

# LM3S2110 Microcontroller

**DATA SHEET** 

## **Legal Disclaimers and Trademark Information**

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH LUMINARY MICRO PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN LUMINARY MICRO'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, LUMINARY MICRO ASSUMES NO LIABILITY WHATSOEVER, AND LUMINARY MICRO DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF LUMINARY MICRO'S PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. LUMINARY MICRO'S PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE-SUSTAINING APPLICATIONS.

Luminary Micro may make changes to specifications and product descriptions at any time, without notice. Contact your local Luminary Micro sales office or your distributor to obtain the latest specifications before placing your product order.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Luminary Micro reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Copyright © 2007 Luminary Micro, Inc. All rights reserved. Stellaris, Luminary Micro, and the Luminary Micro logo are registered trademarks of Luminary Micro, Inc. or its subsidiaries in the United States and other countries. ARM and Thumb are registered trademarks and Cortex is a trademark of ARM Limited. Other names and brands may be claimed as the property of others.

Luminary Micro, Inc. 108 Wild Basin, Suite 350 Austin, TX 78746 Main: +1-512-279-8800 Fax: +1-512-279-8879 http://www.luminarymicro.com







www.DataSheet4U.com

November 29, 2007
Preliminary

## **Table of Contents**

Α	bout	This Document	18
		ce	
		This Manual	
		d Documents	
D	ocume	entation Conventions	18
1		Architectural Overview	20
1.	1	Product Features	20
1.	2	Target Applications	25
1.	3	High-Level Block Diagram	26
44.	<b>4</b> m	Functional Overview	27
1.	4.1	ARM Cortex™-M3	27
1.	4.2	Motor Control Peripherals	27
1.	4.3	Analog Peripherals	28
1.	4.4	Serial Communications Peripherals	28
1.	4.5	System Peripherals	30
1.	4.6	Memory Peripherals	31
1.	4.7	Additional Features	31
1.	4.8	Hardware Details	32
2		ARM Cortex-M3 Processor Core	33
2.	1	Block Diagram	34
2.	2	Functional Description	34
2.	2.1	Serial Wire and JTAG Debug	34
2.	2.2	Embedded Trace Macrocell (ETM)	35
2.	2.3	Trace Port Interface Unit (TPIU)	35
2.	2.4	ROM Table	35
2.	2.5	Memory Protection Unit (MPU)	35
2.	2.6	Nested Vectored Interrupt Controller (NVIC)	35
3		Memory Map	39
4		Interrupts	41
5		JTAG Interface	4:
5.		Block Diagram	
5.		Functional Description	
5.	2.1	JTAG Interface Pins	
5.	2.2	JTAG TAP Controller	
	2.3	Shift Registers	
5.	2.4	Operational Considerations	47
5.	3	Initialization and Configuration	50
5.	4	Register Descriptions	50
5.	4.1	Instruction Register (IR)	50
5.	4.2	Data Registers	
6		System Control	54
6.		Functional Description	
6.	1.1	Device Identification	
6.	1.2	Reset Control	

6.1.3	Power Control	
6.1.4	Clock Control	
6.1.5	System Control	59
6.2	Initialization and Configuration	59
6.3	Register Map	60
6.4	Register Descriptions	61
7	Internal Memory	110
7.1	Block Diagram	110
7.2	Functional Description	110
7.2.1	SRAM Memory	110
7.2.2	Flash Memory	
7.3	Flash Memory Initialization and Configuration	112
7.3.1	Flash Programming	112
7.3.2	Nonvolatile Register Programming	113
7.4	Register Map	
7.5	Flash Register Descriptions (Flash Control Offset)	
7.6	Flash Register Descriptions (System Control Offset)	121
8	General-Purpose Input/Outputs (GPIOs)	134
8.1	Functional Description	
8.1.1	Data Control	
8.1.2	Interrupt Control	
8.1.3	Mode Control	
8.1.4	Commit Control	
8.1.5	Pad Control	
8.1.6	Identification	
8.2	Initialization and Configuration	
8.3	Register Map	
8.4	Register Descriptions	
9	General-Purpose Timers	
9.1	Block Diagram	
9.2	Functional Description	
9.2.1	GPTM Reset Conditions	
9.2.2	32-Bit Timer Operating Modes	
9.2.3	16-Bit Timer Operating Modes	
9.3	Initialization and Configuration	
9.3.1	32-Bit One-Shot/Periodic Timer Mode	
9.3.2 9.3.3	32-Bit Real-Time Clock (RTC) Mode	
9.3.4 9.3.4	16-Bit One-Shot/Periodic Timer Mode	
9.3. <del>4</del> 9.3.5	,	
9.3.6	16-Bit Input Edge Timing Mode	
9.3.0	Register Map	
9. <del>4</del> 9.5	Register Descriptions	
<b>10</b> 10.1	Watchdog Timer  Block Diagram	
10.1	Functional Description	
10.2	Initialization and Configuration	
10.0	minumzunon ana oomigaranon	

10.4	Register Map	212
10.5	Register Descriptions	213
11	Universal Asynchronous Receivers/Transmitters (UARTs)	234
11.1	Block Diagram	
11.2	Functional Description	235
11.2.1	Transmit/Receive Logic	235
11.2.2	Baud-Rate Generation	236
11.2.3	Data Transmission	237
11.2.4	Serial IR (SIR)	237
11.2.5	FIFO Operation	238
11.2.6	Interrupts	238
11.2.7	Loopback Operation	
11.2.8	IrDA SIR block	
11.3	Initialization and Configuration	
11.4	Register Map	
11.5	Register Descriptions	241
12	Synchronous Serial Interface (SSI)	275
12.1	Block Diagram	275
12.2	Functional Description	275
12.2.1	Bit Rate Generation	
12.2.2	FIFO Operation	276
12.2.3	Interrupts	276
12.2.4	Frame Formats	
12.3	Initialization and Configuration	
12.4	Register Map	
12.5	Register Descriptions	286
13	Inter-Integrated Circuit (I <sup>2</sup> C) Interface	
13.1	Block Diagram	
13.2	Functional Description	
13.2.1	I <sup>2</sup> C Bus Functional Overview	
	Available Speed Modes	
	Interrupts	
	Loopback Operation	
	Command Sequence Flow Charts	
13.3	Initialization and Configuration	
	I/O Danistan Man	$\alpha \alpha A$
13.4	I <sup>2</sup> C Register Map	
13.5	Register Descriptions (I <sup>2</sup> C Master)	325
		325
13.5 13.6 <b>14</b>	Register Descriptions (I <sup>2</sup> C Master)  Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module	325 338 <b>347</b>
13.5 13.6	Register Descriptions (I <sup>2</sup> C Master)	325 338 <b>347</b>
13.5 13.6 <b>14</b> 14.1 14.2	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features	325 338 <b>347</b> 347 347
13.5 13.6 <b>14</b> 14.1 14.2 14.3	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features Controller Area Network Block Diagram	325 338 <b>347</b> 347 347 348
13.5 13.6 <b>14</b> 14.1 14.2 14.3	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features Controller Area Network Block Diagram Controller Area Network Functional Description	325 338 <b>347</b> 347 347 348 349
13.5 13.6 <b>14</b> 14.1 14.2 14.3 14.4	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features Controller Area Network Block Diagram Controller Area Network Functional Description Initialization	325 338 <b>347</b> 347 348 349 349
13.5 13.6 <b>14</b> 14.1 14.2 14.3 14.4 14.4.1	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features Controller Area Network Block Diagram Controller Area Network Functional Description Initialization Operation	325 338 <b>347</b> 347 348 349 349 350
13.5 13.6 <b>14</b> 14.1 14.2 14.3	Register Descriptions (I <sup>2</sup> C Master) Register Descriptions (I2C Slave)  Controller Area Network (CAN) Module Controller Area Network Overview Controller Area Network Features Controller Area Network Block Diagram Controller Area Network Functional Description Initialization	325 338 <b>347</b> 347 348 349 349 350 350

	Updating a Transmit Message Object	
14.4.6	Accepting Received Message Objects	. 351
14.4.7	Receiving a Data Frame	. 352
14.4.8	Receiving a Remote Frame	. 352
14.4.9	Receive/Transmit Priority	352
14.4.10	Configuring a Receive Message Object	352
14.4.11	Handling of Received Message Objects	. 353
14.4.12	P. Handling of Interrupts	353
14.4.13	Bit Timing Configuration Error Considerations	. 354
14.4.14	Bit Time and Bit Rate	. 354
14.4.15	Calculating the Bit Timing Parameters	356
14.5	Controller Area Network Register Map	. 358
14.6	Register Descriptions	360
15	Analog Comparators	388
15.1	Block Diagram	
15.2	Functional Description	389
15.2.1	Internal Reference Programming	
15.3	Initialization and Configuration	
15.4	Register Map	
15.5	Register Descriptions	393
16	Pulse Width Modulator (PWM)	401
16.1	Block Diagram	
16.2	Functional Description	
16.2.1	PWM Timer	
	PWM Comparators	
	PWM Signal Generator	
	Dead-Band Generator	
	Interrupt Selector	
16.2.6	Synchronization Methods	
16.2.7		
16.2.8	Output Control Block	405
16.3	Initialization and Configuration	
16.4	Register Map	406
16.5	Register Descriptions	407
17	Pin Diagram	436
18	Signal Tables	437
19	Operating Characteristics	449
20	Electrical Characteristics	450
20.1	DC Characteristics	450
20.1.1	Maximum Ratings	450
20.1.2	Recommended DC Operating Conditions	. 450
	On-Chip Low Drop-Out (LDO) Regulator Characteristics	
20.1.4	Power Specifications	451
20.1.5	Flash Memory Characteristics	452
20.2	AC Characteristics	. 453
20.2.1	Load Conditions	453
20.22	Clocks	453

20.2.3	Analog Comparator	454
20.2.4	l <sup>2</sup> C	454
20.2.5	Synchronous Serial Interface (SSI)	455
20.2.6	JTAG and Boundary Scan	456
20.2.7	General-Purpose I/O	458
20.2.8	Reset	458
21	Package Information	461
Α	Serial Flash Loader	463
A.1	Serial Flash Loader	
A.2	Interfaces	463
A.2.1	UART	463
A.2.2	SSI	463
A.3	Packet Handling	464
A.3.1	Packet Format	464
A.3.2	Sending Packets	464
A.3.3	Receiving Packets	464
A.4	Commands	465
A.4.1	COMMAND_PING (0X20)	
A.4.2	COMMAND_GET_STATUS (0x23)	
A.4.3	COMMAND_DOWNLOAD (0x21)	465
A.4.4	COMMAND_SEND_DATA (0x24)	
A.4.5	COMMAND_RUN (0x22)	
A.4.6	COMMAND_RESET (0x25)	466
В	Register Quick Reference	468
С	Ordering and Contact Information	484
C.1	Ordering Information	
C.2	Kits	484
C.3	Company Information	484
C.4	Support Information	485

# **List of Figures**

Figure 1-1.	Stellaris® 2000 Series High-Level Block Diagram	26
Figure 2-1.	CPU Block Diagram	34
Figure 2-2.	TPIU Block Diagram	35
Figure 5-1.	JTAG Module Block Diagram	44
Figure 5-2.	Test Access Port State Machine	47
Figure 5-3.	IDCODE Register Format	52
Figure 5-4.	BYPASS Register Format	53
Figure 5-5.	Boundary Scan Register Format	53
Figure 6-1.	External Circuitry to Extend Reset	55
Figure 7-1.	Flash Block Diagram	110
Figure 8-1.	GPIO Port Block Diagram	135
Figure 8-2.	GPIODATA Write Example	136
Figure 8-3.	GPIODATA Read Example	136
Figure 9-1.	GPTM Module Block Diagram	
Figure 9-2.	16-Bit Input Edge Count Mode Example	180
Figure 9-3.	16-Bit Input Edge Time Mode Example	181
Figure 9-4.	16-Bit PWM Mode Example	182
Figure 10-1.	WDT Module Block Diagram	211
Figure 11-1.	UART Module Block Diagram	235
Figure 11-2.	UART Character Frame	236
Figure 11-3.	IrDA Data Modulation	238
Figure 12-1.	SSI Module Block Diagram	275
Figure 12-2.	TI Synchronous Serial Frame Format (Single Transfer)	278
Figure 12-3.	TI Synchronous Serial Frame Format (Continuous Transfer)	278
Figure 12-4.	Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0	279
Figure 12-5.	Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0	279
Figure 12-6.	Freescale SPI Frame Format with SPO=0 and SPH=1	280
Figure 12-7.	Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0	
Figure 12-8.	Freescale SPI Frame Format (Continuous Transfer) with SPO=1 and SPH=0	
Figure 12-9.	Freescale SPI Frame Format with SPO=1 and SPH=1	282
Figure 12-10.	MICROWIRE Frame Format (Single Frame)	283
Figure 12-11.	MICROWIRE Frame Format (Continuous Transfer)	284
•	MICROWIRE Frame Format, SSIFss Input Setup and Hold Requirements	
•	I <sup>2</sup> C Block Diagram	
Figure 13-2.	I <sup>2</sup> C Bus Configuration	313
Figure 13-3.	START and STOP Conditions	313
Figure 13-4.	Complete Data Transfer with a 7-Bit Address	314
Figure 13-5.	R/S Bit in First Byte	
Figure 13-6.	Data Validity During Bit Transfer on the I <sup>2</sup> C Bus	314
Figure 13-7.	Master Single SEND	317
Figure 13-8.	Master Single RECEIVE	318
Figure 13-9.	Master Burst SEND	319
	Master Burst RECEIVE	
Figure 13-11.	Master Burst RECEIVE after Burst SEND	321
Figure 13-12.	Master Burst SEND after Burst RECEIVE	322

Figure 13-13.	Slave Command Sequence	. 323
Figure 14-1.	CAN Module Block Diagram	. 348
Figure 14-2.	CAN Bit Time	. 355
Figure 15-1.	Analog Comparator Module Block Diagram	. 389
Figure 15-2.	Structure of Comparator Unit	. 390
Figure 15-3.	Comparator Internal Reference Structure	. 391
Figure 16-1.	PWM Module Block Diagram	. 401
Figure 16-2.	PWM Count-Down Mode	. 402
Figure 16-3.	PWM Count-Up/Down Mode	. 403
Figure 16-4.	PWM Generation Example In Count-Up/Down Mode	. 403
Figure 16-5.	PWM Dead-Band Generator	. 404
Figure 17-1.	Pin Connection Diagram	. 436
Figure 20-1.	Load Conditions	. 453
Figure 20-2.	I <sup>2</sup> C Timing	. 455
Figure 20-3.	SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement	. 455
Figure 20-4.	SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer	. 456
Figure 20-5.	SSI Timing for SPI Frame Format (FRF=00), with SPH=1	. 456
Figure 20-6.	JTAG Test Clock Input Timing	. 457
Figure 20-7.	JTAG Test Access Port (TAP) Timing	. 458
Figure 20-8.	JTAG TRST Timing	. 458
Figure 20-9.	External Reset Timing (RST)	. 459
Figure 20-10.	Power-On Reset Timing	. 459
Figure 20-11.	Brown-Out Reset Timing	. 459
Figure 20-12.	Software Reset Timing	. 460
_	Watchdog Reset Timing	
Figure 21-1.	100-Pin LQFP Package	. 461

## **List of Tables**

Table 1.	Documentation Conventions	
Table 3-1.	Memory Map	39
Table 4-1.	Exception Types	41
Table 4-2.	Interrupts	
Table 5-1.	JTAG Port Pins Reset State	45
Table 5-2.	JTAG Instruction Register Commands	50
Table 6-1.	System Control Register Map	60
Table 7-1.	Flash Protection Policy Combinations	112
Table 7-2.	Flash Resident Registers	113
Table 7-3.	Flash Register Map	113
Table 8-1.	GPIO Pad Configuration Examples	137
Table 8-2.	GPIO Interrupt Configuration Example	138
Table 8-3.	GPIO Register Map	139
Table 9-1.	Available CCP Pins	176
Table 9-2.	16-Bit Timer With Prescaler Configurations	179
Table 9-3.	Timers Register Map	185
Table 10-1.	Watchdog Timer Register Map	212
Table 11-1.	UART Register Map	240
Table 12-1.	SSI Register Map	285
Table 13-1.	Examples of I <sup>2</sup> C Master Timer Period versus Speed Mode	315
Table 13-2.	Inter-Integrated Circuit (I <sup>2</sup> C) Interface Register Map	324
Table 13-3.	Write Field Decoding for I2CMCS[3:0] Field (Sheet 1 of 3)	329
Table 14-1.	Transmit Message Object Bit Settings	351
Table 14-2.	Receive Message Object Bit Settings	353
Table 14-3.	CAN Protocol Ranges	355
Table 14-4.	CAN Register Map	358
Table 15-1.	Comparator 0 Operating Modes	390
Table 15-2.	Comparator 1 Operating Modes	390
Table 15-3.	Comparator 2 Operating Modes	391
Table 15-4.	Internal Reference Voltage and ACREFCTL Field Values	391
Table 15-5.	Analog Comparators Register Map	393
Table 16-1.	PWM Register Map	406
Table 18-1.	Signals by Pin Number	437
Table 18-2.	Signals by Signal Name	441
Table 18-3.	Signals by Function, Except for GPIO	444
Table 18-4.	GPIO Pins and Alternate Functions	447
Table 19-1.	Temperature Characteristics	449
Table 19-2.	Thermal Characteristics	449
Table 20-1.	Maximum Ratings	450
Table 20-2.	Recommended DC Operating Conditions	450
Table 20-3.	LDO Regulator Characteristics	451
Table 20-4.	Detailed Power Specifications	452
Table 20-5.	Flash Memory Characteristics	452
Table 20-6.	Phase Locked Loop (PLL) Characteristics	453
Table 20-7.	Clock Characteristics	453
Table 20-8.	Crystal Characteristics	453

Table 20-9.	Analog Comparator Characteristics	454
Table 20-10.	Analog Comparator Voltage Reference Characteristics	454
Table 20-11.	I <sup>2</sup> C Characteristics	454
Table 20-12.	SSI Characteristics	455
Table 20-13.	JTAG Characteristics	456
Table 20-14.	GPIO Characteristics	458
Table 20-15.	Reset Characteristics	458
Table C-1.	Part Ordering Information	484

