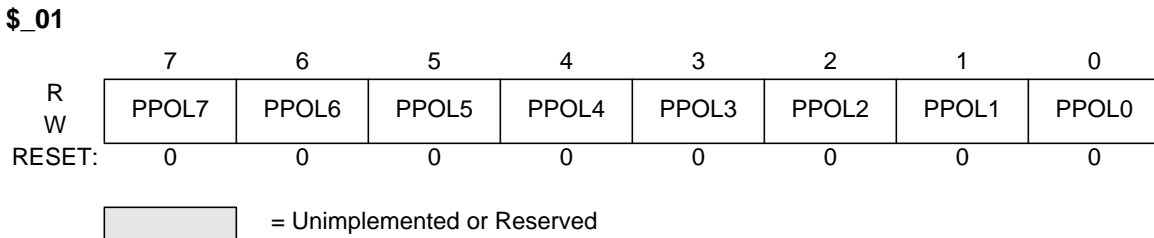


Figure 3-2 PWM Polarity Register (PWMPOL)

Read: anytime

Write: anytime

NOTE: *PPOLx register bits can be written anytime. If the polarity is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition*

PPOL7 — Pulse Width Channel 7 Polarity

- 1 = PWM channel 7 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 7 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL6 — Pulse Width Channel 6 Polarity

- 1 = PWM channel 6 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 6 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL5 — Pulse Width Channel 5 Polarity

- 1 = PWM channel 5 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 5 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL4 — Pulse Width Channel 4 Polarity

- 1 = PWM channel 4 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 4 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL3 — Pulse Width Channel 3 Polarity

- 1 = PWM channel 3 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 3 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL2 — Pulse Width Channel 2 Polarity

- 1 = PWM channel 2 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 2 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL1 — Pulse Width Channel 1 Polarity

- 1 = PWM channel 1 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 1 output is low at the beginning of the period, then goes high when the duty count is reached.

PPOL0 — Pulse Width Channel 0 Polarity

- 1 = PWM channel 0 output is high at the beginning of the period, then goes low when the duty count is reached.
- 0 = PWM channel 0 output is low at the beginning of the period, then goes high when the duty count is reached.

3.3.3 PWM Clock Select Register (PWMCLK)

Each PWM channel has a choice of two clocks to use as the clock source for that channel as described below.

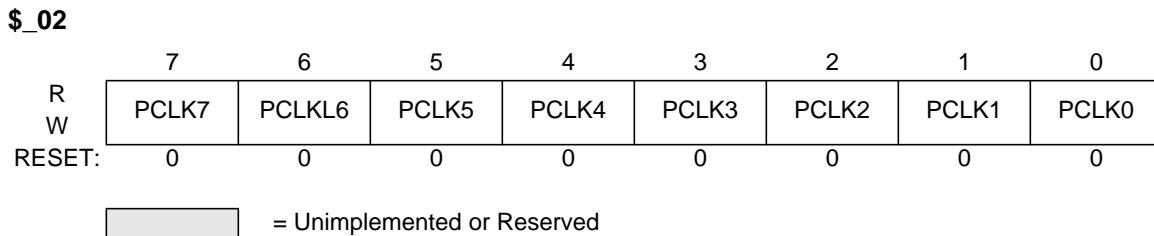


Figure 3-3 PWM Clock Select Register (PWMCLK)

Read: anytime

Write: anytime

NOTE: Register bits PCLK0 to PCLK7 can be written anytime. If a clock select is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.

PCLK7 — Pulse Width Channel 7 Clock Select

- 1 = Clock SB is the clock source for PWM channel 7.
- 0 = Clock B is the clock source for PWM channel 7.

PCLK6 — Pulse Width Channel 6 Clock Select

- 1 = Clock SB is the clock source for PWM channel 6.
- 0 = Clock B is the clock source for PWM channel 6.

PCLK5 — Pulse Width Channel 5 Clock Select

- 1 = Clock SA is the clock source for PWM channel 5.
- 0 = Clock A is the clock source for PWM channel 5.

PCLK4 — Pulse Width Channel 4 Clock Select

- 1 = Clock SA is the clock source for PWM channel 4.
- 0 = Clock A is the clock source for PWM channel 4.

PCLK3 — Pulse Width Channel 3 Clock Select

- 1 = Clock SB is the clock source for PWM channel 3.
- 0 = Clock B is the clock source for PWM channel 3.

PCLK2 — Pulse Width Channel 2 Clock Select

- 1 = Clock SB is the clock source for PWM channel 2.
- 0 = Clock B is the clock source for PWM channel 2.

PCLK1 — Pulse Width Channel 1 Clock Select

- 1 = Clock SA is the clock source for PWM channel 1.
- 0 = Clock A is the clock source for PWM channel 1.

PCLK0 — Pulse Width Channel 0 Clock Select

- 1 = Clock SA is the clock source for PWM channel 0.

0 = Clock A is the clock source for PWM channel 0.

3.3.4 PWM Prescale Clock Select Register (PWMPRCLK)

This register selects the prescale clock source for clocks A and B independently.

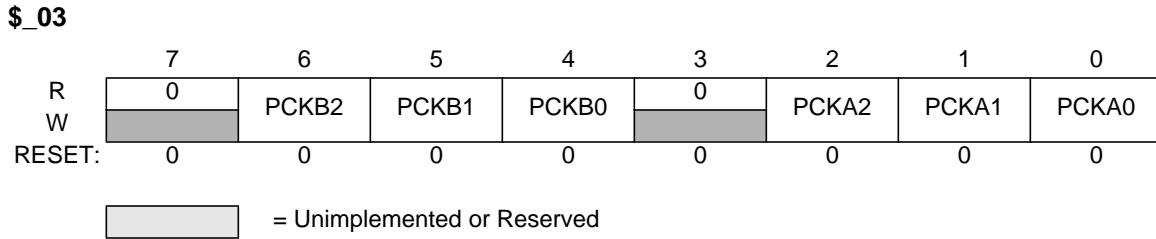


Figure 3-4 PWM Prescale Clock Select Register (PWMPRCLK)

Read: anytime

Write: anytime

NOTE: *PCKB2-0 and PCKA2-0 register bits can be written anytime. If the clock pre-scale is changed while a PWM signal is being generated, a truncated or stretched pulse can occur during the transition.*

PCKB2 - PCKB0 — Prescaler Select for Clock B

Clock B is one of two clock sources which can be used for channels 2, 3, 6 or 7. These three bits determine the rate of clock B, as shown in the following table.

Table 3-2 Clock B Prescaler Selects

PCKB2	PCKB1	PCKB0	Value of Clock B
0	0	0	bus clock
0	0	1	bus clock / 2
0	1	0	bus clock / 4
0	1	1	bus clock / 8
1	0	0	bus clock / 16
1	0	1	bus clock / 32
1	1	0	bus clock / 64
1	1	1	bus clock / 128

PCKA2 - PCKA0 — Prescaler Select for Clock A

Clock A is one of two clock sources which can be used for channels 0, 1, 4 or 5. These three bits determine the rate of clock A, as shown in the following table.

Table 3-3 Clock A Prescaler Selects

PCKA2	PCKA1	PCKA0	Value of Clock A
0	0	0	bus clock
0	0	1	bus clock / 2
0	1	0	bus clock / 4
0	1	1	bus clock / 8
1	0	0	bus clock / 16
1	0	1	bus clock / 32
1	1	0	bus clock / 64
1	1	1	bus clock / 128

3.3.5 PWM Center Align Enable Register (PWMCAE)

The PWMCAE register contains eight control bits for the selection of center aligned outputs or left aligned outputs for each PWM channel. If the CAEx bit is set to a one, the corresponding PWM output will be center aligned. If the CAEx bit is cleared, the corresponding PWM output will be left aligned. Reference **4.2.5 Left Aligned Outputs** and **4.2.6 Center Aligned Outputs** for a more detailed description of the PWM output modes.

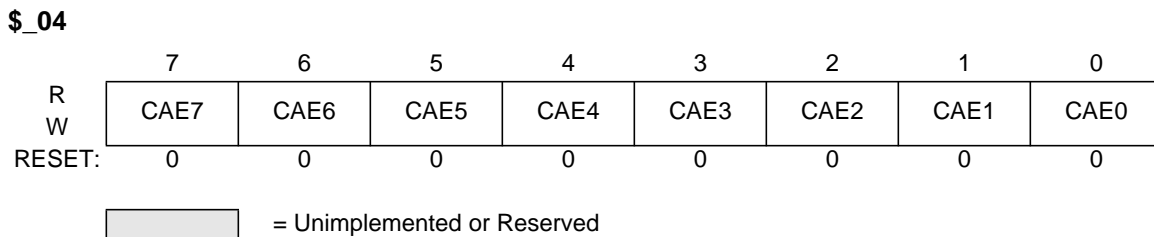


Figure 3-5 PWM Center Align Enable Register (PWMCAE)

Read: anytime

Write: anytime

NOTE: Write these bits only when the corresponding channel is disabled.

CAE7 — Center Aligned Output Mode on channel 7
 1 = Channel 7 operates in Center Aligned Output Mode.
 0 = Channel 7 operates in Left Aligned Output Mode.

CAE6 — Center Aligned Output Mode on channel 6

1 = Channel 6 operates in Center Aligned Output Mode.

0 = Channel 6 operates in Left Aligned Output Mode.

CAE5 — Center Aligned Output Mode on channel 5

1 = Channel 5 operates in Center Aligned Output Mode.

0 = Channel 5 operates in Left Aligned Output Mode.

CAE4 — Center Aligned Output Mode on channel 4

1 = Channel 4 operates in Center Aligned Output Mode.

0 = Channel 4 operates in Left Aligned Output Mode.

CAE3 — Center Aligned Output Mode on channel 3

1 = Channel 3 operates in Center Aligned Output Mode.

0 = Channel 3 operates in Left Aligned Output Mode.

CAE2 — Center Aligned Output Mode on channel 2

1 = Channel 2 operates in Center Aligned Output Mode.

0 = Channel 2 operates in Left Aligned Output Mode.

CAE1 — Center Aligned Output Mode on channel 1

1 = Channel 1 operates in Center Aligned Output Mode.

0 = Channel 1 operates in Left Aligned Output Mode.

CAE0 — Center Aligned Output Mode on channel 0

1 = Channel 0 operates in Center Aligned Output Mode.

0 = Channel 0 operates in Left Aligned Output Mode.

3.3.6 PWM Control Register (PWMCTL)

The PWMCTL register provides for various control of the PWM module.



Figure 3-6 PWM Control Register (PWMCTL)

Read: anytime

Write: anytime

There are three control bits for concatenation, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. When channels 6 and 7 are concatenated, channel 6 registers become the high order bytes of the double byte channel. When channels 4 and 5 are concatenated, channel 4 registers

become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double byte channel.

Reference **4.2.7 PWM 16-Bit Functions** for a more detailed description of the concatenation PWM Function.

NOTE: *Change these bits only when both corresponding channels are disabled.*

CON67 — Concatenate channels 6 and 7

- 1 = Channels 6 and 7 are concatenated to create one 16-bit PWM channel. Channel 6 becomes the high order byte and channel 7 becomes the low order byte. Channel 7 output pin is used as the output for this 16-bit PWM (bit 7 of port PWMP). Channel 7 clock select control-bit determines the clock source, channel 7 polarity bit determines the polarity, channel 7 enable bit enables the output and channel 7 center aligned enable bit determines the output mode.
- 0 = Channels 6 and 7 are separate 8-bit PWMs.

CON45 — Concatenate channels 4 and 5

- 1 = Channels 4 and 5 are concatenated to create one 16-bit PWM channel. Channel 4 becomes the high order byte and channel 5 becomes the low order byte. Channel 5 output pin is used as the output for this 16-bit PWM (bit 5 of port PWMP). Channel 5 clock select control-bit determines the clock source, channel 5 polarity bit determines the polarity, channel 5 enable bit enables the output and channel 5 center aligned enable bit determines the output mode.
- 0 = Channels 4 and 5 are separate 8-bit PWMs.

CON23 — Concatenate channels 2 and 3

- 1 = Channels 2 and 3 are concatenated to create one 16-bit PWM channel. Channel 2 becomes the high order byte and channel 3 becomes the low order byte. Channel 3 output pin is used as the output for this 16-bit PWM (bit 3 of port PWMP). Channel 3 clock select control-bit determines the clock source, channel 3 polarity bit determines the polarity, channel 3 enable bit enables the output and channel 3 center aligned enable bit determines the output mode.
- 0 = Channels 2 and 3 are separate 8-bit PWMs.

CON01 — Concatenate channels 0 and 1

- 1 = Channels 0 and 1 are concatenated to create one 16-bit PWM channel. Channel 0 becomes the high order byte and channel 1 becomes the low order byte. Channel 1 output pin is used as the output for this 16-bit PWM (bit 1 of port PWMP). Channel 1 clock select control-bit determines the clock source, channel 1 polarity bit determines the polarity, channel 1 enable bit enables the output and channel 1 center aligned enable bit determines the output mode.
- 0 = Channels 0 and 1 are separate 8-bit PWMs.

PSWAI — PWM Stops in Wait Mode

Enabling this bit allows for lower power consumption in Wait Mode by disabling the input clock to the prescaler.

- 1 = Stop the input clock to the prescaler whenever the MCU is in Wait Mode.
- 0 = Allow the clock to the prescaler to continue while in wait mode.

PFRZ — PWM Counters Stop in Freeze Mode

In Freeze Mode, there is an option to disable the input clock to the prescaler by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode the input clock to the prescaler is disabled. This feature is useful during emulation as it allows the PWM function to be suspended. In this way, the counters of the PWM can be stopped while in freeze mode so that once normal program flow is continued, the counters are re-enabled to simulate real-time operations. Since the registers can still be accessed in this mode, to re-enable the prescaler clock, either disable the PFRZ bit or exit freeze mode.

- 1 = Disable PWM input clock to the prescaler whenever the part is in freeze mode. This is useful for emulation.
- 0 = Allow PWM to continue while in freeze mode.

3.3.7 Reserved Register (PWMTST)

This register is reserved for factory testing of the PWM module and is not available in normal modes.

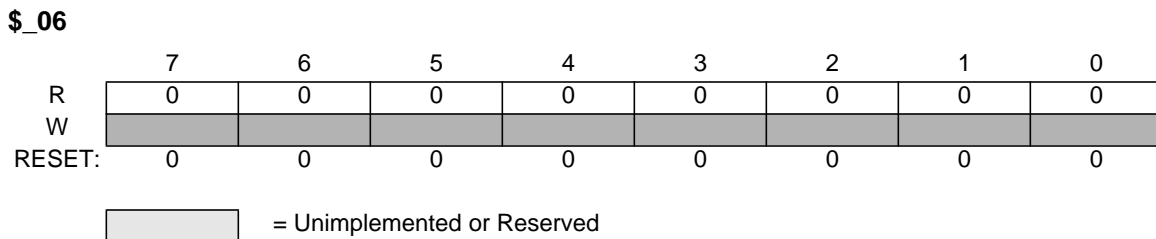


Figure 3-7 Reserved Register (PWMTST)

Read: always read \$00 in normal modes

Write: unimplemented in normal modes

NOTE: Writing to this register when in special modes can alter the PWM functionality.

3.3.8 Reserved Register (PWMPRSC)

This register is reserved for factory testing of the PWM module and is not available in normal modes.

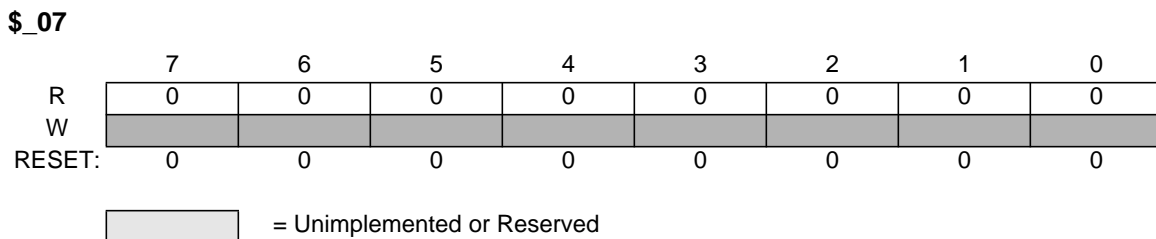


Figure 3-8 Reserved Register (PWMPRSC)

Read: always read \$00 in normal modes

Write: unimplemented in normal modes

NOTE: Writing to this register when in special modes can alter the PWM functionality.

3.3.9 PWM Scale A Register (PWMSCLA)

PWMSCLA is the programmable scale value used in scaling clock A to generate clock SA. Clock SA is generated by taking clock A, dividing it by the value in the PWMSCLA register and dividing that by two.

$$\text{Clock SA} = \text{Clock A} / (2 * \text{PWMSCLA})$$

NOTE: When PWMSCLA = \$00, PWMSCLA value is considered a full scale value of 256. Clock A is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLA).

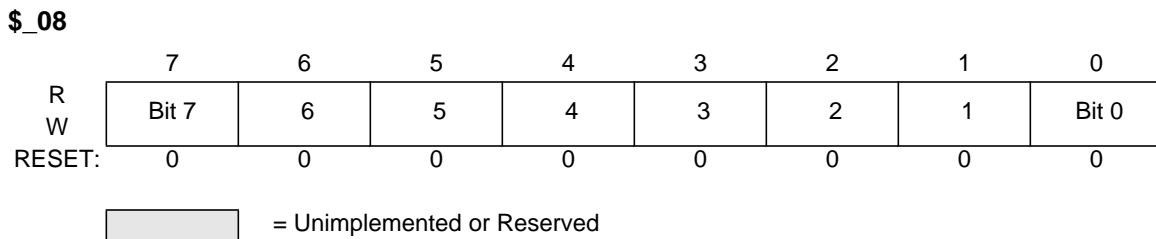


Figure 3-9 PWM Scale A Register (PWMSCLA)

Read: anytime

Write: anytime (causes the scale counter to load the PWMSCLA value)

3.3.10 PWM Scale B Register (PWMSCLB)

PWMSCLB is the programmable scale value used in scaling clock B to generate clock SB. Clock SB is generated by taking clock B, dividing it by the value in the PWMSCLB register and dividing that by two.

$$\text{Clock SB} = \text{Clock B} / (2 * \text{PWMSCLB})$$

NOTE: When PWMSCLB = \$00, PWMSCLB value is considered a full scale value of 256. Clock B is thus divided by 512.

Any value written to this register will cause the scale counter to load the new scale value (PWMSCLB).

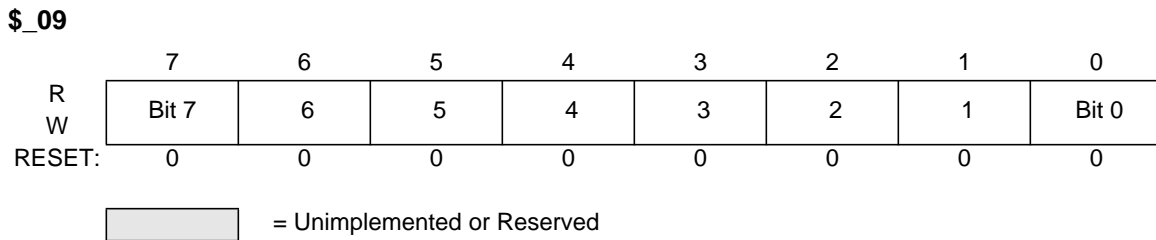


Figure 3-10 PWM Scale B Register (PWMSCLB)

Read: anytime

Write: anytime (causes the scale counter to load the PWMSCLB value).

3.3.11 Reserved Registers (PWMSCNTx)

The registers PWMSCNTA and PWMSCNTB are reserved for factory testing of the PWM module and are not available in normal modes.

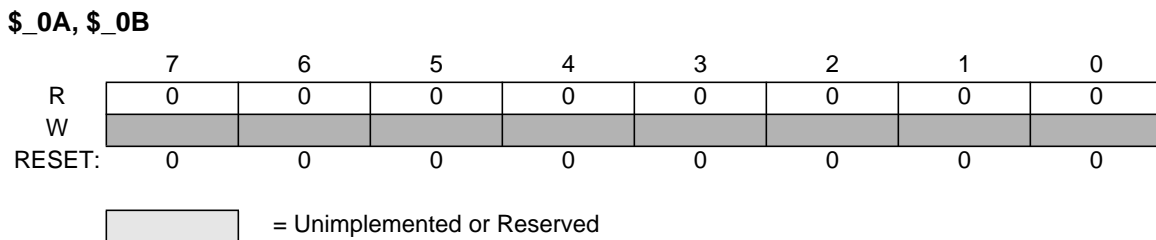


Figure 3-11 Reserved Registers (PWMSCNTx)

Read: always read \$00 in normal modes

Write: unimplemented in normal modes

NOTE: Writing to these registers when in special modes can alter the PWM functionality.

3.3.12 PWM Channel Counter Registers (PWMCNTx)

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source. The counter can be read at any time without affecting the count or the operation of the PWM channel. In left aligned output mode, the counter counts from 0 to the value in the period register - 1. In center aligned output mode, the counter counts from 0 up to the value in the period register and then back down to 0.

Any value written to the counter causes the counter to reset to \$00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. The counter is also cleared at the end of the effective period (see Sections 4.2.5 Left Aligned Outputs and 4.2.6 Center Aligned Outputs for more details). When the channel is disabled (PWME_x=0), the PWMCNT_x register does not count. When a channel becomes enabled

(PWME_x=1), the associated PWM counter starts at the count in the PWMCNT_x register. For more detailed information on the operation of the counters, reference **4.2.4 PWM Timer Counters**.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

NOTE: Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.

\$_0C = PWMCNT0
\$_0D = PWMCNT1
\$_0E = PWMCNT2
\$_0F = PWMCNT3
\$_10 = PWMCNT4
\$_11 = PWMCNT5
\$_12 = PWMCNT6
\$_13 = PWMCNT7

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W	0	0	0	0	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 3-12 PWM Channel Counter Registers (PWMCNT_x)

Read: anytime

Write: anytime (any value written causes PWM counter to be reset to \$00).

3.3.13 PWM Channel Period Registers (PWMPER_x)

There is a dedicated period register for each channel. The value in this register determines the period of the associated PWM channel.

The period registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to \$00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period register will go directly to the latches as well as the buffer.

NOTE: Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active period due to the double buffering scheme.

Reference **4.2.3 PWM Period and Duty** for more information.

To calculate the output period, take the selected clock source period for the channel of interest (A, B, SA, or SB) and multiply it by the value in the period register for that channel:

- Left Aligned Output (CAEx=0)
- $PWM_x \text{ Period} = \text{Channel Clock Period} * PWM_{PERx} \text{ Center Aligned Output (CAEx=1)}$
 $PWM_x \text{ Period} = \text{Channel Clock Period} * (2 * PWM_{PERx})$

For Boundary Case programming values, please refer to Section **4.2.8 PWM Boundary Cases**

- \$_14 = PWMPER0**
- \$_15 = PWMPER1**
- \$_16 = PWMPER2**
- \$_17 = PWMPER3**
- \$_18 = PWMPER4**
- \$_19 = PWMPER5**
- \$_1A = PWMPER6**
- \$_1B = PWMPER7**

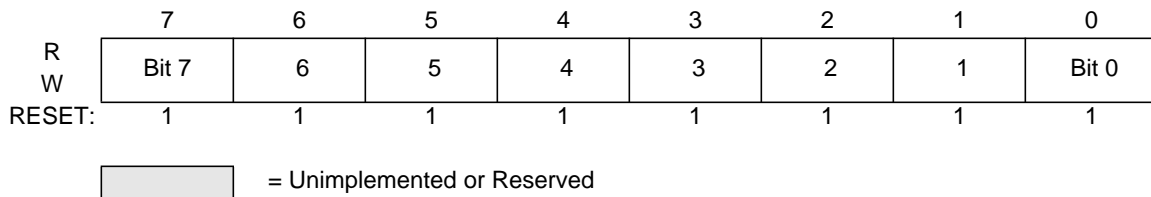


Figure 3-13 PWM Channel Period Registers (PWMPERx)

Read: anytime

Write: anytime

3.3.14 PWM Channel Duty Registers (PWMDTYx)

There is a dedicated duty register for each channel. The value in this register determines the duty of the associated PWM channel. The duty value is compared to the counter and if it is equal to the counter value a match occurs and the output changes state.

The duty registers for each channel are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to \$00)
- The channel is disabled

In this way, the output of the PWM will always be either the old duty waveform or the new duty waveform, not some variation in between. If the channel is not enabled, then writes to the duty register will go directly to the latches as well as the buffer.

NOTE: Reads of this register return the most recent value written. Reads do not necessarily return the value of the currently active duty due to the double buffering scheme.

Reference **4.2.3 PWM Period and Duty** for more information.

NOTE: Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time. If the polarity bit is one, the output starts high and then goes low when the duty count is reached, so the duty registers contain a count of the high time. If the polarity bit is zero, the output starts low and then goes high when the duty count is reached, so the duty registers contain a count of the low time.

To calculate the output duty cycle (high time as a% of period) for a particular channel:

- Polarity = 0 (PPOLx=0)

$$\text{Duty Cycle} = [(PWMPER_x - PWMDTY_x) / PWMPER_x] * 100\%$$
- Polarity = 1 (PPOLx=1)

$$\text{Duty Cycle} = [PWMDTY_x / PWMPER_x] * 100\%$$
- For Boundary Case programming values, please refer to Section **4.2.8 PWM Boundary Cases**.

\$_1C = PWMDTY0
 \$_1D = PWMDTY1
 \$_1E = PWMDTY2
 \$_1F = PWMDTY3
 \$_20 = PWMDTY4
 \$_21 = PWMDTY5
 \$_22 = PWMDTY6
 \$_23 = PWMDTY7

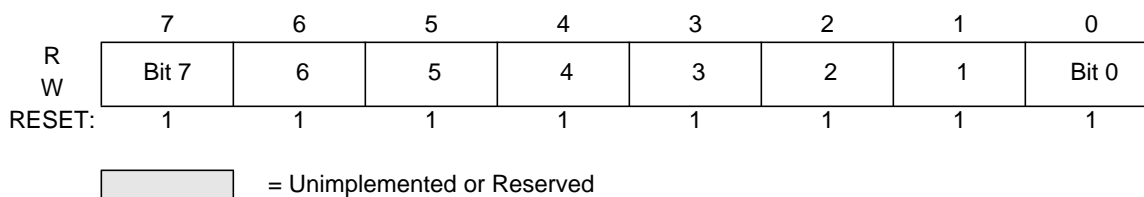


Figure 3-14 PWM Channel Duty Registers (PWMDTYx)

Read: anytime

Write: anytime

3.3.15 PWM Shutdown Register (PWMSDN)

The PWMSDN register provides for the shutdown functionality of the PWM module in the emergency cases. For proper operation channel 7 must be driven to the active level for a minimum of two bus clocks.



Figure 3-15 PWM Shutdown Register (PWMSDN)

Read: anytime

Write: anytime

PWM7ENA — PWM emergency shutdown Enable

If this bit is logic 1 the pin associated with channel 7 is forced to input and the emergency shutdown feature is enabled. All the other bits in this register are meaningful only if PWM7ENA = 1.

1 = PWM emergency feature is enabled.

0 = PWM emergency feature disabled.

PWM7INL — PWM shutdown active input level for channel 7.

If the emergency shutdown feature is enabled (PWM7ENA = 1), this bit determines the active level of the PWM7channel.

1 = Active level is high

0 = Active level is low

PWM7IN — PWM channel 7 input status.

This reflects the current status of the PWM7 pin.

PWMLVL — PWM shutdown output Level.

If active level as defined by the PWM7IN input, gets asserted all enabled PWM channels are immediately driven to the level defined by PWMLVL.

1 = PWM outputs are forced to 1.

0 = PWM outputs are forced to 0

PWMRSTRT — PWM Restart.

The PWM can only be restarted if the PWM channel input 7 is de-asserted. After writing a logic 1 to the PWMRSTRT bit (trigger event) the PWM channels start running after the corresponding counter passes next “counter == 0” phase.

Also if the PWM7ENA bit is reset to 0, the PWM do not start before the counter passes \$00.
The bit is always read as “0”.

PWMIE — PWM Interrupt Enable

If interrupt is enabled an interrupt to the CPU is asserted.

1 = PWM interrupt is enabled.

0 = PWM interrupt is disabled.

PWMIF — PWM Interrupt Flag

Any change from passive to asserted (active) state or from active to passive state will be flagged by setting the PWMIF flag = 1. The flag is cleared by writing a logic 1 to it. Writing a 0 has no effect.

1 = change on PWM7IN input

0 = No change on PWM7IN input.

Section 4 Functional Description

4.1 PWM Clock Select

There are four available clocks called clock A, clock B, clock SA (Scaled A), and clock SB (Scaled B). These four clocks are based on the bus clock.

Clock A and B can be software selected to be 1, 1/2, 1/4, 1/8,..., 1/64, 1/128 times the bus clock. Clock SA uses clock A as an input and divides it further with a reloadable counter. Similarly, Clock SB uses clock B as an input and divides it further with a reloadable counter. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB. Each PWM channel has the capability of selecting one of two clocks, either the pre-scaled clock (clock A or B) or the scaled clock (clock SA or SB).

The block diagram in **Figure 4-1** shows the four different clocks and how the scaled clocks are created.