# **List of Registers**

System Co	ntrol	54
Register 1:	Device Identification 0 (DID0), offset 0x000	62
Register 2:	Brown-Out Reset Control (PBORCTL), offset 0x030	64
Register 3:	LDO Power Control (LDOPCTL), offset 0x034	65
Register 4:	Raw Interrupt Status (RIS), offset 0x050	66
Register 5:	Interrupt Mask Control (IMC), offset 0x054	67
Register 6:	Masked Interrupt Status and Clear (MISC), offset 0x058	68
Register 7:	Reset Cause (RESC), offset 0x05C	69
Register 8:	Run-Mode Clock Configuration (RCC), offset 0x060	70
Register 9:	XTAL to PLL Translation (PLLCFG), offset 0x064	
Register 10:	Run-Mode Clock Configuration 2 (RCC2), offset 0x070	
Register 11:	Deep Sleep Clock Configuration (DSLPCLKCFG), offset 0x144	77
Register 12:	Device Identification 1 (DID1), offset 0x004	
Register 13:	Device Capabilities 0 (DC0), offset 0x008	80
Register 14:	Device Capabilities 1 (DC1), offset 0x010	
Register 15:	Device Capabilities 2 (DC2), offset 0x014	83
Register 16:	Device Capabilities 3 (DC3), offset 0x018	
Register 17:	Device Capabilities 4 (DC4), offset 0x01C	87
Register 18:	Run Mode Clock Gating Control Register 0 (RCGC0), offset 0x100	
Register 19:	Sleep Mode Clock Gating Control Register 0 (SCGC0), offset 0x110	
Register 20:	Deep Sleep Mode Clock Gating Control Register 0 (DCGC0), offset 0x120	92
Register 21:	Run Mode Clock Gating Control Register 1 (RCGC1), offset 0x104	
Register 22:	Sleep Mode Clock Gating Control Register 1 (SCGC1), offset 0x114	
Register 23:	Deep Sleep Mode Clock Gating Control Register 1 (DCGC1), offset 0x124	
Register 24:	Run Mode Clock Gating Control Register 2 (RCGC2), offset 0x108	100
Register 25:	Sleep Mode Clock Gating Control Register 2 (SCGC2), offset 0x118	
Register 26:	Deep Sleep Mode Clock Gating Control Register 2 (DCGC2), offset 0x128	
Register 27:	Software Reset Control 0 (SRCR0), offset 0x040	
Register 28:	Software Reset Control 1 (SRCR1), offset 0x044	
Register 29:	Software Reset Control 2 (SRCR2), offset 0x048	109
Internal Me	mory	110
Register 1:	Flash Memory Address (FMA), offset 0x000	115
Register 2:	Flash Memory Data (FMD), offset 0x004	116
Register 3:	Flash Memory Control (FMC), offset 0x008	
Register 4:	Flash Controller Raw Interrupt Status (FCRIS), offset 0x00C	119
Register 5:	Flash Controller Interrupt Mask (FCIM), offset 0x010	
Register 6:	Flash Controller Masked Interrupt Status and Clear (FCMISC), offset 0x014	121
Register 7:	USec Reload (USECRL), offset 0x140	
Register 8:	Flash Memory Protection Read Enable 0 (FMPRE0), offset 0x130 and 0x200	123
Register 9:	Flash Memory Protection Program Enable 0 (FMPPE0), offset 0x134 and 0x400	
Register 10:	User Debug (USER_DBG), offset 0x1D0	
Register 11:	User Register 0 (USER_REG0), offset 0x1E0	
Register 12:	User Register 1 (USER_REG1), offset 0x1E4	
Register 13:	Flash Memory Protection Read Enable 1 (FMPRE1), offset 0x204	
Register 14:	Flash Memory Protection Read Enable 2 (FMPRE2), offset 0x208	129

Register 15:	Flash Memory Protection Read Enable 3 (FMPRE3), offset 0x20C	130
Register 16:	Flash Memory Protection Program Enable 1 (FMPPE1), offset 0x404	
Register 17:	Flash Memory Protection Program Enable 2 (FMPPE2), offset 0x408	132
Register 18:	Flash Memory Protection Program Enable 3 (FMPPE3), offset 0x40C	133
General-Pu	rpose Input/Outputs (GPIOs)	134
Register 1:	GPIO Data (GPIODATA), offset 0x000	
Register 2:	GPIO Direction (GPIODIR), offset 0x400	142
Register 3:	GPIO Interrupt Sense (GPIOIS), offset 0x404	143
Register 4:	GPIO Interrupt Both Edges (GPIOIBE), offset 0x408	144
Register 5:	GPIO Interrupt Event (GPIOIEV), offset 0x40C	145
Register 6:	GPIO Interrupt Mask (GPIOIM), offset 0x410	146
Register 7:	GPIO Raw Interrupt Status (GPIORIS), offset 0x414	147
Register 8:	GPIO Masked Interrupt Status (GPIOMIS), offset 0x418	148
Register 9:	GPIO Interrupt Clear (GPIOICR), offset 0x41C	149
Register 10:	GPIO Alternate Function Select (GPIOAFSEL), offset 0x420	150
Register 11:	GPIO 2-mA Drive Select (GPIODR2R), offset 0x500	152
Register 12:	GPIO 4-mA Drive Select (GPIODR4R), offset 0x504	153
Register 13:	GPIO 8-mA Drive Select (GPIODR8R), offset 0x508	154
Register 14:	GPIO Open Drain Select (GPIOODR), offset 0x50C	155
Register 15:	GPIO Pull-Up Select (GPIOPUR), offset 0x510	156
Register 16:	GPIO Pull-Down Select (GPIOPDR), offset 0x514	157
Register 17:	GPIO Slew Rate Control Select (GPIOSLR), offset 0x518	158
Register 18:	GPIO Digital Enable (GPIODEN), offset 0x51C	
Register 19:	GPIO Lock (GPIOLOCK), offset 0x520	160
Register 20:	GPIO Commit (GPIOCR), offset 0x524	
Register 21:	GPIO Peripheral Identification 4 (GPIOPeriphID4), offset 0xFD0	163
Register 22:	GPIO Peripheral Identification 5 (GPIOPeriphID5), offset 0xFD4	164
Register 23:	GPIO Peripheral Identification 6 (GPIOPeriphID6), offset 0xFD8	165
Register 24:	GPIO Peripheral Identification 7 (GPIOPeriphID7), offset 0xFDC	166
Register 25:	GPIO Peripheral Identification 0 (GPIOPeriphID0), offset 0xFE0	167
Register 26:	GPIO Peripheral Identification 1 (GPIOPeriphID1), offset 0xFE4	
Register 27:	GPIO Peripheral Identification 2 (GPIOPeriphID2), offset 0xFE8	169
Register 28:	GPIO Peripheral Identification 3 (GPIOPeriphID3), offset 0xFEC	170
Register 29:	GPIO PrimeCell Identification 0 (GPIOPCellID0), offset 0xFF0	171
Register 30:	GPIO PrimeCell Identification 1 (GPIOPCelIID1), offset 0xFF4	
Register 31:	GPIO PrimeCell Identification 2 (GPIOPCelIID2), offset 0xFF8	173
Register 32:	GPIO PrimeCell Identification 3 (GPIOPCellID3), offset 0xFFC	
General-Pu	rpose Timers	175
Register 1:	GPTM Configuration (GPTMCFG), offset 0x000	
Register 2:	GPTM TimerA Mode (GPTMTAMR), offset 0x004	
Register 3:	GPTM TimerB Mode (GPTMTBMR), offset 0x008	
Register 4:	GPTM Control (GPTMCTL), offset 0x00C	
Register 5:	GPTM Interrupt Mask (GPTMIMR), offset 0x018	
Register 6:	GPTM Raw Interrupt Status (GPTMRIS), offset 0x01C	
Register 7:	GPTM Masked Interrupt Status (GPTMMIS), offset 0x020	
Register 8:	GPTM Interrupt Clear (GPTMICR), offset 0x024	
Register 9:	GPTM TimerA Interval Load (GPTMTAILR), offset 0x028	
Register 10:	GPTM TimerB Interval Load (GPTMTBILR), offset 0x02C	
-,-,	, , , , , ,	

Register 11:	GPTM TimerA Match (GPTMTAMATCHR), offset 0x030	203
Register 12:	GPTM TimerB Match (GPTMTBMATCHR), offset 0x034	204
Register 13:	GPTM TimerA Prescale (GPTMTAPR), offset 0x038	205
Register 14:	GPTM TimerB Prescale (GPTMTBPR), offset 0x03C	206
Register 15:	GPTM TimerA Prescale Match (GPTMTAPMR), offset 0x040	207
Register 16:	GPTM TimerB Prescale Match (GPTMTBPMR), offset 0x044	208
Register 17:	GPTM TimerA (GPTMTAR), offset 0x048	209
Register 18:	GPTM TimerB (GPTMTBR), offset 0x04C	210
Watchdog <sup>-</sup>	Timer	211
Register 1:	Watchdog Load (WDTLOAD), offset 0x000	
Register 2:	Watchdog Value (WDTVALUE), offset 0x004	
Register 3:	Watchdog Control (WDTCTL), offset 0x008	
Register 4:	Watchdog Interrupt Clear (WDTICR), offset 0x00C	
Register 5:	Watchdog Raw Interrupt Status (WDTRIS), offset 0x010	
Register 6:	Watchdog Masked Interrupt Status (WDTMIS), offset 0x014	
Register 7:	Watchdog Test (WDTTEST), offset 0x418	
Register 8:	Watchdog Lock (WDTLOCK), offset 0xC00	
Register 9:	Watchdog Peripheral Identification 4 (WDTPeriphID4), offset 0xFD0	
Register 10:	Watchdog Peripheral Identification 5 (WDTPeriphID5), offset 0xFD4	
Register 11:	Watchdog Peripheral Identification 6 (WDTPeriphID6), offset 0xFD8	
Register 12:	Watchdog Peripheral Identification 7 (WDTPeriphID7), offset 0xFDC	
Register 13:	Watchdog Peripheral Identification 0 (WDTPeriphID0), offset 0xFE0	
Register 14:	Watchdog Peripheral Identification 1 (WDTPeriphID1), offset 0xFE4	
Register 15:	Watchdog Peripheral Identification 2 (WDTPeriphID2), offset 0xFE8	
Register 16:	Watchdog Peripheral Identification 3 (WDTPeriphID3), offset 0xFEC	229
Register 17:	Watchdog PrimeCell Identification 0 (WDTPCellID0), offset 0xFF0	230
Register 18:	Watchdog PrimeCell Identification 1 (WDTPCellID1), offset 0xFF4	231
Register 19:	Watchdog PrimeCell Identification 2 (WDTPCellID2), offset 0xFF8	232
Register 20:	Watchdog PrimeCell Identification 3 (WDTPCellID3 ), offset 0xFFC	233
Universal A	Asynchronous Receivers/Transmitters (UARTs)	234
Register 1:	UART Data (UARTDR), offset 0x000	
Register 2:	UART Receive Status/Error Clear (UARTRSR/UARTECR), offset 0x004	
Register 3:	UART Flag (UARTFR), offset 0x018	
Register 4:	UART IrDA Low-Power Register (UARTILPR), offset 0x020	
Register 5:	UART Integer Baud-Rate Divisor (UARTIBRD), offset 0x024	
Register 6:	UART Fractional Baud-Rate Divisor (UARTFBRD), offset 0x028	
Register 7:	UART Line Control (UARTLCRH), offset 0x02C	
Register 8:	UART Control (UARTCTL), offset 0x030	
Register 9:	UART Interrupt FIFO Level Select (UARTIFLS), offset 0x034	
Register 10:	UART Interrupt Mask (UARTIM), offset 0x038	
Register 11:	UART Raw Interrupt Status (UARTRIS), offset 0x03C	
Register 12:	UART Masked Interrupt Status (UARTMIS), offset 0x040	
Register 13:	UART Interrupt Clear (UARTICR), offset 0x044	
Register 14:	UART Peripheral Identification 4 (UARTPeriphID4), offset 0xFD0	
Register 15:	UART Peripheral Identification 5 (UARTPeriphID5), offset 0xFD4	
Register 16:	UART Peripheral Identification 6 (UARTPeriphID6), offset 0xFD8	
Register 17:	UART Peripheral Identification 7 (UARTPeriphID7), offset 0xFDC	
Register 18:	UART Peripheral Identification 0 (UARTPeriphID0), offset 0xFE0	

Register 19:	UART Peripheral Identification 1 (UARTPeriphID1), offset 0xFE4	268
Register 20:	UART Peripheral Identification 2 (UARTPeriphID2), offset 0xFE8	269
Register 21:	UART Peripheral Identification 3 (UARTPeriphID3), offset 0xFEC	270
Register 22:	UART PrimeCell Identification 0 (UARTPCellID0), offset 0xFF0	
Register 23:	UART PrimeCell Identification 1 (UARTPCellID1), offset 0xFF4	
Register 24:	UART PrimeCell Identification 2 (UARTPCellID2), offset 0xFF8	
Register 25:	UART PrimeCell Identification 3 (UARTPCellID3), offset 0xFFC	274
Synchrono	us Serial Interface (SSI)	275
Register 1:	SSI Control 0 (SSICR0), offset 0x000	287
Register 2:	SSI Control 1 (SSICR1), offset 0x004	
Register 3:	SSI Data (SSIDR), offset 0x008	
Register 4:	SSI Status (SSISR), offset 0x00C	
Register 5:	SSI Clock Prescale (SSICPSR), offset 0x010	
Register 6:	SSI Interrupt Mask (SSIIM), offset 0x014	
Register 7:	SSI Raw Interrupt Status (SSIRIS), offset 0x018	
Register 8:	SSI Masked Interrupt Status (SSIMIS), offset 0x01C	
Register 9:	SSI Interrupt Clear (SSIICR), offset 0x020	
Register 10:	SSI Peripheral Identification 4 (SSIPeriphID4), offset 0xFD0	
Register 11:	SSI Peripheral Identification 5 (SSIPeriphID5), offset 0xFD4	
Register 12:	SSI Peripheral Identification 6 (SSIPeriphID6), offset 0xFD8	
Register 13:	SSI Peripheral Identification 7 (SSIPeriphID7), offset 0xFDC	
Register 14:	SSI Peripheral Identification 0 (SSIPeriphID0), offset 0xFE0	
Register 15:	SSI Peripheral Identification 1 (SSIPeriphID1), offset 0xFE4	
Register 16:	SSI Peripheral Identification 2 (SSIPeriphID2), offset 0xFE8	
Register 17:	SSI Peripheral Identification 3 (SSIPeriphID3), offset 0xFEC	
Register 18:	SSI PrimeCell Identification 0 (SSIPCellID0), offset 0xFF0	
Register 19:	SSI PrimeCell Identification 1 (SSIPCellID1), offset 0xFF4	
Register 20: Register 21:	SSI PrimeCell Identification 3 (SSIPCellID3), offset 0xFFC	
ŭ		
•	ated Circuit (I <sup>2</sup> C) Interface	
Register 1:	I <sup>2</sup> C Master Slave Address (I2CMSA), offset 0x000	
Register 2:	I <sup>2</sup> C Master Control/Status (I2CMCS), offset 0x004	
Register 3:	I <sup>2</sup> C Master Data (I2CMDR), offset 0x008	
Register 4:	I <sup>2</sup> C Master Timer Period (I2CMTPR), offset 0x00C	
Register 5:	I <sup>2</sup> C Master Interrupt Mask (I2CMIMR), offset 0x010	
Register 6:	I <sup>2</sup> C Master Raw Interrupt Status (I2CMRIS), offset 0x014	
Register 7:	I <sup>2</sup> C Master Masked Interrupt Status (I2CMMIS), offset 0x018	
Register 8:	I <sup>2</sup> C Master Interrupt Clear (I2CMICR), offset 0x01C	
Register 9:	I <sup>2</sup> C Master Configuration (I2CMCR), offset 0x020	337
Register 10:	I <sup>2</sup> C Slave Own Address (I2CSOAR), offset 0x000	
Register 11:	I <sup>2</sup> C Slave Control/Status (I2CSCSR), offset 0x004	340
Register 12:	I <sup>2</sup> C Slave Data (I2CSDR), offset 0x008	342
Register 13:	I <sup>2</sup> C Slave Interrupt Mask (I2CSIMR), offset 0x00C	
Register 14:	I <sup>2</sup> C Slave Raw Interrupt Status (I2CSRIS), offset 0x010	344
Register 15:	I <sup>2</sup> C Slave Masked Interrupt Status (I2CSMIS), offset 0x014	345
Register 16:	I <sup>2</sup> C Slave Interrupt Clear (I2CSICR), offset 0x018	346