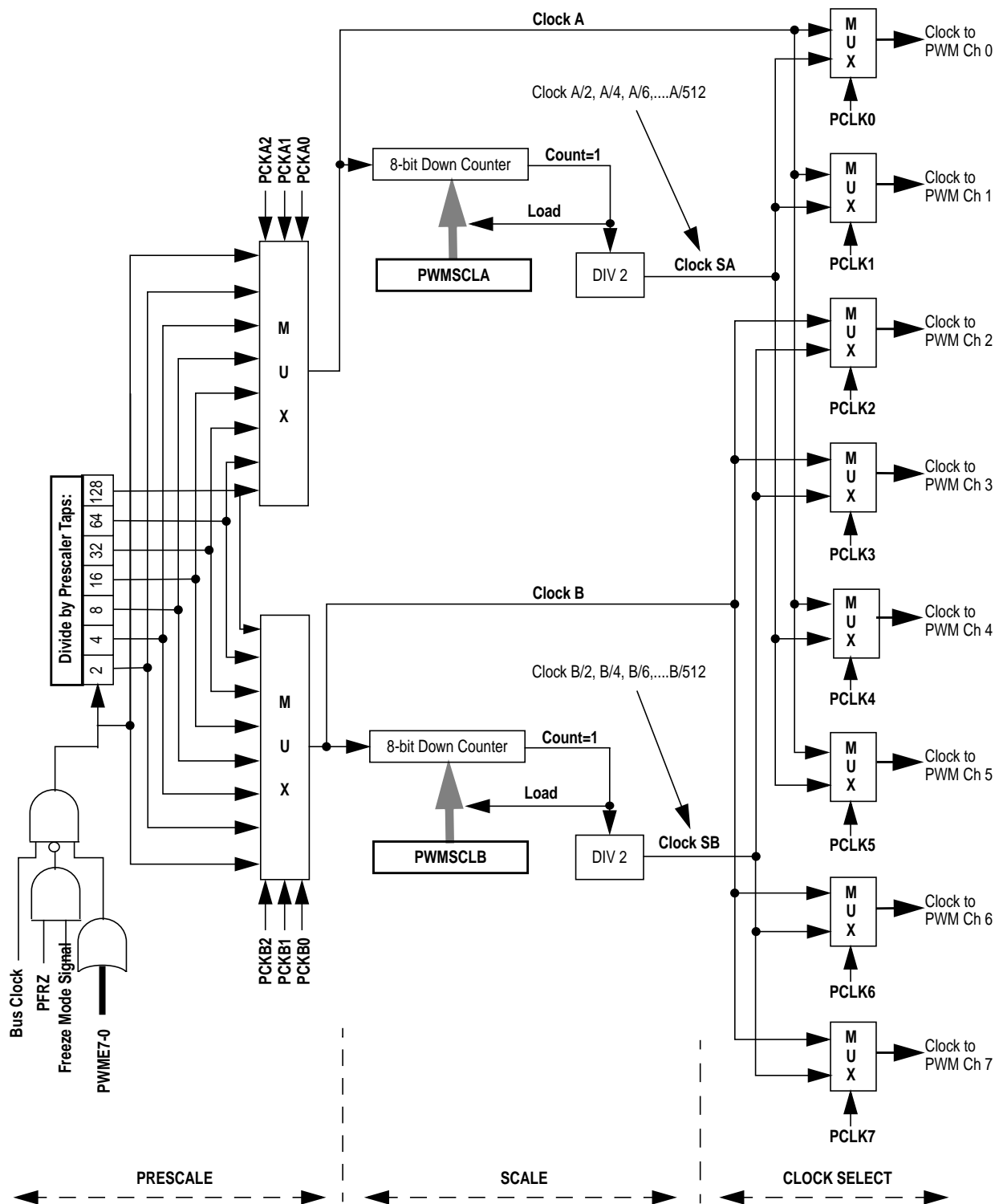


Figure 4-1 PWM Clock Select Block Diagram

4.1.1 Prescale

The input clock to the PWM prescaler is the bus clock. It can be disabled whenever the part is in freeze mode by setting the PFRZ bit in the PWMCTL register. If this bit is set, whenever the MCU is in freeze mode (Freeze Mode Signal active) the input clock to the prescaler is disabled. This is useful for emulation in order to freeze the PWM. The input clock can also be disabled when all eight PWM channels are disabled (PWME7-0=0). This is useful for reducing power by disabling the prescale counter.

Clock A and clock B are scaled values of the input clock. The value is software selectable for both clock A and clock B and has options of 1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, or 1/128 times the bus clock. The value selected for clock A is determined by the PCKA2, PCKA1, PCKA0 bits in the PWMPRCLK register. The value selected for clock B is determined by the PCKB2, PCKB1, PCKB0 bits also in the PWMPRCLK register.

4.1.2 Clock Scale

The scaled A clock uses clock A as an input and divides it further with a user programmable value and then divides this by 2. The scaled B clock uses clock B as an input and divides it further with a user programmable value and then divides this by 2. The rates available for clock SA are software selectable to be clock A divided by 2, 4, 6, 8,..., or 512 in increments of divide by 2. Similar rates are available for clock SB.

Clock A is used as an input to an 8-bit down counter. This down counter loads a user programmable scale value from the scale register (PWMSCLA). When the down counter reaches one, two things happen; a pulse is output and the 8-bit counter is re-loaded. The output signal from this circuit is further divided by two. This gives a greater range with only a slight reduction in granularity. Clock SA equals Clock A divided by two times the value in the PWMSCLA register.

NOTE: $\text{Clock SA} = \text{Clock A} / (2 * \text{PWMSCLA})$

*When PWMSCLA = \$00, PWMSCLA value is considered a full scale value of 256.
Clock A is thus divided by 512.*

Similarly, Clock B is used as an input to an 8-bit down counter followed by a divide by two producing clock SB. Thus, clock SB equals Clock B divided by two times the value in the PWMSCLB register.

NOTE: $\text{Clock SB} = \text{Clock B} / (2 * \text{PWMSCLB})$

*When PWMSCLB = \$00, PWMSCLB value is considered a full scale value of 256.
Clock B is thus divided by 512.*

As an example, consider the case in which the user writes \$FF into the PWMSCLA register. Clock A for this case will be E divided by 4. A pulse will occur at a rate of once every 255x4 E cycles. Passing this through the divide by two circuit produces a clock signal at an E divided by 2040 rate. Similarly, a value of \$01 in the PWMSCLA register when clock A is E divided by 4 will produce a clock at an E divided by 8 rate.

Writing to PWMSCLA or PWMSCLB causes the associated 8-bit down counter to be re-loaded. Otherwise, when changing rates the counter would have to count down to \$01 before counting at the proper rate. Forcing the associated counter to re-load the scale register value every time PWMSCLA or PWMSCLB is written prevents this.

NOTE: *Writing to the scale registers while channels are operating can cause irregularities in the PWM outputs.*

4.1.3 Clock Select

Each PWM channel has the capability of selecting one of two clocks. For channels 0, 1, 4, and 5 the clock choices are clock A or clock SA. For channels 2, 3, 6, and 7 the choices are clock B or clock SB. The clock selection is done with the PCLKx control bits in the PWMCLK register.

NOTE: *Changing clock control bits while channels are operating can cause irregularities in the PWM outputs.*

4.2 PWM Channel Timers

The main part of the PWM module are the actual timers. Each of the timer channels has a counter, a period register and a duty register (each are 8-bit). The waveform output period is controlled by a match between the period register and the value in the counter. The duty is controlled by a match between the duty register and the counter value and causes the state of the output to change during the period. The starting polarity of the output is also selectable on a per channel basis. Shown below in **Figure 4-2** is the block diagram for the PWM timer.

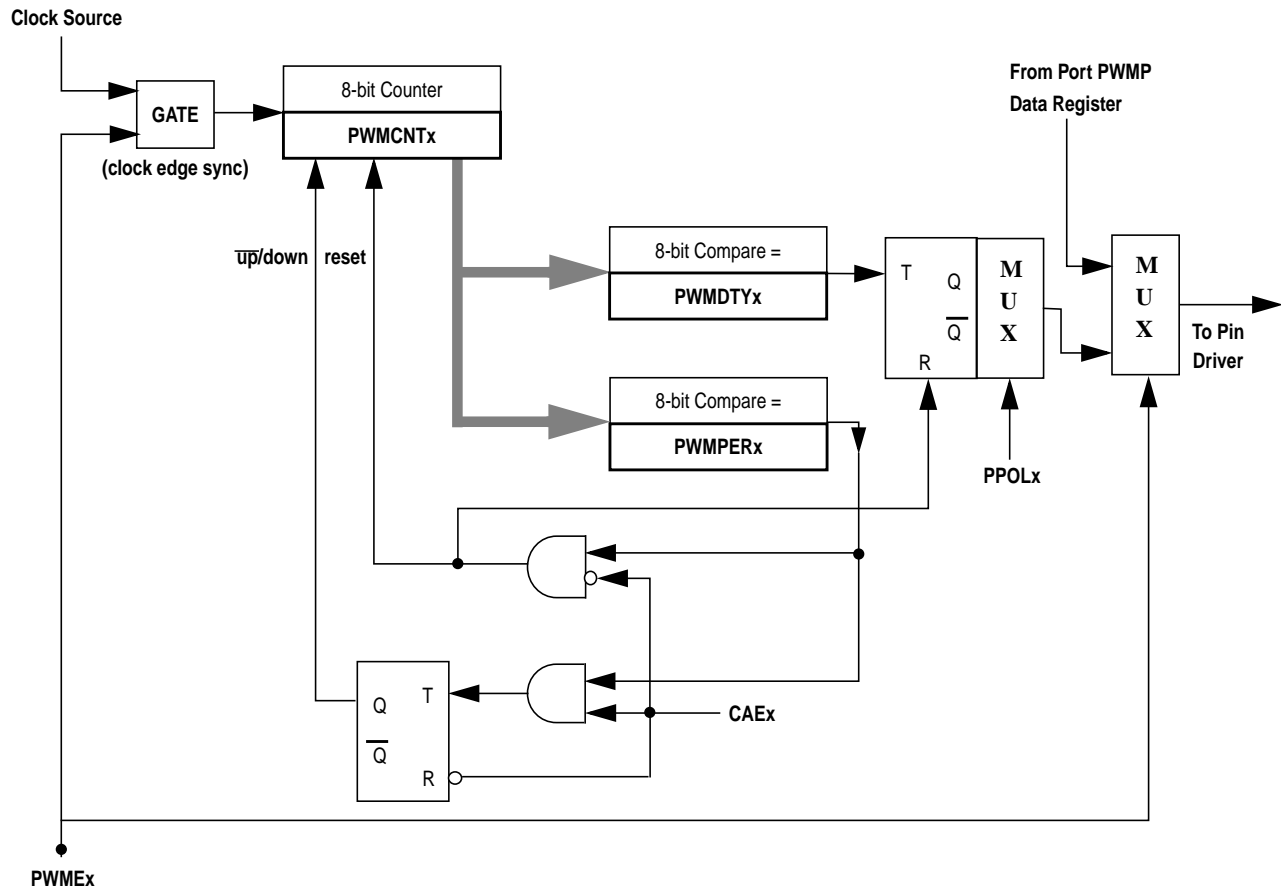


Figure 4-2 PWM Timer Channel Block Diagram

4.2.1 PWM Enable

Each PWM channel has an enable bit (PWME_x) to start its waveform output. When any of the PWME_x bits are set (PWME_x=1), the associated PWM output signal is enabled immediately. However, the actual PWM waveform is not available on the associated PWM output until its clock source begins its next cycle due to the synchronization of PWME_x and the clock source. An exception to this is when channels are concatenated. Refer to **Section 4.2.7 PWM 16-Bit Functions** for more detail.

NOTE: *The first PWM cycle after enabling the channel can be irregular.*

On the front end of the PWM timer, the clock is enabled to the PWM circuit by the PWME_x bit being high. There is an edge-synchronizing circuit to guarantee that the clock will only be enabled or disabled at an edge. When the channel is disabled (PWME_x=0), the counter for the channel does not count.

4.2.2 PWM Polarity

Each channel has a polarity bit to allow starting a waveform cycle with a high or low signal. This is shown on the block diagram as a Mux select of either the Q output or the \bar{Q} output of the PWM output flip flop.

When one of the bits in the PWMPOL register is set, the associated PWM channel output is high at the beginning of the waveform, then goes low when the duty count is reached. Conversely, if the polarity bit is zero, the output starts low and then goes high when the duty count is reached.

4.2.3 PWM Period and Duty

Dedicated period and duty registers exist for each channel and are double buffered so that if they change while the channel is enabled, the change will NOT take effect until one of the following occurs:

- The effective period ends
- The counter is written (counter resets to \$00)
- The channel is disabled

In this way, the output of the PWM will always be either the old waveform or the new waveform, not some variation in between. If the channel is not enabled, then writes to the period and duty registers will go directly to the latches as well as the buffer.

A change in duty or period can be forced into effect “immediately” by writing the new value to the duty and/or period registers and then writing to the counter. This forces the counter to reset and the new duty and/or period values to be latched. In addition, since the counter is readable it is possible to know where the count is with respect to the duty value and software can be used to make adjustments

NOTE: *When forcing a new period or duty into effect immediately, an irregular PWM cycle can occur.*

NOTE: *Depending on the polarity bit, the duty registers will contain the count of either the high time or the low time.*

4.2.4 PWM Timer Counters

Each channel has a dedicated 8-bit up/down counter which runs at the rate of the selected clock source (reference **Section 4.1 PWM Clock Select** for the available clock sources and rates). The counter compares to two registers, a duty register and a period register as shown in **Figure 4-2 PWM Timer Channel Block Diagram**. When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register behaves differently depending on what output mode is selected as shown in **Figure 4-2 PWM Timer Channel Block Diagram** and described in **Section 4.2.5 Left Aligned Outputs** and **Section 4.2.6 Center Aligned Outputs**.

Each channel counter can be read at anytime without affecting the count or the operation of the PWM channel.

Any value written to the counter causes the counter to reset to \$00, the counter direction to be set to up, the immediate load of both duty and period registers with values from the buffers, and the output to change according to the polarity bit. When the channel is disabled (PWME_x=0), the counter stops. When a channel becomes enabled (PWME_x=1), the associated PWM counter continues from the count in the PWMCNT_x register. This allows the waveform to continue where it left off when the channel is

re-enabled. When the channel is disabled, writing “0” to the period register will cause the counter to reset on the next selected clock.

NOTE: *If the user wants to start a new “clean” PWM waveform without any “history” from the old waveform, the user must write to channel counter (PWMCNTx) prior to enabling the PWM channel (PWME_x=1).*

Generally, writes to the counter are done prior to enabling a channel in order to start from a known state. However, writing a counter can also be done while the PWM channel is enabled (counting). The effect is similar to writing the counter when the channel is disabled except that the new period is started immediately with the output set according to the polarity bit.

NOTE: *Writing to the counter while the channel is enabled can cause an irregular PWM cycle to occur.*

The counter is cleared at the end of the effective period (see **Section 4.2.5 Left Aligned Outputs** and **Section 4.2.6 Center Aligned Outputs** for more details).

Table 4-1 PWM Timer Counter Conditions

Counter Clears (\$00)	Counter Counts	Counter Stops
When PWMCNTx register written to any value	When PWM channel is enabled (PWME _x =1). Counts from last value in PWMCNTx.	When PWM channel is disabled (PWME _x =0)
Effective period ends		

4.2.5 Left Aligned Outputs

The PWM timer provides the choice of two types of outputs, Left Aligned or Center Aligned outputs. They are selected with the CAEx bits in the PWMCAE register. If the CAEx bit is cleared (CAEx=0), the corresponding PWM output will be left aligned.

In left aligned output mode, the 8-bit counter is configured as an up counter only. It compares to two registers, a duty register and a period register as shown in the block diagram in **Figure 4-2 PWM Timer Channel Block Diagram**. When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register resets the counter and the output flip-flop as shown in **Figure 4-2 PWM Timer Channel Block Diagram** as well as performing a load from the double buffer period and duty register to the associated registers as described in **Section 4.2.3 PWM Period and Duty**. The counter counts from 0 to the value in the period register - 1.

NOTE: *Changing the PWM output mode from Left Aligned Output to Center Aligned Output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.*

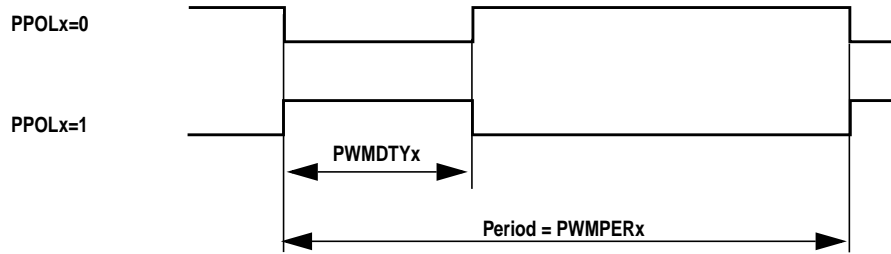


Figure 4-3 PWM Left Aligned Output Waveform

To calculate the output frequency in left aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by the value in the period register for that channel.

- $PWMx \text{ Frequency} = \text{Clock}(A, B, SA, \text{ or } SB) / PWMPERx$
- $PWMx \text{ Duty Cycle (high time as a\% of period):}$
 - Polarity = 0 (PPOLx=0)
 $\text{Duty Cycle} = [(PWMPERx - PWMDTYx) / PWMPERx] * 100\%$
 - Polarity = 1 (PPOLx=1)
 $\text{Duty Cycle} = [PWMDTYx / PWMPERx] * 100\%$

As an example of a left aligned output, consider the following case:

Clock Source = E, where E=10MHz (100ns period)

PPOLx = 0

PWMPERx = 4

PWMDTYx = 1

$PWMx \text{ Frequency} = 10\text{MHz} / 4 = 2.5\text{MHz}$

$PWMx \text{ Period} = 400\text{ns}$

$PWMx \text{ Duty Cycle} = 3/4 * 100\% = 75\%$

Shown below is the output waveform generated.

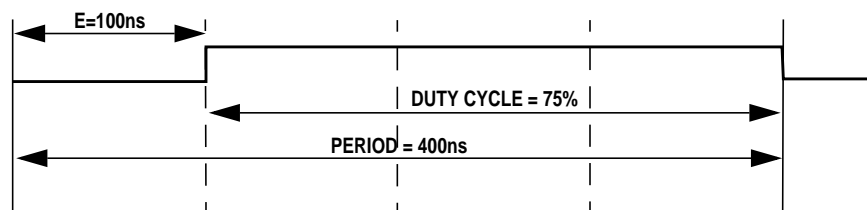


Figure 4-4 PWM Left Aligned Output Example Waveform

4.2.6 Center Aligned Outputs

For Center Aligned Output Mode selection, set the CAEx bit (CAEx=1) in the PWMCAE register and the corresponding PWM output will be center aligned.

The 8-bit counter operates as an up/down counter in this mode and is set to up whenever the counter is equal to \$00. The counter compares to two registers, a duty register and a period register as shown in the block diagram in **Figure 4-2 PWM Timer Channel Block Diagram**. When the PWM counter matches the duty register the output flip-flop changes state causing the PWM waveform to also change state. A match between the PWM counter and the period register changes the counter direction from an up-count to a down-count. When the PWM counter decrements and matches the duty register again, the output flip-flop changes state causing the PWM output to also change state. When the PWM counter decrements and reaches zero, the counter direction changes from a down-count back to an up-count and a load from the double buffer period and duty registers to the associated registers is performed as described in **Section 4.2.3 PWM Period and Duty**. The counter counts from 0 up to the value in the period register and then back down to 0. Thus the effective period is $PWMPER_x * 2$.

NOTE: Changing the PWM output mode from Left Aligned Output to Center Aligned Output (or vice versa) while channels are operating can cause irregularities in the PWM output. It is recommended to program the output mode before enabling the PWM channel.

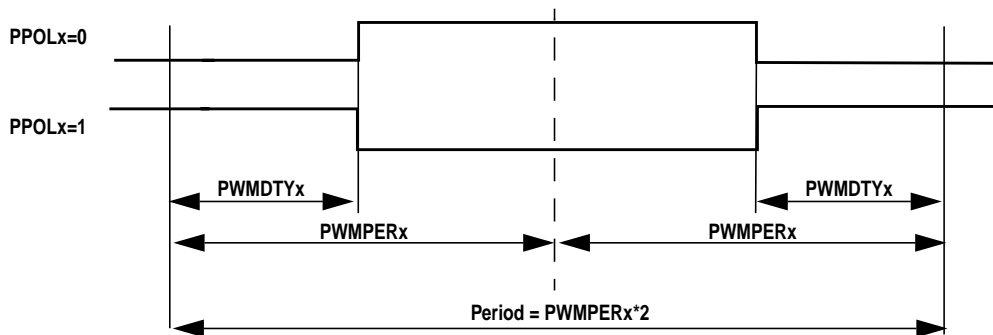


Figure 4-5 PWM Center Aligned Output Waveform

To calculate the output frequency in center aligned output mode for a particular channel, take the selected clock source frequency for the channel (A, B, SA, or SB) and divide it by twice the value in the period register for that channel.

- $PWM_x \text{ Frequency} = \text{Clock}(A, B, SA, \text{ or } SB) / (2 * PWMPER_x)$
- PWMx Duty Cycle (high time as a% of period):
 - Polarity = 0 (PPOLx=0)
Duty Cycle = $[(PWMPER_x - PWMDTY_x) / PWMPER_x] * 100\%$
 - Polarity = 1 (PPOLx=1)

$$\text{Duty Cycle} = [\text{PWMDTY}_x / \text{PWMPER}_x] * 100\%$$

As an example of a center aligned output, consider the following case:

Clock Source = E, where E=10MHz (100ns period)

PPOL_x = 0

PWMPER_x = 4

PWMDTY_x = 1

PWM_x Frequency = 10MHz/8 = 1.25MHz

PWM_x Period = 800ns

PWM_x Duty Cycle = 3/4 * 100% = 75%

Shown below is the output waveform generated.

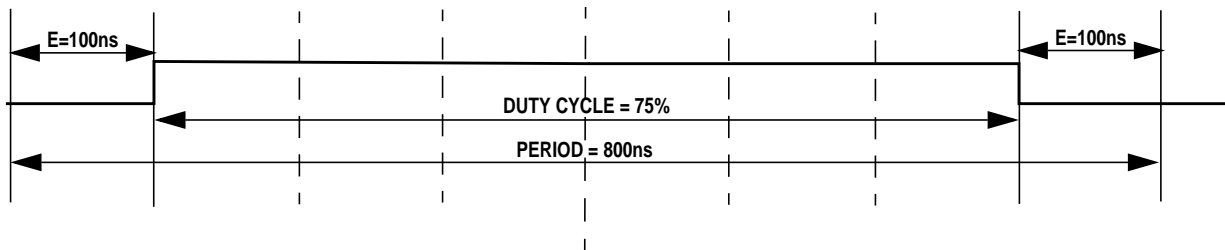


Figure 4-6 PWM Center Aligned Output Example Waveform

4.2.7 PWM 16-Bit Functions

The PWM timer also has the option of generating 8-channels of 8-bits or 4-channels of 16-bits for greater PWM resolution. This 16-bit channel option is achieved through the concatenation of two 8-bit channels.

The PWMCTL register contains four control bits, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. Channels 6 and 7 are concatenated with the CON67 bit, channels 4 and 5 are concatenated with the CON45 bit, channels 2 and 3 are concatenated with the CON23 bit, and channels 0 and 1 are concatenated with the CON01 bit.

NOTE: Change these bits only when both corresponding channels are disabled.

When channels 6 and 7 are concatenated, channel 6 registers become the high order bytes of the double byte channel as shown in **Figure 4-7 PWM 16-Bit Mode**. Similarly, when channels 4 and 5 are concatenated, channel 4 registers become the high order bytes of the double byte channel. When channels 2 and 3 are concatenated, channel 2 registers become the high order bytes of the double byte channel. When channels 0 and 1 are concatenated, channel 0 registers become the high order bytes of the double

byte channel.

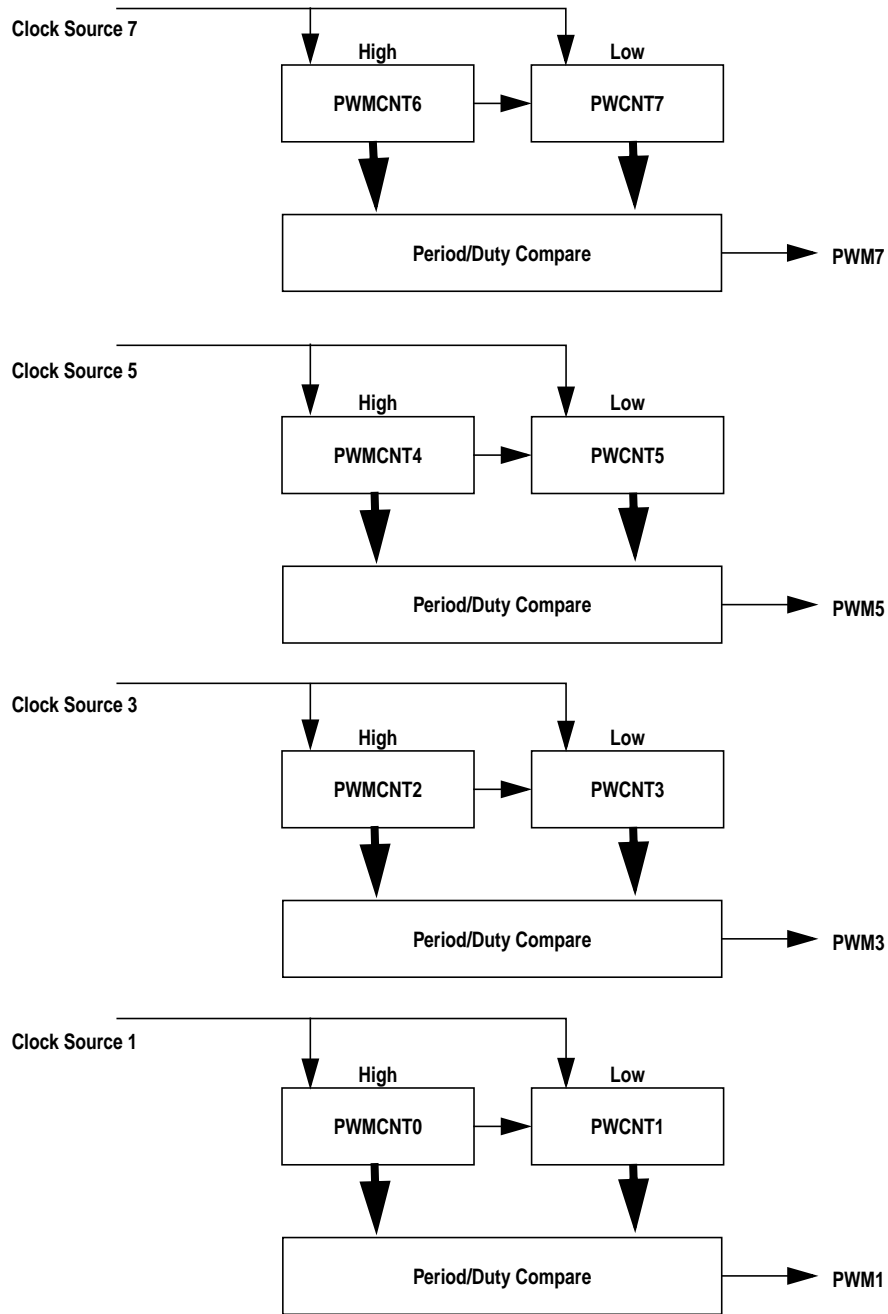


Figure 4-7 PWM 16-Bit Mode

When using the 16-bit concatenated mode, the clock source is determined by the low order 8-bit channel clock select control bits. That is channel 7 when channels 6 and 7 are concatenated, channel 5 when channels 4 and 5 are concatenated, channel 3 when channels 2 and 3 are concatenated, and channel 1 when channels 0 and 1 are concatenated. The resulting PWM is output to the pins of the corresponding low order

8-bit channel as also shown in **Figure 4-7 PWM 16-Bit Mode**. The polarity of the resulting PWM output is controlled by the PPOLx bit of the corresponding low order 8-bit channel as well.

Once concatenated mode is enabled (CONxx bits set in PWMCTL register) then enabling/disabling the corresponding 16-bit PWM channel is controlled by the low order PWMEx bit. In this case, the high order bytes PWMEx bits have no effect and their corresponding PWM output is disabled.

In concatenated mode, writes to the 16-bit counter by using a 16-bit access or writes to either the low or high order byte of the counter will reset the 16-bit counter. Reads of the 16-bit counter must be made by 16-bit access to maintain data coherency.

Either left aligned or center aligned output mode can be used in concatenated mode and is controlled by the low order CAEx bit. The high order CAEx bit has no effect.

The table shown below is used to summarize which channels are used to set the various control bits when in 16-bit mode.

Table 4-2 16-bit Concatenation Mode Summary

CONxx	PWMEx	PPOLx	PCLKx	CAEx	PWMx OUTPUT
CON67	PWME7	PPOL7	PCLK7	CAE7	PWM7
CON45	PWME5	PPOL5	PCLK5	CAE5	PWM5
CON23	PWME3	PPOL3	PCLK3	CAE3	PWM3
CON01	PWME1	PPOL1	PCLK1	CAE1	PWM1

4.2.8 PWM Boundary Cases

The following table summarizes the boundary conditions for the PWM regardless of the output mode (Left Aligned or Center Aligned) and 8-bit (normal) or 16-bit (concatenation).

Table 4-3 PWM Boundary Cases

PWMDTYx	PWMPERx	PPOLx	PWMx Output
\$00 (indicates no duty)	>\$00	1	Always Low
\$00 (indicates no duty)	>\$00	0	Always High
XX	\$00 ¹ (indicates no period)	1	Always High
XX	\$00 ¹ (indicates no period)	0	Always Low
>= PWMPERx	XX	1	Always High
>= PWMPERx	XX	0	Always Low

NOTES:

1. Counter=\$00 and does not count.

Section 5 Resets

5.1 General

The reset state of each individual bit is listed within the Register Description section **Section 3.3 Register Descriptions** which details the registers and their bit-fields. All special functions or modes which are initialized during or just following reset are described within this section.

- The 8-bit up/down counter is configured as an up counter out of reset.
- All the channels are disabled and all the counters don't count.

Section 6 Interrupts

6.1 Interrupt Operation

The PWM module has only one interrupt which is generated at the time of emergency shutdown, if the corresponding enable bit (PWMIE) is set. This bit is the enable for the interrupt. The interrupt flag PWMIF is set whenever the input level of the PWM7 channel changes while PWM7ENA=1 or when PWMENA is being asserted while the level at PWM7 is active.

In STOP mode or WAIT mode (with the PSWAI bit set) the emergency shutdown feature will drive the PWM outputs to their shutdown output levels but the PWMIF flag will not be set.