Controller A	Area Network (CAN) Module	347
Register 1:	CAN Control (CANCTL), offset 0x000	361
Register 2:	CAN Status (CANSTS), offset 0x004	363
Register 3:	CAN Error Counter (CANERR), offset 0x008	366
Register 4:	CAN Bit Timing (CANBIT), offset 0x00C	367
Register 5:	CAN Interrupt (CANINT), offset 0x010	
Register 6:	CAN Test (CANTST), offset 0x014	370
Register 7:	CAN Baud Rate Prescalar Extension (CANBRPE), offset 0x018	372
Register 8:	CAN IF1 Command Request (CANIF1CRQ), offset 0x020	373
Register 9:	CAN IF2 Command Request (CANIF2CRQ), offset 0x080	373
Register 10:	CAN IF1 Command Mask (CANIF1CMSK), offset 0x024	374
Register 11:	CAN IF2 Command Mask (CANIF2CMSK), offset 0x084	374
Register 12:	CAN IF1 Mask 1 (CANIF1MSK1), offset 0x028	
Register 13:	CAN IF2 Mask 1 (CANIF2MSK1), offset 0x088	377
Register 14:	CAN IF1 Mask 2 (CANIF1MSK2), offset 0x02C	378
Register 15:	CAN IF2 Mask 2 (CANIF2MSK2), offset 0x08C	
Register 16:	CAN IF1 Arbitration 1 (CANIF1ARB1), offset 0x030	379
Register 17:	CAN IF2 Arbitration 1 (CANIF2ARB1), offset 0x090	
Register 18:	CAN IF1 Arbitration 2 (CANIF1ARB2), offset 0x034	
Register 19:	CAN IF2 Arbitration 2 (CANIF2ARB2), offset 0x094	
Register 20:	CAN IF1 Message Control (CANIF1MCTL), offset 0x038	
Register 21:	CAN IF2 Message Control (CANIF2MCTL), offset 0x098	
Register 22:	CAN IF1 Data A1 (CANIF1DA1), offset 0x03C	
Register 23:	CAN IF1 Data A2 (CANIF1DA2), offset 0x040	
Register 24:	CAN IF1 Data B1 (CANIF1DB1), offset 0x044	
Register 25:	CAN IF1 Data B2 (CANIF1DB2), offset 0x048	
Register 26:	CAN IF2 Data A1 (CANIF2DA1), offset 0x09C	
Register 27:	CAN IF2 Data A2 (CANIF2DA2), offset 0x0A0	
Register 28:	CAN IF2 Data B1 (CANIF2DB1), offset 0x0A4	
Register 29:	CAN IF2 Data B2 (CANIF2DB2), offset 0x0A8	
Register 30:	CAN Transmission Request 1 (CANTXRQ1), offset 0x100	
Register 31:	CAN Transmission Request 2 (CANTXRQ2), offset 0x104	
Register 32:	CAN New Data 1 (CANNWDA1), offset 0x120	
Register 33:	CAN New Data 2 (CANNWDA2), offset 0x124	
Register 34:	CAN Message 1 Interrupt Pending (CANMSG1INT), offset 0x140	
Register 35:	CAN Message 2 Interrupt Pending (CANMSG2INT), offset 0x144	
Register 36:	CAN Message 1 Valid (CANMSG1VAL), offset 0x160	
Register 37:	CAN Message 2 Valid (CANMSG2VAL), offset 0x164	
•	nparators	
Register 1:	Analog Comparator Masked Interrupt Status (ACMIS), offset 0x00	
Register 2:	Analog Comparator Raw Interrupt Status (ACRIS), offset 0x04	
Register 3:	Analog Comparator Interrupt Enable (ACINTEN), offset 0x04	
Register 4:	Analog Comparator Reference Voltage Control (ACREFCTL), offset 0x10	
Register 5:	Analog Comparator Status 0 (ACSTAT0), offset 0x20	
Register 6:	Analog Comparator Status 1 (ACSTATO), offset 0x40	
Register 7:	Analog Comparator Status 2 (ACSTAT1), offset 0x40	
•	Analog Comparator Control 0 (ACCTL0), offset 0x24	
Register 8: Register 9:	Analog Comparator Control 1 (ACCTL1), offset 0x44	
i vegistei 9.	Analog Comparator Control 1 (ACCTLT), UNSCLUX44	398

Register 10:	Analog Comparator Control 2 (ACCTL2), offset 0x64	399
Pulse Widt	h Modulator (PWM)	401
Register 1:	PWM Master Control (PWMCTL), offset 0x000	
Register 2:	PWM Time Base Sync (PWMSYNC), offset 0x004	409
Register 3:	PWM Output Enable (PWMENABLE), offset 0x008	
Register 4:	PWM Output Inversion (PWMINVERT), offset 0x00C	411
Register 5:	PWM Output Fault (PWMFAULT), offset 0x010	412
Register 6:	PWM Interrupt Enable (PWMINTEN), offset 0x014	413
Register 7:	PWM Raw Interrupt Status (PWMRIS), offset 0x018	414
Register 8:	PWM Interrupt Status and Clear (PWMISC), offset 0x01C	415
Register 9:	PWM Status (PWMSTATUS), offset 0x020	416
Register 10:	PWM0 Control (PWM0CTL), offset 0x040	417
Register 11:	PWM0 Interrupt Enable (PWM0INTEN), offset 0x044	419
Register 12:	PWM0 Raw Interrupt Status (PWM0RIS), offset 0x048	421
Register 13:	PWM0 Interrupt Status and Clear (PWM0ISC), offset 0x04C	422
Register 14:	PWM0 Load (PWM0LOAD), offset 0x050	423
Register 15:	PWM0 Counter (PWM0COUNT), offset 0x054	424
Register 16:	PWM0 Compare A (PWM0CMPA), offset 0x058	425
Register 17:	PWM0 Compare B (PWM0CMPB), offset 0x05C	426
Register 18:	PWM0 Generator A Control (PWM0GENA), offset 0x060	427
Register 19:	PWM0 Generator B Control (PWM0GENB), offset 0x064	430
Register 20:	PWM0 Dead-Band Control (PWM0DBCTL), offset 0x068	433
Register 21:	PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE), offset 0x06C	434
Register 22:	PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL), offset 0x070	435

## **About This Document**

This data sheet provides reference information for the LM3S2110 microcontroller, describing the functional blocks of the system-on-chip (SoC) device designed around the ARM® Cortex™-M3 core.

#### **Audience**

This manual is intended for system software developers, hardware designers, and application developers.

#### **About This Manual**

w.DataSheet4U.com

This document is organized into sections that correspond to each major feature.

#### **Related Documents**

The following documents are referenced by the data sheet, and available on the documentation CD or from the Luminary Micro web site at www.luminarymicro.com:

- ARM® Cortex™-M3 Technical Reference Manual
- ARM® CoreSight Technical Reference Manual
- ARM® v7-M Architecture Application Level Reference Manual

The following related documents are also referenced:

IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture

This documentation list was current as of publication date. Please check the Luminary Micro web site for additional documentation, including application notes and white papers.

#### **Documentation Conventions**

This document uses the conventions shown in Table 1 on page 18.

**Table 1. Documentation Conventions** 

Notation	Meaning	
General Register Notation		
REGISTER	APB registers are indicated in uppercase bold. For example, <b>PBORCTL</b> is the Power-On and Brown-Out Reset Control register. If a register name contains a lowercase n, it represents more than one register. For example, <b>SRCRn</b> represents any (or all) of the three Software Reset Control registers: <b>SRCR0</b> , <b>SRCR1</b> , and <b>SRCR2</b> .	
bit	A single bit in a register.	
bit field	Two or more consecutive and related bits.	
offset 0xnnn	A hexadecimal increment to a register's address, relative to that module's base address as specified in "Memory Map" on page 39.	
Register N	Registers are numbered consecutively throughout the document to aid in referencing them. The register number has no meaning to software.	

Notation	Meaning
reserved	Register bits marked <i>reserved</i> are reserved for future use. In most cases, reserved bits are set to 0; however, user software should not rely on the value of a reserved bit. To provide software compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
yy:xx	The range of register bits inclusive from xx to yy. For example, 31:15 means bits 15 through 31 in that register.
Register Bit/Field Types	This value in the register bit diagram indicates whether software running on the controller can change the value of the bit field.
RC	Software can read this field. The bit or field is cleared by hardware after reading the bit/field.
RO	Software can read this field. Always write the chip reset value.
R/W	Software can read or write this field.
R/W1C	Software can read or write this field. A write of a 0 to a W1C bit does not affect the bit value in the register. A write of a 1 clears the value of the bit in the register; the remaining bits remain unchanged.
	This register type is primarily used for clearing interrupt status bits where the read operation provides the interrupt status and the write of the read value clears only the interrupts being reported at the time the register was read.
W1C	Software can write this field. A write of a 0 to a W1C bit does not affect the bit value in the register. A write of a 1 clears the value of the bit in the register; the remaining bits remain unchanged. A read of the register returns no meaningful data.
	This register is typically used to clear the corresponding bit in an interrupt register.
WO	Only a write by software is valid; a read of the register returns no meaningful data.
Register Bit/Field Reset Value	This value in the register bit diagram shows the bit/field value after any reset, unless noted.
0	Bit cleared to 0 on chip reset.
1	Bit set to 1 on chip reset.
-	Nondeterministic.
Pin/Signal Notation	
[]	Pin alternate function; a pin defaults to the signal without the brackets.
pin	Refers to the physical connection on the package.
signal	Refers to the electrical signal encoding of a pin.
assert a signal	Change the value of the signal from the logically False state to the logically True state. For active High signals, the asserted signal value is 1 (High); for active Low signals, the asserted signal value is 0 (Low). The active polarity (High or Low) is defined by the signal name (see SIGNAL and SIGNAL below).
deassert a signal	Change the value of the signal from the logically True state to the logically False state.
SIGNAL	Signal names are in uppercase and in the Courier font. An overbar on a signal name indicates that it is active Low. To assert SIGNAL is to drive it Low; to deassert SIGNAL is to drive it High.
SIGNAL	Signal names are in uppercase and in the Courier font. An active High signal has no overbar. To assert SIGNAL is to drive it High; to deassert SIGNAL is to drive it Low.
Numbers	
X	An uppercase X indicates any of several values is allowed, where X can be any legal pattern. For example, a binary value of 0X00 can be either 0100 or 0000, a hex value of 0xX is 0x0 or 0x1, and so on.
0x	Hexadecimal numbers have a prefix of 0x. For example, 0x00FF is the hexadecimal number FF.
	All other numbers within register tables are assumed to be binary. Within conceptual information, binary numbers are indicated with a b suffix, for example, 1011b, and decimal numbers are written without a prefix or suffix.

MAN Data Sheet III con

## 1 Architectural Overview

The Luminary Micro Stellaris<sup>®</sup> family of microcontrollers—the first ARM® Cortex<sup>™</sup>-M3 based controllers—brings high-performance 32-bit computing to cost-sensitive embedded microcontroller applications. These pioneering parts deliver customers 32-bit performance at a cost equivalent to legacy 8- and 16-bit devices, all in a package with a small footprint.

The Stellaris<sup>®</sup> family offers efficient performance and extensive integration, favorably positioning the device into cost-conscious applications requiring significant control-processing and connectivity capabilities. The Stellaris<sup>®</sup> LM3S1000 series extends the Stellaris<sup>®</sup> family with larger on-chip memories, enhanced power management, and expanded I/O and control capabilities. The Stellaris<sup>®</sup> LM3S2000 series, designed for Controller Area Network (CAN) applications, extends the Stellaris family with Bosch CAN networking technology, the golden standard in short-haul industrial networks. The Stellaris<sup>®</sup> LM3S2000 series also marks the first integration of CAN capabilities with the revolutionary Cortex-M3 core. The Stellaris<sup>®</sup> LM3S6000 series combines both a 10/100 Ethernet Media Access Control (MAC) and Physical (PHY) layer, marking the first time that integrated connectivity is available with an ARM Cortex-M3 MCU and the only integrated 10/100 Ethernet MAC and PHY available in an ARM architecture MCU. The Stellaris<sup>®</sup> LM3S8000 series combines Bosch Controller Area Network technology with both a 10/100 Ethernet Media Access Control (MAC) and Physical (PHY) layer.

The LM3S2110 microcontroller is targeted for industrial applications, including remote monitoring, electronic point-of-sale machines, test and measurement equipment, network appliances and switches, factory automation, HVAC and building control, gaming equipment, motion control, medical instrumentation, and fire and security.

In addition, the LM3S2110 microcontroller offers the advantages of ARM's widely available development tools, System-on-Chip (SoC) infrastructure IP applications, and a large user community. Additionally, the microcontroller uses ARM's Thumb®-compatible Thumb-2 instruction set to reduce memory requirements and, thereby, cost. Finally, the LM3S2110 microcontroller is code-compatible to all members of the extensive Stellaris® family; providing flexibility to fit our customers' precise needs.

Luminary Micro offers a complete solution to get to market quickly, with evaluation and development boards, white papers and application notes, an easy-to-use peripheral driver library, and a strong support, sales, and distributor network.

#### 1.1 Product Features

The LM3S2110 microcontroller includes the following product features:

- 32-Bit RISC Performance
  - 32-bit ARM® Cortex™-M3 v7M architecture optimized for small-footprint embedded applications
  - System timer (SysTick), providing a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism
  - Thumb®-compatible Thumb-2-only instruction set processor core for high code density
  - 25-MHz operation

Hardware-division and single-cycle-multiplication

- Integrated Nested Vectored Interrupt Controller (NVIC) providing deterministic interrupt handling
- 26 interrupts with eight priority levels
- Memory protection unit (MPU), providing a privileged mode for protected operating system functionality
- Unaligned data access, enabling data to be efficiently packed into memory
- Atomic bit manipulation (bit-banding), delivering maximum memory utilization and streamlined peripheral control

#### Internal Memory

- 64 KB single-cycle flash
  - User-managed flash block protection on a 2-KB block basis
  - User-managed flash data programming
  - User-defined and managed flash-protection block
- 16 KB single-cycle SRAM

#### General-Purpose Timers

- Three General-Purpose Timer Modules (GPTM), each of which provides two 16-bit timers.
   Each GPTM can be configured to operate independently:
  - As a single 32-bit timer
  - As one 32-bit Real-Time Clock (RTC) to event capture
  - For Pulse Width Modulation (PWM)
- 32-bit Timer modes
  - Programmable one-shot timer
  - Programmable periodic timer
  - Real-Time Clock when using an external 32.768-KHz clock as the input
  - User-enabled stalling in periodic and one-shot mode when the controller asserts the CPU Halt flag during debug
- 16-bit Timer modes
  - · General-purpose timer function with an 8-bit prescaler
  - Programmable one-shot timer
  - Programmable periodic timer
  - User-enabled stalling when the controller asserts CPU Halt flag during debug

November 29, 2007 21

- 16-bit Input Capture modes
  - Input edge count capture
  - Input edge time capture
- 16-bit PWM mode
  - Simple PWM mode with software-programmable output inversion of the PWM signal
- ARM FiRM-compliant Watchdog Timer
  - 32-bit down counter with a programmable load register
  - Separate watchdog clock with an enable
  - Programmable interrupt generation logic with interrupt masking
  - Lock register protection from runaway software
  - Reset generation logic with an enable/disable
  - User-enabled stalling when the controller asserts the CPU Halt flag during debug
- Controller Area Network (CAN)
  - Supports CAN protocol version 2.0 part A/B
  - Bit rates up to 1Mb/s
  - 32 message objects, each with its own identifier mask
  - Maskable interrupt
  - Disable automatic retransmission mode for TTCAN
  - Programmable loop-back mode for self-test operation
- Synchronous Serial Interface (SSI)
  - Master or slave operation
  - Programmable clock bit rate and prescale
  - Separate transmit and receive FIFOs, 16 bits wide, 8 locations deep
  - Programmable interface operation for Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces
  - Programmable data frame size from 4 to 16 bits
  - Internal loopback test mode for diagnostic/debug testing
- UART
  - Fully programmable 16C550-type UART with IrDA support

- Separate 16x8 transmit (TX) and 16x12 receive (RX) FIFOs to reduce CPU interrupt service loading
- Programmable baud-rate generator with fractional divider
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of 1/8, 1/4, 1/2, 3/4, and 7/8
- Standard asynchronous communication bits for start, stop, and parity
- False-start-bit detection
- Line-break generation and detection

#### Analog Comparators

- Three independent integrated analog comparators
- Configurable for output to: drive an output pin or generate an interrupt
- Compare external pin input to external pin input or to internal programmable voltage reference

#### I<sup>2</sup>C

- Master and slave receive and transmit operation with transmission speed up to 100 Kbps in Standard mode and 400 Kbps in Fast mode
- Interrupt generation
- Master with arbitration and clock synchronization, multimaster support, and 7-bit addressing mode

#### PWM

- One PWM generator blocks, each with one 16-bit counter, two comparators, a PWM generator, and a dead-band generator
- One 16-bit counter
  - · Runs in Down or Up/Down mode
  - Output frequency controlled by a 16-bit load value
  - · Load value updates can be synchronized
  - · Produces output signals at zero and load value
- Two PWM comparators
  - · Comparator value updates can be synchronized
  - · Produces output signals on match
- PWM generator

- Output PWM signal is constructed based on actions taken as a result of the counter and PWM comparator output signals
- · Produces two independent PWM signals
- Dead-band generator
  - Produces two PWM signals with programmable dead-band delays suitable for driving a half-H bridge
  - · Can be bypassed, leaving input PWM signals unmodified
- Flexible output control block with PWM output enable of each PWM signal
  - PWM output enable of each PWM signal
  - Optional output inversion of each PWM signal (polarity control)
  - Optional fault handling for each PWM signal
  - Synchronization of timers in the PWM generator blocks
  - Synchronization of timer/comparator updates across the PWM generator blocks
  - Interrupt status summary of the PWM generator blocks

#### GPIOs

- 11-40 GPIOs, depending on configuration
- 5-V-tolerant input/outputs
- Programmable interrupt generation as either edge-triggered or level-sensitive
- Bit masking in both read and write operations through address lines
- Programmable control for GPIO pad configuration:
  - Weak pull-up or pull-down resistors
  - 2-mA, 4-mA, and 8-mA pad drive
  - Slew rate control for the 8-mA drive
  - Open drain enables
  - Digital input enables

#### Power

- On-chip Low Drop-Out (LDO) voltage regulator, with programmable output user-adjustable from 2.25 V to 2.75 V
- Low-power options on controller: Sleep and Deep-sleep modes
- Low-power options for peripherals: software controls shutdown of individual peripherals

- User-enabled LDO unregulated voltage detection and automatic reset
- 3.3-V supply brown-out detection and reporting via interrupt or reset
- Flexible Reset Sources
  - Power-on reset (POR)
  - Reset pin assertion
  - Brown-out (BOR) detector alerts to system power drops
  - Software reset
  - Watchdog timer reset
  - Internal low drop-out (LDO) regulator output goes unregulated
- Additional Features
  - Six reset sources
  - Programmable clock source control
  - Clock gating to individual peripherals for power savings
  - IEEE 1149.1-1990 compliant Test Access Port (TAP) controller
  - Debug access via JTAG and Serial Wire interfaces
  - Full JTAG boundary scan
- Industrial-range 100-pin RoHS-compliant LQFP package

## 1.2 Target Applications

- Remote monitoring
- Electronic point-of-sale (POS) machines
- Test and measurement equipment
- Network appliances and switches
- Factory automation
- HVAC and building control
- Gaming equipment
- Motion control
- Medical instrumentation
- Fire and security
- Power and energy

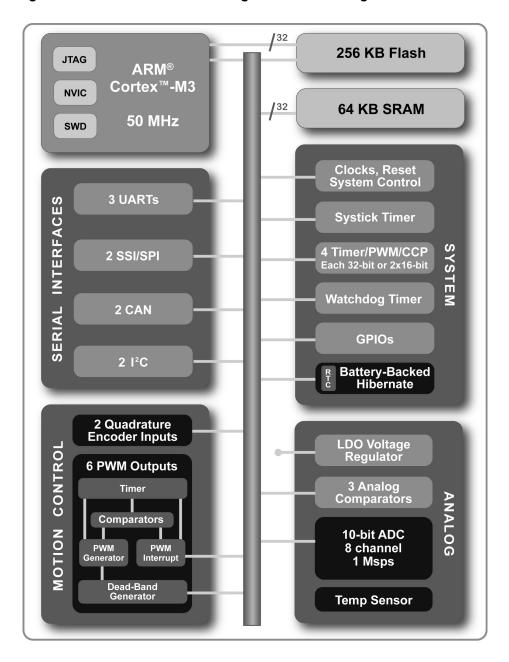
November 29, 2007 25

#### Transportation

## 1.3 High-Level Block Diagram

Figure 1-1 on page 26 represents the full set of features in the Stellaris<sup>®</sup> 2000 series of devices; not all features may be available on the LM3S2110 microcontroller.