A description of the registers involved and affected due to this interrupt is explained in **Section 3.3.15 PWM Shutdown Register (PWMSDN)**.

The PWM block only generates the interrupt and does not service it. The interrupt signal name is PWM Interrupt Signal.

User Guide End Sheet

**FINAL PAGE OF
54
PAGES**

Chapter 1

Serial Communications Interface (S12SCIV2)

Block Description

1.1 Introduction

This block guide provide an overview of serial communication interface (SCI) module. The SCI allows asynchronous serial communications with peripheral devices and other CPUs.

1.1.1 Glossary

IRQ — Interrupt Request

LSB — Least Significant Bit

MSB — Most Significant Bit

NRZ — Non-Return-to-Zero

RZI — Return-to-Zero-Inverted

RXD — Receive Pin

SCI — Serial Communication Interface

TXD — Transmit Pin

1.1.2 Features

The SCI includes these distinctive features:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable transmitter output parity
- Two receiver wake up methods:
 - Idle line wake-up
 - Address mark wake-up
- Interrupt-driven operation with eight flags:
 - Transmitter empty

- Transmission complete
- Receiver full
- Idle receiver input
- Receiver overrun
- Noise error
- Framing error
- Parity error
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

1.1.3 Modes of Operation

The SCI operation is the same independent of device resource mapping and bus interface mode. Different power modes are available to facilitate power saving.

1.1.3.1 Run Mode

Normal mode of operation.

1.1.3.2 Wait Mode

SCI operation in wait mode depends on the state of the SCISWAI bit in the SCI control register 1 (SCICR1).

- If SCISWAI is clear, the SCI operates normally when the CPU is in wait mode.
- If SCISWAI is set, SCI clock generation ceases and the SCI module enters a power-conservation state when the CPU is in wait mode. Setting SCISWAI does not affect the state of the receiver enable bit, RE, or the transmitter enable bit, TE.
- If SCISWAI is set, any transmission or reception in progress stops at wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of wait mode. Exiting wait mode by reset aborts any transmission or reception in progress and resets the SCI.

1.1.3.3 Stop Mode

The SCI is inactive during stop mode for reduced power consumption. The STOP instruction does not affect the SCI register states, but the SCI module clock will be disabled. The SCI operation resumes from where it left off after an external interrupt brings the CPU out of stop mode. Exiting stop mode by reset aborts any transmission or reception in progress and resets the SCI.

1.1.4 Block Diagram

Figure 1-1 is a high level block diagram of the SCI module, showing the interaction of various functional blocks.

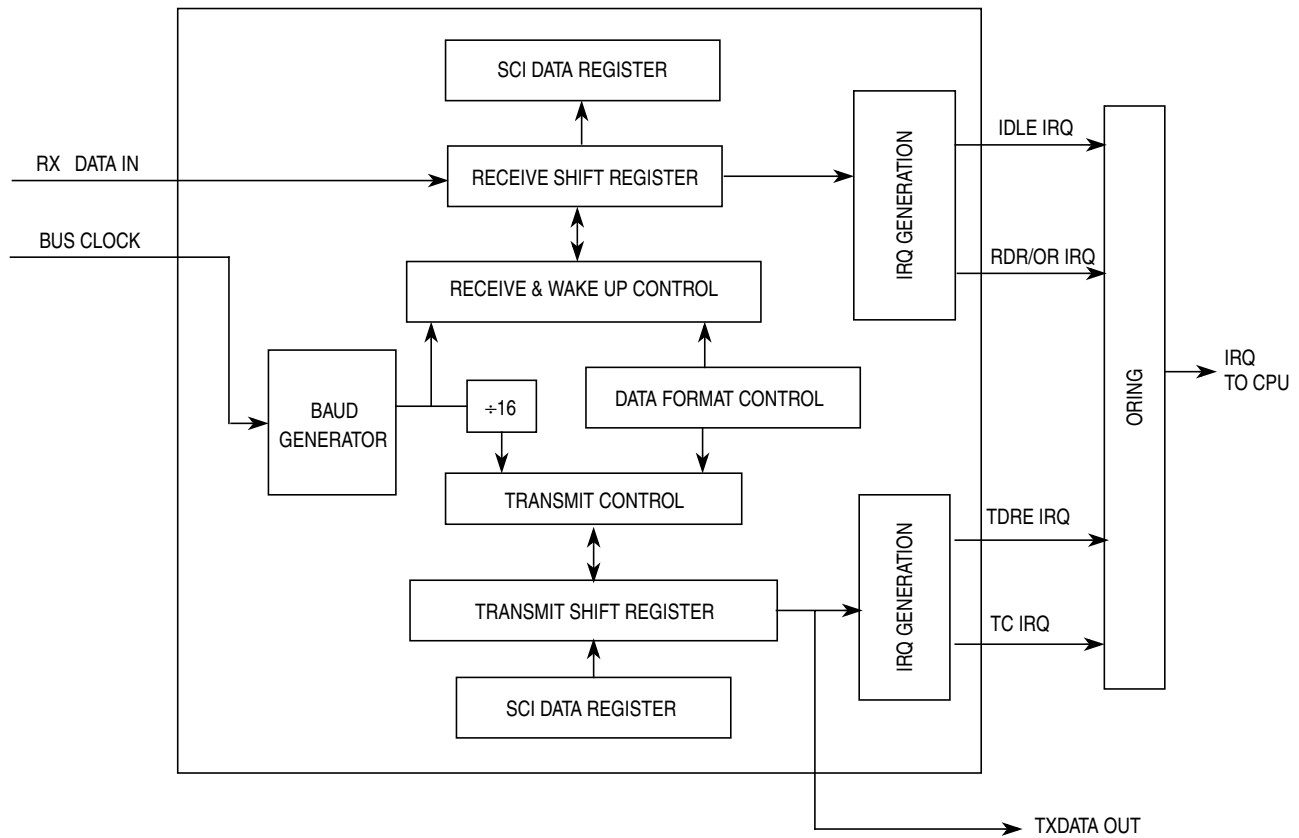


Figure 1-1. SCI Block Diagram

1.2 External Signal Description

The SCI module has a total of two external pins:

1.2.1 TXD-SCI Transmit Pin

This pin serves as transmit data output of SCI.

1.2.2 RXD-SCI Receive Pin

This pin serves as receive data input of the SCI.

1.3 Memory Map and Registers

This section provides a detailed description of all memory and registers.

1.3.1 Module Memory Map

The memory map for the SCI module is given below in [Figure 1-2](#). The Address listed for each register is the address offset. The total address for each register is the sum of the base address for the SCI module and the address offset for each register.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	SCIBDH	R	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
		W								
0x0001	SCIBDL	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		W								
0x0002	SCICR1	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		W								
0x0003	SCICR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		W								
0x0004	SCISR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		W								
0x0005	SCISR2	R	0	0	0	0	0	BRK13	TXDIR	RAF
		W								
0x0006	SCIDRH	R	R8	T8	0	0	0	0	0	0
		W								
0x0007	SCIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

= Unimplemented or Reserved

Figure 1-2. SCI Register Summary

1.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Writes to a reserved register location do not have any effect and reads of these locations return a zero. Details of register bit and field function follow the register diagrams, in bit order.

1.3.2.1 SCI Baud Rate Registers (SCIBDH and SCHBDL)

Module Base + 0x_0000

	7	6	5	4	3	2	1	0
R	0	0	0	SBR12	SBR11	SBR10	SBR9	SBR8
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x_0001

	7	6	5	4	3	2	1	0
R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
W								
Reset	0	0	0	0	0	1	0	0

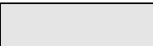
 = Unimplemented or Reserved

Figure 1-3. SCI Baud Rate Registers (SCIBDH and SCIBDL)

The SCI Baud Rate Register is used by the counter to determine the baud rate of the SCI. The formula for calculating the baud rate is:

$$\text{SCI baud rate} = \text{SCI module clock} / (16 \times \text{BR})$$

where:

BR is the content of the SCI baud rate registers, bits SBR12 through SBR0. The baud rate registers can contain a value from 1 to 8191.

Read: Anytime. If only SCIBDH is written to, a read will not return the correct data until SCIBDL is written to as well, following a write to SCIBDH.

Write: Anytime

Table 1-1. SCIBDH AND SCIBDL Field Descriptions

Field	Description
4–0 7–0 SBR[12:0]	<p>SCI Baud Rate Bits — The baud rate for the SCI is determined by these 13 bits.</p> <p>Note: The baud rate generator is disabled until the TE bit or the RE bit is set for the first time after reset. The baud rate generator is disabled when BR = 0.</p> <p>Writing to SCIBDH has no effect without writing to SCIBDL, since writing to SCIBDH puts the data in a temporary location until SCIBDL is written to.</p>

1.3.2.2 SCI Control Register 1 (SCICR1)

Module Base + 0x_0002

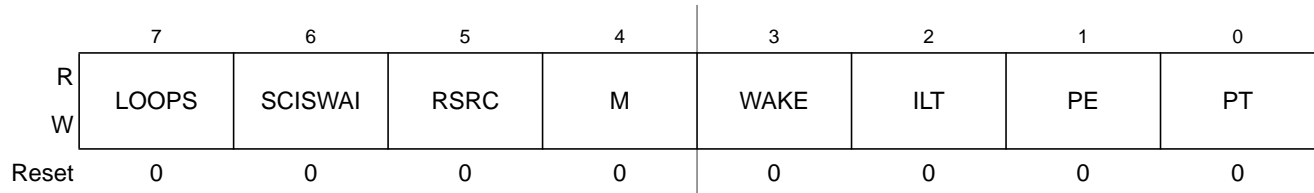


Figure 1-4. SCI Control Register 1 (SCICR1)

Read: Anytime

Write: Anytime

Table 1-2. SCICR1 Field Descriptions

Field	Description
7 LOOPS	Loop Select Bit — LOOPS enables loop operation. In loop operation, the RXD pin is disconnected from the SCI and the transmitter output is internally connected to the receiver input. Both the transmitter and the receiver must be enabled to use the loop function. See Table 1-3. 0 Normal operation enabled 1 Loop operation enabled Note: The receiver input is determined by the RSRC bit.
6 SCISWAI	SCI Stop in Wait Mode Bit — SCISWAI disables the SCI in wait mode. 0 SCI enabled in wait mode 1 SCI disabled in wait mode
5 RSRC	Receiver Source Bit — When LOOPS = 1, the RSRC bit determines the source for the receiver shift register input. 0 Receiver input internally connected to transmitter output 1 Receiver input connected externally to transmitter
4 M	Data Format Mode Bit — MODE determines whether data characters are eight or nine bits long. 0 One start bit, eight data bits, one stop bit 1 One start bit, nine data bits, one stop bit
3 WAKE	Wakeup Condition Bit — WAKE determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received data character or an idle condition on the RXD. 0 Idle line wakeup 1 Address mark wakeup
2 ILT	Idle Line Type Bit — ILT determines when the receiver starts counting logic 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. 0 Idle character bit count begins after start bit 1 Idle character bit count begins after stop bit
1 PE	Parity Enable Bit — PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position. 0 Parity function disabled 1 Parity function enabled
0 PT	Parity Type Bit — PT determines whether the SCI generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity 1 Odd parity

Table 1-3. Loop Functions

LOOPS	RSRC	Function
0	x	Normal operation
1	0	Loop mode with Rx input internally connected to Tx output
1	1	Single-wire mode with Rx input connected to TXD

1.3.2.3 SCI Control Register 2 (SCICR2)

Module Base + 0x_0003

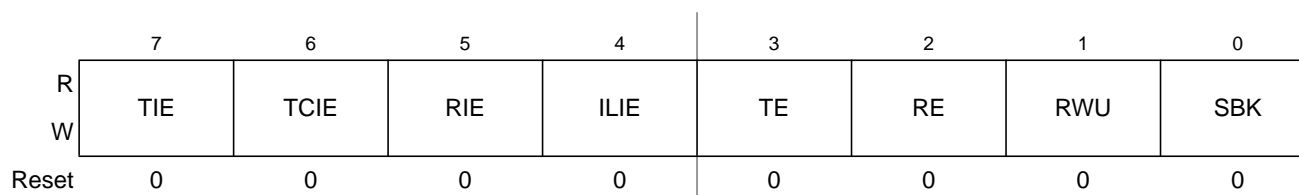


Figure 1-5. SCI Control Register 2 (SCICR2)

Read: Anytime

Write: Anytime

Table 1-4. SCICR2 Field Descriptions

Field	Description
7 TIE	Transmitter Interrupt Enable Bit — TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests. 0 TDRE interrupt requests disabled 1 TDRE interrupt requests enabled
6 TCIE	Transmission Complete Interrupt Enable Bit — TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 TC interrupt requests disabled 1 TC interrupt requests enabled
5 RIE	Receiver Full Interrupt Enable Bit — RIE enables the receive data register full flag, RDRF, or the overrun flag, OR, to generate interrupt requests. 0 RDRF and OR interrupt requests disabled 1 RDRF and OR interrupt requests enabled
4 ILIE	Idle Line Interrupt Enable Bit — ILIE enables the idle line flag, IDLE, to generate interrupt requests. 0 IDLE interrupt requests disabled 1 IDLE interrupt requests enabled
3 TE	Transmitter Enable Bit — TE enables the SCI transmitter and configures the TXD pin as being controlled by the SCI. The TE bit can be used to queue an idle preamble. 0 Transmitter disabled 1 Transmitter enabled
2 RE	Receiver Enable Bit — RE enables the SCI receiver. 0 Receiver disabled 1 Receiver enabled

Table 1-4. SCICR2 Field Descriptions (continued)

Field	Description
1 RWU	Receiver Wakeup Bit — Standby state 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	Send Break Bit — Toggling SBK sends one break character (10 or 11 logic 0s, respectively 13 or 14 logics 0s if BRK13 is set). Toggling implies clearing the SBK bit before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10 or 11 bits, respectively 13 or 14 bits). 0 No break characters 1 Transmit break characters

1.3.2.4 SCI Status Register 1 (SCISR1)

The SCISR1 and SCISR2 registers provides inputs to the MCU for generation of SCI interrupts. Also, these registers can be polled by the MCU to check the status of these bits. The flag-clearing procedures require that the status register be read followed by a read or write to the SCI Data Register. It is permissible to execute other instructions between the two steps as long as it does not compromise the handling of I/O, but the order of operations is important for flag clearing.

Module Base + 0x_0004

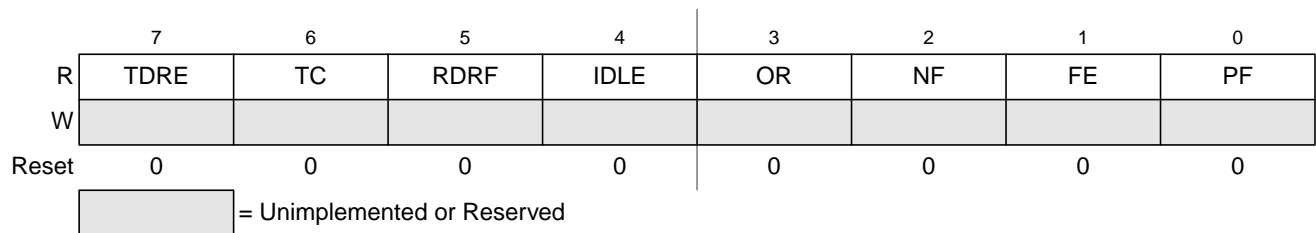


Figure 1-6. SCI Status Register 1 (SCISR1)

Read: Anytime

Write: Has no meaning or effect

Table 1-5. SCISR1 Field Descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag — TDRE is set when the transmit shift register receives a byte from the SCI data register. When TDRE is 1, the transmit data register (SCIDRH/L) is empty and can receive a new value to transmit. Clear TDRE by reading SCI status register 1 (SCISR1), with TDRE set and then writing to SCI data register low (SCIDRL). 0 No byte transferred to transmit shift register 1 Byte transferred to transmit shift register; transmit data register empty
6 TC	Transmit Complete Flag — TC is set low when there is a transmission in progress or when a preamble or break character is loaded. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD out signal becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL). TC is cleared automatically when data, preamble, or break is queued and ready to be sent. TC is cleared in the event of a simultaneous set and clear of the TC flag (transmission not complete). 0 Transmission in progress 1 No transmission in progress

Table 1-5. SCISR1 Field Descriptions (continued)

Field	Description
5 RDRF	<p>Receive Data Register Full Flag — RDRF is set when the data in the receive shift register transfers to the SCI data register. Clear RDRF by reading SCI status register 1 (SCISR1) with RDRF set and then reading SCI data register low (SCIDRL).</p> <p>0 Data not available in SCI data register 1 Received data available in SCI data register</p>
4 IDLE	<p>Idle Line Flag — IDLE is set when 10 consecutive logic 1s (if M=0) or 11 consecutive logic 1s (if M=1) appear on the receiver input. Once the IDLE flag is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag. Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL).</p> <p>0 Receiver input is either active now or has never become active since the IDLE flag was last cleared 1 Receiver input has become idle</p> <p>Note: When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag.</p>
3 OR	<p>Overrun Flag — OR is set when software fails to read the SCI data register before the receive shift register receives the next frame. The OR bit is set immediately after the stop bit has been completely received for the second frame. The data in the shift register is lost, but the data already in the SCI data registers is not affected. Clear OR by reading SCI status register 1 (SCISR1) with OR set and then reading SCI data register low (SCIDRL).</p> <p>0 No overrun 1 Overrun</p> <p>Note: OR flag may read back as set when RDRF flag is clear. This may happen if the following sequence of events occurs:</p> <ol style="list-style-type: none"> 1. After the first frame is received, read status register SCISR1 (returns RDRF set and OR flag clear); 2. Receive second frame without reading the first frame in the data register (the second frame is not received and OR flag is set); 3. Read data register SCIDRL (returns first frame and clears RDRF flag in the status register); 4. Read status register SCISR1 (returns RDRF clear and OR set). <p>Event 3 may be at exactly the same time as event 2 or any time after. When this happens, a dummy SCIDRL read following event 4 will be required to clear the OR flag if further frames are to be received.</p>
2 NF	<p>Noise Flag — NF is set when the SCI detects noise on the receiver input. NF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear NF by reading SCI status register 1 (SCISR1), and then reading SCI data register low (SCIDRL).</p> <p>0 No noise 1 Noise</p>
1 FE	<p>Framing Error Flag — FE is set when a logic 0 is accepted as the stop bit. FE bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading SCI status register 1 (SCISR1) with FE set and then reading the SCI data register low (SCIDRL).</p> <p>0 No framing error 1 Framing error</p>
0 PF	<p>Parity Error Flag — PF is set when the parity enable bit (PE) is set and the parity of the received data does not match the parity type bit (PT). PF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear PF by reading SCI status register 1 (SCISR1), and then reading SCI data register low (SCIDRL).</p> <p>0 No parity error 1 Parity error</p>

1.3.2.5 SCI Status Register 2 (SCISR2)

Module Base + 0x_0005

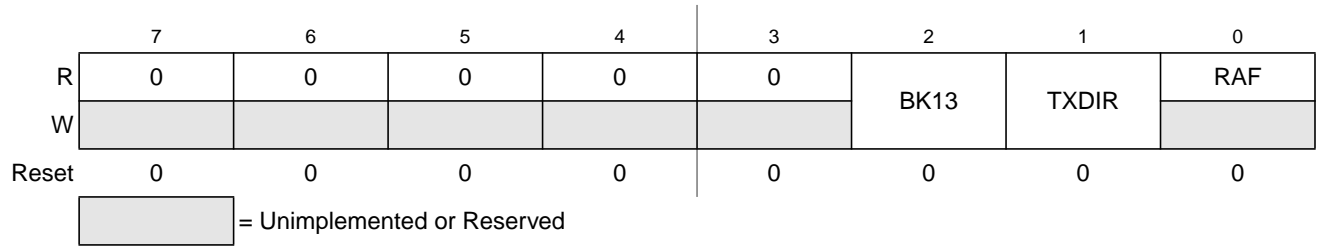


Figure 1-7. SCI Status Register 2 (SCISR2)

Read: Anytime

Write: Anytime; writing accesses SCI status register 2; writing to any bits except TXDIR and BRK13 (SCISR2[1] & [2]) has no effect

Table 1-6. SCISR2 Field Descriptions

Field	Description
2 BK13	Break Transmit Character Length — This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bits long. The detection of a framing error is not affected by this bit. 0 Break Character is 10 or 11 bit long 1 Break character is 13 or 14 bit long
1 TXDIR	Transmitter Pin Data Direction in Single-Wire Mode. — This bit determines whether the TXD pin is going to be used as an input or output, in the Single-Wire mode of operation. This bit is only relevant in the Single-Wire mode of operation. 0 TXD pin to be used as an input in Single-Wire mode 1 TXD pin to be used as an output in Single-Wire mode
0 RAF	Receiver Active Flag — RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 No reception in progress 1 Reception in progress

1.3.2.6 SCI Data Registers (SCIDRH and SCIDRL)

Module Base + 0x_0006

	7	6	5	4	3	2	1	0
R	R8	T8	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Module Base + 0x_0007

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

 = Unimplemented or Reserved

Figure 1-8. SCI Data Registers (SCIDRH and SCIDRL)

Read: Anytime; reading accesses SCI receive data register

Write: Anytime; writing accesses SCI transmit data register; writing to R8 has no effect

Table 1-7. SCIDRH AND SCIDRL Field Descriptions

Field	Description
7 R8	Received Bit 8 — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).
6 T8	Transmit Bit 8 — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).
7–0 R[7:0] T[7:0]	Received Bits — Received bits seven through zero for 9-bit or 8-bit data formats Transmit Bits — Transmit bits seven through zero for 9-bit or 8-bit formats

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten.

In 8-bit data format, only SCI data register low (SCIDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCIDRH), then SCIDRL.

1.4 Functional Description

This section provides a complete functional description of the SCI block, detailing the operation of the design from the end user perspective in a number of subsections.

Figure 1-9 shows the structure of the SCI module. The SCI allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

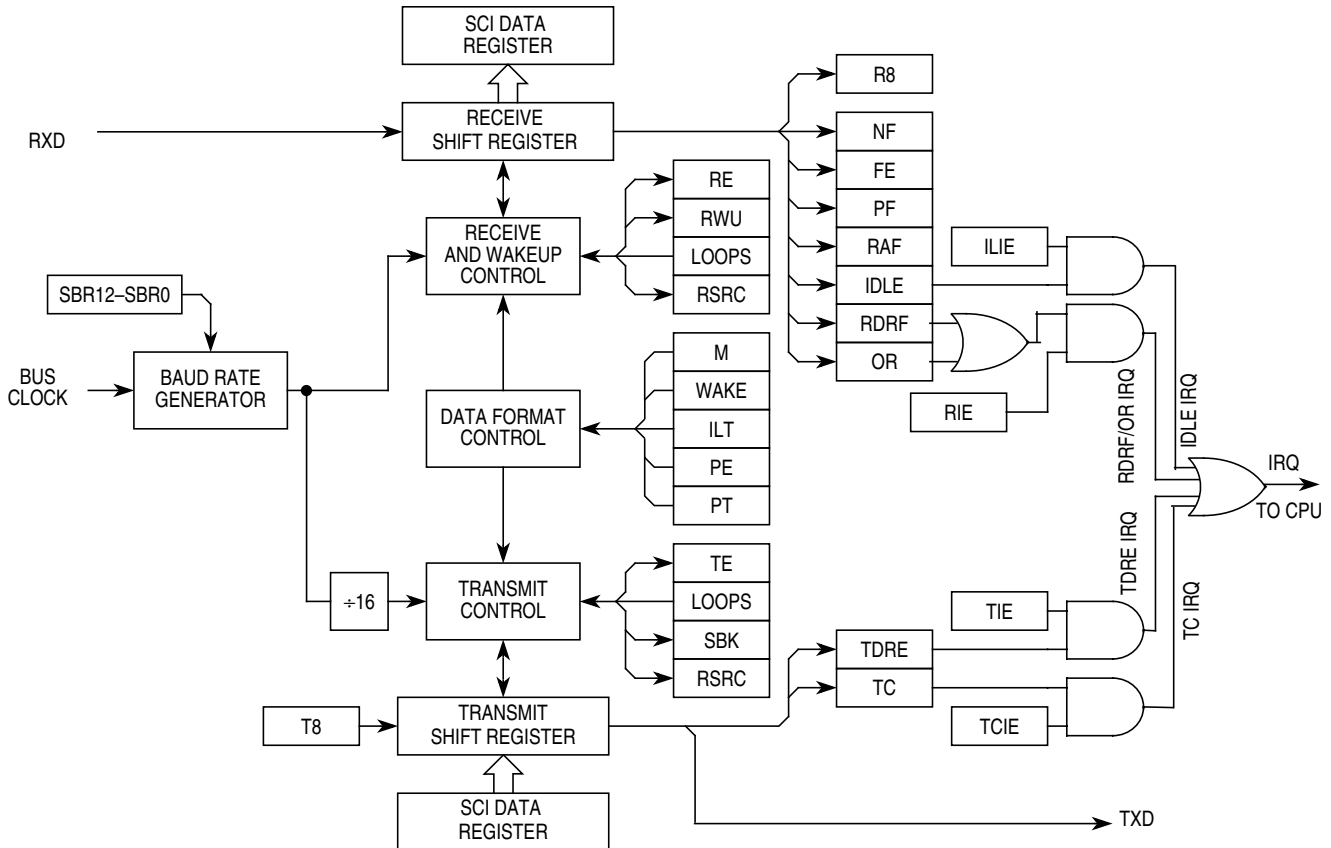


Figure 1-9. SCI Block Diagram

1.4.1 Data Format

The SCI uses the standard NRZ mark/space data format illustrated in [Figure 1-10](#) below.

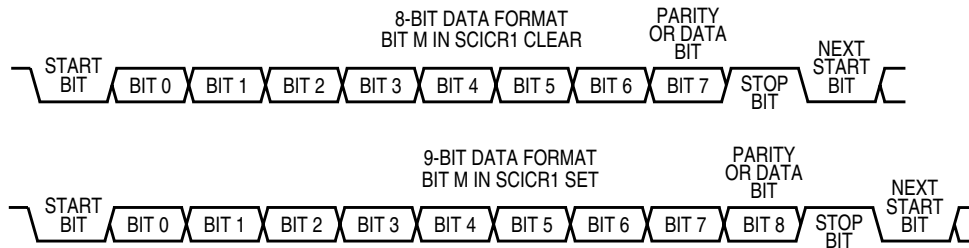


Figure 1-10. SCI Data Formats

Each data character is contained in a frame that includes a start bit, eight or nine data bits, and a stop bit. Clearing the M bit in SCI control register 1 configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Setting the M bit configures the SCI for nine-bit data characters. A frame with nine data bits has a total of 11 bits

Table 1-8. Example of 8-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	8	0	0	1
1	7	0	1	1
1	7	1 ¹	0	1

¹ The address bit identifies the frame as an address character. See [Section 1.4.4.6, "Receiver Wakeup"](#).

When the SCI is configured for 9-bit data characters, the ninth data bit is the T8 bit in SCI data register high (SCIDRH). It remains unchanged after transmission and can be used repeatedly without rewriting it. A frame with nine data bits has a total of 11 bits.

Table 1-9. Example of 9-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	9	0	0	1
1	8	0	1	1
1	8	1 ¹	0	1

¹ The address bit identifies the frame as an address character. See [Section 1.4.4.6, "Receiver Wakeup"](#).

1.4.2 Baud Rate Generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 8191 written to the SBR12–SBR0 bits determines the module clock divisor. The SBR bits are in the SCI baud rate registers (SCIBDH and SCIBDL). The baud rate clock is synchronized with the bus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to one source of error:

Integer division of the module clock may not give the exact target frequency.

Table 1-10 lists some examples of achieving target baud rates with a module clock frequency of 25 MHz

SCI baud rate = SCI module clock / (16 * SCIBR[12:0])

Table 1-10. Baud Rates (Example: Module Clock = 25 MHz)

Bits SBR[12-0]	Receiver Clock (Hz)	Transmitter Clock (Hz)	Target Baud Rate	Error (%)
41	609,756.1	38,109.8	38,400	.76
81	308,642.0	19,290.1	19,200	.47
163	153,374.2	9585.9	9600	.16
326	76,687.1	4792.9	4800	.15
651	38,402.5	2400.2	2400	.01
1302	19,201.2	1200.1	1200	.01
2604	9600.6	600.0	600	.00
5208	4800.0	300.0	300	.00

1.4.3 Transmitter

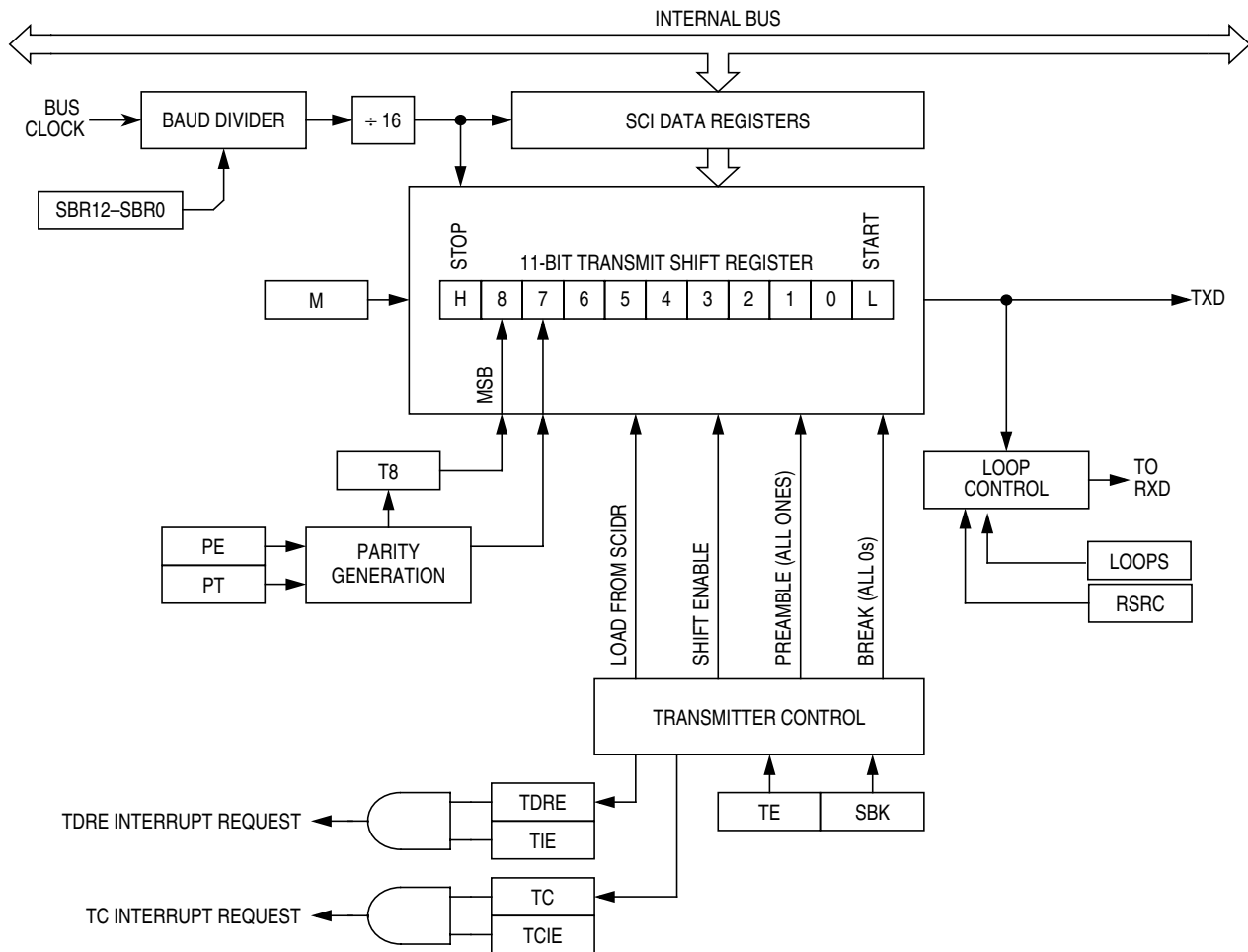


Figure 1-11. Transmitter Block Diagram

1.4.3.1 Transmitter Character Length

The SCI transmitter can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When transmitting 9-bit data, bit T8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

1.4.3.2 Character Transmission

To transmit data, the MCU writes the data bits to the SCI data registers (SCIDRH/SCIDRL), which in turn are transferred to the transmitter shift register. The transmit shift register then shifts a frame out through the **Tx output** signal, after it has prefaced them with a start bit and appended them with a stop bit. The SCI data registers (SCIDRH and SCIDRL) are the write-only buffers between the internal data bus and the transmit shift register.

The SCI also sets a flag, the transmit data register empty flag (TDRE), every time it transfers data from the buffer (SCIDRH/L) to the transmitter shift register. The transmit driver routine may respond to this flag

by writing another byte to the Transmitter buffer (SCIDRH/SCIDRL), while the shift register is still shifting out the first byte.

To initiate an SCI transmission:

1. Configure the SCI:
 - a) Select a baud rate. Write this value to the SCI baud registers (SCIBDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the SCIBDH has no effect without also writing to SCIBDL.
 - b) Write to SCICR1 to configure word length, parity, and other configuration bits (LOOPS,RSRC,M,WAKE,ILT,PE,PT).
 - c) Enable the transmitter, interrupts, receive, and wake up as required, by writing to the SCICR2 register bits (TIE,TCIE,RIE,ILIE,TE,RE,RWU,SBK). A preamble or idle character will now be shifted out of the transmitter shift register.
2. Transmit Procedure for Each Byte:
 - a. Poll the TDRE flag by reading the SCISR1 or responding to the TDRE interrupt. Keep in mind that the TDRE bit resets to one.
 - d) If the TDRE flag is set, write the data to be transmitted to SCIDRH/L, where the ninth bit is written to the T8 bit in SCIDRH if the SCI is in 9-bit data format. A new transmission will not result until the TDRE flag has been cleared.
3. Repeat step 2 for each subsequent transmission.

NOTE

The TDRE flag is set when the shift register is loaded with the next data to be transmitted from SCIDRH/L, which happens, generally speaking, a little over half-way through the stop bit of the previous frame. Specifically, this transfer occurs 9/16ths of a bit time AFTER the start of the stop bit of the previous frame.

Writing the TE bit from 0 to 1 automatically loads the transmit shift register with a preamble of 10 logic 1s (if M = 0) or 11 logic 1s (if M = 1). After the preamble shifts out, control logic transfers the data from the SCI data register into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit position of the transmit shift register. A logic 1 stop bit goes into the most significant bit position.

Hardware supports odd or even parity. When parity is enabled, the most significant bit (msb) of the data character is the parity bit.

The transmit data register empty flag, TDRE, in SCI status register 1 (SCISR1) becomes set when the SCI data register transfers a byte to the transmit shift register. The TDRE flag indicates that the SCI data register can accept new data from the internal data bus. If the transmit interrupt enable bit, TIE, in SCI control register 2 (SCICR2) is also set, the TDRE flag generates a transmitter interrupt request.

When the transmit shift register is not transmitting a frame, the **Tx output** signal goes to the idle condition, logic 1. If at any time software clears the TE bit in SCI control register 2 (SCICR2), the transmitter enable signal goes low and the transmit signal goes idle.

If software clears TE while a transmission is in progress ($TC = 0$), the frame in the transmit shift register continues to shift out. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last byte of the first message to SCIDRH/L.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the TE bit.
4. Write the first byte of the second message to SCIDRH/L.

1.4.3.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCI control register 2 (SCICR2) loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCI control register 1 (SCICR1). As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next frame.

The SCI recognizes a break character when a start bit is followed by eight or nine logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on SCI registers:

- Sets the framing error flag, FE
- Sets the receive data register full flag, RDRF
- Clears the SCI data registers (SCIDRH/L)
- May set the overrun flag, OR, noise flag, NF, parity error flag, PE, or the receiver active flag, RAF (see [Section 1.3.2.4, “SCI Status Register 1 \(SCISR1\)”](#) and [Section 1.3.2.5, “SCI Status Register 2 \(SCISR2\)”](#))

1.4.3.4 Idle Characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCI control register 1 (SCICR1). The preamble is a synchronizing idle character that begins the first transmission initiated after writing the TE bit from 0 to 1.

If the TE bit is cleared during a transmission, the **Tx output** signal becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the frame currently being transmitted.

NOTE

When queuing an idle character, return the TE bit to logic 1 before the stop bit of the current frame shifts out through the **Tx output** signal. Setting TE after the stop bit appears on **Tx output signal** causes data previously written to the SCI data register to be lost. Toggle the TE bit for a queued idle character while the TDRE flag is set and immediately before writing the next byte to the SCI data register.

NOTE

If the TE bit is clear and the transmission is complete, the SCI is not the master of the TXD pin

1.4.4 Receiver

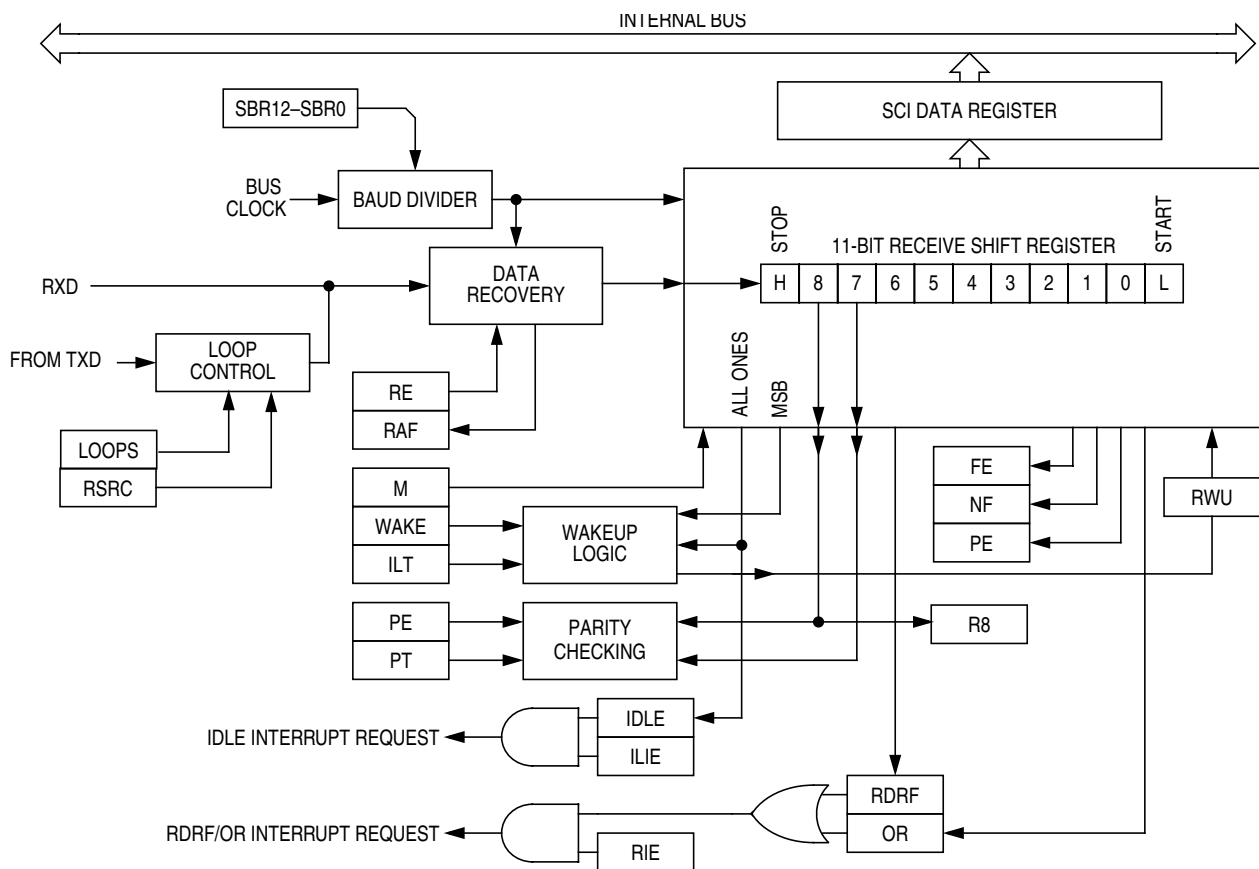


Figure 1-12. SCI Receiver Block Diagram

1.4.4.1 Receiver Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

1.4.4.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the **Rx input** signal. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCISR1) becomes set, indicating that the received byte can be read. If the receive interrupt enable bit, RIE, in SCI control register 2 (SCICR2) is also set, the RDRF flag generates an RDRF interrupt request.

1.4.4.3 Data Sampling

The receiver samples the **Rx input** signal at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see [Figure 1-13](#)) is re-synchronized:

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s. When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

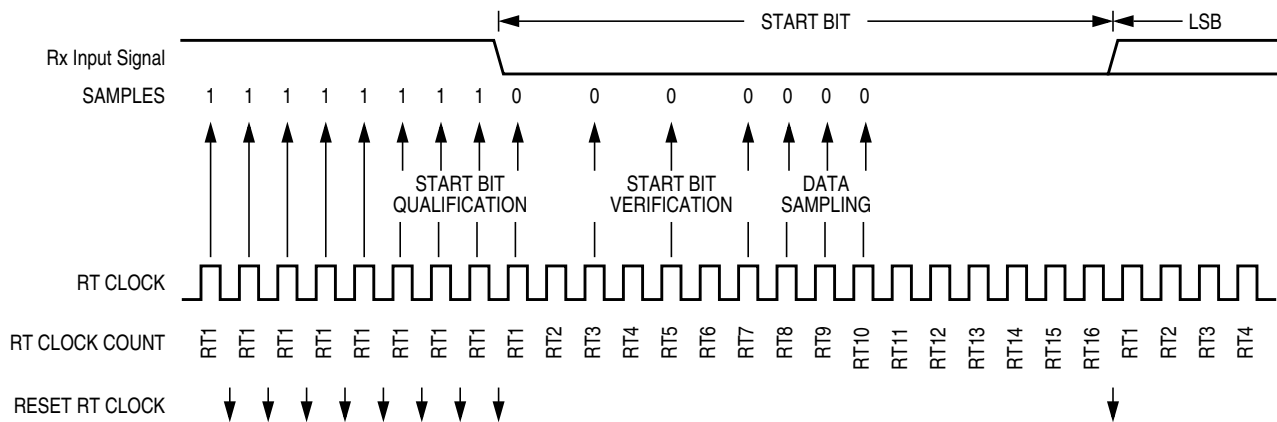


Figure 1-13. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. [Table 1-11](#) summarizes the results of the start bit verification samples.

Table 1-11. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0

Table 1-11. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 1-12](#) summarizes the results of the data bit samples.

Table 1-12. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 1-13](#) summarizes the results of the stop bit samples.

Table 1-13. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In [Figure 1-14](#) the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

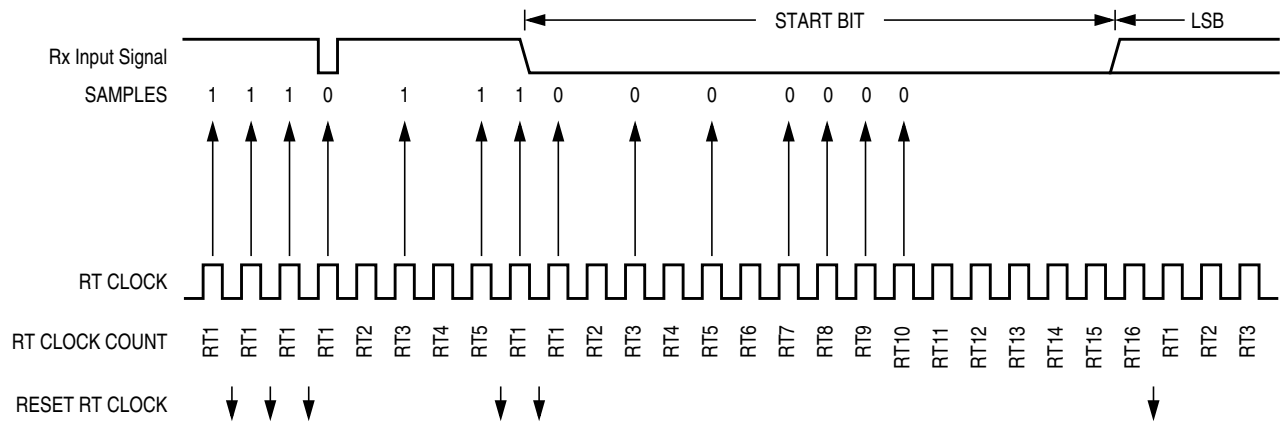


Figure 1-14. Start Bit Search Example 1

In [Figure 1-15](#), verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

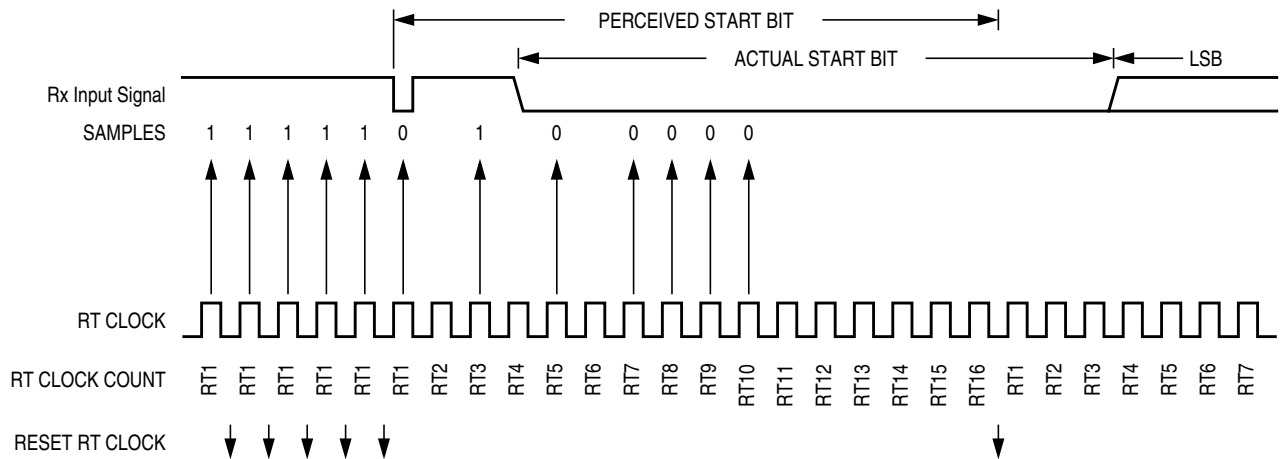


Figure 1-15. Start Bit Search Example 2

In Figure 1-16, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

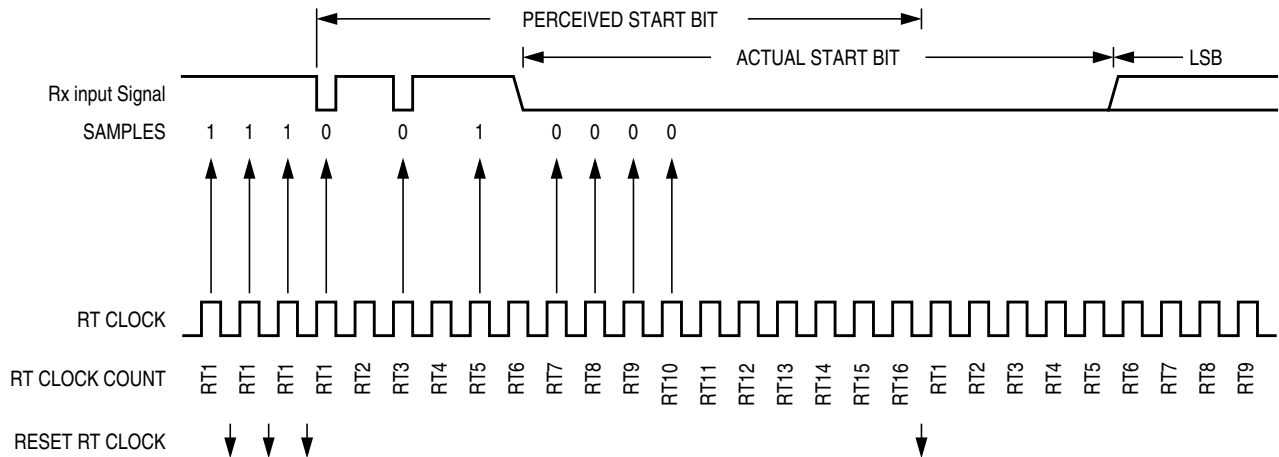


Figure 1-16. Start Bit Search Example 3

Figure 1-17 shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

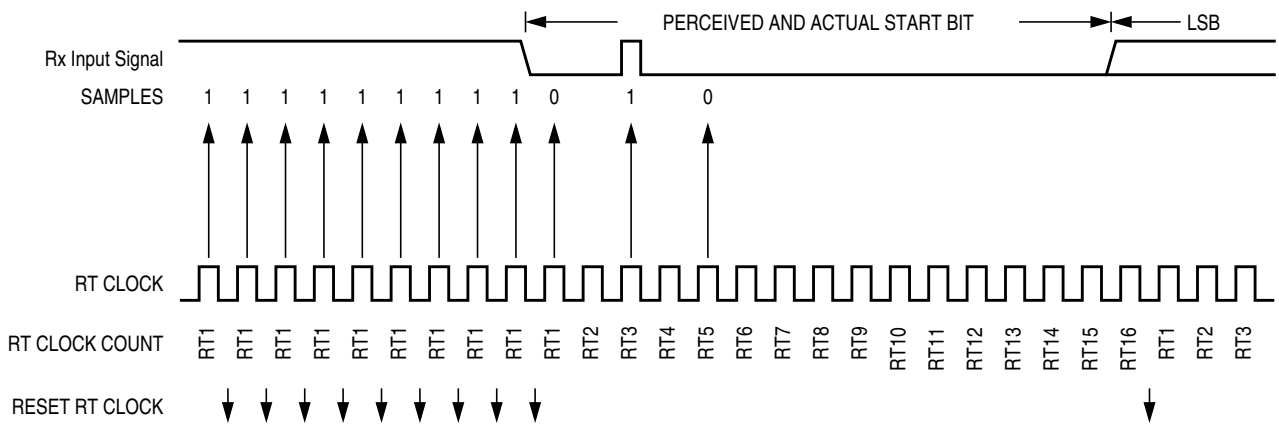


Figure 1-17. Start Bit Search Example 4

1.4.4.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, FE, in SCI status register 1 (SCISR1). A break character also sets the FE flag because a break character has no stop bit. The FE flag is set at the same time that the RDRF flag is set.

1.4.4.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic zero.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Re synchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

1.4.4.5.1 Slow Data Tolerance

Figure 1-20 shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

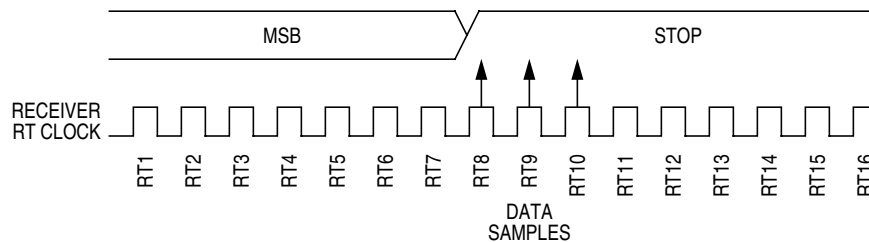


Figure 1-20. Slow Data

Let's take RTr as receiver RT clock and RTt as transmitter RT clock.

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 7 RTr cycles = 151 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 1-20, the receiver counts 151 RTr cycles at the point when the count of the transmitting device is 9 bit times x 16 RTt cycles = 144 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((151 - 144) / 151) \times 100 = 4.63\%$$

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 7 RTr cycles = 167 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in [Figure 1-20](#), the receiver counts 167 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((167 - 160) / 167) \times 100 = 4.19\%$$

1.4.4.5.2 Fast Data Tolerance

[Figure 1-21](#) shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

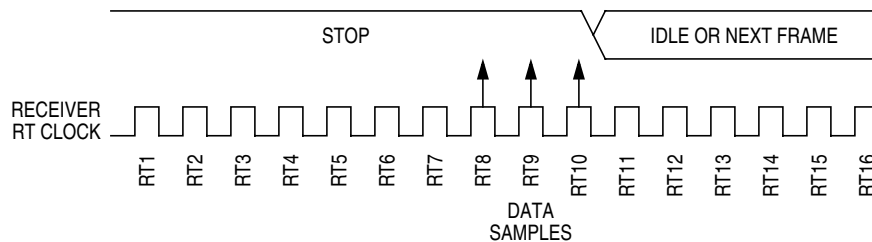


Figure 1-21. Fast Data

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 10 RTr cycles = 154 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in [Figure 1-21](#), the receiver counts 154 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$$((160 - 154) / 160) \times 100 = 3.75\%$$

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 10 RTr cycles = 170 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in [Figure 1-21](#), the receiver counts 170 RTr cycles at the point when the count of the transmitting device is 11 bit times x 16 RTt cycles = 176 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$$((176 - 170) / 176) \times 100 = 3.40\%$$

1.4.4.6 Receiver Wakeup

To enable the SCI to ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCI control register 2 (SCICR2) puts the receiver into standby state during which receiver interrupts are disabled. The SCI will still load the receive data into the SCIDRH/L registers, but it will not set the RDRF flag.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in SCI control register 1 (SCICR1) determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either idle line wakeup or address mark wakeup.

1.4.4.6.1 Idle Input Line Wakeup (WAKE = 0)

In this wakeup method, an idle condition on the **Rx Input** signal clears the RWU bit and wakes up the SCI. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another idle character appears on the **Rx Input** signal.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

The idle character that wakes a receiver does not set the receiver idle bit, IDLE, or the receive data register full flag, RDRF.

The idle line type bit, ILT, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit. ILT is in SCI control register 1 (SCICR1).

1.4.4.6.2 Address Mark Wakeup (WAKE = 1)

In this wakeup method, a logic 1 in the most significant bit (msb) position of a frame clears the RWU bit and wakes up the SCI. The logic 1 in the msb position marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the **Rx Input** signal.

The logic 1 msb of an address frame clears the receiver's RWU bit before the stop bit is received and sets the RDRF flag.

Address mark wakeup allows messages to contain idle characters but requires that the msb be reserved for use in address frames. {sci_wake }

NOTE

With the WAKE bit clear, setting the RWU bit after the **Rx Input** signal has been idle can cause the receiver to wake up immediately.

1.4.5 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI. The SCI uses the TXD pin for both receiving and transmitting.

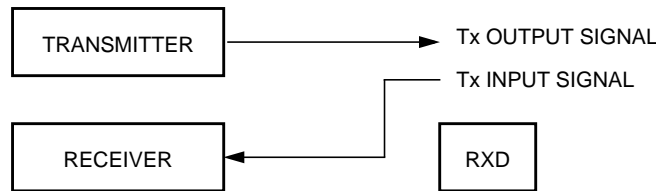


Figure 1-22. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the **Rx Input** signal to the receiver. Setting the RSRC bit connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1). The TXDIR bit (SCISR2[1]) determines whether the TXD pin is going to be used as an input (TXDIR = 0) or an output (TXDIR = 1) in this mode of operation.

1.4.6 Loop Operation

In loop operation the transmitter output goes to the receiver input. The **Rx Input** signal is disconnected from the SCI.

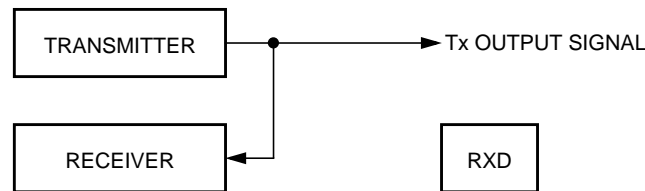


Figure 1-23. Loop Operation (LOOPS = 1, RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the **Rx Input** signal to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

1.5 Initialization Information

1.5.1 Reset Initialization

The reset state of each individual bit is listed in [Section 1.3, “Memory Map and Registers”](#) which details the registers and their bit fields. All special functions or modes which are initialized during or just following reset are described within this section.

1.5.2 Interrupt Operation

1.5.2.1 System Level Interrupt Sources

There are five interrupt sources that can generate an SCI interrupt in to the CPU. They are listed in [Table 1-14](#).

Table 1-14. SCI Interrupt Source

Interrupt Source	Flag	Local Enable
Transmitter	TDRE	TIE
Transmitter	TC	TCIE
Receiver	RDRF	RIE
	OR	
Receiver	IDLE	ILIE

1.5.2.2 Interrupt Descriptions

The SCI only originates interrupt requests. The following is a description of how the SCI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are chip dependent. The SCI only has a single interrupt line (**SCI Interrupt Signal**, active high operation) and all the following interrupts, when generated, are ORed together and issued through that port.

1.5.2.2.1 TDRE Description

The TDRE interrupt is set high by the SCI when the transmit shift register receives a byte from the SCI data register. A TDRE interrupt indicates that the transmit data register (SCIDRH/L) is empty and that a new byte can be written to the SCIDRH/L for transmission. Clear TDRE by reading SCI status register 1 with TDRE set and then writing to SCI data register low (SCIDRL).

1.5.2.2.2 TC Description

The TC interrupt is set by the SCI when a transmission has been completed. A TC interrupt indicates that there is no transmission in progress. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL). TC is cleared automatically when data, preamble, or break is queued and ready to be sent.

1.5.2.2.3 RDRF Description

The RDRF interrupt is set when the data in the receive shift register transfers to the SCI data register. A RDRF interrupt indicates that the received data has been transferred to the SCI data register and that the byte can now be read by the MCU. The RDRF interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

1.5.2.2.4 OR Description

The OR interrupt is set when software fails to read the SCI data register before the receive shift register receives the next frame. The newly acquired data in the shift register will be lost in this case, but the data already in the SCI data registers is not affected. The OR interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

1.5.2.3 IDLE Description

The IDLE interrupt is set when 10 consecutive logic 1s (if $M = 0$) or 11 consecutive logic 1s (if $M = 1$) appear on the receiver input. Once the IDLE is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag. Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL).

1.5.3 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

SPI

Block User Guide

V02.07

Original Release Date: 21 JAN 2000

Revised: 11 Dec 2002

Motorola, Inc.