Figure 1-1. Stellaris® 2000 Series High-Level Block Diagram



#### 1.4 Functional Overview

The following sections provide an overview of the features of the LM3S2110 microcontroller. The page number in parenthesis indicates where that feature is discussed in detail. Ordering and support information can be found in "Ordering and Contact Information" on page 484.

#### 1.4.1 ARM Cortex™-M3

#### 1.4.1.1 Processor Core (see page 33)

All members of the Stellaris<sup>®</sup> product family, including the LM3S2110 microcontroller, are designed around an ARM Cortex<sup>™</sup>-M3 processor core. The ARM Cortex-M3 processor provides the core for a high-performance, low-cost platform that meets the needs of minimal memory implementation, reduced pin count, and low-power consumption, while delivering outstanding computational performance and exceptional system response to interrupts.

"ARM Cortex-M3 Processor Core" on page 33 provides an overview of the ARM core; the core is detailed in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*.

#### 1.4.1.2 System Timer (SysTick)

Cortex-M3 includes an integrated system timer, SysTick. SysTick provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

- An RTOS tick timer which fires at a programmable rate (for example, 100 Hz) and invokes a SysTick routine.
- A high-speed alarm timer using the system clock.
- A variable rate alarm or signal timer—the duration is range-dependent on the reference clock used and the dynamic range of the counter.
- A simple counter. Software can use this to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

#### 1.4.1.3 Nested Vectored Interrupt Controller (NVIC)

The LM3S2110 controller includes the ARM Nested Vectored Interrupt Controller (NVIC) on the ARM Cortex-M3 core. The NVIC and Cortex-M3 prioritize and handle all exceptions. All exceptions are handled in Handler Mode. The processor state is automatically stored to the stack on an exception, and automatically restored from the stack at the end of the Interrupt Service Routine (ISR). The vector is fetched in parallel to the state saving, which enables efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration. Software can set eight priority levels on 7 exceptions (system handlers) and 26 interrupts.

"Interrupts" on page 41 provides an overview of the NVIC controller and the interrupt map. Exceptions and interrupts are detailed in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*.

#### 1.4.2 Motor Control Peripherals

To enhance motor control, the LM3S2110 controller features Pulse Width Modulation (PWM) outputs.

November 29, 2007 27

#### 1.4.2.1 PWM

Pulse width modulation (PWM) is a powerful technique for digitally encoding analog signal levels. High-resolution counters are used to generate a square wave, and the duty cycle of the square wave is modulated to encode an analog signal. Typical applications include switching power supplies and motor control.

On the LM3S2110, PWM motion control functionality can be achieved through:

- Dedicated, flexible motion control hardware using the PWM pins
- The motion control features of the general-purpose timers using the CCP pins

#### PWM Pins (see page 401)

The LM3S2110 PWM module consists of one PWM generator blocks and a control block. Each PWM generator block contains one timer (16-bit down or up/down counter), two comparators, a PWM signal generator, a dead-band generator, and an interrupt. The control block determines the polarity of the PWM signals, and which signals are passed through to the pins.

Each PWM generator block produces two PWM signals that can either be independent signals or a single pair of complementary signals with dead-band delays inserted. The output of the PWM generation blocks are managed by the output control block before being passed to the device pins.

#### CCP Pins (see page 181)

The General-Purpose Timer Module's CCP (Capture Compare PWM) pins are software programmable to support a simple PWM mode with a software-programmable output inversion of the PWM signal.

#### 1.4.3 Analog Peripherals

For support of analog signals, the LM3S2110 microcontroller offers three analog comparators.

#### 1.4.3.1 Analog Comparators (see page 388)

An analog comparator is a peripheral that compares two analog voltages, and provides a logical output that signals the comparison result.

The LM3S2110 microcontroller provides three independent integrated analog comparators that can be configured to drive an output or generate an interrupt .

A comparator can compare a test voltage against any one of these voltages:

- An individual external reference voltage
- A shared single external reference voltage
- A shared internal reference voltage

The comparator can provide its output to a device pin, acting as a replacement for an analog comparator on the board, or it can be used to signal the application via interrupts to cause it to start capturing a sample sequence.

#### 1.4.4 Serial Communications Peripherals

The LM3S2110 controller supports both asynchronous and synchronous serial communications with:

One fully programmable 16C550-type UART

- One SSI module
- One I<sup>2</sup>C module
- One CAN unit

#### 1.4.4.1 UART (see page 234)

A Universal Asynchronous Receiver/Transmitter (UART) is an integrated circuit used for RS-232C serial communications, containing a transmitter (parallel-to-serial converter) and a receiver (serial-to-parallel converter), each clocked separately.

The LM3S2110 controller includes one fully programmable 16C550-type UARTthat supports data transfer speeds up to 460.8 Kbps. (Although similar in functionality to a 16C550 UART, it is not register-compatible.) In addition, each UART is capable of supporting IrDA.

Separate 16x8 transmit (TX) and 16x12 receive (RX) FIFOs reduce CPU interrupt service loading. The UART can generate individually masked interrupts from the RX, TX, modem status, and error conditions. The module provides a single combined interrupt when any of the interrupts are asserted and are unmasked.

#### 1.4.4.2 SSI (see page 275)

Synchronous Serial Interface (SSI) is a four-wire bi-directional communications interface.

The LM3S2110 controller includes one SSI module that provides the functionality for synchronous serial communications with peripheral devices, and can be configured to use the Freescale SPI, MICROWIRE, or TI synchronous serial interface frame formats. The size of the data frame is also configurable, and can be set between 4 and 16 bits, inclusive.

The SSI module performs serial-to-parallel conversion on data received from a peripheral device, and parallel-to-serial conversion on data transmitted to a peripheral device. The TX and RX paths are buffered with internal FIFOs, allowing up to eight 16-bit values to be stored independently.

The SSI module can be configured as either a master or slave device. As a slave device, the SSI module can also be configured to disable its output, which allows a master device to be coupled with multiple slave devices.

The SSI module also includes a programmable bit rate clock divider and prescaler to generate the output serial clock derived from the SSI module's input clock. Bit rates are generated based on the input clock and the maximum bit rate is determined by the connected peripheral.

#### 1.4.4.3 $I^2C$ (see page 312)

The Inter-Integrated Circuit ( $I^2C$ ) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL).

The I<sup>2</sup>C bus interfaces to external I<sup>2</sup>C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I<sup>2</sup>C bus may also be used for system testing and diagnostic purposes in product development and manufacture.

The LM3S2110 controller includes one I<sup>2</sup>C module that provides the ability to communicate to other IC devices over an I<sup>2</sup>C bus. The I<sup>2</sup>C bus supports devices that can both transmit and receive (write and read) data.

Devices on the I<sup>2</sup>C bus can be designated as either a master or a slave. The I<sup>2</sup>C module supports both sending and receiving data as either a master or a slave, and also supports the simultaneous operation as both a master and a slave. The four I<sup>2</sup>C modes are: Master Transmit, Master Receive, Slave Transmit, and Slave Receive.

November 29, 2007 29

A Stellaris<sup>®</sup> I<sup>2</sup>C module can operate at two speeds: Standard (100 Kbps) and Fast (400 Kbps).

Both the I<sup>2</sup>C master and slave can generate interrupts. The I<sup>2</sup>C master generates interrupts when a transmit or receive operation completes (or aborts due to an error). The I<sup>2</sup>C slave generates interrupts when data has been sent or requested by a master.

#### 1.4.4.4 Controller Area Network (see page 347)

Controller Area Network (CAN) is a multicast shared serial-bus standard for connecting electronic control units (ECUs). CAN was specifically designed to be robust in electromagnetically noisy environments and can utilize a differential balanced line like RS-485 or a more robust twisted-pair wire. Originally created for automotive purposes, now it is used in many embedded control applications (for example, industrial or medical). Bit rates up to 1Mb/s are possible at network lengths below 40 meters. Decreased bit rates allow longer network distances (for example, 125 Kb/s at 500m).

A transmitter sends a message to all CAN nodes (broadcasting). Each node decides on the basis of the identifier received whether it should process the message. The identifier also determines the priority that the message enjoys in competition for bus access. Each CAN message can transmit from 0 to 8 bytes of user information. The LM3S2110 includes one CAN units.

#### 1.4.5 System Peripherals

#### 1.4.5.1 Programmable GPIOs (see page 134)

General-purpose input/output (GPIO) pins offer flexibility for a variety of connections.

The Stellaris<sup>®</sup> GPIO module is composed of eight physical GPIO blocks, each corresponding to an individual GPIO port. The GPIO module is FiRM-compliant (compliant to the ARM Foundation IP for Real-Time Microcontrollers specification) and supports 11-40 programmable input/output pins. The number of GPIOs available depends on the peripherals being used (see "Signal Tables" on page 437 for the signals available to each GPIO pin).

The GPIO module features programmable interrupt generation as either edge-triggered or level-sensitive on all pins, programmable control for GPIO pad configuration, and bit masking in both read and write operations through address lines.

#### 1.4.5.2 Three Programmable Timers (see page 175)

Programmable timers can be used to count or time external events that drive the Timer input pins.

The Stellaris<sup>®</sup> General-Purpose Timer Module (GPTM) contains three GPTM blocks. Each GPTM block provides two 16-bit timers/counters that can be configured to operate independently as timers or event counters, or configured to operate as one 32-bit timer or one 32-bit Real-Time Clock (RTC).

When configured in 32-bit mode, a timer can run as a Real-Time Clock (RTC), one-shot timer or periodic timer. When in 16-bit mode, a timer can run as a one-shot timer or periodic timer, and can extend its precision by using an 8-bit prescaler. A 16-bit timer can also be configured for event capture or Pulse Width Modulation (PWM) generation.

#### 1.4.5.3 Watchdog Timer (see page 211)

A watchdog timer can generate nonmaskable interrupts (NMIs) or a reset when a time-out value is reached. The watchdog timer is used to regain control when a system has failed due to a software error or to the failure of an external device to respond in the expected way.

The Stellaris<sup>®</sup> Watchdog Timer module consists of a 32-bit down counter, a programmable load register, interrupt generation logic, and a locking register.

www.DataSheet4U.com

vww.DataSheet4U

The Watchdog Timer can be configured to generate an interrupt to the controller on its first time-out, and to generate a reset signal on its second time-out. Once the Watchdog Timer has been configured, the lock register can be written to prevent the timer configuration from being inadvertently altered.

#### 1.4.6 Memory Peripherals

The LM3S2110 controller offers both single-cycle SRAM and single-cycle Flash memory.

#### 1.4.6.1 SRAM (see page 110)

The LM3S2110 static random access memory (SRAM) controller supports 16 KB SRAM. The internal SRAM of the Stellaris<sup>®</sup> devices is located at offset 0x0000.0000 of the device memory map. To reduce the number of time-consuming read-modify-write (RMW) operations, ARM has introduced *bit-banding* technology in the new Cortex-M3 processor. With a bit-band-enabled processor, certain regions in the memory map (SRAM and peripheral space) can use address aliases to access individual bits in a single, atomic operation.

#### 1.4.6.2 Flash (see page 111)

The LM3S2110 Flash controller supports 64 KB of flash memory. The flash is organized as a set of 1-KB blocks that can be individually erased. Erasing a block causes the entire contents of the block to be reset to all 1s. These blocks are paired into a set of 2-KB blocks that can be individually protected. The blocks can be marked as read-only or execute-only, providing different levels of code protection. Read-only blocks cannot be erased or programmed, protecting the contents of those blocks from being modified. Execute-only blocks cannot be erased or programmed, and can only be read by the controller instruction fetch mechanism, protecting the contents of those blocks from being read by either the controller or by a debugger.

#### 1.4.7 Additional Features

#### 1.4.7.1 Memory Map (see page 39)

A memory map lists the location of instructions and data in memory. The memory map for the LM3S2110 controller can be found in "Memory Map" on page 39. Register addresses are given as a hexadecimal increment, relative to the module's base address as shown in the memory map.

The ARM® Cortex™-M3 Technical Reference Manual provides further information on the memory map.

#### 1.4.7.2 JTAG TAP Controller (see page 43)

The Joint Test Action Group (JTAG) port provides a standardized serial interface for controlling the Test Access Port (TAP) and associated test logic. The TAP, JTAG instruction register, and JTAG data registers can be used to test the interconnects of assembled printed circuit boards, obtain manufacturing information on the components, and observe and/or control the inputs and outputs of the controller during normal operation. The JTAG port provides a high degree of testability and chip-level access at a low cost.

The JTAG port is comprised of the standard five pins: TRST, TCK, TMS, TDI, and TDO. Data is transmitted serially into the controller on TDI and out of the controller on TDO. The interpretation of this data is dependent on the current state of the TAP controller. For detailed information on the operation of the JTAG port and TAP controller, please refer to the IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture.

The Luminary Micro JTAG controller works with the ARM JTAG controller built into the Cortex-M3 core. This is implemented by multiplexing the  $\tiny{TDO}$  outputs from both JTAG controllers. ARM JTAG instructions select the ARM  $\tiny{TDO}$  output while Luminary Micro JTAG instructions select the Luminary

November 29, 2007

Preliminary

Micro TDO outputs. The multiplexer is controlled by the Luminary Micro JTAG controller, which has comprehensive programming for the ARM, Luminary Micro, and unimplemented JTAG instructions.

#### 1.4.7.3 System Control and Clocks (see page 54)

System control determines the overall operation of the device. It provides information about the device, controls the clocking of the device and individual peripherals, and handles reset detection and reporting.

#### 1.4.8 Hardware Details

Details on the pins and package can be found in the following sections:

- "Pin Diagram" on page 436
- "Signal Tables" on page 437
- "Operating Characteristics" on page 449
- "Electrical Characteristics" on page 450
- "Package Information" on page 461

ww.DataSheet4U.com

## 2 ARM Cortex-M3 Processor Core

The ARM Cortex-M3 processor provides the core for a high-performance, low-cost platform that meets the needs of minimal memory implementation, reduced pin count, and low power consumption, while delivering outstanding computational performance and exceptional system response to interrupts. Features include:

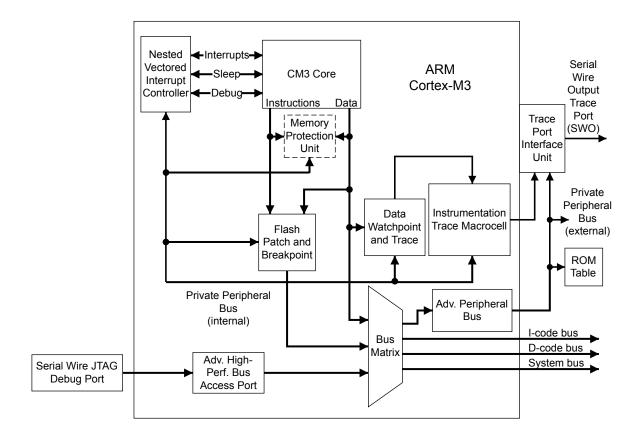
- Compact core.
- Thumb-2 instruction set, delivering the high-performance expected of an ARM core in the memory size usually associated with 8- and 16-bit devices; typically in the range of a few kilobytes of memory for microcontroller class applications.
- Rapid application execution through Harvard architecture characterized by separate buses for instruction and data.
- Exceptional interrupt handling, by implementing the register manipulations required for handling an interrupt in hardware.
- Memory protection unit (MPU) to provide a privileged mode of operation for complex applications.
- Migration from the ARM7<sup>TM</sup> processor family for better performance and power efficiency.
- Full-featured debug solution with a:
  - Serial Wire JTAG Debug Port (SWJ-DP)
  - Flash Patch and Breakpoint (FPB) unit for implementing breakpoints
  - Data Watchpoint and Trigger (DWT) unit for implementing watchpoints, trigger resources, and system profiling
  - Instrumentation Trace Macrocell (ITM) for support of printf style debugging
  - Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer

The Stellaris<sup>®</sup> family of microcontrollers builds on this core to bring high-performance 32-bit computing to cost-sensitive embedded microcontroller applications, such as factory automation and control, industrial control power devices, building and home automation, and stepper motors.

For more information on the ARM Cortex-M3 processor core, see the *ARM*® *Cortex*™-*M3 Technical Reference Manual*. For information on SWJ-DP, see the *ARM*® *CoreSight Technical Reference Manual*.

### 2.1 Block Diagram

Figure 2-1. CPU Block Diagram



## 2.2 Functional Description

Important: The ARM® Cortex™-M3 Technical Reference Manual describes all the features of an ARM Cortex-M3 in detail. However, these features differ based on the implementation. This section describes the Stellaris® implementation.

Luminary Micro has implemented the ARM Cortex-M3 core as shown in Figure 2-1 on page 34. As noted in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*, several Cortex-M3 components are flexible in their implementation: SW/JTAG-DP, ETM, TPIU, the ROM table, the MPU, and the Nested Vectored Interrupt Controller (NVIC). Each of these is addressed in the sections that follow.