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
0.1	21 Jan 2000			This spec is based on the Barracuda, with modifications to change the module from 16 bit to 8 bit.
0.2	1 Mar 2000			Template of this document changed as per Version 2.0 SRS.
0.3	14 Jun 2000			<ul style="list-style-type: none"> - Signal names are changed as per the SRS2.0 - SPE bit remains set in the Mode Fault error case - Slave SPI does not support div2 and div4 cases
0.4	31 Aug 2000			<ul style="list-style-type: none"> - Electrical spec added - SPIF flag is cleared by a read access to the status register followed by read access to the data register.
0.5	13 Mar 2001	13 Mar 2001		- Incorporated feedback regarding format of the document.
0.6	13 Mar 2001	19 Mar 2001		- Incorporated changes as a result of internal discussions and clarification of SRS2
0.7	6 July 2001	6 July 2001		<ul style="list-style-type: none"> - Line is added with respect to SPTEF bit to make spec more clear. - Landscape pages have been removed from pdf. - Extra blank pages have been removed.
0.8	19 July 2001	19 July 2001		- Line is added with respect to SPE bit to make spec more clear.
V02.02	26 July 2001			<ul style="list-style-type: none"> -Added Document Names -variable definitions and Names have been hidden -Changed chapter 3.9 Errata to Note
V02.03	28 Sept 2001			- Corrected the status of SPE when MODF is set
V02.04	11 Dec 2001			<ul style="list-style-type: none"> - Added a note for slave operation for MUCts00531 - Updated Block Diagram
V02.05	13 Feb 2002	13 Feb 2002		<ul style="list-style-type: none"> - Cleaned up revision history, summarized three entries with Version Number V2.04 into one - Section 4.6.2 Cleaned up Figures in Table 4-1 - Removed note 4.9, because in slave mode baud rates DIV2 and DIV4 are not supported
V02.06	06 Mar 2002	06 Mar 2002		- Document format update
V02.07	11 Dec 2002	11 Dec 2002		Section 3.3.5 - Added Note

Table of Contents

Section 1 Introduction

1.1	Overview	9
1.2	Features	9
1.3	Modes of Operation	9
1.4	Block Diagram	10

Section 2 Signal Description

2.1	Overview	11
2.2	Detailed Signal Descriptions	11
2.2.1	MOSI	11
2.2.2	MISO	11
2.2.3	\overline{SS}	11
2.2.4	SCK	11

Section 3 Memory Map and Registers

3.1	Overview	13
3.2	Module Memory Map	13
3.3	Register Descriptions	13
3.3.1	SPI Control Register 1	14
3.3.2	SPI Control Register 2	15
3.3.3	SPI Baud Rate Register	16
3.3.4	SPI Status Register	19
3.3.5	SPI Data Register	20

Section 4 Functional Description

4.1	General	21
4.2	Master Mode	21
4.3	Slave Mode	22
4.4	Transmission Formats	23
4.4.1	Clock Phase and Polarity Controls	23
4.4.2	CPHA = 0 Transfer Format	24
4.4.3	CPHA = 1 Transfer Format	25
4.5	SPI Baud Rate Generation	27

4.6	Special Features.	27
4.6.1	\overline{SS} Output.	27
4.6.2	Bidirectional Mode (MOMI or SISO).	28
4.7	Error Conditions	28
4.7.1	Mode Fault Error	29
4.8	Low Power Mode Options	29
4.8.1	SPI in Run Mode	29
4.8.2	SPI in Wait Mode	29
4.8.3	SPI in Stop Mode	30

Section 5 Reset

5.1	General.	31
-----	------------------	----

Section 6 Interrupts

6.1	Interrupt Operation	33
6.1.1	MODF.	33
6.1.2	SPIF.	33
6.1.3	SPTEF	33

List of Figures

Figure 1-1	SPI Block Diagram.	10
Figure 3-2	SPI Control Register 1 (SPICR1).	14
Figure 3-3	SPI Control Register 2 (SPICR2).	15
Figure 3-4	SPI Baud Rate Register (SPIBR)	16
Figure 3-5	SPI Status Register (SPISR)	19
Figure 3-6	SPI Data Register (SPIDR)	20
Figure 4-1	Master/Slave Transfer Block Diagram.	23
Figure 4-2	SPI Clock Format 0 (CPHA = 0)	25
Figure 4-3	SPI Clock Format 1 (CPHA = 1)	26
Figure 4-4	Baud Rate Divisor Equation.	27

List of Tables

Table 3-1	Module Memory Map	13
Table 3-2	\overline{SS} Input / Output Selection	15
Table 3-3	Bidirectional Pin Configurations	16
Table 3-4	SPI Baud Rate Selection	17
Table 4-1	Normal Mode and Bidirectional Mode	28

Section 1 Introduction

1.1 Overview

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or the SPI operation can be interrupt driven.

1.2 Features

The SPI includes these distinctive features:

- Master mode and slave mode
- Bi-directional mode
- Slave select output
- Mode fault error flag with CPU interrupt capability
- Double-buffered operation
- Serial clock with programmable polarity and phase
- Control of SPI operation during wait mode

1.3 Modes of Operation

The SPI functions in three modes, run, wait, and stop.

- Run Mode

This is the basic mode of operation.

- Wait Mode

SPI operation in wait mode is a configurable low power mode. Depending on the state of internal bits, the SPI can operate normally when the CPU is in wait mode or the SPI clock generation can be turned off and the SPI module enters a power conservation state during wait mode. During wait mode, any master transmission in progress stops if the SPISWAI bit is set in the SPI control register². Reception and transmission of a byte as slave continues so that the slave is synchronized to the master.

- Stop Mode

The SPI is inactive in stop mode for reduced power consumption. The STOP instruction does not affect or depend on SPI register states. Again, reception and transmission of a byte as slave continues to stay synchronized with the master.

This is a high level description only, detailed descriptions of operating modes are contained in section **4.8 Low Power Mode Options**.

1.4 Block Diagram

Figure 1-1 is a general block diagram of the SPI.

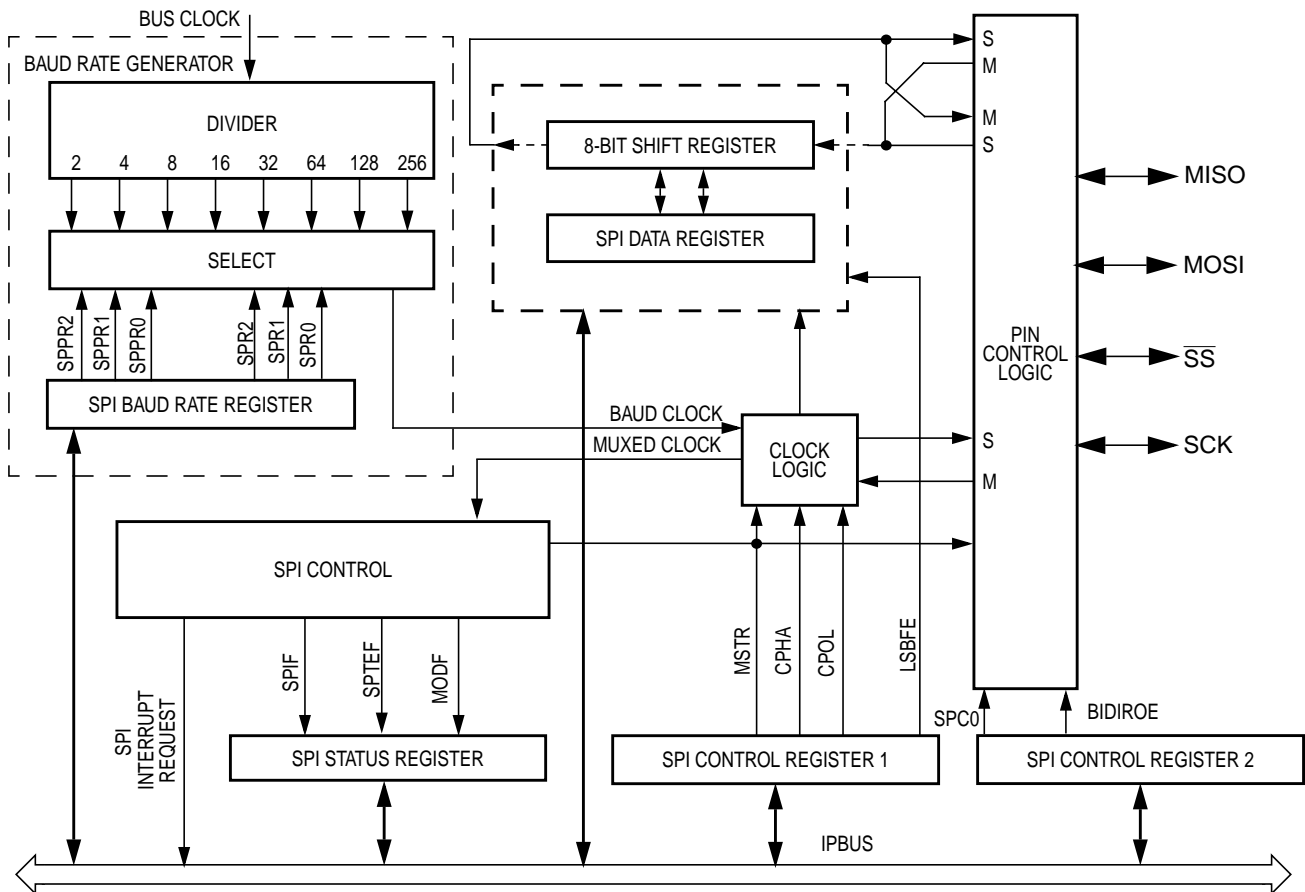


Figure 1-1 SPI Block Diagram

Section 2 Signal Description

2.1 Overview

This section lists the name and description of all ports including inputs and outputs that do, or may, connect off chip. The SPI module has a total of 4 external pins.

2.2 Detailed Signal Descriptions

2.2.1 MOSI

This pin is used to transmit data out of the SPI module when it is configured as a Master and receive data when it is configured as slave.

2.2.2 MISO

This pin is used to transmit data out of the SPI module when it is configured as a Slave and receive data when it is configured as Master.

2.2.3 \overline{SS}

This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place.

2.2.4 SCK

This pin is used to output the clock with respect to which the SPI transfers data or receive clock in case of Slave.

Section 3 Memory Map and Registers

3.1 Overview

This section provides a detailed description of all memory and registers accessible to the end user.

3.2 Module Memory Map

The memory map for the SPI is given below in **Table 3-1**. The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reads from the reserved bits return zeros and writes to the reserved bits have no effect.

Table 3-1 Module Memory Map

Address	Use	Access
\$__0	SPI Control Register 1 (SPICR1)	Read / Write
\$__1	SPI Control Register 2 (SPICR2)	Read / Write ¹
\$__2	SPI Baud Rate Register (SPIBR)	Read / Write ¹
\$__3	SPI Status Register (SPISR)	Read ²
\$__4	Reserved	— ^{2 3}
\$__5	SPI Data Register (SPIDR)	Read / Write
\$__6	Reserved	— ^{2 3}
\$__7	Reserved	— ^{2 3}

NOTES:

1. Certain bits are non-writable.
2. Writes to this register are ignored.
3. Reading from this register returns all zeros.

3.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

3.3.1 SPI Control Register 1

Register Address: \$__0

	Bit 7	6	5	4	3	2	1	Bit 0
R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
W								
Reset:	0	0	0	0	0	1	0	0

Figure 3-2 SPI Control Register 1 (SPICR1)

Read: anytime

Write: anytime

SPIE — SPI Interrupt Enable Bit

This bit enables SPI interrupts each time the SPIF or MODF status flag is set.

- 1 = SPI interrupts enabled.
- 0 = SPI interrupts disabled.

SPE — SPI System Enable Bit

This bit enables the SPI system and dedicates the SPI port pins to SPI system functions.

- 1 = SPI port pins are dedicated to SPI functions.
- 0 = SPI disabled (lower power consumption).

SPTIE — SPI Transmit Interrupt Enable

This bit enables SPI interrupt generated each time the SPTEF flag is set.

- 1 = SPTEF interrupt enabled.
- 0 = SPTEF interrupt disabled.

MSTR — SPI Master/Slave Mode Select Bit

- 1 = Master mode
- 0 = Slave mode

CPOL — SPI Clock Polarity Bit

This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values.

- 1 = Active-low clocks selected; SCK idles high
- 0 = Active-high clocks selected; SCK idles low

CPHA — SPI Clock Phase Bit

This bit is used to shift the SCK serial clock.

- 1 = The first SCK edge is issued at the beginning of the 8-cycle transfer operation
- 0 = The first SCK edge is issued one-half cycle into the 8-cycle transfer operation

SSOE — Slave Select Output Enable

The \overline{SS} output feature is enabled only in the master mode by asserting the SSOE as shown in **Table 3-2**.

Table 3-2 \overline{SS} Input / Output Selection

MOD FEN	SSOE	Master Mode	Slave Mode
0	0	\overline{SS} not used by SPI	\overline{SS} input
0	1	\overline{SS} not used by SPI	\overline{SS} input
1	0	\overline{SS} input with MODF feature	\overline{SS} input
1	1	\overline{SS} output	\overline{SS} input

LSBFE — SPI LSB-First Enable

This bit does not affect the position of the msb and lsb in the data register. Reads and writes of the data register always have the msb in bit 7.

- 1 = Data is transferred least significant bit first.
- 0 = Data is transferred most significant bit first.

3.3.2 SPI Control Register 2

Register Address: \$__1

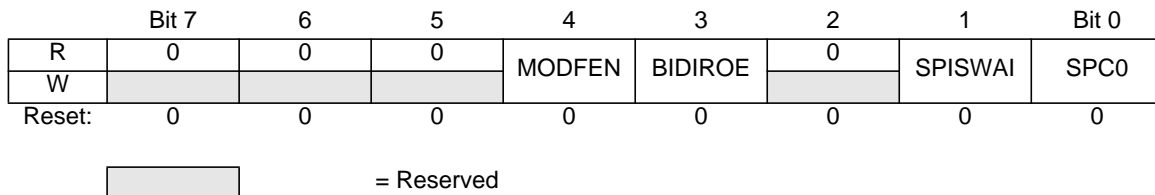


Figure 3-3 SPI Control Register 2 (SPICR2)

Read: anytime

Write: anytime; writes to the reserved bits have no effect

MODFEN — Mode Fault Enable Bit

This bit when set allows the MODF flag to be set. If the MODF flag is set, clearing the MODFEN does not clear the MODF flag. If the SPI is enabled as master and the MODFEN bit is low, then the \overline{SS} pin is not used by the SPI.

When the SPI is enabled as a slave, the \overline{SS} is available only as an input regardless of the value of MODFEN.

- 1 = Enable setting the MODF error
- 0 = Disable the MODF error

BIDIROE — Output enable in the Bidirectional mode of operation

This bit along with the MSTR bit of SPCR1 is used to enable the output buffer when the SPI is configured in bidirectional mode.

- 1 = Output buffer enabled
- 0 = Output buffer disabled

SPISWAI — SPI Stop in Wait Mode Bit

This bit is used for power conservation while in wait mode.

- 1 = Stop SPI clock generation when in wait mode
- 0 = SPI clock operates normally in wait mode

SPC0 — Serial Pin Control Bit 0

With the MSTR control bit, this bit enables bidirectional pin configurations as shown in **Table 3-3**.

Table 3-3 Bidirectional Pin Configurations

Pin Mode		SPC0	MSTR	MISO ¹	MOSI ²	SCK ³	\overline{SS} ⁴
A	Normal	0	0	Slave Out	Slave In	SCK in	\overline{SS} in
B			1	Master In	Master Out	SCK out	\overline{SS} I/O
C	Bidirectional	1	0	Slave I/O	—	SCK in	\overline{SS} In
D			1	—	Master I/O	SCK out	\overline{SS} I/O

NOTES:

1. Slave output is enabled if BIDIROE bit = 1, \overline{SS} = 0, and MSTR = 0 (C)
2. Master output is enabled if BIDIROE bit = 1 and MSTR = 1 (D)
3. SCK output is enabled if MSTR = 1 (B, D)
4. \overline{SS} output is enabled if MODFEN bit = 1, SSOE = 1, and MSTR = 1 (B, D).

3.3.3 SPI Baud Rate Register

Register Address: \$__2

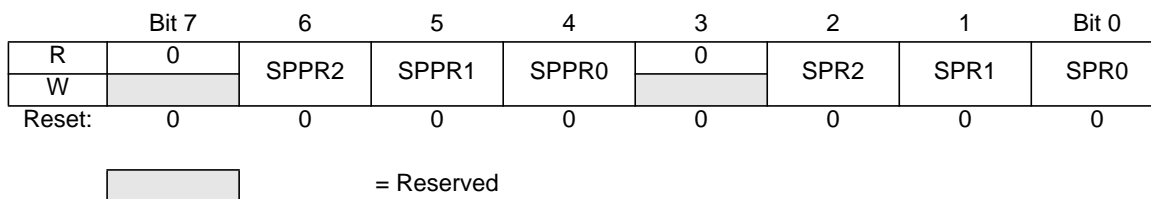


Figure 3-4 SPI Baud Rate Register (SPIBR)

Read: anytime

Write: anytime; writes to the reserved bits have no effect

NOTE: Writing to this register during data transfers may cause spurious results.

SPPR2–SPPR0 — SPI Baud Rate Preselection Bits

SPR2–SPR0 — SPI Baud Rate Selection Bits

These bits specify the SPI baud rates as shown in the table below

The baud rate divisor equation is as follows

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \cdot 2^{(\text{SPR} + 1)}$$

$$\text{Baud Rate} = \text{Bus clock} / \text{BaudRateDivisor}$$

Table 3-4 SPI Baud Rate Selection

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	SPI Module Clock Divisor
0	0	0	0	0	0	2
0	0	0	0	0	1	4
0	0	0	0	1	0	8
0	0	0	0	1	1	16
0	0	0	1	0	0	32
0	0	0	1	0	1	64
0	0	0	1	1	0	128
0	0	0	1	1	1	256
0	0	1	0	0	0	4
0	0	1	0	0	1	8
0	0	1	0	1	0	16
0	0	1	0	1	1	32
0	0	1	1	0	0	64
0	0	1	1	0	1	128
0	0	1	1	1	0	256
0	0	1	1	1	1	512
0	1	0	0	0	0	6
0	1	0	0	0	1	12
0	1	0	0	1	0	24
0	1	0	0	1	1	48
0	1	0	1	0	0	96
0	1	0	1	0	1	192
0	1	0	1	1	0	384
0	1	0	1	1	1	768
0	1	1	0	0	0	8
0	1	1	0	0	1	16
0	1	1	0	1	0	32
0	1	1	0	1	1	64
0	1	1	1	0	0	128
0	1	1	1	0	1	256

Table 3-4 SPI Baud Rate Selection

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	SPI Module Clock Divisor
0	1	1	1	1	0	512
0	1	1	1	1	1	1024
1	0	0	0	0	0	10
1	0	0	0	0	1	20
1	0	0	0	1	0	40
1	0	0	0	1	1	80
1	0	0	1	0	0	160
1	0	0	1	0	1	320
1	0	0	1	1	0	640
1	0	0	1	1	1	1280
1	0	1	0	0	0	12
1	0	1	0	0	1	24
1	0	1	0	1	0	48
1	0	1	0	1	1	96
1	0	1	1	0	0	192
1	0	1	1	0	1	384
1	0	1	1	1	0	768
1	0	1	1	1	1	1536
1	1	0	0	0	0	14
1	1	0	0	0	1	28
1	1	0	0	1	0	56
1	1	0	0	1	1	112
1	1	0	1	0	0	224
1	1	0	1	0	1	448
1	1	0	1	1	0	896
1	1	0	1	1	1	1792
1	1	1	0	0	0	16
1	1	1	0	0	1	32
1	1	1	0	1	0	64
1	1	1	0	1	1	128
1	1	1	1	0	0	256
1	1	1	1	0	1	512
1	1	1	1	1	0	1024
1	1	1	1	1	1	2048

NOTE: *DIV2 and DIV4 are not supported in slave mode of SPI.*

3.3.4 SPI Status Register

Register Address: \$__3

	Bit 7	6	5	4	3	2	1	Bit 0
R	SPIF	0	SPTEF	MODF	0	0	0	0
W								
Reset:	0	0	1	0	0	0	0	0

= Reserved

Figure 3-5 SPI Status Register (SPISR)

Read: anytime

Write: has no effect

SPIF — SPIF Interrupt Flag

This bit is set after the eighth SCK cycle in a data transfer and is cleared by reading the SPISR register (with SPIF set) followed by a read access to the SPI data register.

1 = New data Copied to SPIDR

0 = Transfer not yet complete

SPTEF — SPI Transmit Empty Interrupt Flag

This bit is set when there is room in the transmit data buffer. It is cleared by reading SPISR with SPTEF set, followed by writing a data value to the transmit buffer at SPIDR. SPISR must be read with SPTEF=1 before writing data to SPIDR or the SPIDR write will be ignored. SPTEF generates an SPTEF CPU interrupt request if the SPTIE bit in the SPICR1 is also set. SPTEF is automatically set when a data byte transfers from the transmit buffer into the transmit shift register. For an idle SPI (no data in the transmit buffer or the shift register and no transfer in progress), data written to SPIDR is transferred to the shifter almost immediately so SPTEF is set within two bus cycles allowing a second 8-bit data value to be queued into the transmit buffer. After completion of the transfer of the value in the shift register, the queued value from the transmit buffer will automatically move to the shifter and SPTEF will be set to indicate there is room for new data in the transmit buffer. If no new data is waiting in the transmit buffer, SPTEF simply remains set and no data moves from the buffer to the shifter.

1 = SPI Data register empty

0 = SPI Data register not empty

NOTE: Do not write to the SPI data register unless the SPTEF bit is high. Any such write to the SPI Data Register before reading SPTEF=1 is effectively ignored

MODF — Mode Fault Flag

This bit is set if the \overline{SS} input becomes low while the SPI is configured as a master. The flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to the SPI control register 1. The MODF flag is set only if the MODFEN bit of SPICR2 register is set. Refer to MODFEN bit description in **3.3.2 SPI Control Register 2**.

1 = Mode fault has occurred.

0 = Mode fault has not occurred.

3.3.5 SPI Data Register

Register Address: \$__5

	Bit 7	6	5	4	3	2	1	Bit 0
R	Bit 7	6	5	4	3	2	2	Bit 0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 3-6 SPI Data Register (SPIDR)

Read: anytime; normally read only after SPIF is set

Write: anytime; see SPTEF

The SPI Data register is both the input and output register for SPI data. A write to this register allows a data byte to be queued and transmitted. For a SPI configured as a master, a queued data byte is transmitted immediately after the previous transmission has completed. The SPI Transmitter empty flag in SPISR indicates when the SPI data register is ready to accept new data.

NOTE: Do not write to the SPI data register unless the SPTEF bit is high.

Reading the data can occur anytime from after the SPIF is set to before the end of the next transfer. If the SPIF is not serviced by the end of the successive transfers, those data bytes are lost and the data within the SPIDR retains the first byte until SPIF is serviced.

NOTE: After reset the content of the SPI Shift Register is undefined until a data byte is stored into SPIDR.

Section 4 Functional Description

4.1 General

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI control register 1. While SPE is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select (\overline{SS})
- Serial clock (SCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

The main element of the SPI system is the SPI data register. The 8-bit data register in the master and the 8-bit data register in the slave are linked by the MOSI and MISO pins to form a distributed 16-bit register. When a data transfer operation is performed, this 16-bit register is serially shifted eight bit positions by the SCK clock from the master; data is exchanged between the master and the slave. Data written to the master SPI data register becomes the output data for the slave, and data read from the master SPI data register after a transfer operation is the input data from the slave.

A write to the SPI data register puts data into the transmit buffer if the previous transmission was complete. When a transfer is complete, received data is moved into a receive data register. Data may be read from this double-buffered system any time before the next transfer is complete. This 8-bit data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A single SPI register address is used for reading data from the read data buffer and for writing data to the shifter.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI control register 1 select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by shifting the clock by a half cycle or by not shifting the clock (see **4.4 Transmission Formats**).

The SPI can be configured to operate as a master or as a slave. When MSTR in SPI control register 1 is set, the master mode is selected; when the MSTR bit is clear, the slave mode is selected.

4.2 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by writing to the master SPI data register. If the shift register is empty, the byte immediately transfers to the shift register. The byte begins shifting out on the MOSI pin under the control of the serial clock.

The SPR2, SPR1, and SPR0 baud rate selection bits in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI baud rate register control the baud rate generator and determine the

speed of the shift register. The SCK pin is the SPI clock output. Through the SCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.

In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and MSTR control bits.

The \overline{SS} pin is normally an input which should remain in the inactive high state. However, in the master mode, if both MODFEN bit and SSOE bit are set, then the \overline{SS} pin is the slave select output.

The \overline{SS} output becomes low during each transmission and is high when the SPI is in the idling state. If the \overline{SS} input becomes low while the SPI is configured as a master, it indicates a mode fault error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. In this case, the SPI immediately clears the output buffer enables associated with the MISO, MOSI (or MOMI), and SCK pins so that these pins become inputs. This mode fault error also clears the MSTR control bit and sets the mode fault (MODF) flag in the SPI status register. If the SPI interrupt enable bit (SPIE) is set when the MODF bit gets set, then an SPI interrupt sequence is also requested

When a write to the SPI data register in the master occurs, there is a half SCK-cycle delay. After the delay, SCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI control register 1 (see **4.4 Transmission Formats**).

4.3 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI control register 1 is clear. In slave mode, SCK is the SPI clock input from the master, and \overline{SS} is the slave select input. Before a data transmission occurs, the \overline{SS} pin of the slave SPI must be at logic 0. \overline{SS} must remain low until the transmission is complete.

In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit in SPI control register 2 and the MSTR control bit. While in slave mode, the \overline{SS} input controls the serial data output pin; if \overline{SS} is high (not selected), the serial data output pin is high impedance, and, if \overline{SS} is low the first bit in the SPI data register is driven out of the serial data output pin. Also, if the slave is not selected (\overline{SS} is high), then the SCK input is ignored and no internal shifting of the SPI shift register takes place.

Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

NOTE: *When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.*

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI control register 1 is clear, odd numbered edges on the SCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB of the SPI shifter.

If the CPHA bit is set, even numbered edges on the SCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB of the SPI shifter.

NOTE: *In slave mode, the control bits CPHA and CPOL of the SPI should be configured only when SPI is disabled else it may lead to incorrect data transfer. Care must be taken to avoid driver collisions in SPI disabled state, because SPI port pins are then not under the control of the SPI. For recommended actions refer to the Device User Guide, SPI section.*

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the \overline{SS} input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth shift, the transfer is considered complete and the received data is transferred into the SPI data register. To indicate transfer is complete, the SPIF flag in the SPI status register is set.

4.4 Transmission Formats

During an SPI transmission, data is transmitted (shifted out serially) and received (shifted in serially) simultaneously. The serial clock (SCK) synchronizes shifting and sampling of the information on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. Optionally, on a master SPI device, the slave select line can be used to indicate multiple-master bus contention.

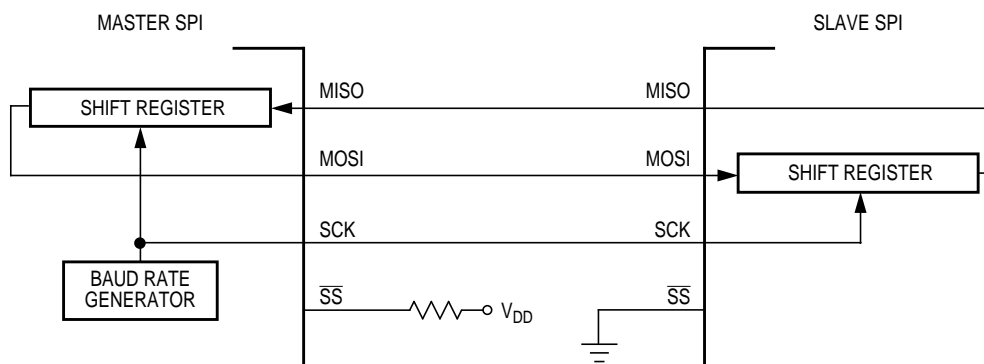


Figure 4-1 Master/Slave Transfer Block Diagram

4.4.1 Clock Phase and Polarity Controls

Using two bits in the SPI control register1, software selects one of four combinations of serial clock phase and polarity.

The CPOL clock polarity control bit specifies an active high or low clock and has no significant effect on the transmission format.

The CPHA clock phase control bit selects one of two fundamentally different transmission formats.

Clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

NOTE: *It is recommended that software writes to the SPI control register to change CPHA, CPOL or MSTR bits only in the idle state of the SPI. If these bits are changed during the transmission, data transmission gets corrupted.*

4.4.2 CPHA = 0 Transfer Format

The first edge on the SCK line is used to clock the first data bit of slave into the master and the first data bit of master into the slave. In some peripherals, the first bit of the slave's data is available at the slave data out pin as soon as the slave is selected. In this format, the first SCK edge is not issued until a half cycle into the 8-cycle transfer operation. The first edge of SCK is delayed a half cycle by clearing the CPHA bit.

The SCK output from the master remains in the inactive state for a half SCK period before the first edge appears. A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the LSB of the shifter.

After this second edge, the next bit of the SPI master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for a total of 16 edges on the SCK line, with data being latched on odd numbered edges and shifted on even numbered edges.

Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After the 16th (last) SCK edge:

- Data that was previously in the master SPI data register should now be in the slave data register and the data that was in the slave data register should be in the master.
- The SPIF flag in the SPI status register is set indicating that the transfer is complete.

Figure 4-2 is a timing diagram of an SPI transfer where CPHA = 0. SCK waveforms are shown for CPOL = 0 and CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave and the MOSI signal is the output from the master. The \overline{SS} pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

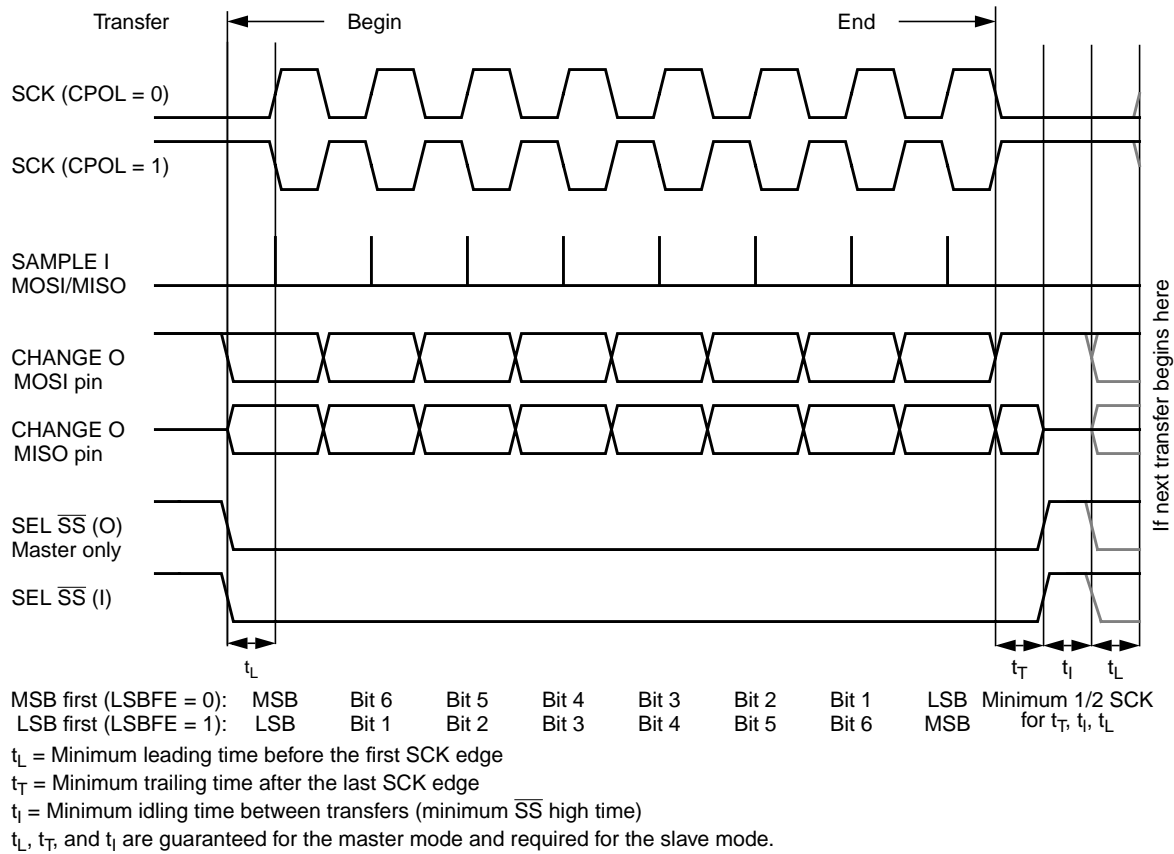


Figure 4-2 SPI Clock Format 0 (CPHA = 0)

The \overline{SS} line should be deasserted at least for minimum idle time (half SCK cycle) between the successive transfers (\overline{SS} should not be tied low all the times in this mode). If \overline{SS} is not deasserted between the successive transmission then the new byte written to the data register would not be transmitted, instead the last received byte would be transmitted.

4.4.3 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin; the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the 8-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its most significant data bit to the serial data input pin of the master.

A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB of the SPI shifter. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pins on the slave.

This process continues for a total of 16 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

Data reception is double buffered; data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After the 16th SCK edge:

- Data that was previously in the SPI data register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 4-3 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram since the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The \overline{SS} line is the slave select input to the slave. The \overline{SS} pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

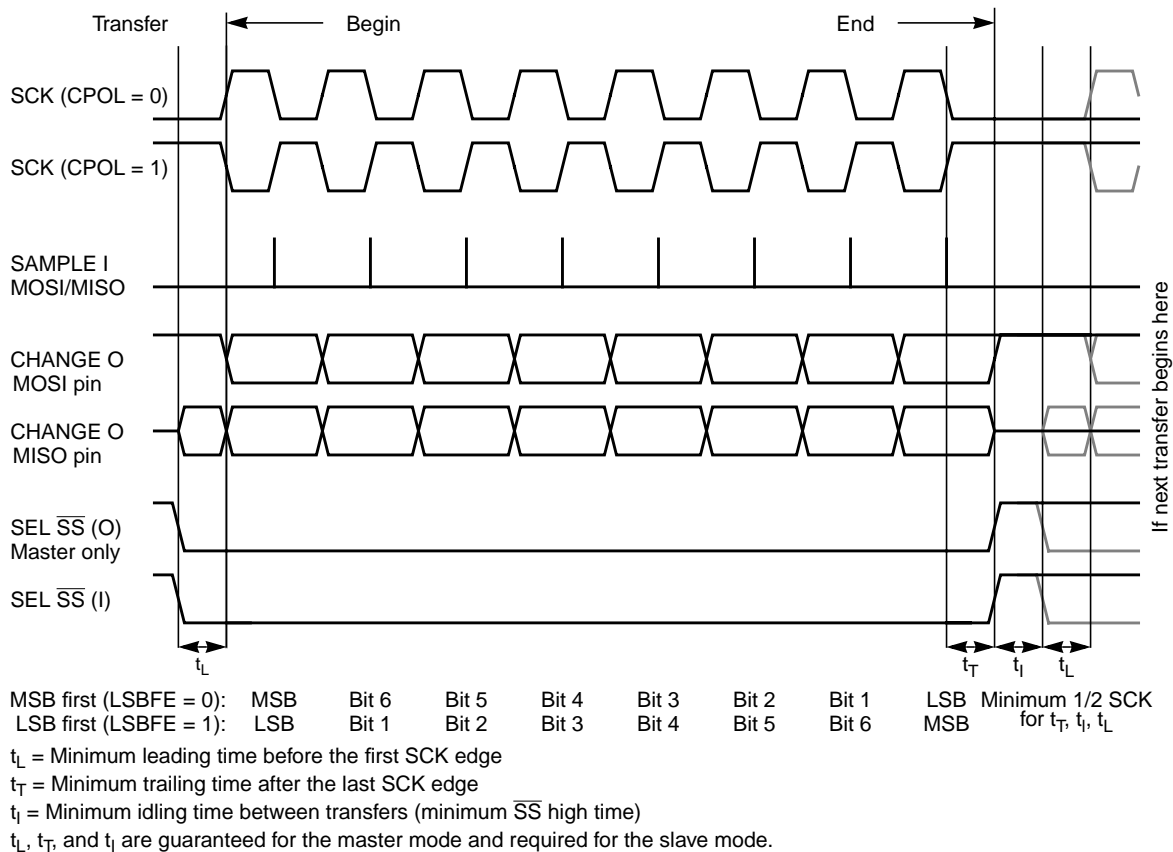


Figure 4-3 SPI Clock Format 1 (CPHA = 1)

The \overline{SS} line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set after the last SCK edge in a data transfer operation to indicate that the transfer is complete though transfer is actually complete half SCK cycle later.

4.5 SPI Baud Rate Generation

Baud rate generation consists of a series of divider stages. Six bits in the SPI baud rate register (SPPR2, SPPR1, SPPR0, SPR2, SPR1, and SPR0) determine the divisor to the SPI module clock which results in the SPI baud rate.

The SPI clock rate is determined by the product of the value in the baud rate preselection bits (SPPR2–SPPR0) and the value in the baud rate selection bits (SPR2–SPR0). The module clock divisor equation is shown in **Figure 4-4**.

When all bits are clear (the default condition), the SPI module clock is divided by 2. When the selection bits (SPR2–SPR0) are 001 and the preselection bits (SPPR2–SPPR0) are 000, the module clock divisor becomes 4. When the selection bits are 010, the module clock divisor becomes 8 etc.

When the preselection bits are 001, the divisor determined by the selection bits is multiplied by 2. When the preselection bits are 010, the divisor is multiplied by 3, etc. The two sets of selects allows the clock to be divided by a non-power of two to achieve other baud rates such as divide by 6, divide by 10, etc.

The baud rate generator is activated only when the SPI is in the master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease I_{DD} current.

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \cdot 2^{(\text{SPR} + 1)}$$

Figure 4-4 Baud Rate Divisor Equation

4.6 Special Features

4.6.1 \overline{SS} Output

The \overline{SS} output feature automatically drives the \overline{SS} pin low during transmission to select external devices and drives it high during idle to deselect external devices. When \overline{SS} output is selected, the \overline{SS} output pin is connected to the \overline{SS} input pin of the external device.

The \overline{SS} output is available only in master mode during normal SPI operation by asserting SSOE and MODFEN bit as shown in **Table 3-2**.

The mode fault feature is disabled while \overline{SS} output is enabled.

NOTE: Care must be taken when using the \overline{SS} output feature in a multimaster system since the mode fault feature is not available for detecting system errors between masters.

4.6.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see **Table 4-1 Normal Mode and Bidirectional Mode**). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in the master mode and MOSI pin in the slave mode are not used by the SPI in Bidirectional Mode.

Table 4-1 Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

The SCK is output for the master mode and input for the slave mode.

The \overline{SS} is the input or output for the master mode, and it is always the input for the slave mode.

The bidirectional mode does not affect SCK and \overline{SS} functions.

4.7 Error Conditions

The SPI has one error condition:

- Mode fault error

4.7.1 Mode Fault Error

If the \overline{SS} input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. This condition is not permitted in normal operation; the MODF bit in the SPI status register is set automatically provided the MODFEN bit is set.

In the special case where the MODFEN bit is cleared, the \overline{SS} pin is a general purpose input/output pin for the SPI system configured in master mode. In this special case, the mode error function is inhibited and MODF remains cleared. In case the SPI system is configured as a slave, the \overline{SS} pin is a dedicated input pin. Mode fault error doesn't occur in slave mode.

When a mode fault error occurs, the MSTR bit in control register SPICR1 is cleared, MODF bit in the status register is set and the output enable for the SCK, MISO and MOSI pins are de-asserted. So SCK, MISO and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver.

If the mode fault error occurs in the bidirectional mode for a SPI system configured in master mode, output enable of the MOMI (MOSI in bidirectional mode) is cleared if it was set but MISO (SISO) is not affected. No mode fault error occurs in the bidirectional mode for SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to SPI control register 1.

4.8 Low Power Mode Options

4.8.1 SPI in Run Mode

In run mode with the SPI system enable (SPE) bit in the SPI control register clear, the SPI system is in a low-power, disabled state. SPI registers can still be accessed, but clocks to the core of this module are disabled.

4.8.2 SPI in Wait Mode

SPI operation in wait mode depends upon the state of the SPISWAI bit in SPI control register 2.

- If SPISWAI is clear, the SPI operates normally when the CPU is in wait mode
- If SPISWAI is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
 - If SPISWAI is set and the SPI is configured for master, any transmission and reception in progress stops at wait mode entry. The transmission and reception resumes when the SPI exits wait mode.
 - If SPISWAI is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SCK continues to be driven from the master. This keeps the slave synchronized to the master and the SCK.

If the master transmits several bytes while the slave is in wait mode, the slave will continue to send out bytes consistent with the its operation mode at the start of wait mode (i.e. If the slave is currently sending its SPIDR to the master, it will continue to send the same byte. Else if the slave is currently sending the last received byte from the master, it will continue to send each previous master byte).

NOTE: *Care must be taken when expecting data from a master while the slave is in wait or stop mode. Even though the shift register will continue to operate, the rest of the SPI is shut down (i.e. a SPIF interrupt will **not** be generated until exiting stop or wait mode). Also, the byte from the shift register will not be copied into the SPIDR register until after the slave SPI has exited wait or stop mode. A SPIF flag and SPIDR copy is only generated if wait mode is entered or exited during a transmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither a SPIF nor a SPIDR copy will occur.*

4.8.3 SPI in Stop Mode

Stop mode is dependent on the system. The SPI enters stop mode when the module clock is disabled (held high or low). If the SPI is in master mode and exchanging data when the CPU enters stop mode, the transmission is frozen until the CPU exits stop mode. After stop, data to and from the external SPI is exchanged correctly. In slave mode, the SPI will stay synchronized with the master.

The stop mode is equivalent to the wait mode with the SPISWAI bit set except that the stop mode is dependent on the system and cannot be controlled with the SPISWAI bit.

Section 5 Reset

5.1 General

The reset values of registers and signals are described in the Memory Map and Registers section of the End User Guide (see **Section 3 Memory Map and Registers**) which details the registers and their bit-fields.

- If a data transmission occurs in slave mode after reset without a write to SPIDR, it will transmit garbage, or the byte last received from the master before the reset.
- Reading from the SPIDR after reset will always read a byte of zeros.

Section 6 Interrupts

The SPI only originates interrupt requests. The following is a description of how the SPI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

6.1 Interrupt Operation

6.1.1 MODF

MODF occurs when the master detects an error on the \overline{SS} pin. The master SPI must be configured for the MODF feature (see **Table 3-2 SS Input / Output Selection**). Once MODF is set, the current transfer is halted and the following bit is changed:

- MSTR=0, The master bit in SPICR1 resets.

The MODF interrupt is reflected in the status register MODF flag. Clearing the flag will also clear the interrupt. This interrupt will stay active while the MODF flag is set. MODF has an automatic clearing process which is described in **3.3.4 SPI Status Register**.

6.1.2 SPIF

SPIF occurs when the SPI receives/transmits the last SCK edge in a data transfer operation. Once SPIF is set, it does not clear until it is serviced. SPIF has an automatic clearing process which is described in **3.3.4 SPI Status Register**. In the event that the SPIF is not serviced before the end of the next transfer (i.e. SPIF remains active throughout another transfer), the latter transfers will be ignored and no new data will be copied into the SPIDR.

6.1.3 SPTEF

SPTEF occurs when the SPI Data register transfers a byte into the transmit buffer. Once SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process which is described in **3.3.4 SPI Status Register**.

User Guide End Sheet

**FINAL PAGE OF
36
PAGES**

Appendix A Electrical Characteristics

A.1 General

This introduction is intended to give an overview on several common topics like power supply, current injection etc.

A.1.1 Parameter Classification

The electrical parameters shown in this supplement are guaranteed by various methods. To give the customer a better understanding the following classification is used and the parameters are tagged accordingly in the tables where appropriate.

P:

Those parameters are guaranteed during production testing on each individual device.

C:

Those parameters are achieved by the design characterization by measuring a statistically relevant sample size across process variations. They are regularly verified by production monitors.

T:

Those parameters are achieved by design characterization on a small sample size from typical devices. All values shown in the typical column are within this category.

D:

Those parameters are derived mainly from simulations.

A.1.2 Power Supply

The MC9S12DT128 utilizes several pins to supply power to the I/O ports, A/D converter, oscillator, PLL and internal logic.

The VDDA, VSSA pair supplies the A/D converter and the resistor ladder of the internal voltage regulator.

The VDDX, VSSX, VDDR and VSSR pairs supply the I/O pins, VDDR supplies also the internal voltage regulator.

VDD1, VSS1, VDD2 and VSS2 are the supply pins for the digital logic, VDDPLL, VSSPLL supply the oscillator and the PLL.

VSS1 and VSS2 are internally connected by metal.

VDDA, VDDX, VDDR as well as VSSA, VSSX, VSSR are connected by anti-parallel diodes for ESD protection.

NOTE: *In the following context VDD5 is used for either VDDA, VDDR and VDDX; VSS5 is used for either VSSA, VSSR and VSSX unless otherwise noted. IDD5 denotes the sum of the currents flowing into the VDDA, VDDX and VDDR pins. VDD is used for VDD1, VDD2 and VDDPLL, VSS is used for VSS1, VSS2 and VSSPLL. IDD is used for the sum of the currents flowing into VDD1 and VDD2.*

A.1.3 Pins

There are four groups of functional pins.

A.1.3.1 5V I/O pins

Those I/O pins have a nominal level of 5V. This class of pins is comprised of all port I/O pins, the analog inputs, BKGD pin and the RESET inputs. The internal structure of all those pins is identical, however some of the functionality may be disabled. E.g. for the analog inputs the output drivers, pull-up and pull-down resistors are disabled permanently.

A.1.3.2 Analog Reference

This class is made up by the two VRH and VRL pins.

A.1.3.3 Oscillator

The pins XFC, EXTAL, XTAL dedicated to the oscillator have a nominal 2.5V level. They are supplied by VDDPLL.

A.1.3.4 TEST

This pin is used for production testing only.

A.1.3.5 VREGEN

This pin is used to enable the on chip voltage regulator.

A.1.4 Current Injection

Power supply must maintain regulation within operating V_{DD5} or V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{in} > V_{DD5}$) is greater than I_{DD5} , the injection current may flow out of VDD5 and could result in external power supply going out of regulation. Insure external VDD5 load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power; e.g. if no system clock is present, or if clock rate is very low which would reduce overall power consumption.

A.1.5 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only. A functional operation under or outside those maxima is not guaranteed. Stress beyond those limits may affect the reliability or cause permanent damage of the device.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS5} or V_{DD5}).

Table A-1 Absolute Maximum Ratings¹

Num	Rating	Symbol	Min	Max	Unit
1	I/O, Regulator and Analog Supply Voltage	V_{DD5}	-0.3	6.0	V
2	Digital Logic Supply Voltage ²	V_{DD}	-0.3	3.0	V
3	PLL Supply Voltage ⁽²⁾	V_{DDPLL}	-0.3	3.0	V
4	Voltage difference VDDX to VDDR and VDDA	ΔV_{DDX}	-0.3	0.3	V
5	Voltage difference VSSX to VSSR and VSSA	ΔV_{SSX}	-0.3	0.3	V
6	Digital I/O Input Voltage	V_{IN}	-0.3	6.0	V
7	Analog Reference	V_{RH}, V_{RL}	-0.3	6.0	V
8	XFC, EXTAL, XTAL inputs	V_{ILV}	-0.3	3.0	V
9	TEST input	V_{TEST}	-0.3	10.0	V
10	Instantaneous Maximum Current Single pin limit for all digital I/O pins ³	I_D	-25	+25	mA
11	Instantaneous Maximum Current Single pin limit for XFC, EXTAL, XTAL ⁴	I_{DL}	-25	+25	mA
12	Instantaneous Maximum Current Single pin limit for TEST ⁵	I_{DT}	-0.25	0	mA
13	Storage Temperature Range	T_{stg}	-65	155	°C

NOTES:

- Beyond absolute maximum ratings device might be damaged.
- The device contains an internal voltage regulator to generate the logic and PLL supply out of the I/O supply. The absolute maximum ratings apply when the device is powered from an external source.
- All digital I/O pins are internally clamped to V_{SSX} and V_{DDX} , V_{SSR} and V_{DDR} or V_{SSA} and V_{DDA} .
- Those pins are internally clamped to V_{SSPLL} and V_{DDPLL} .
- This pin is clamped low to V_{SSX} , but not clamped high. This pin must be tied low in applications.

A.1.6 ESD Protection and Latch-up Immunity

All ESD testing is in conformity with CDF-AEC-Q100 Stress test qualification for Automotive Grade Integrated Circuits. During the device qualification ESD stresses were performed for the Human Body Model (HBM), the Machine Model (MM) and the Charge Device Model.

A device will be defined as a failure if after exposure to ESD pulses the device no longer meets the device specification. Complete DC parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-2 ESD and Latch-up Test Conditions

Model	Description	Symbol	Value	Unit
Human Body	Series Resistance	R1	1500	Ohm
	Storage Capacitance	C	100	pF
	Number of Pulse per pin positive negative	–	3 3	
Machine	Series Resistance	R1	0	Ohm
	Storage Capacitance	C	200	pF
	Number of Pulse per pin positive negative	–	3 3	
Latch-up	Minimum input voltage limit		–2.5	V
	Maximum input voltage limit		7.5	V

Table A-3 ESD and Latch-Up Protection Characteristics

Num	C	Rating	Symbol	Min	Max	Unit
1	C	Human Body Model (HBM)	V_{HBM}	2000	–	V
2	C	Machine Model (MM)	V_{MM}	200	–	V
3	C	Charge Device Model (CDM)	V_{CDM}	500	–	V
4	C	Latch-up Current at 125°C positive negative	I_{LAT}	+100 –100	–	mA
5	C	Latch-up Current at 27°C positive negative	I_{LAT}	+200 –200	–	mA

A.1.7 Operating Conditions

This chapter describes the operating conditions of the device. Unless otherwise noted those conditions apply to all the following data.

NOTE: Please refer to the temperature rating of the device (C, V, M) with regards to the ambient temperature T_A and the junction temperature T_J . For power dissipation

calculations refer to **Section A.1.8 Power Dissipation and Thermal Characteristics**.

Table A-4 Operating Conditions

Rating	Symbol	Min	Typ	Max	Unit
I/O, Regulator and Analog Supply Voltage	V_{DD5}	4.5	5	5.25	V
Digital Logic Supply Voltage ¹	V_{DD}	2.35	2.5	2.75	V
PLL Supply Voltage ¹	V_{DDPLL}	2.25	2.5	2.75	V
Voltage Difference VDDX to VDDR and VDDA	Δ_{VDDX}	-0.1	0	0.1	V
Voltage Difference VSSX to VSSR and VSSA	Δ_{VSSX}	-0.1	0	0.1	V
Bus Frequency	f_{bus}	0.25 ²	-	25	MHz
MC9S12DT128C					
Operating Junction Temperature Range	T_J	-40	-	100	°C
Operating Ambient Temperature Range ³	T_A	-40	27	85	°C
MC9S12DT128V					
Operating Junction Temperature Range	T_J	-40	-	120	°C
Operating Ambient Temperature Range ³	T_A	-40	27	105	°C
MC9S12DT128M					
Operating Junction Temperature Range	T_J	-40	-	140	°C
Operating Ambient Temperature Range ³	T_A	-40	27	125	°C

NOTES:

1. The device contains an internal voltage regulator to generate the logic and PLL supply out of the I/O supply. The given operating range applies when this regulator is disabled and the device is powered from an external source.
2. Some blocks e.g. ATD (conversion) and NVMs (program/erase) require higher bus frequencies for proper operation.
3. Please refer to **Section A.1.8 Power Dissipation and Thermal Characteristics** for more details about the relation between ambient temperature T_A and device junction temperature T_J .

A.1.8 Power Dissipation and Thermal Characteristics

Power dissipation and thermal characteristics are closely related. The user must assure that the maximum operating junction temperature is not exceeded. The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \Theta_{JA})$$

T_J = Junction Temperature, [°C]

T_A = Ambient Temperature, [°C]

P_D = Total Chip Power Dissipation, [W]

Θ_{JA} = Package Thermal Resistance, [$^{\circ}\text{C}/\text{W}$]

The total power dissipation can be calculated from:

$$P_D = P_{INT} + P_{IO}$$

P_{INT} = Chip Internal Power Dissipation, [W]

Two cases with internal voltage regulator enabled and disabled must be considered:

1. Internal Voltage Regulator disabled

$$P_{INT} = I_{DD} \cdot V_{DD} + I_{DDPLL} \cdot V_{DDPLL} + I_{DDA} \cdot V_{DDA}$$

$$P_{IO} = \sum_i R_{DSON} \cdot I_{IO_i}^2$$

Which is the sum of all output currents on I/O ports associated with VDDX and VDDR.

For R_{DSON} is valid:

$$R_{DSON} = \frac{V_{OL}}{I_{OL}}; \text{for outputs driven low}$$

respectively

$$R_{DSON} = \frac{V_{DD5} - V_{OH}}{I_{OH}}; \text{for outputs driven high}$$

2. Internal voltage regulator enabled

$$P_{INT} = I_{DDR} \cdot V_{DDR} + I_{DDA} \cdot V_{DDA}$$

I_{DDR} is the current shown in **(Table A-7)** and not the overall current flowing into VDDR, which additionally contains the current flowing into the external loads with output high.

$$P_{IO} = \sum_i R_{DSON} \cdot I_{IO_i}^2$$

Which is the sum of all output currents on I/O ports associated with VDDX and VDDR.

Table A-5 Thermal Package Characteristics¹

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	T	Thermal Resistance LQFP112, single sided PCB ²	θ_{JA}	–	–	54	°C/W
2	T	Thermal Resistance LQFP112, double sided PCB with 2 internal planes ³	θ_{JA}	–	–	41	°C/W
3	T	Junction to Board LQFP112	θ_{JB}	–	–	31	°C/W
4	T	Junction to Case LQFP112	θ_{JC}	–	–	11	°C/W
5	T	Junction to Package Top LQFP112	Ψ_{JT}	–	–	2	°C/W
6	T	Thermal Resistance QFP 80, single sided PCB	θ_{JA}	–	–	51	°C/W
7	T	Thermal Resistance QFP 80, double sided PCB with 2 internal planes	θ_{JA}	–	–	41	°C/W
8	T	Junction to Board QFP80	θ_{JB}	–	–	27	°C/W
9	T	Junction to Case QFP80	θ_{JC}	–	–	14	°C/W
10	T	Junction to Package Top QFP80	Ψ_{JT}	–	–	3	°C/W

NOTES:

1. The values for thermal resistance are achieved by package simulations
2. PC Board according to EIA/JEDEC Standard 51-3
3. PC Board according to EIA/JEDEC Standard 51-7

A.1.9 I/O Characteristics

This section describes the characteristics of all 5V I/O pins. All parameters are not always applicable, e.g. not all pins feature pull up/down resistances.

Table A-6 5V I/O Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Input High Voltage	V_{IH}	$0.65 \cdot V_{DD5}$	–		V
	T	Input High Voltage	V_{IH}	–	–	$V_{DD5} + 0.3$	
2	P	Input Low Voltage	V_{IL}	–	–	$0.35 \cdot V_{DD5}$	V
	T	Input Low Voltage	V_{IL}	$V_{SS5} - 0.3$	–	–	V
3	C	Input Hysteresis	V_{HYS}		250		mV
4	P	Input Leakage Current (pins in high ohmic input mode) $V_{in} = V_{DD5}$ or V_{SS5}	I_{in}	-1.0	–	1.0	μA
5	C	Output High Voltage (pins in output mode) Partial Drive $I_{OH} = -2.0mA$	V_{OH}	$V_{DD5} - 0.8$	–	–	V
	P	Full Drive $I_{OH} = -10.0mA$					
6	C	Output Low Voltage (pins in output mode) Partial Drive $I_{OL} = +2.0mA$	V_{OL}	–	–	0.8	V
	P	Full Drive $I_{OL} = +10.0mA$					
7	P	Internal Pull Up Device Current, tested at V_{IL} Max.	I_{PUL}	–	–	-130	μA
8	C	Internal Pull Up Device Current, tested at V_{IH} Min.	I_{PUH}	-10	–	–	μA
9	P	Internal Pull Down Device Current, tested at V_{IH} Min.	I_{PDH}	–	–	130	μA
10	C	Internal Pull Down Device Current, tested at V_{IL} Max.	I_{PDL}	10	–	–	μA
11	D	Input Capacitance	C_{in}		6	–	pF
12	T	Injection current ¹ Single Pin limit Total Device Limit. Sum of all injected currents	I_{ICS} I_{ICP}	-2.5 -25	–	2.5 25	mA
13	P	Port H, J, P Interrupt Input Pulse filtered ²	t_{PULSE}			3	μs
14	P	Port H, J, P Interrupt Input Pulse passed ²	t_{PULSE}	10			μs

NOTES:

1. Refer to **Section A.1.4 Current Injection**, for more details
2. Parameter only applies in STOP or Pseudo STOP mode.

A.1.10 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

A.1.10.1 Measurement Conditions

All measurements are without output loads. Unless otherwise noted the currents are measured in single chip mode, internal voltage regulator enabled and at 25MHz bus frequency using a 4MHz oscillator in Colpitts mode. Production testing is performed using a square wave signal at the EXTAL input.

A.1.10.2 Additional Remarks

In expanded modes the currents flowing in the system are highly dependent on the load at the address, data and control signals as well as on the duty cycle of those signals. No generally applicable numbers can be given. A very good estimate is to take the single chip currents and add the currents due to the external loads.

Table A-7 Supply Current Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Run supply currents Single Chip, Internal regulator enabled	I_{DD5}			55	mA
2	P P	Wait Supply current All modules enabled, PLL on only RTI enabled ⁽¹⁾	I_{DDW}			30 5	mA
3	C P C C P C P C P	Pseudo Stop Current (RTI and COP disabled) ^{1, 2} -40°C 27°C 70°C 85°C "C" Temp Option 100°C 105°C "V" Temp Option 120°C 125°C "M" Temp Option 140°C	I_{DDPS}		370 400 450 550 600 650 800 850 1200	500 1600 2100 5000	μA
4	C C C C C C C	Pseudo Stop Current (RTI and COP enabled) ^{(1), (2)} -40°C 27°C 70°C 85°C 105°C 125°C 140°C	I_{DDPS}		570 600 650 750 850 1200 1500		μA
5	C P C C P C P C P	Stop Current ⁽²⁾ -40°C 27°C 70°C 85°C "C" Temp Option 100°C 105°C "V" Temp Option 120°C 125°C "M" Temp Option 140°C	I_{DD5}		12 25 100 100 130 160 200 350 400 600	100 1200 1700 5000	μA

NOTES:

1. PLL off, Oscillator in Colpitts Mode
2. At those low power dissipation levels $T_J = T_A$ can be assumed

A.2 ATD Characteristics

This section describes the characteristics of the analog to digital converter.

A.2.1 ATD Operating Characteristics

The **(Table A-8)** shows conditions under which the ATD operates.

The following constraints exist to obtain full-scale, full range results:

$V_{SSA} \leq V_{RL} \leq V_{IN} \leq V_{RH} \leq V_{DDA}$. This constraint exists since the sample buffer amplifier can not drive beyond the power supply levels that it ties to. If the input level goes outside of this range it will effectively be clipped.

Table A-8 ATD Operating Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	D	Reference Potential Low High	V_{RL} V_{RH}	V_{SSA} $V_{DDA}/2$		$V_{DDA}/2$ V_{DDA}	V V
2	C	Differential Reference Voltage ¹	$V_{RH}-V_{RL}$	4.50	5.00	5.25	V
3	D	ATD Clock Frequency	f_{ATDCLK}	0.5		2.0	MHz
4	D	ATD 10-Bit Conversion Period Clock Cycles ² Conv, Time at 2.0MHz ATD Clock f_{ATDCLK}	N_{CONV10} T_{CONV10}	14 7		28 14	Cycles μs
5	D	ATD 8-Bit Conversion Period Clock Cycles ⁽²⁾ Conv, Time at 2.0MHz ATD Clock f_{ATDCLK}	N_{CONV8} T_{CONV8}	12 6		26 13	Cycles μs
6	D	Stop Recovery Time ($V_{DDA}=5.0$ Volts)	t_{SR}			20	μs
7	P	Reference Supply current (Both ATD modules on)	I_{REF}			0.75	mA
8	P	Reference Supply current (Only one ATD module on)	I_{REF}			0.375	mA

NOTES:

1. Full accuracy is not guaranteed when differential voltage is less than 4.50V
2. The minimum time assumes a final sample period of 2 ATD clocks cycles while the maximum time assumes a final sample period of 16 ATD clocks.

A.2.2 Factors influencing accuracy

Three factors – source resistance, source capacitance and current injection – have an influence on the accuracy of the ATD.

A.2.2.1 Source Resistance:

Due to the input pin leakage current as specified in **(Table A-6)** in conjunction with the source resistance there will be a voltage drop from the signal source to the ATD input. The maximum source resistance R_S

specifies results in an error of less than 1/2 LSB (2.5mV) at the maximum leakage current. If device or operating conditions are less than worst case or leakage-induced error is acceptable, larger values of source resistance is allowed.

A.2.2.2 Source capacitance

When sampling an additional internal capacitor is switched to the input. This can cause a voltage drop due to charge sharing with the external and the pin capacitance. For a maximum sampling error of the input voltage $\leq 1\text{LSB}$, then the external filter capacitor, $C_f \geq 1024 * (C_{INS} - C_{INN})$.