#### 2.2.1 Serial Wire and JTAG Debug

Luminary Micro has replaced the ARM SW-DP and JTAG-DP with the ARM CoreSight™-compliant Serial Wire JTAG Debug Port (SWJ-DP) interface. This means Chapter 12, "Debug Port," of the *ARM*® *Cortex™-M3 Technical Reference Manual* does not apply to Stellaris<sup>®</sup> devices.

The SWJ-DP interface combines the SWD and JTAG debug ports into one module. See the CoreSight™ Design Kit Technical Reference Manual for details on SWJ-DP.

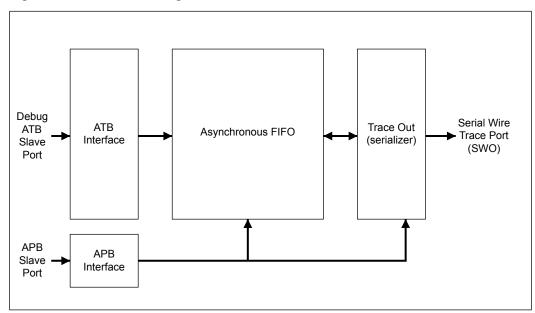
#### 2.2.2 Embedded Trace Macrocell (ETM)

ETM was not implemented in the Stellaris<sup>®</sup> devices. This means Chapters 15 and 16 of the *ARM*® *Cortex*™-*M3 Technical Reference Manual* can be ignored.

#### 2.2.3 Trace Port Interface Unit (TPIU)

The TPIU acts as a bridge between the Cortex-M3 trace data from the ITM, and an off-chip Trace Port Analyzer. The Stellaris<sup>®</sup> devices have implemented TPIU as shown in Figure 2-2 on page 35. This is similar to the non-ETM version described in the *ARM® Cortex™-M3 Technical Reference Manual*, however, SWJ-DP only provides SWV output for the TPIU.

Figure 2-2. TPIU Block Diagram



#### 2.2.4 ROM Table

The default ROM table was implemented as described in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*.

#### 2.2.5 Memory Protection Unit (MPU)

The Memory Protection Unit (MPU) is included on the LM3S2110 controller and supports the standard ARMv7 Protected Memory System Architecture (PMSA) model. The MPU provides full support for protection regions, overlapping protection regions, access permissions, and exporting memory attributes to the system.

#### 2.2.6 Nested Vectored Interrupt Controller (NVIC)

The Nested Vectored Interrupt Controller (NVIC):

- Facilitates low-latency exception and interrupt handling
- Controls power management
- Implements system control registers

November 29, 2007 35

The NVIC supports up to 240 dynamically reprioritizable interrupts each with up to 256 levels of priority. The NVIC and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts. The NVIC maintains knowledge of the stacked (nested) interrupts to enable tail-chaining of interrupts.

You can only fully access the NVIC from privileged mode, but you can pend interrupts in user-mode if you enable the Configuration Control Register (see the ARM® Cortex™-M3 Technical Reference Manual). Any other user-mode access causes a bus fault.

All NVIC registers are accessible using byte, halfword, and word unless otherwise stated.

All NVIC registers and system debug registers are little endian regardless of the endianness state of the processor.

#### 2.2.6.1 Interrupts

The ARM® Cortex™-M3 Technical Reference Manual describes the maximum number of interrupts and interrupt priorities. The LM3S2110 microcontroller supports 26 interrupts with eight priority levels.

#### 2.2.6.2 System Timer (SysTick)

Cortex-M3 includes an integrated system timer, SysTick. SysTick provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

- An RTOS tick timer which fires at a programmable rate (for example, 100 Hz) and invokes a SysTick routine.
- A high-speed alarm timer using the system clock.
- A variable rate alarm or signal timer—the duration is range-dependent on the reference clock used and the dynamic range of the counter.
- A simple counter. Software can use this to measure time to completion and time used.
- An internal clock source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

#### **Functional Description**

The timer consists of three registers:

- A control and status counter to configure its clock, enable the counter, enable the SysTick interrupt, and determine counter status.
- The reload value for the counter, used to provide the counter's wrap value.
- The current value of the counter.

A fourth register, the SysTick Calibration Value Register, is not implemented in the Stellaris<sup>®</sup> devices.

When enabled, the timer counts down from the reload value to zero, reloads (wraps) to the value in the SysTick Reload Value register on the next clock edge, then decrements on subsequent clocks. Writing a value of zero to the Reload Value register disables the counter on the next wrap. When the counter reaches zero, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

Writing to the Current Value register clears the register and the COUNTFLAG status bit. The write does not trigger the SysTick exception logic. On a read, the current value is the value of the register at the time the register is accessed.

If the core is in debug state (halted), the counter will not decrement. The timer is clocked with respect to a reference clock. The reference clock can be the core clock or an external clock source.

### SysTick Control and Status Register

Use the SysTick Control and Status Register to enable the SysTick features. The reset is 0x0000.0000.

Bit/Field	Name	Туре	Reset	Description	
31:17	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.	
16	COUNTFLAG	R/W	0	Returns 1 if timer counted to 0 since last time this was read. Clears on read by application. If read by the debugger using the DAP, this bit is cleared on read-onl if the MasterType bit in the AHB-AP Control Register is set to 0. Otherwise, the COUNTFLAG bit is not changed by the debugger read.	
15:3	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.	
2	CLKSOURCE	R/W	0	0 = external reference clock. (Not implemented for Stellaris microcontrollers.) 1 = core clock.	
				If no reference clock is provided, it is held at 1 and so gives the same time as the core clock. The core clock must be at least 2.5 times faster than the reference clock. If it is not, the count values are unpredictable.	
1	TICKINT	R/W	0	1 = counting down to 0 pends the SysTick handler.	
				0 = counting down to 0 does not pend the SysTick handler. Software can use the COUNTFLAG to determine if ever counted to 0.	
0	ENABLE	R/W	0	1 = counter operates in a multi-shot way. That is, counter loads with the Reload value and then begins counting down. On reaching 0, it sets the COUNTFLAG to 1 and optionally pends the SysTick handler, based on TICKINT. It then loads the Reload value again, and begins counting.	
				0 = counter disabled.	

#### SysTick Reload Value Register

Use the SysTick Reload Value Register to specify the start value to load into the current value register when the counter reaches 0. It can be any value between 1 and 0x00FF.FFFF. A start value of 0 is possible, but has no effect because the SysTick interrupt and COUNTFLAG are activated when counting from 1 to 0.

Therefore, as a multi-shot timer, repeated over and over, it fires every N+1 clock pulse, where N is any value from 1 to 0x00FF.FFFF. So, if the tick interrupt is required every 100 clock pulses, 99 must be written into the RELOAD. If a new value is written on each tick interrupt, so treated as single shot, then the actual count down must be written. For example, if a tick is next required after 400 clock pulses, 400 must be written into the RELOAD.

Bit/Field	Name	Type	Reset	Description
31:24	reserved	RO		Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

November 29, 2007 37

Bit/Field	Name	Type	Reset	Description
23:0	RELOAD	W1C	-	Value to load into the SysTick Current Value Register when the counter reaches 0.

### SysTick Current Value Register

Use the SysTick Current Value Register to find the current value in the register.

Bit/Field	Name	Туре	Reset	Description
31:24	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:0	CURRENT	W1C	-	Current value at the time the register is accessed. No read-modify-write protection is provided, so change with care.
				This register is write-clear. Writing to it with any value clears the register to 0. Clearing this register also clears the COUNTFLAG bit of the SysTick Control and Status Register.

ww DataSheet4U.com

### SysTick Calibration Value Register

The SysTick Calibration Value register is not implemented.

# **Memory Map**

The memory map for the LM3S2110 controller is provided in Table 3-1 on page 39.

In this manual, register addresses are given as a hexadecimal increment, relative to the module's base address as shown in the memory map. See also Chapter 4, "Memory Map" in the ARM® Cortex™-M3 Technical Reference Manual.

Important: In Table 3-1 on page 39, addresses not listed are reserved.

Table 3-1. Memory Map<sup>a</sup>

Start	End	Description	For details on registers, see page	
Memory		'		
0x0000.0000	0x0000.FFFF	On-chip flash <sup>b</sup>	114	
0x2000.0000	0x2000.3FFF	Bit-banded on-chip SRAM <sup>c</sup>	114	
0x2010.0000	0x21FF.FFFF	Reserved non-bit-banded SRAM space	-	
0x2200.0000	0x23FF.FFFF	Bit-band alias of 0x2000.0000 through 0x200F.FFFF	110	
0x2400.0000	0x3FFF.FFFF	Reserved non-bit-banded SRAM space	-	
FiRM Peripherals	'			
0x4000.0000	0x4000.0FFF	Watchdog timer	213	
0x4000.4000	0x4000.4FFF	GPIO Port A	140	
0x4000.5000	0x4000.5FFF	GPIO Port B	140	
0x4000.6000	0x4000.6FFF	GPIO Port C	140	
0x4000.7000	0x4000.7FFF	GPIO Port D	140	
0x4000.8000	0x4000.8FFF	SSI0	286	
0x4000.C000	0x4000.CFFF	UART0	241	
Peripherals	'			
0x4002.0000	0x4002.07FF	I2C Master 0	325	
0x4002.0800	0x4002.0FFF	I2C Slave 0	338	
0x4002.4000	0x4002.4FFF	GPIO Port E	140	
0x4002.5000	0x4002.5FFF	GPIO Port F	140	
0x4002.6000	0x4002.6FFF	GPIO Port G	140	
0x4002.7000	0x4002.7FFF	GPIO Port H	140	
0x4002.8000	0x4002.8FFF	PWM	407	
0x4003.0000	0x4003.0FFF	Timer0	186	
0x4003.1000	0x4003.1FFF	Timer1	186	
0x4003.2000	0x4003.2FFF	Timer2	186	
0x4003.C000	0x4003.CFFF	Analog Comparators	388	
0x4004.0000	0x4004.0FFF	CAN0 Controller	360	
0x400F.D000	0x400F.DFFF	Flash control	114	
0x400F.E000	0x400F.EFFF	System control	61	
0x4200.0000	0x43FF.FFFF	Bit-banded alias of 0x4000.0000 through 0x400F.FFFF	-	
Private Peripheral B	us	•	•	

Start	End	Description	For details on registers, see page
0xE000.0000	0xE000.0FFF	Instrumentation Trace Macrocell (ITM)	ARM®
0xE000.1000	0xE000.1FFF	Data Watchpoint and Trace (DWT)	Cortex™-M3 Technical
0xE000.2000	0xE000.2FFF	Flash Patch and Breakpoint (FPB)	Reference
0xE000.3000	0xE000.DFFF	Reserved	Manual
0xE000.E000	0xE000.EFFF	Nested Vectored Interrupt Controller (NVIC)	
0xE000.F000	0xE003.FFFF	Reserved	
0xE004.0000	0xE004.0FFF	Trace Port Interface Unit (TPIU)	
0xE004.1000	0xE004.1FFF	Reserved	-
0xE004.2000	0xE00F.FFFF	Reserved	-
0xE010.0000	0xFFFF.FFFF	Reserved for vendor peripherals	-

a. All reserved space returns a bus fault when read or written.

b. The unavailable flash will bus fault throughout this range.

c. The unavailable SRAM will bus fault throughout this range.

www.DataSheet4U.com

# 4 Interrupts

The ARM Cortex-M3 processor and the Nested Vectored Interrupt Controller (NVIC) prioritize and handle all exceptions. All exceptions are handled in Handler Mode. The processor state is automatically stored to the stack on an exception, and automatically restored from the stack at the end of the Interrupt Service Routine (ISR). The vector is fetched in parallel to the state saving, which enables efficient interrupt entry. The processor supports tail-chaining, which enables back-to-back interrupts to be performed without the overhead of state saving and restoration.

Table 4-1 on page 41 lists all the exceptions. Software can set eight priority levels on seven of these exceptions (system handlers) as well as on 26 interrupts (listed in Table 4-2 on page 42).

Priorities on the system handlers are set with the NVIC System Handler Priority registers. Interrupts are enabled through the NVIC Interrupt Set Enable register and prioritized with the NVIC Interrupt Priority registers. You can also group priorities by splitting priority levels into pre-emption priorities and subpriorities. All the interrupt registers are described in Chapter 8, "Nested Vectored Interrupt Controller" in the *ARM® Cortex™-M3 Technical Reference Manual*.

Internally, the highest user-settable priority (0) is treated as fourth priority, after a Reset, NMI, and a Hard Fault. Note that 0 is the default priority for all the settable priorities.

If you assign the same priority level to two or more interrupts, their hardware priority (the lower the position number) determines the order in which the processor activates them. For example, if both GPIO Port A and GPIO Port B are priority level 1, then GPIO Port A has higher priority.

See Chapter 5, "Exceptions" and Chapter 8, "Nested Vectored Interrupt Controller" in the *ARM*® *Cortex*™-*M3 Technical Reference Manual* for more information on exceptions and interrupts.

Note: In Table 4-2 on page 42 interrupts not listed are reserved.

**Table 4-1. Exception Types** 

Exception Type	Position	<b>Priority</b> <sup>a</sup>	Description
-	0	-	Stack top is loaded from first entry of vector table on reset.
Reset	1	-3 (highest)	Invoked on power up and warm reset. On first instruction, drops to lowest priority (and then is called the base level of activation). This is asynchronous.
Non-Maskable Interrupt (NMI)	2	-2	Cannot be stopped or preempted by any exception but reset. This is asynchronous.
			An NMI is only producible by software, using the NVIC <b>Interrupt Control State</b> register.
Hard Fault	3	-1	All classes of Fault, when the fault cannot activate due to priority or the configurable fault handler has been disabled. This is synchronous.
Memory Management	4	settable	MPU mismatch, including access violation and no match. This is synchronous.
			The priority of this exception can be changed.
Bus Fault	5	settable	Pre-fetch fault, memory access fault, and other address/memory related faults. This is synchronous when precise and asynchronous when imprecise.
			You can enable or disable this fault.
Usage Fault	6	settable	Usage fault, such as undefined instruction executed or illegal state transition attempt. This is synchronous.
-	7-10	-	Reserved.
SVCall	11	settable	System service call with SVC instruction. This is synchronous.

November 29, 2007

Proliminary

Exception Type	Position	<b>Priority</b> <sup>a</sup>	Description
Debug Monitor	12	settable	Debug monitor (when not halting). This is synchronous, but only active when enabled. It does not activate if lower priority than the current activation.
-	13	-	Reserved.
PendSV	14	settable	Pendable request for system service. This is asynchronous and only pended by software.
SysTick	15	settable	System tick timer has fired. This is asynchronous.
Interrupts	16 and above	settable	Asserted from outside the ARM Cortex-M3 core and fed through the NVIC (prioritized). These are all asynchronous. Table 4-2 on page 42 lists the interrupts on the LM3S2110 controller.

a. 0 is the default priority for all the settable priorities.

### Table 4-2. Interrupts

Interrupt (Bit in Interrupt Registers)	Description
0	GPIO Port A
1	GPIO Port B
2	GPIO Port C
3	GPIO Port D
4	GPIO Port E
5	UARTO
7	SSI0
8	I2C0
9	PWM Fault
10	PWM Generator 0
18	Watchdog timer
19	Timer0 A
20	Timer0 B
21	Timer1 A
22	Timer1 B
23	Timer2 A
24	Timer2 B
25	Analog Comparator 0
26	Analog Comparator 1
27	Analog Comparator 2
28	System Control
29	Flash Control
30	GPIO Port F
31	GPIO Port G
32	GPIO Port H
39	CAN0

# 5 JTAG Interface

The Joint Test Action Group (JTAG) port is an IEEE standard that defines a Test Access Port and Boundary Scan Architecture for digital integrated circuits and provides a standardized serial interface for controlling the associated test logic. The TAP, Instruction Register (IR), and Data Registers (DR) can be used to test the interconnections of assembled printed circuit boards and obtain manufacturing information on the components. The JTAG Port also provides a means of accessing and controlling design-for-test features such as I/O pin observation and control, scan testing, and debugging.

The JTAG port is comprised of the standard five pins: TRST, TCK, TMS, TDI, and TDO. Data is transmitted serially into the controller on TDI and out of the controller on TDO. The interpretation of this data is dependent on the current state of the TAP controller. For detailed information on the operation of the JTAG port and TAP controller, please refer to the IEEE Standard 1149.1-Test Access Port and Boundary-Scan Architecture.

The Luminary Micro JTAG controller works with the ARM JTAG controller built into the Cortex-M3 core. This is implemented by multiplexing the TDO outputs from both JTAG controllers. ARM JTAG instructions select the ARM TDO output while Luminary Micro JTAG instructions select the Luminary Micro TDO outputs. The multiplexer is controlled by the Luminary Micro JTAG controller, which has comprehensive programming for the ARM, Luminary Micro, and unimplemented JTAG instructions.

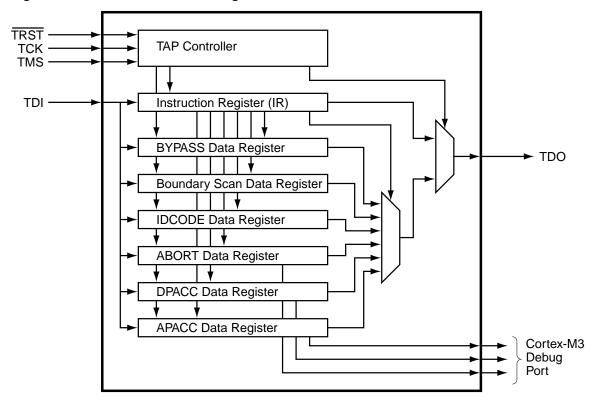
The JTAG module has the following features:

- IEEE 1149.1-1990 compatible Test Access Port (TAP) controller
- Four-bit Instruction Register (IR) chain for storing JTAG instructions
- IEEE standard instructions:
  - BYPASS instruction
  - IDCODE instruction
  - SAMPLE/PRELOAD instruction
  - EXTEST instruction
  - INTEST instruction
- ARM additional instructions:
  - APACC instruction
  - DPACC instruction
  - ABORT instruction
- Integrated ARM Serial Wire Debug (SWD)

See the ARM® Cortex™-M3 Technical Reference Manual for more information on the ARM JTAG controller.