A.2.2.3 Current injection

There are two cases to consider.

1. A current is injected into the channel being converted. The channel being stressed has conversion values of \$3FF (\$FF in 8-bit mode) for analog inputs greater than V_{RH} and \$000 for values less than V_{RL} unless the current is higher than specified as disruptive conditions.
2. Current is injected into pins in the neighborhood of the channel being converted. A portion of this current is picked up by the channel (coupling ratio K), This additional current impacts the accuracy of the conversion depending on the source resistance.

The additional input voltage error on the converted channel can be calculated as $V_{ERR} = K * R_S * I_{INJ}$, with I_{INJ} being the sum of the currents injected into the two pins adjacent to the converted channel.

Table A-9 ATD Electrical Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	C	Max input Source Resistance	R_S	-	-	1	K Ω
2	T	Total Input Capacitance Non Sampling Sampling	C_{INN} C_{INS}			10 22	pF
3	C	Disruptive Analog Input Current	I_{NA}	-2.5		2.5	mA
4	C	Coupling Ratio positive current injection	K_p			10^{-4}	A/A
5	C	Coupling Ratio negative current injection	K_n			10^{-2}	A/A

A.2.3 ATD accuracy

(Table A-10) specifies the ATD conversion performance excluding any errors due to current injection, input capacitance and source resistance.

Table A-10 ATD Conversion Performance

Conditions are shown in (Table A-4) unless otherwise noted $V_{REF} = V_{RH} - V_{RL} = 5.12V$. Resulting to one 8 bit count = 20mV and one 10 bit count = 5mV $f_{ATDCLK} = 2.0MHz$							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	10-Bit Resolution	LSB		5		mV
2	P	10-Bit Differential Nonlinearity	DNL	-1		1	Counts
3	P	10-Bit Integral Nonlinearity	INL	-2.5	±1.5	2.5	Counts
4	P	10-Bit Absolute Error ¹	AE	-3	±2.0	3	Counts
5	P	8-Bit Resolution	LSB		20		mV
6	P	8-Bit Differential Nonlinearity	DNL	-0.5		0.5	Counts
7	P	8-Bit Integral Nonlinearity	INL	-1.0	±0.5	1.0	Counts
8	P	8-Bit Absolute Error ⁽¹⁾	AE	-1.5	±1.0	1.5	Counts

NOTES:

1. These values include the quantization error which is inherently 1/2 count for any A/D converter.

For the following definitions see also **Figure A-1**.

Differential Non-Linearity (DNL) is defined as the difference between two adjacent switching steps.

$$DNL(i) = \frac{V_i - V_{i-1}}{1LSB} - 1$$

The Integral Non-Linearity (INL) is defined as the sum of all DNLs:

$$INL(n) = \sum_{i=1}^n DNL(i) = \frac{V_n - V_0}{1LSB} - n$$

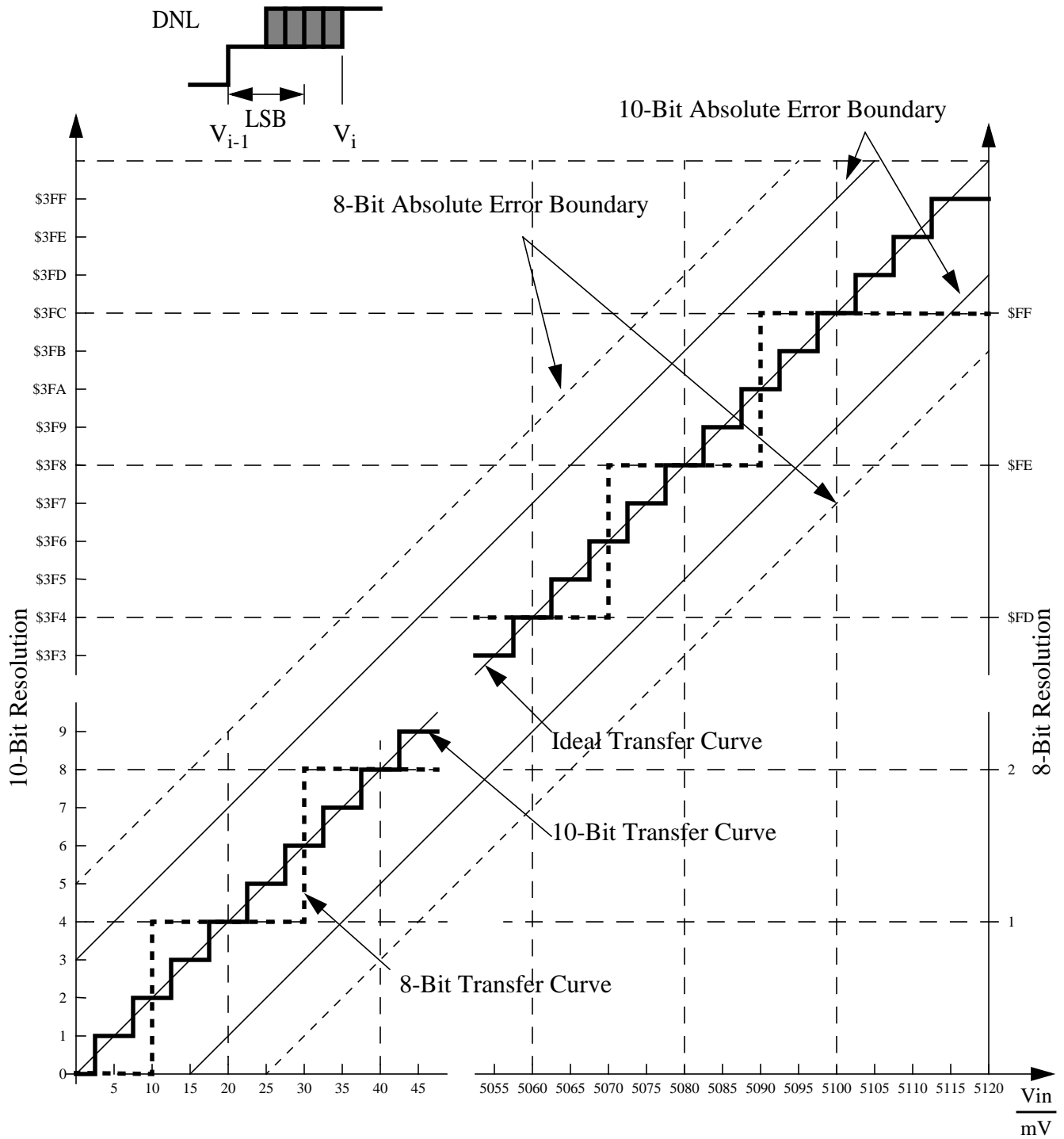


Figure A-1 ATD Accuracy Definitions

NOTE: *Figure A-1 shows only definitions, for specification values refer to **Table A-10**.*

A.3 NVM, Flash and EEPROM

NOTE: Unless otherwise noted the abbreviation NVM (Non Volatile Memory) is used for both Flash and EEPROM.

A.3.1 NVM timing

The time base for all NVM program or erase operations is derived from the oscillator. A minimum oscillator frequency f_{NVMOSC} is required for performing program or erase operations. The NVM modules do not have any means to monitor the frequency and will not prevent program or erase operation at frequencies above or below the specified minimum. Attempting to program or erase the NVM modules at a lower frequency a full program or erase transition is not assured.

The Flash and EEPROM program and erase operations are timed using a clock derived from the oscillator using the FCLKDIV and ECLKDIV registers respectively. The frequency of this clock must be set within the limits specified as f_{NVMOP} .

The minimum program and erase times shown in **(Table A-11)** are calculated for maximum f_{NVMOP} and maximum f_{bus} . The maximum times are calculated for minimum f_{NVMOP} and a f_{bus} of 2MHz.

A.3.1.1 Single Word Programming

The programming time for single word programming is dependant on the bus frequency as a well as on the frequency f_{NVMOP} and can be calculated according to the following formula.

$$t_{\text{swpgm}} = 9 \cdot \frac{1}{f_{\text{NVMOP}}} + 25 \cdot \frac{1}{f_{\text{bus}}}$$

A.3.1.2 Row Programming

This applies only to the Flash where up to 32 words in a row can be programmed consecutively by keeping the command pipeline filled. The time to program a consecutive word can be calculated as:

$$t_{\text{bwp gm}} = 4 \cdot \frac{1}{f_{\text{NVMOP}}} + 9 \cdot \frac{1}{f_{\text{bus}}}$$

The time to program a whole row is:

$$t_{\text{brpgm}} = t_{\text{swpgm}} + 31 \cdot t_{\text{bwp gm}}$$

Row programming is more than 2 times faster than single word programming.

A.3.1.3 Sector Erase

Erasing a 512 byte Flash sector or a 4 byte EEPROM sector takes:

$$t_{era} \approx 4000 \cdot \frac{1}{f_{NVMOP}}$$

The setup time can be ignored for this operation.

A.3.1.4 Mass Erase

Erasing a NVM block takes:

$$t_{mass} \approx 20000 \cdot \frac{1}{f_{NVMOP}}$$

The setup time can be ignored for this operation.

A.3.1.5 Blank Check

The time it takes to perform a blank check on the Flash or EEPROM is dependant on the location of the first non-blank word starting at relative address zero. It takes one bus cycle per word to verify plus a setup of the command.

$$t_{check} \approx \text{location} \cdot t_{cyc} + 10 \cdot t_{cyc}$$

Table A-11 NVM Timing Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	D	External Oscillator Clock	f_{NVMOSC}	0.5		50 ¹	MHz
2	D	Bus frequency for Programming or Erase Operations	f_{NVMBUS}	1			MHz
3	D	Operating Frequency	f_{NVMOP}	150		200	kHz
4	P	Single Word Programming Time	t_{swpgm}	46 ²		74.5 ³	μ s
5	D	Flash Row Programming consecutive word ⁴	t_{bwpgm}	20.4 ⁽²⁾		31 ⁽³⁾	μ s
6	D	Flash Row Programming Time for 32 Words ⁽⁴⁾	t_{brpgm}	678.4 ⁽²⁾		1035.5 ⁽³⁾	μ s
7	P	Sector Erase Time	t_{era}	20 ⁵		26.7 ⁽³⁾	ms
8	P	Mass Erase Time	t_{mass}	100 ⁽⁵⁾		133 ⁽³⁾	ms
9	D	Blank Check Time Flash per block	t_{check}	11 ⁶		32778 ⁷	t_{cyc}
10	D	Blank Check Time EEPROM per block	t_{check}	11 ⁽⁶⁾		1034 ⁽⁷⁾	t_{cyc}

NOTES:

1. Restrictions for oscillator in crystal mode apply!
2. Minimum Programming times are achieved under maximum NVM operating frequency f_{NVMOP} and maximum bus frequency f_{bus} .
3. Maximum Erase and Programming times are achieved under particular combinations of f_{NVMOP} and bus frequency f_{bus} . Refer to formulae in Sections **Section A.3.1.1 Single Word Programming-** **Section A.3.1.4 Mass Erase** for guidance.
4. Row Programming operations are not applicable to EEPROM
5. Minimum Erase times are achieved under maximum NVM operating frequency f_{NVMOP} .
6. Minimum time, if first word in the array is not blank
7. Maximum time to complete check on an erased block

A.3.2 NVM Reliability

The reliability of the NVM blocks is guaranteed by stress test during qualification, constant process monitors and burn-in to screen early life failures.

The failure rates for data retention and program/erase cycling are specified at the operating conditions noted.

The program/erase cycle count on the sector is incremented every time a sector or mass erase event is executed.

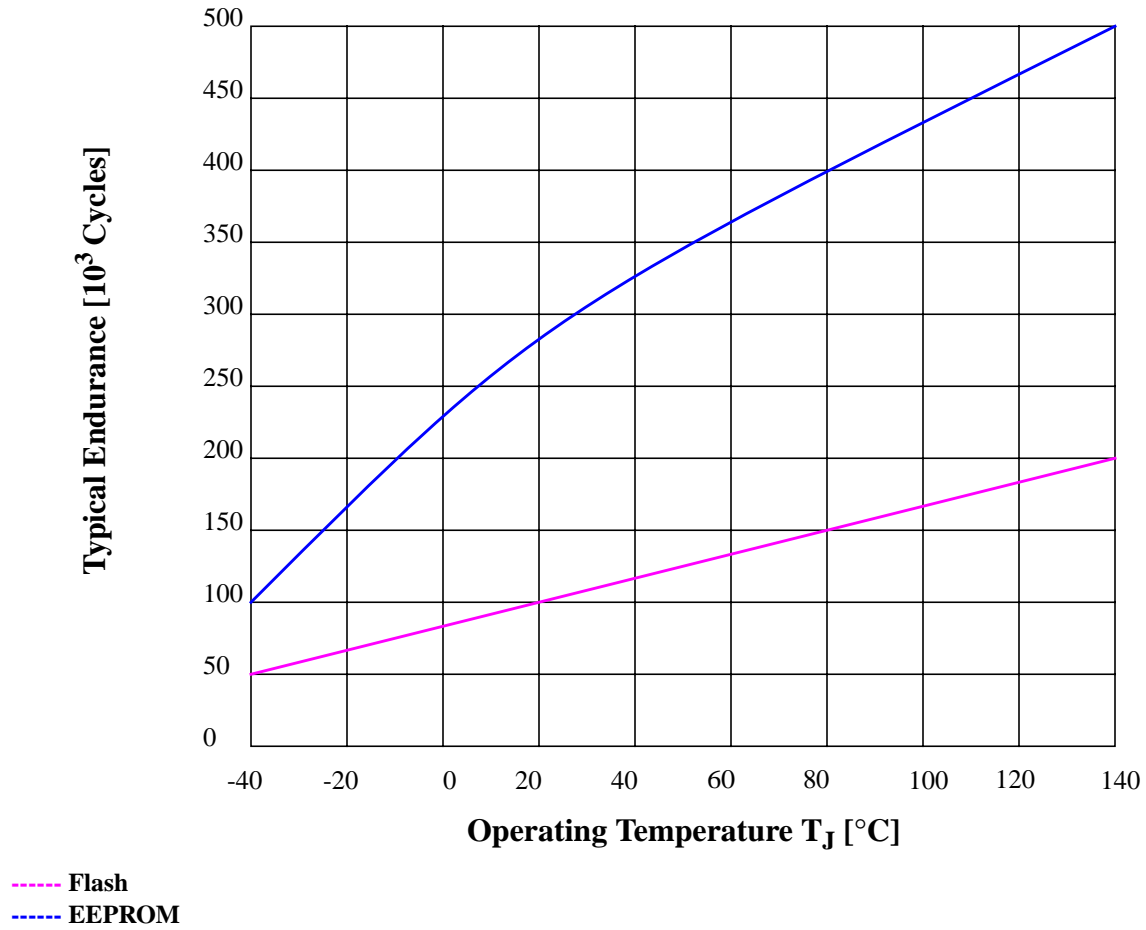
Table A-12 NVM Reliability Characteristics¹

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
Flash Reliability Characteristics							
1	C	Data retention after 10,000 program/erase cycles at an average junction temperature of $T_{Javg} \leq 85^{\circ}\text{C}$	t_{FLRET}	15	100^2	—	Years
2	C	Data retention with <100 program/erase cycles at an average junction temperature $T_{Javg} \leq 85^{\circ}\text{C}$		20	100^2	—	
3	C	Number of program/erase cycles ($-40^{\circ}\text{C} \leq T_J \leq 0^{\circ}\text{C}$)	n_{FL}	10,000	—	—	Cycles
4	C	Number of program/erase cycles ($0^{\circ}\text{C} \leq T_J \leq 140^{\circ}\text{C}$)		10,000	$100,000^3$	—	
EEPROM Reliability Characteristics							
5	C	Data retention after up to 100,000 program/erase cycles at an average junction temperature of $T_{Javg} \leq 85^{\circ}\text{C}$	t_{EEPRET}	15	100^2	—	Years
6	C	Data retention with <100 program/erase cycles at an average junction temperature $T_{Javg} \leq 85^{\circ}\text{C}$		20	100^2	—	
7	C	Number of program/erase cycles ($-40^{\circ}\text{C} \leq T_J \leq 0^{\circ}\text{C}$)	n_{EEP}	10,000	—	—	Cycles
8	C	Number of program/erase cycles ($0^{\circ}\text{C} < T_J \leq 140^{\circ}\text{C}$)		100,000	$300,000^3$	—	

NOTES:

- T_{Javg} will not exceed 85°C considering a typical temperature profile over the lifetime of a consumer, industrial or automotive application.
- Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how Freescale defines Typical Data Retention, please refer to Engineering Bulletin EB618.
- Spec table quotes typical endurance evaluated at 25°C for this product family, typical endurance at various temperature can be estimated using the graph below. For additional information on how Freescale defines Typical Endurance, please refer to Engineering Bulletin EB619.

Figure A-2 Typical Endurance vs Temperature



A.4 Voltage Regulator

The on-chip voltage regulator is intended to supply the internal logic and oscillator circuits. No external DC load is allowed.

Table A-13 Voltage Regulator Recommended Load Capacitances

Rating	Symbol	Min	Typ	Max	Unit
Load Capacitance on VDD1, 2	C_{LVDD}		220		nF
Load Capacitance on VDDPLL	$C_{LVDDfcPLL}$		220		nF

A.5 Reset, Oscillator and PLL

This section summarizes the electrical characteristics of the various startup scenarios for Oscillator and Phase-Locked-Loop (PLL).

A.5.1 Startup

(Table A-14) summarizes several startup characteristics explained in this section. Detailed description of the startup behavior can be found in the Clock and Reset Generator (CRG) Block User Guide.

Table A-14 Startup Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	T	POR release level	V_{PORR}			2.07	V
2	T	POR assert level	V_{PORA}	0.97			V
3	D	Reset input pulse width, minimum input time	PW_{RSTL}	2			t_{osc}
4	D	Startup from Reset	n_{RST}	192		196	n_{osc}
5	D	Interrupt pulse width, \overline{IRQ} edge-sensitive mode	PW_{IRQ}	20			ns
6	D	Wait recovery startup time	t_{WRS}			14	t_{cyc}

A.5.1.1 POR

The release level V_{PORR} and the assert level V_{PORA} are derived from the V_{DD} Supply. They are also valid if the device is powered externally. After releasing the POR reset the oscillator and the clock quality check are started. If after a time t_{CQOUT} no valid oscillation is detected, the MCU will start using the internal self clock. The fastest startup time possible is given by n_{uposc} .

A.5.1.2 SRAM Data Retention

Provided an appropriate external reset signal is applied to the MCU, preventing the CPU from executing code when V_{DD5} is out of specification limits, the SRAM contents integrity is guaranteed if after the reset the PORF bit in the CRG Flags Register has not been set.

A.5.1.3 External Reset

When external reset is asserted for a time greater than PW_{RSTL} the CRG module generates an internal reset, and the CPU starts fetching the reset vector without doing a clock quality check, if there was an oscillation before reset.

A.5.1.4 Stop Recovery

Out of STOP the controller can be woken up by an external interrupt. A clock quality check as after POR is performed before releasing the clocks to the system.

A.5.1.5 Pseudo Stop and Wait Recovery

The recovery from Pseudo STOP and Wait are essentially the same since the oscillator was not stopped in both modes. The controller can be woken up by internal or external interrupts. After t_{WRS} the CPU starts fetching the interrupt vector.

A.5.2 Oscillator

The device features an internal Colpitts and Pierce oscillator. The selection of Colpitts oscillator or Pierce oscillator/external clock depends on the XCLKS signal which is sampled during reset. By asserting the \overline{XCLKS} input during reset this oscillator can be bypassed allowing the input of a square wave. Before asserting the oscillator to the internal system clocks the quality of the oscillation is checked for each start from either power-on, STOP or oscillator fail. t_{CQOUT} specifies the maximum time before switching to the internal self clock mode after POR or STOP if a proper oscillation is not detected. The quality check also determines the minimum oscillator start-up time t_{UPOSC} . The device also features a clock monitor. A

Clock Monitor Failure is asserted if the frequency of the incoming clock signal is below the Assert Frequency f_{CMFA} .

Table A-15 Oscillator Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1a	C	Crystal oscillator range (Colpitts)	f_{OSC}	0.5		16	MHz
1b	C	Crystal oscillator range (Pierce) ¹	f_{OSC}	0.5		40	MHz
2	P	Startup Current	i_{OSC}	100			μA
3	C	Oscillator start-up time (Colpitts)	t_{UOSC}		8^2	100^3	ms
4	D	Clock Quality check time-out	t_{CQOUT}	0.45		2.5	s
5	P	Clock Monitor Failure Assert Frequency	f_{CMFA}	50	100	200	KHz
6	P	External square wave input frequency ⁴	f_{EXT}	0.5		50	MHz
7	D	External square wave pulse width low	t_{EXTL}	9.5			ns
8	D	External square wave pulse width high	t_{EXTH}	9.5			ns
9	D	External square wave rise time	t_{EXTR}			1	ns
10	D	External square wave fall time	t_{EXTF}			1	ns
11	D	Input Capacitance (EXTAL, XTAL pins)	C_{IN}		7		pF
12	C	DC Operating Bias in Colpitts Configuration on EXTAL Pin	V_{DCBIAS}		1.1		V
13	P	EXTAL Pin Input High Voltage ⁴	$V_{IH,EXTAL}$	$0.75 \cdot V_{DDPLL}$			V
	T	EXTAL Pin Input High Voltage ⁴	$V_{IH,EXTAL}$			$V_{DDPLL} + 0.3$	V
14	P	EXTAL Pin Input Low Voltage ⁴	$V_{IL,EXTAL}$			$0.25 \cdot V_{DDPLL}$	V
	T	EXTAL Pin Input Low Voltage ⁴	$V_{IL,EXTAL}$	$V_{SSPLL} - 0.3$			V
15	C	EXTAL Pin Input Hysteresis ⁴	$V_{HYS,EXTAL}$		250		mV

NOTES:

1. Depending on the crystal a damping series resistor might be necessary
2. $f_{osc} = 4\text{MHz}$, $C = 22\text{pF}$.
3. Maximum value is for extreme cases using high Q, low frequency crystals
4. $XCLKS = 0$ during reset

A.5.3 Phase Locked Loop

The oscillator provides the reference clock for the PLL. The PLL's Voltage Controlled Oscillator (VCO) is also the system clock source in self clock mode.

A.5.3.1 XFC Component Selection

This section describes the selection of the XFC components to achieve a good filter characteristics.

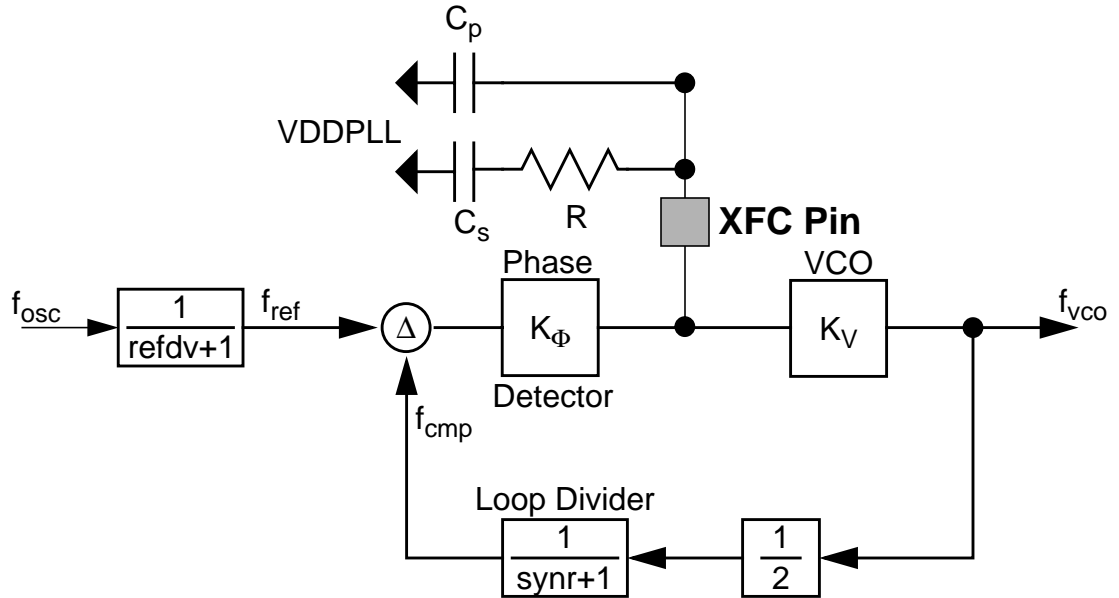


Figure A-3 Basic PLL functional diagram

The following procedure can be used to calculate the resistance and capacitance values using typical values for K_1 , f_1 and i_{ch} from **(Table A-16)**.

The grey boxes show the calculation for $f_{VCO} = 50\text{MHz}$ and $f_{ref} = 1\text{MHz}$. E.g., these frequencies are used for $f_{OSC} = 4\text{MHz}$ and a 25MHz bus clock.

The VCO Gain at the desired VCO frequency is approximated by:

$$K_V = K_1 \cdot e^{\frac{(f_1 - f_{VCO})}{K_1 \cdot 1V}} = -100 \cdot e^{\frac{(60 - 50)}{-100}} = -90.48\text{MHz/V}$$

The phase detector relationship is given by:

$$K_\Phi = -|i_{ch}| \cdot K_V = 316.7\text{Hz}/\Omega$$

i_{ch} is the current in tracking mode.

The loop bandwidth f_C should be chosen to fulfill the Gardner's stability criteria by at least a factor of 10, typical values are 50. $\zeta = 0.9$ ensures a good transient response.

$$f_C < \frac{2 \cdot \zeta \cdot f_{ref}}{\pi \cdot (\zeta + \sqrt{1 + \zeta^2})} \cdot \frac{1}{10} \rightarrow f_C < \frac{f_{ref}}{4 \cdot 10}; (\zeta = 0.9)$$

$$f_C < 25\text{kHz}$$

And finally the frequency relationship is defined as

$$n = \frac{f_{VCO}}{f_{ref}} = 2 \cdot (\text{synr} + 1) = 50$$

With the above values the resistance can be calculated. The example is shown for a loop bandwidth $f_C=10\text{KHz}$:

$$R = \frac{2 \cdot \pi \cdot n \cdot f_C}{K_\Phi} = 2 \cdot \pi \cdot 50 \cdot 10\text{kHz} / (316.7\text{Hz}/\Omega) = 9.9\text{k}\Omega \approx 10\text{k}\Omega$$

The capacitance C_s can now be calculated as:

$$C_s = \frac{2 \cdot \zeta^2}{\pi \cdot f_C \cdot R} \approx \frac{0.516}{f_C \cdot R}; (\zeta = 0.9) = 5.19\text{nF} \approx 4.7\text{nF}$$

The capacitance C_p should be chosen in the range of:

$$C_s/20 \leq C_p \leq C_s/10 \quad C_p = 470\text{pF}$$

A.5.3.2 Jitter Information

The basic functionality of the PLL is shown in **Figure A-3**. With each transition of the clock f_{cmp} , the deviation from the reference clock f_{ref} is measured and input voltage to the VCO is adjusted accordingly. The adjustment is done continuously with no abrupt changes in the clock output frequency. Noise, voltage, temperature and other factors cause slight variations in the control loop resulting in a clock jitter. This jitter affects the real minimum and maximum clock periods as illustrated in **Figure A-4**.

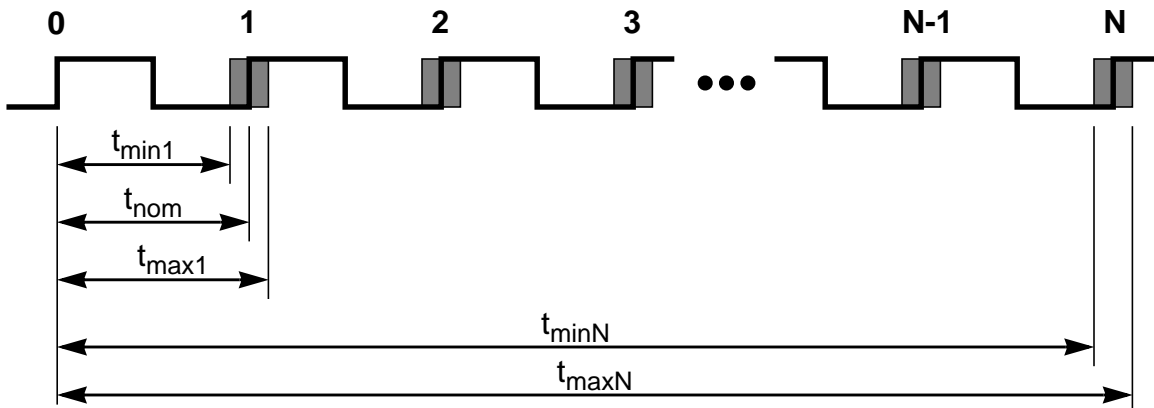


Figure A-4 Jitter Definitions

The relative deviation of t_{nom} is at its maximum for one clock period, and decreases towards zero for larger number of clock periods (N).

Defining the jitter as:

$$J(N) = \max\left(\left|1 - \frac{t_{max}(N)}{N \cdot t_{nom}}\right|, \left|1 - \frac{t_{min}(N)}{N \cdot t_{nom}}\right|\right)$$

For $N < 100$, the following equation is a good fit for the maximum jitter:

$$J(N) = \frac{j_1}{\sqrt{N}} + j_2$$

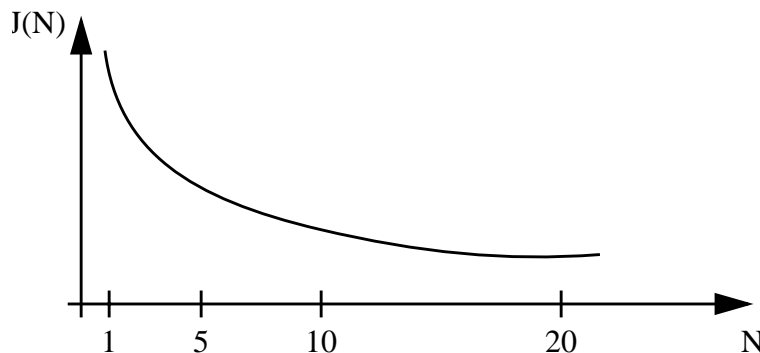


Figure A-5 Maximum bus clock jitter approximation

This is very important to notice with respect to timers, serial modules where a pre-scaler will eliminate the effect of the jitter to a large extent.