# 5.1 Block Diagram

Figure 5-1. JTAG Module Block Diagram



# **5.2** Functional Description

A high-level conceptual drawing of the JTAG module is shown in Figure 5-1 on page 44. The JTAG module is composed of the Test Access Port (TAP) controller and serial shift chains with parallel update registers. The TAP controller is a simple state machine controlled by the TRST, TCK and TMS inputs. The current state of the TAP controller depends on the current value of TRST and the sequence of values captured on TMS at the rising edge of TCK. The TAP controller determines when the serial shift chains capture new data, shift data from TDI towards TDO, and update the parallel load registers. The current state of the TAP controller also determines whether the Instruction Register (IR) chain or one of the Data Register (DR) chains is being accessed.

The serial shift chains with parallel load registers are comprised of a single Instruction Register (IR) chain and multiple Data Register (DR) chains. The current instruction loaded in the parallel load register determines which DR chain is captured, shifted, or updated during the sequencing of the TAP controller.

Some instructions, like EXTEST and INTEST, operate on data currently in a DR chain and do not capture, shift, or update any of the chains. Instructions that are not implemented decode to the BYPASS instruction to ensure that the serial path between TDI and TDO is always connected (see Table 5-2 on page 50 for a list of implemented instructions).

See "JTAG and Boundary Scan" on page 456 for JTAG timing diagrams.

44 November 29, 2007

### 5.2.1 JTAG Interface Pins

The JTAG interface consists of five standard pins: TRST, TCK, TMS, TDI, and TDO. These pins and their associated reset state are given in Table 5-1 on page 45. Detailed information on each pin follows.

Table 5-1. JTAG Port Pins Reset State

Pin Name	Data Direction	Internal Pull-Up	Internal Pull-Down	Drive Strength	Drive Value
TRST	Input	Enabled	Disabled	N/A	N/A
TCK	Input	Enabled	Disabled	N/A	N/A
TMS	Input	Enabled	Disabled	N/A	N/A
TDI	Input	Enabled	Disabled	N/A	N/A
TDO	Output	Enabled	Disabled	2-mA driver	High-Z

SHEEL4U.COIII

### 5.2.1.1 Test Reset Input (TRST)

The  $\overline{\mathtt{TRST}}$  pin is an asynchronous active Low input signal for initializing and resetting the JTAG TAP controller and associated JTAG circuitry. When  $\overline{\mathtt{TRST}}$  is asserted, the TAP controller resets to the Test-Logic-Reset state and remains there while  $\overline{\mathtt{TRST}}$  is asserted. When the TAP controller enters the Test-Logic-Reset state, the JTAG Instruction Register (IR) resets to the default instruction, IDCODE.

By default, the internal pull-up resistor on the TRST pin is enabled after reset. Changes to the pull-up resistor settings on GPIO Port B should ensure that the internal pull-up resistor remains enabled on PB7/TRST; otherwise JTAG communication could be lost.

### 5.2.1.2 Test Clock Input (TCK)

The  ${ t TCK}$  pin is the clock for the JTAG module. This clock is provided so the test logic can operate independently of any other system clocks. In addition, it ensures that multiple JTAG TAP controllers that are daisy-chained together can synchronously communicate serial test data between components. During normal operation,  ${ t TCK}$  is driven by a free-running clock with a nominal 50% duty cycle. When necessary,  ${ t TCK}$  can be stopped at 0 or 1 for extended periods of time. While  ${ t TCK}$  is stopped at 0 or 1, the state of the TAP controller does not change and data in the JTAG Instruction and Data Registers is not lost.

By default, the internal pull-up resistor on the  ${ t TCK}$  pin is enabled after reset. This assures that no clocking occurs if the pin is not driven from an external source. The internal pull-up and pull-down resistors can be turned off to save internal power as long as the  ${ t TCK}$  pin is constantly being driven by an external source.

### 5.2.1.3 Test Mode Select (TMS)

The TMS pin selects the next state of the JTAG TAP controller. TMS is sampled on the rising edge of TCK. Depending on the current TAP state and the sampled value of TMS, the next state is entered. Because the TMS pin is sampled on the rising edge of TCK, the *IEEE Standard 1149.1* expects the value on TMS to change on the falling edge of TCK.

Holding TMS high for five consecutive TCK cycles drives the TAP controller state machine to the Test-Logic-Reset state. When the TAP controller enters the Test-Logic-Reset state, the JTAG Instruction Register (IR) resets to the default instruction, IDCODE. Therefore, this sequence can be used as a reset mechanism, similar to asserting TRST. The JTAG Test Access Port state machine can be seen in its entirety in Figure 5-2 on page 47.

By default, the internal pull-up resistor on the TMS pin is enabled after reset. Changes to the pull-up resistor settings on GPIO Port C should ensure that the internal pull-up resistor remains enabled on PC1/TMS; otherwise JTAG communication could be lost.

#### 5.2.1.4 Test Data Input (TDI)

The TDI pin provides a stream of serial information to the IR chain and the DR chains. TDI is sampled on the rising edge of TCK and, depending on the current TAP state and the current instruction, presents this data to the proper shift register chain. Because the TDI pin is sampled on the rising edge of TCK, the IEEE Standard 1149.1 expects the value on TDI to change on the falling edge of TCK.

By default, the internal pull-up resistor on the TDI pin is enabled after reset. Changes to the pull-up resistor settings on GPIO Port C should ensure that the internal pull-up resistor remains enabled on PC2/TDI: otherwise JTAG communication could be lost.

#### 5.2.1.5 Test Data Output (TDO)

The TDO pin provides an output stream of serial information from the IR chain or the DR chains. The value of TDO depends on the current TAP state, the current instruction, and the data in the chain being accessed. In order to save power when the JTAG port is not being used, the TDO pin is placed in an inactive drive state when not actively shifting out data. Because TDO can be connected to the TDI of another controller in a daisy-chain configuration, the IEEE Standard 1149.1 expects the value on TDO to change on the falling edge of TCK.

By default, the internal pull-up resistor on the TDO pin is enabled after reset. This assures that the pin remains at a constant logic level when the JTAG port is not being used. The internal pull-up and pull-down resistors can be turned off to save internal power if a High-Z output value is acceptable during certain TAP controller states.

#### 5.2.2 JTAG TAP Controller

The JTAG TAP controller state machine is shown in Figure 5-2 on page 47. The TAP controller state machine is reset to the Test-Logic-Reset state on the assertion of a Power-On-Reset (POR) or the assertion of TRST. Asserting the correct sequence on the TMS pin allows the JTAG module to shift in new instructions, shift in data, or idle during extended testing sequences. For detailed information on the function of the TAP controller and the operations that occur in each state, please refer to IEEE Standard 1149.1.

46 November 29, 2007 **Preliminary** 

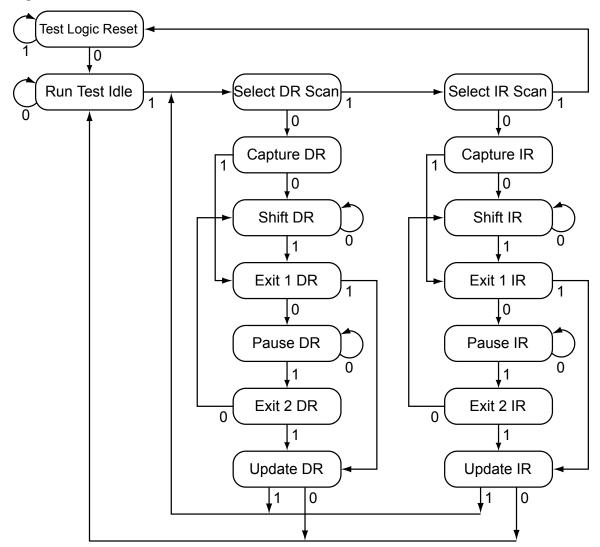


Figure 5-2. Test Access Port State Machine

### 5.2.3 Shift Registers

The Shift Registers consist of a serial shift register chain and a parallel load register. The serial shift register chain samples specific information during the TAP controller's CAPTURE states and allows this information to be shifted out of TDO during the TAP controller's SHIFT states. While the sampled data is being shifted out of the chain on TDO, new data is being shifted into the serial shift register on TDI. This new data is stored in the parallel load register during the TAP controller's UPDATE states. Each of the shift registers is discussed in detail in "Register Descriptions" on page 50.

### 5.2.4 Operational Considerations

There are certain operational considerations when using the JTAG module. Because the JTAG pins can be programmed to be GPIOs, board configuration and reset conditions on these pins must be considered. In addition, because the JTAG module has integrated ARM Serial Wire Debug, the method for switching between these two operational modes is described below.

### 5.2.4.1 GPIO Functionality

When the controller is reset with either a POR or  $\overline{RST}$ , the JTAG/SWD port pins default to their JTAG/SWD configurations. The default configuration includes enabling digital functionality (setting **GPIODEN** to 1), enabling the pull-up resistors (setting **GPIOPUR** to 1), and enabling the alternate hardware function (setting **GPIOAFSEL** to 1) for the PB7 and PC[3:0] JTAG/SWD pins.

It is possible for software to configure these pins as GPIOs after reset by writing 0s to PB7 and PC[3:0] in the **GPIOAFSEL** register. If the user does not require the JTAG/SWD port for debugging or board-level testing, this provides five more GPIOs for use in the design.

Caution – If the JTAG pins are used as GPIOs in a design, PB7 and PC2 cannot have external pull-down resistors connected to both of them at the same time. If both pins are pulled Low during reset, the controller has unpredictable behavior. If this happens, remove one or both of the pull-down resistors, and apply  $\overline{\text{RST}}$  or power-cycle the part.

In addition, it is possible to create a software sequence that prevents the debugger from connecting to the Stellaris® microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. This may lock the debugger out of the part. This can be avoided with a software routine that restores JTAG functionality based on an external or software trigger.

The commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 150) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 160) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 161) have been set to 1.

#### Recovering a "Locked" Device

If software configures any of the JTAG/SWD pins as GPIO and loses the ability to communicate with the debugger, there is a debug sequence that can be used to recover the device. Performing a total of ten JTAG-to-SWD and SWD-to-JTAG switch sequences while holding the device in reset mass erases the flash memory. The sequence to recover the device is:

- 1. Assert and hold the RST signal.
- 2. Perform the JTAG-to-SWD switch sequence.
- 3. Perform the SWD-to-JTAG switch sequence.
- Perform the JTAG-to-SWD switch sequence.
- 5. Perform the SWD-to-JTAG switch sequence.
- Perform the JTAG-to-SWD switch sequence.
- **7.** Perform the SWD-to-JTAG switch sequence.
- 8. Perform the JTAG-to-SWD switch sequence.
- Perform the SWD-to-JTAG switch sequence.
- 10. Perform the JTAG-to-SWD switch sequence.
- 11. Perform the SWD-to-JTAG switch sequence.

November 29, 2007

#### 12. Release the RST signal.

The JTAG-to-SWD and SWD-to-JTAG switch sequences are described in "ARM Serial Wire Debug (SWD)" on page 49. When performing switch sequences for the purpose of recovering the debug capabilities of the device, only steps 1 and 2 of the switch sequence need to be performed.

### 5.2.4.2 ARM Serial Wire Debug (SWD)

In order to seamlessly integrate the ARM Serial Wire Debug (SWD) functionality, a serial-wire debugger must be able to connect to the Cortex-M3 core without having to perform, or have any knowledge of, JTAG cycles. This is accomplished with a SWD preamble that is issued before the SWD session begins.

The preamble used to enable the SWD interface of the SWJ-DP module starts with the TAP controller in the Test-Logic-Reset state. From here, the preamble sequences the TAP controller through the following states: Run Test Idle, Select DR, Select IR, Test Logic Reset, Test Logic Reset, Run Test Idle, Run Test Idle, Select DR, Select IR, Test Logic Reset, Test Logic Reset, Run Test Idle, Run Test Idle, Select DR, Select IR, and Test Logic Reset states.

Stepping through this sequences of the TAP state machine enables the SWD interface and disables the JTAG interface. For more information on this operation and the SWD interface, see the *ARM*® *Cortex*™-*M3 Technical Reference Manual* and the *ARM*® *CoreSight Technical Reference Manual*.

Because this sequence is a valid series of JTAG operations that could be issued, the ARM JTAG TAP controller is not fully compliant to the *IEEE Standard 1149.1*. This is the only instance where the ARM JTAG TAP controller does not meet full compliance with the specification. Due to the low probability of this sequence occurring during normal operation of the TAP controller, it should not affect normal performance of the JTAG interface.

#### JTAG-to-SWD Switching

To switch the operating mode of the Debug Access Port (DAP) from JTAG to SWD mode, the external debug hardware must send a switch sequence to the device. The 16-bit switch sequence for switching to SWD mode is defined as b1110011110011110, transmitted LSB first. This can also be represented as 16'hE79E when transmitted LSB first. The complete switch sequence should consist of the following transactions on the TCK/SWCLK and TMS/SWDIO signals:

- 1. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO set to 1. This ensures that both JTAG and SWD are in their reset/idle states.
- 2. Send the 16-bit JTAG-to-SWD switch sequence, 16'hE79E.
- 3. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO set to 1. This ensures that if SWJ-DP was already in SWD mode, before sending the switch sequence, the SWD goes into the line reset state.

#### SWD-to-JTAG Switching

To switch the operating mode of the Debug Access Port (DAP) from SWD to JTAG mode, the external debug hardware must send a switch sequence to the device. The 16-bit switch sequence for switching to JTAG mode is defined as b1110011110011110, transmitted LSB first. This can also be represented as 16'hE73C when transmitted LSB first. The complete switch sequence should consist of the following transactions on the TCK/SWCLK and TMS/SWDIO signals:

1. Send at least 50 TCK/SWCLK cycles with TMS/SWDIO set to 1. This ensures that both JTAG and SWD are in their reset/idle states.

- 2. Send the 16-bit SWD-to-JTAG switch sequence, 16'hE73C.
- 3. Send at least 5 TCK/SWCLK cycles with TMS/SWDIO set to 1. This ensures that if SWJ-DP was already in JTAG mode, before sending the switch sequence, the JTAG goes into the Test Logic Reset state.

### 5.3 Initialization and Configuration

After a Power-On-Reset or an external reset ( $\overline{\mathbb{RST}}$ ), the JTAG pins are automatically configured for JTAG communication. No user-defined initialization or configuration is needed. However, if the user application changes these pins to their GPIO function, they must be configured back to their JTAG functionality before JTAG communication can be restored. This is done by enabling the five JTAG pins (PB7 and PC[3:0]) for their alternate function using the **GPIOAFSEL** register.

.DataSheet4U.com

### 5.4 Register Descriptions

There are no APB-accessible registers in the JTAG TAP Controller or Shift Register chains. The registers within the JTAG controller are all accessed serially through the TAP Controller. The registers can be broken down into two main categories: Instruction Registers and Data Registers.

### 5.4.1 Instruction Register (IR)

The JTAG TAP Instruction Register (IR) is a four-bit serial scan chain with a parallel load register connected between the JTAG TDI and TDO pins. When the TAP Controller is placed in the correct states, bits can be shifted into the Instruction Register. Once these bits have been shifted into the chain and updated, they are interpreted as the current instruction. The decode of the Instruction Register bits is shown in Table 5-2 on page 50. A detailed explanation of each instruction, along with its associated Data Register, follows.

IR[3:0]	Instruction	Description
0000	EXTEST	Drives the values preloaded into the Boundary Scan Chain by the SAMPLE/PRELOAD instruction onto the pads.
0001	INTEST	Drives the values preloaded into the Boundary Scan Chain by the SAMPLE/PRELOAD instruction into the controller.
0010	SAMPLE / PRELOAD	Captures the current I/O values and shifts the sampled values out of the Boundary Scan Chain while new preload data is shifted in.
1000	ABORT	Shifts data into the ARM Debug Port Abort Register.
1010	DPACC	Shifts data into and out of the ARM DP Access Register.
1011	APACC	Shifts data into and out of the ARM AC Access Register.
1110	IDCODE	Loads manufacturing information defined by the <i>IEEE Standard 1149.1</i> into the IDCODE chain and shifts it out.
1111	BYPASS	Connects TDI to TDO through a single Shift Register chain.
All Others	Reserved	Defaults to the BYPASS instruction to ensure that TDI is always connected to TDO.

#### 5.4.1.1 EXTEST Instruction

The EXTEST instruction does not have an associated Data Register chain. The EXTEST instruction uses the data that has been preloaded into the Boundary Scan Data Register using the SAMPLE/PRELOAD instruction. When the EXTEST instruction is present in the Instruction Register, the preloaded data in the Boundary Scan Data Register associated with the outputs and output enables are used to drive the GPIO pads rather than the signals coming from the core. This allows

tests to be developed that drive known values out of the controller, which can be used to verify connectivity.

#### 5.4.1.2 INTEST Instruction

The INTEST instruction does not have an associated Data Register chain. The INTEST instruction uses the data that has been preloaded into the Boundary Scan Data Register using the SAMPLE/PRELOAD instruction. When the INTEST instruction is present in the Instruction Register, the preloaded data in the Boundary Scan Data Register associated with the inputs are used to drive the signals going into the core rather than the signals coming from the GPIO pads. This allows tests to be developed that drive known values into the controller, which can be used for testing. It is important to note that although the  $\overline{\tt RST}$  input pin is on the Boundary Scan Data Register chain, it is only observable.

#### 5.4.1.3 SAMPLE/PRELOAD Instruction

The SAMPLE/PRELOAD instruction connects the Boundary Scan Data Register chain between TDI and TDO. This instruction samples the current state of the pad pins for observation and preloads new test data. Each GPIO pad has an associated input, output, and output enable signal. When the TAP controller enters the Capture DR state during this instruction, the input, output, and output-enable signals to each of the GPIO pads are captured. These samples are serially shifted out of TDO while the TAP controller is in the Shift DR state and can be used for observation or comparison in various tests.

While these samples of the inputs, outputs, and output enables are being shifted out of the Boundary Scan Data Register, new data is being shifted into the Boundary Scan Data Register from TDI. Once the new data has been shifted into the Boundary Scan Data Register, the data is saved in the parallel load registers when the TAP controller enters the Update DR state. This update of the parallel load register preloads data into the Boundary Scan Data Register that is associated with each input, output, and output enable. This preloaded data can be used with the EXTEST and INTEST instructions to drive data into or out of the controller. Please see "Boundary Scan Data Register" on page 53 for more information.