Table A-16 PLL Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Self Clock Mode frequency	f_{SCM}	1		5.5	MHz
2	D	VCO locking range	f_{VCO}	8		50	MHz
3	D	Lock Detector transition from Acquisition to Tracking mode	$ \Delta_{trk} $	3		4	% ¹
4	D	Lock Detection	$ \Delta_{Lock} $	0		1.5	% ⁽¹⁾
5	D	Un-Lock Detection	$ \Delta_{unt} $	0.5		2.5	% ⁽¹⁾
6	D	Lock Detector transition from Tracking to Acquisition mode	$ \Delta_{unt} $	6		8	% ⁽¹⁾
7	C	PLLON Total Stabilization delay (Auto Mode) ²	t_{stab}		0.5		ms
8	D	PLLON Acquisition mode stabilization delay ⁽²⁾	t_{acq}		0.3		ms
9	D	PLLON Tracking mode stabilization delay ⁽²⁾	t_{al}		0.2		ms
10	D	Fitting parameter VCO loop gain	K_1		-100		MHz/V
11	D	Fitting parameter VCO loop frequency	f_1		60		MHz
12	D	Charge pump current acquisition mode	$ i_{ch} $		38.5		μA
13	D	Charge pump current tracking mode	$ i_{ch} $		3.5		μA
14	C	Jitter fit parameter 1 ⁽²⁾	j_1			1.1	%
15	C	Jitter fit parameter 2 ⁽²⁾	j_2			0.13	%

NOTES:

1. % deviation from target frequency
2. $f_{OSC} = 4\text{MHz}$, $f_{BUS} = 25\text{MHz}$ equivalent $f_{VCO} = 50\text{MHz}$: REFDV = #03, SYNRR = #018, Cs = 4.7nF, Cp = 470pF, Rs = 10K Ω .

A.6 MSCAN

Table A-17 MSCAN Wake-up Pulse Characteristics

Conditions are shown in (Table A-4) unless otherwise noted							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	MSCAN Wake-up dominant pulse filtered	t_{WUP}			2	μs
2	P	MSCAN Wake-up dominant pulse pass	t_{WUP}	5			μs

A.7 SPI

A.7.1 Master Mode

Figure A-6 and Figure A-7 illustrate the master mode timing. Timing values are shown in (Table A-18).

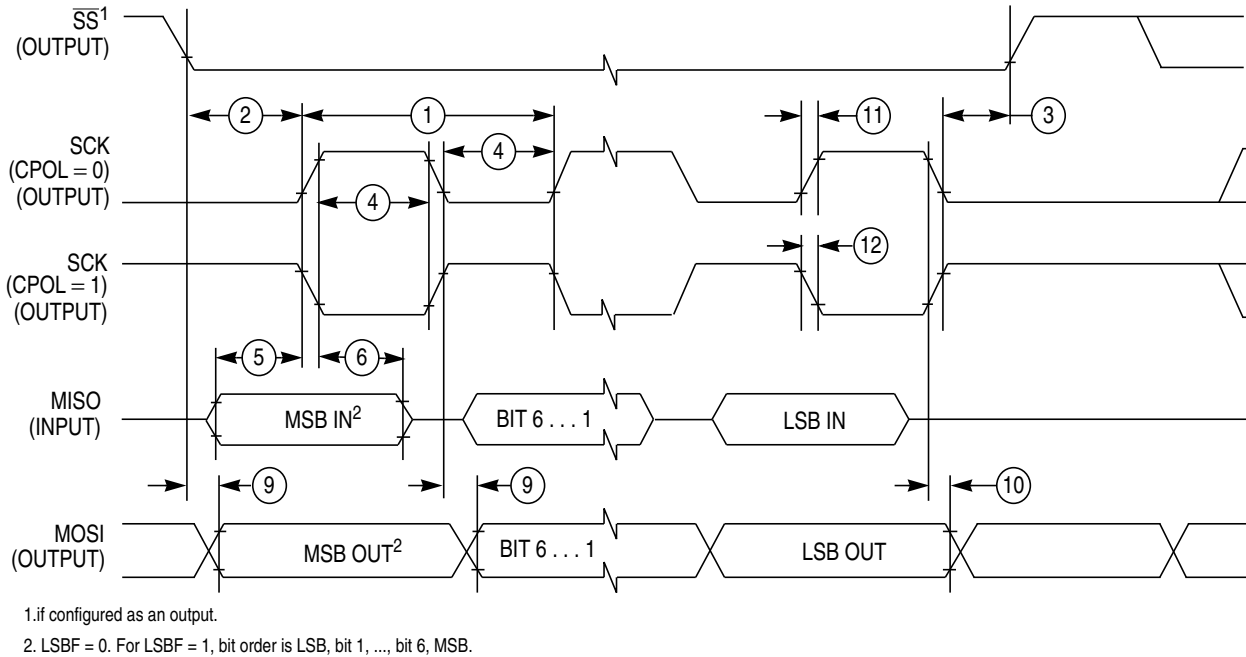
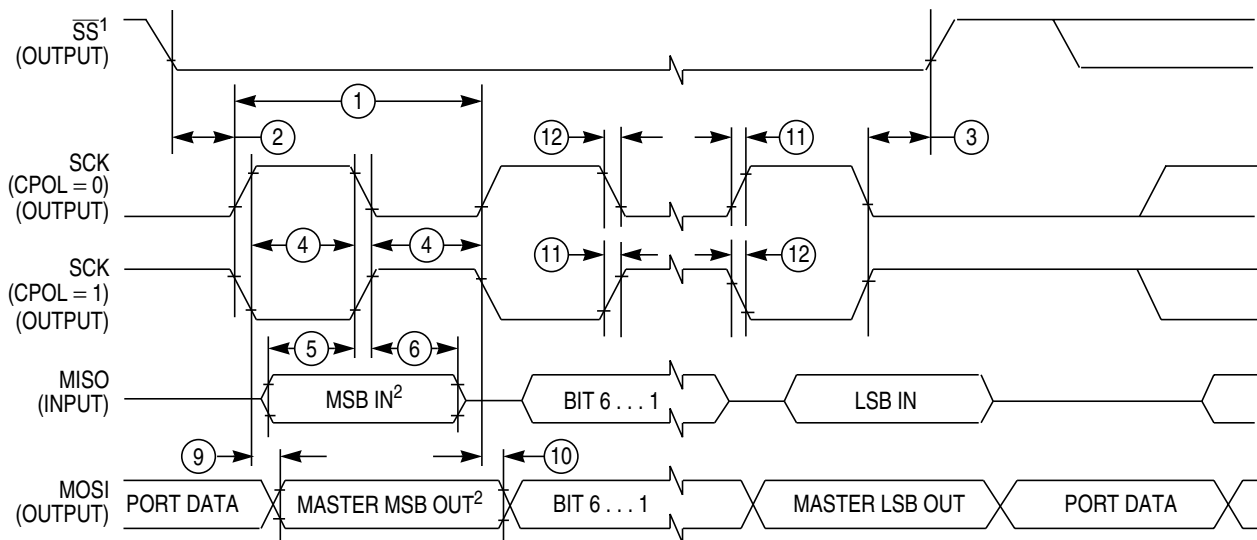


Figure A-6 SPI Master Timing (CPHA = 0)



- 1. If configured as output
- 2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, ..., bit 6, MSB.

Figure A-7 SPI Master Timing (CPHA =1)

Table A-18 SPI Master Mode Timing Characteristics¹

Conditions are shown in (Table A-4) unless otherwise noted, $C_{LOAD} = 200pF$ on all outputs

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Operating Frequency	f_{op}	DC		1/2	f_{bus}
1	P	SCK Period $t_{sck} = 1./f_{op}$	t_{sck}	4		2048	t_{bus}
2	D	Enable Lead Time	t_{lead}	1/2		—	t_{sck}
3	D	Enable Lag Time	t_{lag}	1/2			t_{sck}
4	D	Clock (SCK) High or Low Time	t_{wsck}	$t_{bus} - 30$		$1024 t_{bus}$	ns
5	D	Data Setup Time (Inputs)	t_{su}	25			ns
6	D	Data Hold Time (Inputs)	t_{hi}	0			ns
9	D	Data Valid (after SCK Edge)	t_v			25	ns
10	D	Data Hold Time (Outputs)	t_{ho}	0			ns
11	D	Rise Time Inputs and Outputs	t_r			25	ns
12	D	Fall Time Inputs and Outputs	t_f			25	ns

NOTES:

- 1. The numbers 7, 8 in the column labeled "Num" are missing. This has been done on purpose to be consistent between the Master and the Slave timing shown in (Table A-19).

A.7.2 Slave Mode

Figure A-8 and Figure A-9 illustrate the slave mode timing. Timing values are shown in (Table A-19).

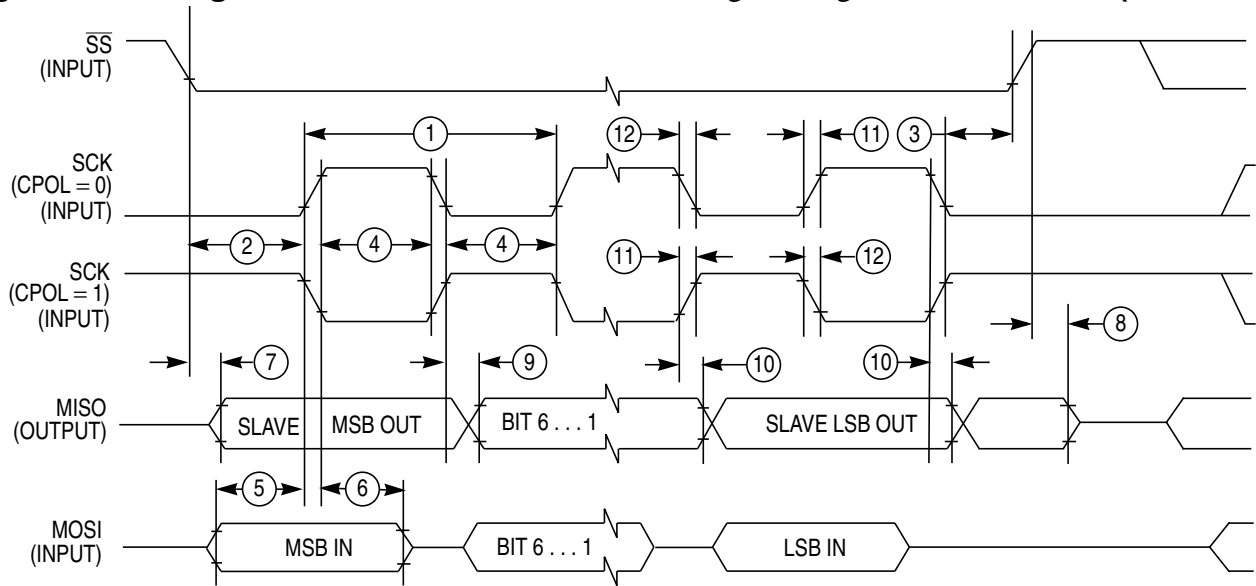


Figure A-8 SPI Slave Timing (CPHA = 0)

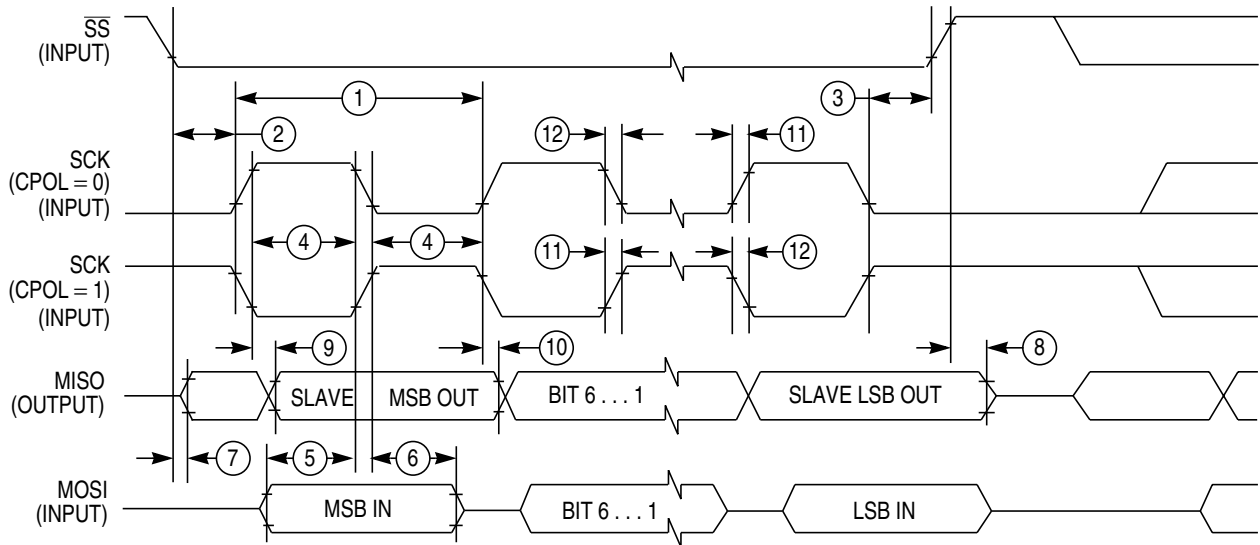


Figure A-9 SPI Slave Timing (CPHA = 1)

Table A-19 SPI Slave Mode Timing Characteristics

Conditions are shown in (Table A-4) unless otherwise noted, CLOAD = 200pF on all outputs							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Operating Frequency	f_{op}	DC		1/4	f_{bus}
1	P	SCK Period $t_{sck} = 1./f_{op}$	t_{sck}	4		2048	t_{bus}
2	D	Enable Lead Time	t_{lead}	1			t_{cyc}
3	D	Enable Lag Time	t_{lag}	1			t_{cyc}
4	D	Clock (SCK) High or Low Time	t_{wsck}	$t_{cyc} - 30$			ns
5	D	Data Setup Time (Inputs)	t_{su}	25			ns
6	D	Data Hold Time (Inputs)	t_{hi}	25			ns
7	D	Slave Access Time	t_a			1	t_{cyc}
8	D	Slave MISO Disable Time	t_{dis}			1	t_{cyc}
9	D	Data Valid (after SCK Edge)	t_v			25	ns
10	D	Data Hold Time (Outputs)	t_{ho}	0			ns
11	D	Rise Time Inputs and Outputs	t_r			25	ns
12	D	Fall Time Inputs and Outputs	t_f			25	ns

A.8 External Bus Timing

A timing diagram of the external multiplexed-bus is illustrated in **Figure A-10** with the actual timing values shown on table (**Table A-20**). All major bus signals are included in the diagram. While both a data write and data read cycle are shown, only one or the other would occur on a particular bus cycle.

A.8.1 General Multiplexed Bus Timing

The expanded bus timings are highly dependent on the load conditions. The timing parameters shown assume a balanced load across all outputs.

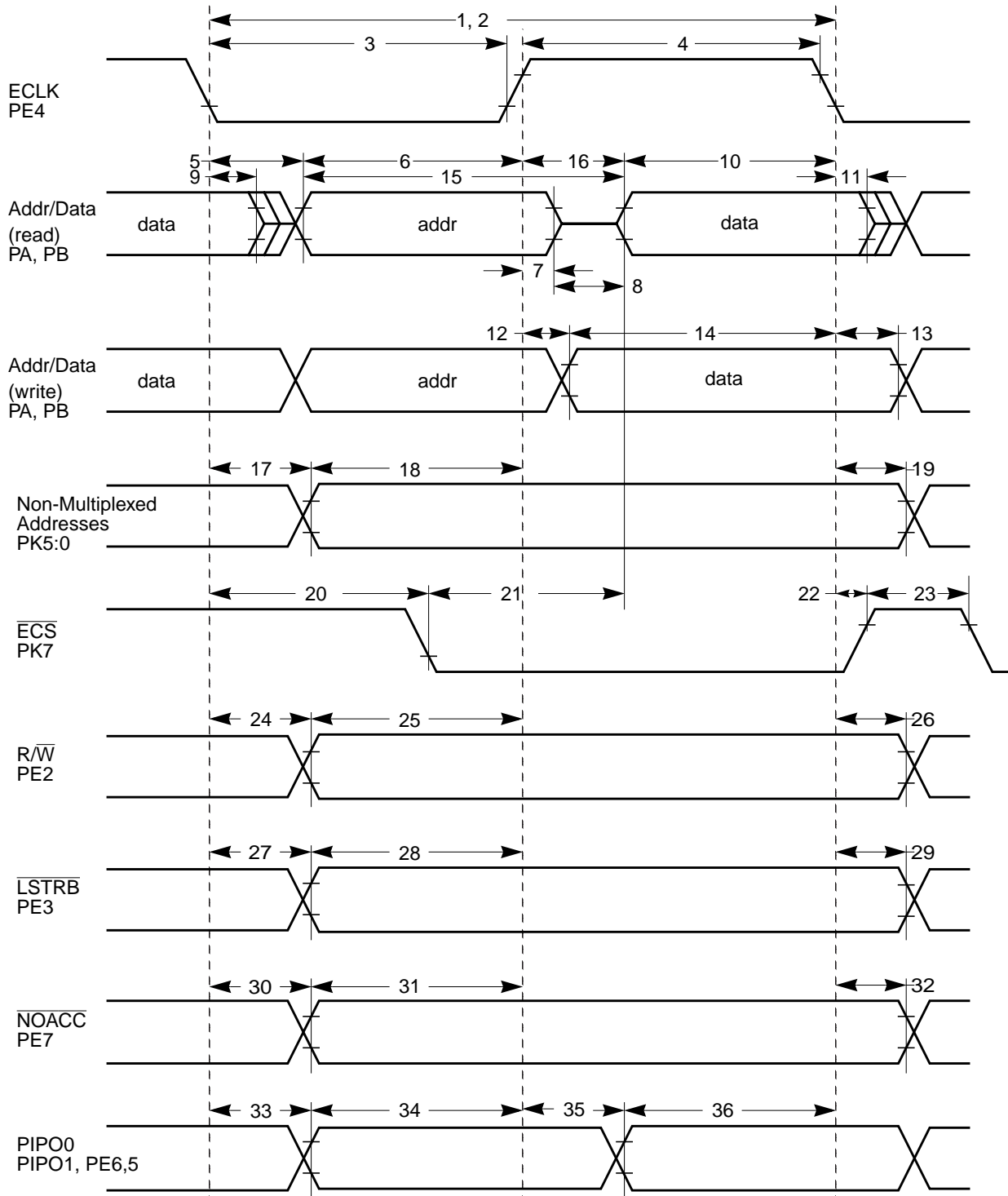


Figure A-10 General External Bus Timing

Table A-20 Expanded Bus Timing Characteristics

Conditions are shown in (Table A-4) unless otherwise noted, C _{LOAD} = 50pF							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Frequency of operation (E-clock)	f _o	0		25.0	MHz
2	P	Cycle time	t _{cyc}	40			ns
3	D	Pulse width, E low	PW _{EL}	19			ns
4	D	Pulse width, E high ¹	PW _{EH}	19			ns
5	D	Address delay time	t _{AD}			8	ns
6	D	Address valid time to E rise (PW _{EL} -t _{AD})	t _{AV}	11			ns
7	D	Muxed address hold time	t _{MAH}	2			ns
8	D	Address hold to data valid	t _{AHDS}	7			ns
9	D	Data hold to address	t _{DHA}	2			ns
10	D	Read data setup time	t _{DSR}	13			ns
11	D	Read data hold time	t _{DHR}	0			ns
12	D	Write data delay time	t _{DDW}			7	ns
13	D	Write data hold time	t _{DHW}	2			ns
14	D	Write data setup time ⁽¹⁾ (PW _{EH} -t _{DDW})	t _{DSW}	12			ns
15	D	Address access time ⁽¹⁾ (t _{cyc} -t _{AD} -t _{DSR})	t _{ACCA}	19			ns
16	D	E high access time ⁽¹⁾ (PW _{EH} -t _{DSR})	t _{ACCE}	6			ns
17	D	Non-multiplexed address delay time	t _{NAD}			6	ns
18	D	Non-muxed address valid to E rise (PW _{EL} -t _{NAD})	t _{NAV}	15			ns
19	D	Non-multiplexed address hold time	t _{NAH}	2			ns
20	D	Chip select delay time	t _{CSD}			16	ns
21	D	Chip select access time ⁽¹⁾ (t _{cyc} -t _{CSD} -t _{DSR})	t _{ACCS}	11			ns
22	D	Chip select hold time	t _{CSH}	2			ns
23	D	Chip select negated time	t _{CSN}	8			ns
24	D	Read/write delay time	t _{RWD}			7	ns
25	D	Read/write valid time to E rise (PW _{EL} -t _{RWD})	t _{RWV}	14			ns
26	D	Read/write hold time	t _{RWH}	2			ns
27	D	Low strobe delay time	t _{LSD}			7	ns
28	D	Low strobe valid time to E rise (PW _{EL} -t _{LSD})	t _{LSV}	14			ns
29	D	Low strobe hold time	t _{LSH}	2			ns
30	D	NOACC strobe delay time	t _{NOD}			7	ns
31	D	NOACC valid time to E rise (PW _{EL} -t _{NOD})	t _{NOV}	14			ns

Table A-20 Expanded Bus Timing Characteristics

Conditions are shown in (Table A-4) unless otherwise noted, C _{LOAD} = 50pF							
Num	C	Rating	Symbol	Min	Typ	Max	Unit
32	D	NOACC hold time	t _{NOH}	2			ns
33	D	IPIPO[1:0] delay time	t _{P0D}	2		7	ns
34	D	IPIPO[1:0] valid time to E rise (PW _{EL} -t _{P0D})	t _{P0V}	11			ns
35	D	IPIPO[1:0] delay time ⁽¹⁾ (PW _{EH} -t _{P1V})	t _{P1D}	2		25	ns
36	D	IPIPO[1:0] valid time to E fall	t _{P1V}	11			ns

NOTES:

1. Affected by clock stretch: add N x t_{cyc} where N=0,1,2 or 3, depending on the number of clock stretches.

Appendix B Package Information

B.1 General

This section provides the physical dimensions of the MC9S12DT128 packages.

B.2 112-pin LQFP package

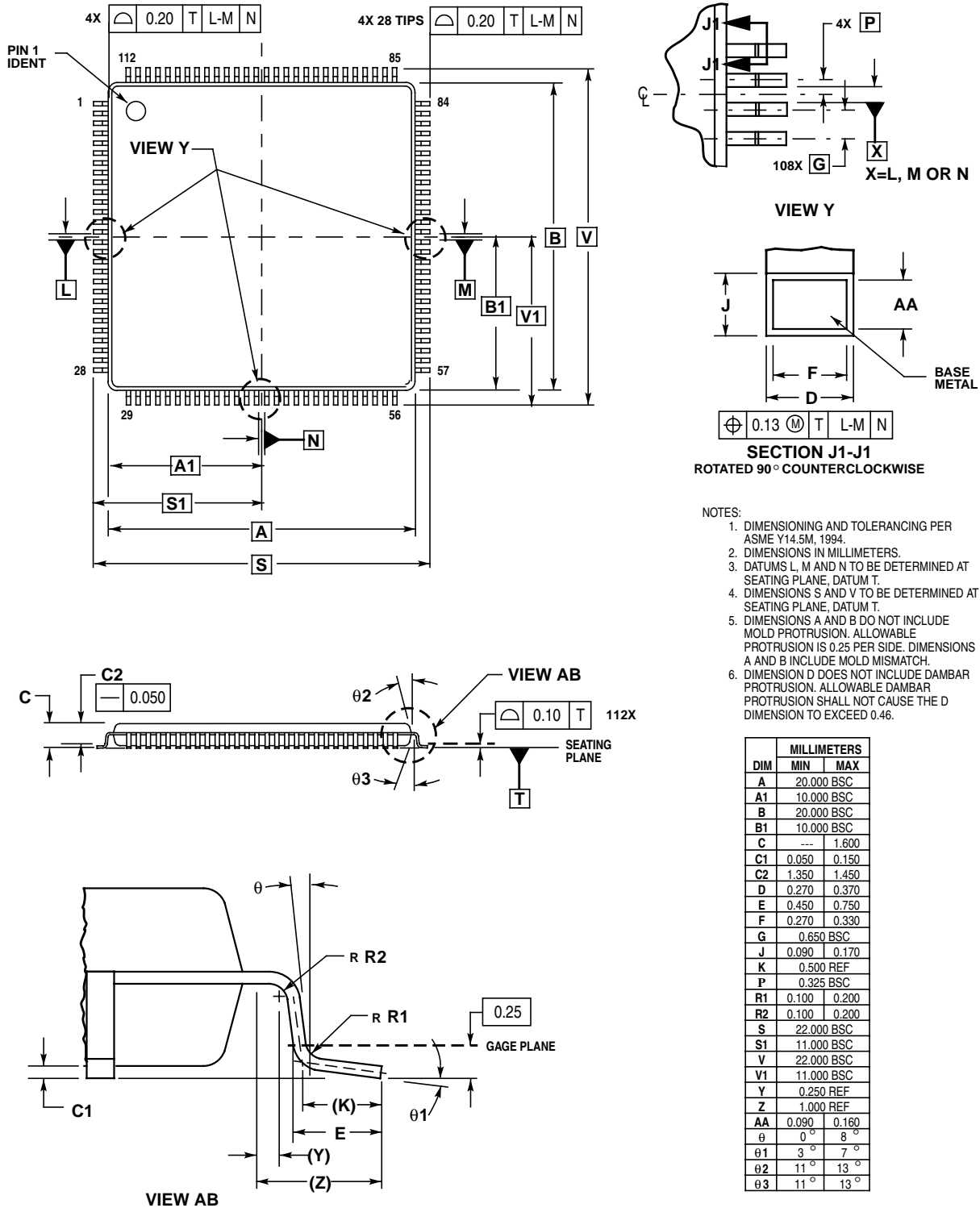


Figure 23-6 112-pin LQFP mechanical dimensions (case no. 987)

B.3 80-pin QFP package

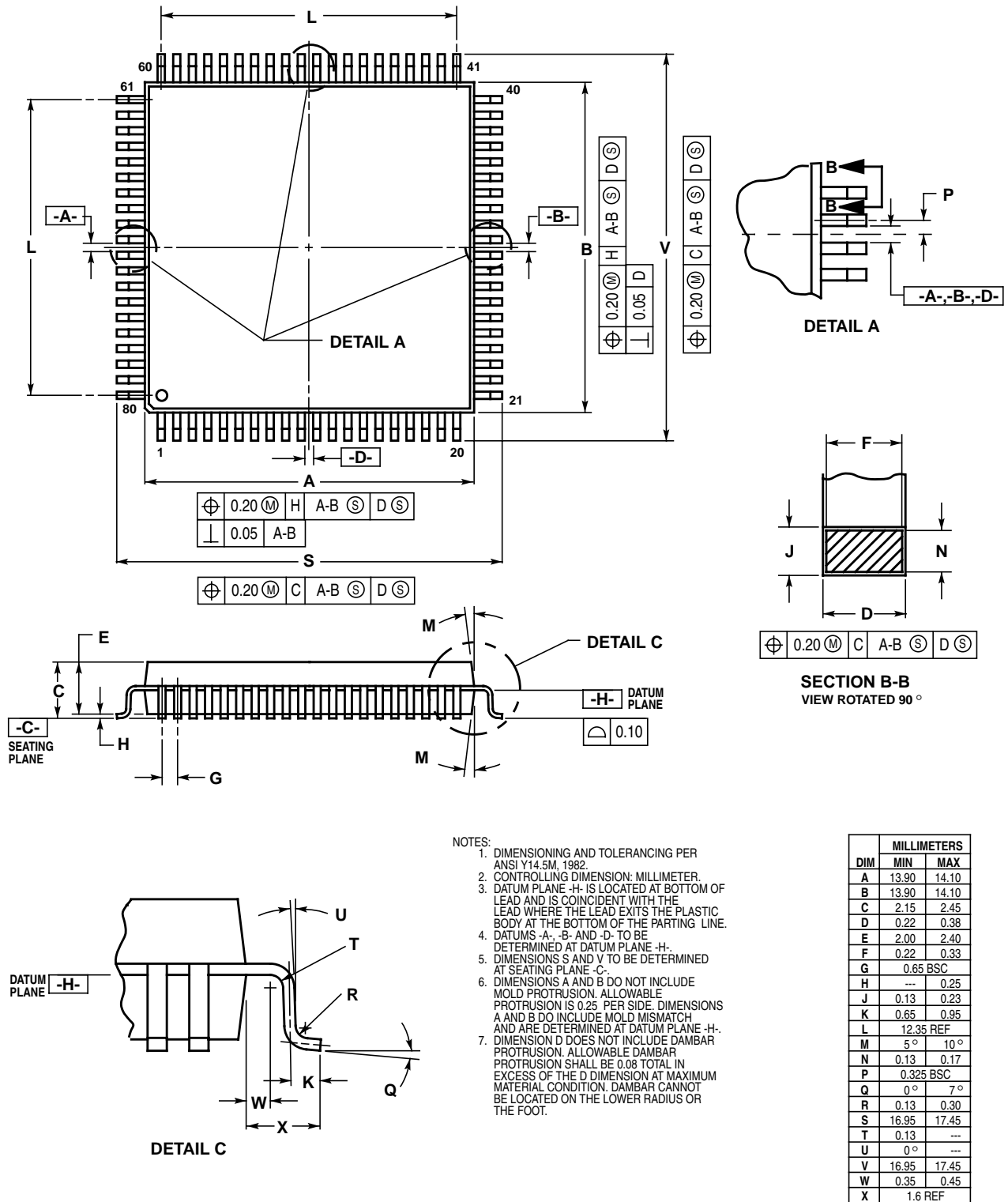


Figure 1 80-pin QFP Mechanical Dimensions (case no. 841B)