#### 5.4.1.4 ABORT Instruction

The ABORT instruction connects the associated ABORT Data Register chain between TDI and TDO. This instruction provides read and write access to the ABORT Register of the ARM Debug Access Port (DAP). Shifting the proper data into this Data Register clears various error bits or initiates a DAP abort of a previous request. Please see the "ABORT Data Register" on page 53 for more information.

#### 5.4.1.5 DPACC Instruction

The DPACC instruction connects the associated DPACC Data Register chain between TDI and TDO. This instruction provides read and write access to the DPACC Register of the ARM Debug Access Port (DAP). Shifting the proper data into this register and reading the data output from this register allows read and write access to the ARM debug and status registers. Please see "DPACC Data Register" on page 53 for more information.

#### 5.4.1.6 APACC Instruction

The APACC instruction connects the associated APACC Data Register chain between TDI and TDO. This instruction provides read and write access to the APACC Register of the ARM Debug Access Port (DAP). Shifting the proper data into this register and reading the data output from this register allows read and write access to internal components and buses through the Debug Port. Please see "APACC Data Register" on page 53 for more information.

#### 5.4.1.7 IDCODE Instruction

The IDCODE instruction connects the associated IDCODE Data Register chain between <code>TDI</code> and <code>TDO</code>. This instruction provides information on the manufacturer, part number, and version of the ARM core. This information can be used by testing equipment and debuggers to automatically configure their input and output data streams. IDCODE is the default instruction that is loaded into the JTAG Instruction Register when a power-on-reset (POR) is asserted, <code>TRST</code> is asserted, or the Test-Logic-Reset state is entered. Please see "IDCODE Data Register" on page 52 for more information.

#### 5.4.1.8 BYPASS Instruction

The BYPASS instruction connects the associated BYPASS Data Register chain between TDI and TDO. This instruction is used to create a minimum length serial path between the TDI and TDO ports. The BYPASS Data Register is a single-bit shift register. This instruction improves test efficiency by allowing components that are not needed for a specific test to be bypassed in the JTAG scan chain by loading them with the BYPASS instruction. Please see "BYPASS Data Register" on page 52 for more information.

### 5.4.2 Data Registers

The JTAG module contains six Data Registers. These include: IDCODE, BYPASS, Boundary Scan, APACC, DPACC, and ABORT serial Data Register chains. Each of these Data Registers is discussed in the following sections.

#### 5.4.2.1 IDCODE Data Register

The format for the 32-bit IDCODE Data Register defined by the *IEEE Standard 1149.1* is shown in Figure 5-3 on page 52. The standard requires that every JTAG-compliant device implement either the IDCODE instruction or the BYPASS instruction as the default instruction. The LSB of the IDCODE Data Register is defined to be a 1 to distinguish it from the BYPASS instruction, which has an LSB of 0. This allows auto configuration test tools to determine which instruction is the default instruction.

The major uses of the JTAG port are for manufacturer testing of component assembly, and program development and debug. To facilitate the use of auto-configuration debug tools, the IDCODE instruction outputs a value of 0x3BA00477. This value indicates an ARM Cortex-M3, Version 1 processor. This allows the debuggers to automatically configure themselves to work correctly with the Cortex-M3 during debug.

Figure 5-3. IDCODE Register Format



### 5.4.2.2 BYPASS Data Register

The format for the 1-bit BYPASS Data Register defined by the *IEEE Standard 1149.1* is shown in Figure 5-4 on page 53. The standard requires that every JTAG-compliant device implement either the BYPASS instruction or the IDCODE instruction as the default instruction. The LSB of the BYPASS Data Register is defined to be a 0 to distinguish it from the IDCODE instruction, which has an LSB of 1. This allows auto configuration test tools to determine which instruction is the default instruction.

.

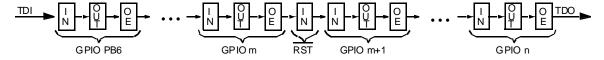
Figure 5-4. BYPASS Register Format

### 5.4.2.3 Boundary Scan Data Register

The format of the Boundary Scan Data Register is shown in Figure 5-5 on page 53. Each GPIO pin, in a counter-clockwise direction from the JTAG port pins, is included in the Boundary Scan Data Register. Each GPIO pin has three associated digital signals that are included in the chain. These signals are input, output, and output enable, and are arranged in that order as can be seen in the figure. In addition to the GPIO pins, the controller reset pin,  $\overline{\text{RST}}$ , is included in the chain. Because the reset pin is always an input, only the input signal is included in the Data Register chain.

When the Boundary Scan Data Register is accessed with the SAMPLE/PRELOAD instruction, the input, output, and output enable from each digital pad are sampled and then shifted out of the chain to be verified. The sampling of these values occurs on the rising edge of <code>TCK</code> in the Capture DR state of the TAP controller. While the sampled data is being shifted out of the Boundary Scan chain in the Shift DR state of the TAP controller, new data can be preloaded into the chain for use with the EXTEST and INTEST instructions. These instructions either force data out of the controller, with the EXTEST instruction, or into the controller, with the INTEST instruction.

Figure 5-5. Boundary Scan Register Format



For detailed information on the order of the input, output, and output enable bits for each of the GPIO ports, please refer to the Stellaris<sup>®</sup> Family Boundary Scan Description Language (BSDL) files, downloadable from www.luminarymicro.com.

### 5.4.2.4 APACC Data Register

The format for the 35-bit APACC Data Register defined by ARM is described in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*.

#### 5.4.2.5 DPACC Data Register

The format for the 35-bit DPACC Data Register defined by ARM is described in the *ARM*® Cortex™-M3 Technical Reference Manual.

### 5.4.2.6 ABORT Data Register

The format for the 35-bit ABORT Data Register defined by ARM is described in the *ARM*® *Cortex*™-*M3 Technical Reference Manual*.

November 29, 2007 53

# 6 System Control

System control determines the overall operation of the device. It provides information about the device, controls the clocking to the core and individual peripherals, and handles reset detection and reporting.

### 6.1 Functional Description

The System Control module provides the following capabilities:

- Device identification, see "Device Identification" on page 54
- Local control, such as reset (see "Reset Control" on page 54), power (see "Power Control" on page 57) and clock control (see "Clock Control" on page 57)
- System control (Run, Sleep, and Deep-Sleep modes), see "System Control" on page 59

#### 6.1.1 Device Identification

Seven read-only registers provide software with information on the microcontroller, such as version, part number, SRAM size, flash size, and other features. See the **DID0**, **DID1**, and **DC0-DC4** registers.

#### 6.1.2 Reset Control

This section discusses aspects of hardware functions during reset as well as system software requirements following the reset sequence.

#### 6.1.2.1 CMOD0 and CMOD1 Test-Mode Control Pins

Two pins, CMOD0 and CMOD1, are defined for use by Luminary Micro for testing the devices during manufacture. They have no end-user function and should not be used. The CMOD pins should be connected to ground.

#### 6.1.2.2 Reset Sources

The controller has five sources of reset:

- External reset input pin (RST) assertion, see "RST Pin Assertion" on page 54.
- 2. Power-on reset (POR), see "Power-On Reset (POR)" on page 55.
- 3. Internal brown-out (BOR) detector, see "Brown-Out Reset (BOR)" on page 55.
- 4. Software-initiated reset (with the software reset registers), see "Software Reset" on page 56.
- 5. A watchdog timer reset condition violation, see "Watchdog Timer Reset" on page 56.

After a reset, the **Reset Cause (RESC)** register is set with the reset cause. The bits in this register are sticky and maintain their state across multiple reset sequences, except when an internal POR is the cause, and then all the other bits in the **RESC** register are cleared except for the POR indicator.

#### 6.1.2.3 RST Pin Assertion

The external reset pin ( $\overline{RST}$ ) resets the controller. This resets the core and all the peripherals except the JTAG TAP controller (see "JTAG Interface" on page 43). The external reset sequence is as follows:

www.DataSheet4U.com

54 November 29, 2007
Preliminary

vww.Da

- 1. The external reset pin (RST) is asserted and then de-asserted.
- 2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, the first instruction designated by the program counter, and begins execution. A few clocks cycles from RST de-assertion to the start of the reset sequence is necessary for synchronization.

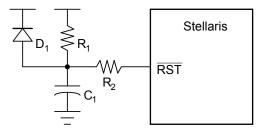
The external reset timing is shown in Figure 20-9 on page 459.

### 6.1.2.4 Power-On Reset (POR)

The Power-On Reset (POR) circuit monitors the power supply voltage ( $V_{DD}$ ). The POR circuit generates a reset signal to the internal logic when the power supply ramp reaches a threshold value ( $V_{TH}$ ). If the application only uses the POR circuit, the  $\overline{\tt RST}$  input needs to be connected to the power supply ( $V_{DD}$ ) through a pull-up resistor (1K to 10K  $\Omega$ ).

The device must be operating within the specified operating parameters at the point when the on-chip power-on reset pulse is complete. The 3.3-V power supply to the device must reach 3.0 V within 10 msec of it crossing 2.0 V to guarantee proper operation. For applications that require the use of an external reset to hold the device in reset longer than the internal POR, the RST input may be used with the circuit as shown in Figure 6-1 on page 55.

Figure 6-1. External Circuitry to Extend Reset



The  $R_1$  and  $C_1$  components define the power-on delay. The  $R_2$  resistor mitigates any leakage from the  $\overline{RST}$  input. The diode (D<sub>1</sub>) discharges  $C_1$  rapidly when the power supply is turned off.

The Power-On Reset sequence is as follows:

- 1. The controller waits for the later of external reset (RST) or internal POR to go inactive.
- 2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, the first instruction designated by the program counter, and begins execution.

The internal POR is only active on the initial power-up of the controller. The Power-On Reset timing is shown in Figure 20-10 on page 459.

Note: The power-on reset also resets the JTAG controller. An external reset does not.

### 6.1.2.5 Brown-Out Reset (BOR)

A drop in the input voltage resulting in the assertion of the internal brown-out detector can be used to reset the controller. This is initially disabled and may be enabled by software.

The system provides a brown-out detection circuit that triggers if the power supply  $(V_{DD})$  drops below a brown-out threshold voltage  $(V_{BTH})$ . If a brown-out condition is detected, the system may generate a controller interrupt or a system reset.

Brown-out resets are controlled with the **Power-On and Brown-Out Reset Control (PBORCTL)** register. The BORIOR bit in the **PBORCTL** register must be set for a brown-out condition to trigger a reset.

The brown-out reset is equivelent to an assertion of the external  $\overline{\mathtt{RST}}$  input and the reset is held active until the proper  $V_{DD}$  level is restored. The **RESC** register can be examined in the reset interrupt handler to determine if a Brown-Out condition was the cause of the reset, thus allowing software to determine what actions are required to recover.

The internal Brown-Out Reset timing is shown in Figure 20-11 on page 459.

#### 6.1.2.6 Software Reset

Software can reset a specific peripheral or generate a reset to the entire system .

Peripherals can be individually reset by software via three registers that control reset signals to each peripheral (see the **SRCRn** registers). If the bit position corresponding to a peripheral is set and subsequently cleared, the peripheral is reset. The encoding of the reset registers is consistent with the encoding of the clock gating control for peripherals and on-chip functions (see "System Control" on page 59). Note that all reset signals for all clocks of the specified unit are asserted as a result of a software-initiated reset.

The entire system can be reset by software by setting the SYSRESETREQ bit in the Cortex-M3 Application Interrupt and Reset Control register resets the entire system including the core. The software-initiated system reset sequence is as follows:

- A software system reset is initiated by writing the SYSRESETREQ bit in the ARM Cortex-M3
   Application Interrupt and Reset Control register.
- 2. An internal reset is asserted.
- The internal reset is deasserted and the controller loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

The software-initiated system reset timing is shown in Figure 20-12 on page 460.

### 6.1.2.7 Watchdog Timer Reset

The watchdog timer module's function is to prevent system hangs. The watchdog timer can be configured to generate an interrupt to the controller on its first time-out, and to generate a reset signal on its second time-out.

After the first time-out event, the 32-bit counter is reloaded with the value of the **Watchdog Timer Load (WDTLOAD)** register, and the timer resumes counting down from that value. If the timer counts down to its zero state again before the first time-out interrupt is cleared, and the reset signal has been enabled, the watchdog timer asserts its reset signal to the system. The watchdog timer reset sequence is as follows:

- 1. The watchdog timer times out for the second time without being serviced.
- 2. An internal reset is asserted.
- The internal reset is released and the controller loads from memory the initial stack pointer, the initial program counter, the first instruction designated by the program counter, and begins execution.

www.DataSheet4U.com

ww DataShoot/III com

The watchdog reset timing is shown in Figure 20-13 on page 460.

### 6.1.3 Power Control

The Stellaris microcontroller provides an integrated LDO regulator that may be used to provide power to the majority of the controller's internal logic. The LDO regulator provides software a mechanism to adjust the regulated value, in small increments (VSTEP), over the range of 2.25 V to 2.75 V (inclusive)—or 2.5 V  $\pm$  10%. The adjustment is made by changing the value of the VADJ field in the **LDO Power Control (LDOPCTL)** register.

Note:

The use of the LDO is optional. The internal logic may be supplied by the on-chip LDO or by an external regulator. If the LDO is used, the LDO output pin is connected to the VDD25 pins on the printed circuit board. The LDO requires decoupling capacitors on the printed circuit board. If an external regulator is used, it is strongly recommended that the external regulator supply the controller only and not be shared with other devices on the printed circuit board.

### 6.1.4 Clock Control

System control determines the control of clocks in this part.

#### 6.1.4.1 Fundamental Clock Sources

There are four clock sources for use in the device:

- Internal Oscillator (IOSC): The internal oscillator is an on-chip clock source. It does not require the use of any external components. The frequency of the internal oscillator is 12 MHz ± 30%. Applications that do not depend on accurate clock sources may use this clock source to reduce system cost. The internal oscillator is the clock source the device uses during and following POR. If the main oscillator is required, software must enable the main oscillator following reset and allow the main oscillator to stabilize before changing the clock reference.
- Main Oscillator: The main oscillator provides a frequency-accurate clock source by one of two means: an external single-ended clock source is connected to the OSC0 input pin, or an external crystal is connected across the OSC0 input and OSC1 output pins. The crystal value allowed depends on whether the main oscillator is used as the clock reference source to the PLL. If so, the crystal must be one of the supported frequencies between 3.579545 MHz through 8.192 MHz (inclusive). If the PLL is not being used, the crystal may be any one of the supported frequencies between 1 MHz and 8.192 MHz. The single-ended clock source range is from DC through the specified speed of the device. The supported crystals are listed in the XTAL bit in the RCC register (see page 70).
- Internal 30-kHz Oscillator: The internal 30-kHz oscillator is similar to the internal oscillator, except that it provides an operational frequency of 30 kHz ± 30%. It is intended for use during Deep-Sleep power-saving modes. This power-savings mode benefits from reduced internal switching and also allows the main oscillator to be powered down.

The internal system clock (sysclk), is derived from any of the four sources plus two others: the output of the internal PLL, and the internal oscillator divided by four (3 MHz  $\pm$  30%). The frequency of the PLL clock reference must be in the range of 3.579545 MHz to 8.192 MHz (inclusive).

The Run-Mode Clock Configuration (RCC) and Run-Mode Clock Configuration 2 (RCC2) registers provide control for the system clock. The RCC2 register is provided to extend fields that offer additional encodings over the RCC register. When used, the RCC2 register field values are

November 29, 2007 57

used by the logic over the corresponding field in the **RCC** register. In particular, **RCC2** provides for a larger assortment of clock configuration options.

### 6.1.4.2 Crystal Configuration for the Main Oscillator (MOSC)

The main oscillator supports the use of a select number of crystals. If the main oscillator is used by the PLL as a reference clock, the supported range of crystals is 3.579545 to 8.192 MHz, otherwise, the range of supported crystals is 1 to 8.192 MHz.

The XTAL bit in the **RCC** register (see page 70) describes the available crystal choices and default programming values.

Software configures the **RCC** register XTAL field with the crystal number. If the PLL is used in the design, the XTAL field value is internally translated to the PLL settings.

### 6.1.4.3 PLL Frequency Configuration

The PLL is disabled by default during power-on reset and is enabled later by software if required. Software configures the PLL input reference clock source, specifies the output divisor to set the system clock frequency, and enables the PLL to drive the output.

If the main oscillator provides the clock reference to the PLL, the translation provided by hardware and used to program the PLL is available for software in the **XTAL to PLL Translation (PLLCFG)** register (see page 74). The internal translation provides a translation within  $\pm$  1% of the targeted PLL VCO frequency.

The Crystal Value field (XTAL) on page 70 describes the available crystal choices and default programming of the **PLLCFG** register. The crystal number is written into the XTAL field of the **Run-Mode Clock Configuration (RCC)** register. Any time the XTAL field changes, the new settings are translated and the internal PLL settings are updated.

#### 6.1.4.4 PLL Modes

The PLL has two modes of operation: Normal and Power-Down

- Normal: The PLL multiplies the input clock reference and drives the output.
- Power-Down: Most of the PLL internal circuitry is disabled and the PLL does not drive the output.

The modes are programmed using the RCC/RCC2 register fields (see page 70 and page 75).

### 6.1.4.5 PLL Operation

If the PLL configuration is changed, the PLL output frequency is unstable until it reconverges (relocks) to the new setting. The time between the configuration change and relock is T<sub>READY</sub> (see Table 20-6 on page 453). During this time, the PLL is not usable as a clock reference.

The PLL is changed by one of the following:

- Change to the XTAL value in the RCC register—writes of the same value do not cause a relock.
- Change in the PLL from Power-Down to Normal mode.

A counter is defined to measure the  $T_{READY}$  requirement. The counter is clocked by the main oscillator. The range of the main oscillator has been taken into account and the down counter is set to 0x1200 (that is, ~600  $\mu$ s at an 8.192 MHz external oscillator clock). Hardware is provided to keep the PLL from being used as a system clock until the  $T_{READY}$  condition is met after one of the

two changes above. It is the user's responsibility to have a stable clock source (like the main oscillator) before the **RCC/RCC2** register is switched to use the PLL.

### 6.1.5 System Control

For power-savings purposes, the **RCGCn**, **SCGCn**, and **DCGCn** registers control the clock gating logic for each peripheral or block in the system while the controller is in Run, Sleep, and Deep-Sleep mode, respectively.

In Run mode, the processor executes code. In Sleep mode, the clock frequency of the active peripherals is unchanged, but the processor is not clocked and therefore no longer executes code. In Deep-Sleep mode, the clock frequency of the active peripherals may change (depending on the Run mode clock configuration) in addition to the processor clock being stopped. An interrupt returns the device to Run mode from one of the sleep modes; the sleep modes are entered on request from the code. Each mode is described in more detail below.

There are four levels of operation for the device defined as:

- Run Mode. Run mode provides normal operation of the processor and all of the peripherals that are currently enabled by the RCGCn registers. The system clock can be any of the available clock sources including the PLL.
- Sleep Mode. Sleep mode is entered by the Cortex-M3 core executing a WFI (Wait for Interrupt) instruction. Any properly configured interrupt event in the system will bring the processor back into Run mode. See the system control NVIC section of the ARM® Cortex™-M3 Technical Reference Manual for more details.
  - In Sleep mode, the Cortex-M3 processor core and the memory subsystem are not clocked. Peripherals are clocked that are enabled in the **SCGCn** register when auto-clock gating is enabled (see the **RCC** register) or the **RCGCn** register when the auto-clock gating is disabled. The system clock has the same source and frequency as that during Run mode.
- Deep-Sleep Mode. Deep-Sleep mode is entered by first writing the Deep Sleep Enable bit in the ARM Cortex-M3 NVIC system control register and then executing a WFI instruction. Any properly configured interrupt event in the system will bring the processor back into Run mode. See the system control NVIC section of the ARM® Cortex™-M3 Technical Reference Manual for more details.

The Cortex-M3 processor core and the memory subsystem are not clocked. Peripherals are clocked that are enabled in the **DCGCn** register when auto-clock gating is enabled (see the **RCC** register) or the **RCGCn** register when auto-clock gating is disabled. The system clock source is the main oscillator by default or the internal oscillator specified in the **DSLPCLKCFG** register if one is enabled. When the **DSLPCLKCFG** register is used, the internal oscillator is powered up, if necessary, and the main oscillator is powered down. If the PLL is running at the time of the WFI instruction, hardware will power the PLL down and override the SYSDIV field of the active **RCC/RCC2** register to be /16 or /64, respectively. When the Deep-Sleep exit event occurs, hardware brings the system clock back to the source and frequency it had at the onset of Deep-Sleep mode before enabling the clocks that had been stopped during the Deep-Sleep duration.

# 6.2 Initialization and Configuration

The PLL is configured using direct register writes to the RCC/RCC2 register. If the RCC2 register is being used, the USERCC2 bit must be set and the appropriate RCC2 bit/field is used. The steps required to successfully change the PLL-based system clock are:

ww DataSheet4U com

- Bypass the PLL and system clock divider by setting the BYPASS bit and clearing the USESYS
  bit in the RCC register. This configures the system to run off a "raw" clock source (using the
  main oscillator or internal oscillator) and allows for the new PLL configuration to be validated
  before switching the system clock to the PLL.
- 2. Select the crystal value (XTAL) and oscillator source (OSCSRC), and clear the PWRDN bit in RCC/RCC2. Setting the XTAL field automatically pulls valid PLL configuration data for the appropriate crystal, and clearing the PWRDN bit powers and enables the PLL and its output.
- 3. Select the desired system divider (SYSDIV) in RCC/RCC2 and set the USESYS bit in RCC. The SYSDIV field determines the system frequency for the microcontroller.
- Wait for the PLL to lock by polling the PLLLRIS bit in the Raw Interrupt Status (RIS) register.
- 5. Enable use of the PLL by clearing the BYPASS bit in RCC/RCC2.

### 6.3 Register Map

Table 6-1 on page 60 lists the System Control registers, grouped by function. The offset listed is a hexadecimal increment to the register's address, relative to the System Control base address of 0x400F.E000.

Note: Spaces in the System Control register space that are not used are reserved for future or internal use by Luminary Micro, Inc. Software should not modify any reserved memory address.

Table 6-1. System Control Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	DID0	RO	-	Device Identification 0	62
0x004	DID1	RO	-	Device Identification 1	78
0x008	DC0	RO	0x003F.001F	Device Capabilities 0	80
0x010	DC1	RO	0x0110.709F	Device Capabilities 1	81
0x014	DC2	RO	0x0707.1011	Device Capabilities 2	83
0x018	DC3	RO	0x0F00.B7C3	Device Capabilities 3	85
0x01C	DC4	RO	0x0000.00FF	Device Capabilities 4	87
0x030	PBORCTL	R/W	0x0000.7FFD	Brown-Out Reset Control	64
0x034	LDOPCTL	R/W	0x0000.0000	LDO Power Control	65
0x040	SRCR0	R/W	0x00000000	Software Reset Control 0	106
0x044	SRCR1	R/W	0x00000000	Software Reset Control 1	107
0x048	SRCR2	R/W	0x00000000	Software Reset Control 2	109
0x050	RIS	RO	0x0000.0000	Raw Interrupt Status	66
0x054	IMC	R/W	0x0000.0000	Interrupt Mask Control	67
0x058	MISC	R/W1C	0x0000.0000	Masked Interrupt Status and Clear	68
0x05C	RESC	R/W	-	Reset Cause	69

60 November 29, 2007

Offset	Name	Туре	Reset	Description	See page
0x060	RCC	R/W	0x07AE.3AD1	Run-Mode Clock Configuration	70
0x064	PLLCFG	RO	-	XTAL to PLL Translation	74
0x070	RCC2	R/W	0x0780.2800	Run-Mode Clock Configuration 2	75
0x100	RCGC0	R/W	0x00000040	Run Mode Clock Gating Control Register 0	88
0x104	RCGC1	R/W	0x00000000	Run Mode Clock Gating Control Register 1	94
0x108	RCGC2	R/W	0x00000000	Run Mode Clock Gating Control Register 2	100
0x110	SCGC0	R/W	0x00000040	Sleep Mode Clock Gating Control Register 0	90
0x114	SCGC1	R/W	0x00000000	Sleep Mode Clock Gating Control Register 1	96
0x118	SCGC2	R/W	0x00000000	Sleep Mode Clock Gating Control Register 2	102
0x120	DCGC0	R/W	0x00000040	Deep Sleep Mode Clock Gating Control Register 0	92
0x124	DCGC1	R/W	0x00000000	Deep Sleep Mode Clock Gating Control Register 1	98
0x128	DCGC2	R/W	0x00000000	Deep Sleep Mode Clock Gating Control Register 2	104
0x144	DSLPCLKCFG	R/W	0x0780.0000	Deep Sleep Clock Configuration	77

**Register Descriptions** 6.4

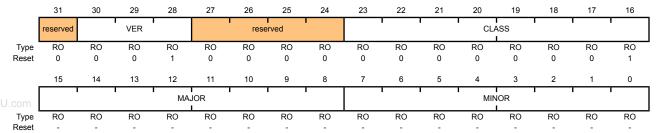
All addresses given are relative to the System Control base address of 0x400F.E000.

### Register 1: Device Identification 0 (DID0), offset 0x000

This register identifies the version of the device.

Device Identification 0 (DID0)

Base 0x400F.E000 Offset 0x000 Type RO, reset -



Bit/Field	Name	Туре	Reset	Description
31	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
30:28	VER	RO	0x1	DID0 Version
				This field defines the $\textbf{DID0}$ register format version. The version number is numeric. The value of the $\mathtt{VER}$ field is encoded as follows:
				Value Description
				0x1 First revision of the <b>DID0</b> register format, for Stellaris® Fury-class devices .
27:24	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
23:16	CLASS	RO	0x1	Device Class

The CLASS field value identifies the internal design from which all mask sets are generated for all devices in a particular product line. The CLASS field value is changed for new product lines, for changes in fab process (for example, a remap or shrink), or any case where the MAJOR OR MINOR fields require differentiation from prior devices. The value of the CLASS field is encoded as follows (all other encodings are reserved):

www.DataSheet4U.com

Value Description

0x0 Stellaris® Sandstorm-class devices.

0x1 Stellaris® Fury-class devices.

Bit/Field	Name	Туре	Reset	Description
15:8	MAJOR	RO	-	Major Revision
				This field specifies the major revision number of the device. The major revision reflects changes to base layers of the design. The major revision number is indicated in the part number as a letter (A for first revision, B for second, and so on). This field is encoded as follows:
				Value Description
				0x0 Revision A (initial device)
				0x1 Revision B (first base layer revision)
				0x2 Revision C (second base layer revision)
				and so on.
7:0	MINOR	RO	-	Minor Revision
				This field specifies the minor revision number of the device. The minor revision reflects changes to the metal layers of the design. The ${\tt MINOR}$ field value is reset when the ${\tt MAJOR}$ field is changed. This field is numeric and is encoded as follows:
				Value Description
				0x0 Initial device, or a major revision update.
				0x1 First metal layer change.
				0x2 Second metal layer change.
				and so on.

November 29, 2007 63

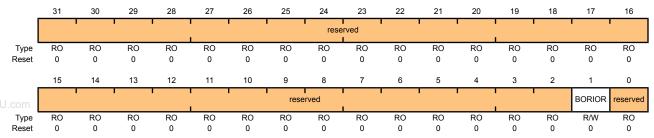
## Register 2: Brown-Out Reset Control (PBORCTL), offset 0x030

This register is responsible for controlling reset conditions after initial power-on reset.

### Brown-Out Reset Control (PBORCTL)

Base 0x400F.E000

Offset 0x030 Type R/W, reset 0x0000.7FFD



Bit/Field	Name	Туре	Reset	Description
31:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORIOR	R/W	0	BOR Interrupt or Reset
				This bit controls how a BOR event is signaled to the controller. If set, a reset is signaled. Otherwise, an interrupt is signaled.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

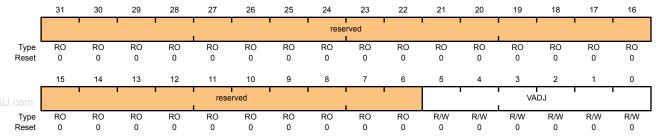
### Register 3: LDO Power Control (LDOPCTL), offset 0x034

The VADJ field in this register adjusts the on-chip output voltage ( $V_{OUT}$ ).

### LDO Power Control (LDOPCTL)

Base 0x400F.E000 Offset 0x034

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:0	VADJ	R/W	0x0	LDO Output Voltage

This field sets the on-chip output voltage. The programming values for the VADJ field are provided below.

Value	$V_{OUT}(V)$
0x00	2.50
0x01	2.45
0x02	2.40
0x03	2.35
0x04	2.30
0x05	2.25
0x06-0x3F	Reserved
0x1B	2.75
0x1C	2.70
0x1D	2.65
0x1E	2.60
0x1F	2.55

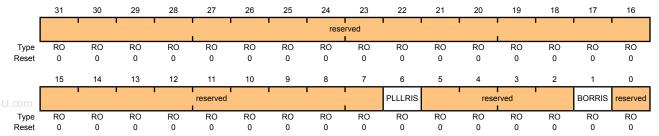
## Register 4: Raw Interrupt Status (RIS), offset 0x050

Central location for system control raw interrupts. These are set and cleared by hardware.

Raw Interrupt Status (RIS)

Base 0x400F.E000 Offset 0x050

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PLLLRIS	RO	0	PLL Lock Raw Interrupt Status
				This bit is set when the PLL $T_{READY}$ Timer asserts.
5:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORRIS	RO	0	Brown-Out Reset Raw Interrupt Status
				This bit is the raw interrupt status for any brown-out conditions. If set, a brown-out condition is currently active. This is an unregistered signal from the brown-out detection circuit. An interrupt is reported if the BORIM bit in the <b>IMC</b> register is set and the BORIOR bit in the <b>PBORCTL</b> register is cleared.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

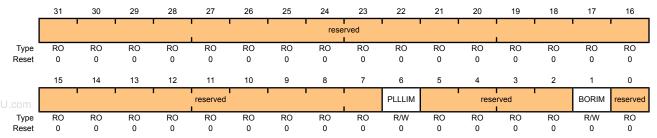
### Register 5: Interrupt Mask Control (IMC), offset 0x054

Central location for system control interrupt masks.

Interrupt Mask Control (IMC)

Base 0x400F.E000

Offset 0x054 Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	PLLLIM	R/W	0	PLL Lock Interrupt Mask
				This bit specifies whether a current limit detection is promoted to a controller interrupt. If set, an interrupt is generated if PLLLRIS in <b>RIS</b> is set; otherwise, an interrupt is not generated.
5:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORIM	R/W	0	Brown-Out Reset Interrupt Mask
				This bit specifies whether a brown-out condition is promoted to a controller interrupt. If set, an interrupt is generated if BORRIS is set; otherwise, an interrupt is not generated.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

## Register 6: Masked Interrupt Status and Clear (MISC), offset 0x058

Central location for system control result of RIS AND IMC to generate an interrupt to the controller. All of the bits are R/W1C and this action also clears the corresponding raw interrupt bit in the RIS register (see page 66).

Masked Interrupt Status and Clear (MISC)

Base 0x400F.E000

Offset 0x058
Type R/W1C, reset 0x0000.0000

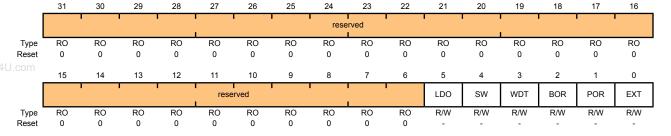
_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'	,				' '	rese	ved	1 1		•			1	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset J.com	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J.00111	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					reserved					PLLLMIS		rese	rved		BORMIS	reserved
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W1C 0	RO 0	RO 0	RO 0	RO 0	R/W1C 0	RO 0
Nosci	Ū	Ü	Ü	U	O	O	Ü	U	Ū	Ü	U	o	Ü	O	O	Ü
Bit/Fi	امام		Name		Type		Reset	Descri	ntion							
וווום	iciu		Name		Турс		Neset	Descri	ption							
31:	7	r	eserved		RO		0	compa	tibility	uld not re with future ross a rea	produ	cts, the v	alue of	a reserv		
6		F	PLLLMIS		R/W1C		0	PLL L	ock Ma	sked Inter	rupt St	atus				
										when the l to this bit	114	EADY time	er asserts	s. The in	terrupt is	cleared
5:2	2	r	eserved		RO		0	compa	tibility '	uld not re with future ross a rea	produ	cts, the v	alue of	a reserv		
1		E	BORMIS		R/W1C		0	BOR N	/lasked	Interrupt	Status					
								The Bo	ORMIS	is simply t	he BOR	ris <b>an[</b>	Ded with	the mas	sk value,	BORIM.
0		ı	eserved		RO		0	compa	tibility	uld not re with future ross a rea	produ	cts, the v	alue of	a reserv		

### Register 7: Reset Cause (RESC), offset 0x05C

This register is set with the reset cause after reset. The bits in this register are sticky and maintain their state across multiple reset sequences, except when an external reset is the cause, and then all the other bits in the **RESC** register are cleared.

Reset Cause (RESC)

Base 0x400F.E000 Offset 0x05C Type R/W, reset -



Bit/Field	Name	Туре	Reset	Description
31:6	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	LDO	R/W	-	LDO Reset
				When set, indicates the LDO circuit has lost regulation and has generated a reset event.
4	SW	R/W	-	Software Reset
				When set, indicates a software reset is the cause of the reset event.
3	WDT	R/W	-	Watchdog Timer Reset
				When set, indicates a watchdog reset is the cause of the reset event.
2	BOR	R/W	-	Brown-Out Reset
				When set, indicates a brown-out reset is the cause of the reset event.
1	POR	R/W	-	Power-On Reset
				When set, indicates a power-on reset is the cause of the reset event.
0	EXT	R/W	-	External Reset
				When set, indicates an external reset (RST assertion) is the cause of

the reset event.

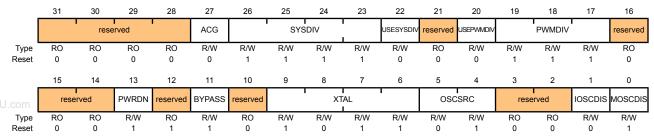
### Register 8: Run-Mode Clock Configuration (RCC), offset 0x060

This register is defined to provide source control and frequency speed.

Run-Mode Clock Configuration (RCC)

Base 0x400F.E000 Offset 0x060

Type R/W, reset 0x07AE.3AD1



Bit/Field	Name	Type	Reset	Description
31:28	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27	ACG	R/W	0	Auto Clock Gating

This bit specifies whether the system uses the Sleep-Mode Clock Gating Control (SCGCn) registers and Deep-Sleep-Mode Clock Gating Control (DCGCn) registers if the controller enters a Sleep or Deep-Sleep mode (respectively). If set, the SCGCn or DCGCn registers are used to control the clocks distributed to the peripherals when the controller is in a sleep mode. Otherwise, the Run-Mode Clock Gating Control (RCGCn) registers are used when the controller enters a sleep mode.

The **RCGCn** registers are always used to control the clocks in Run mode.

This allows peripherals to consume less power when the controller is in a sleep mode and the peripheral is unused.

Bit/Field	Name	Туре	Reset	Description		
26:23	SYSDIV	R/W	0xF	System Clock Divisor		
				Specifies which divisor is use PLL output.	ed to generate the system clock from the	
				The PLL VCO frequency is 400 MHz.		
				Value Divisor (BYPASS=1)	Frequency (BYPASS=0)	
				·	reserved	
				0x1 /2	reserved	
				0x2 /3	reserved	
4U.com				0x3 /4	reserved	
				0x4 /5	reserved	
				0x5 /6	reserved	
				0x6 /7	reserved	
				0x7 /8	25 MHz	
				0x8 /9	22.22 MHz	
				0x9 /10	20 MHz	
				0xA /11	18.18 MHz	
				0xB /12	16.67 MHz	
				0xC /13	15.38 MHz	
				0xD /14	14.29 MHz	
				0xE /15	13.33 MHz	
				0xF /16	12.5 MHz (default)	
				page 70), the SYSDIV value i	Clock Configuration (RCC) register (see is MINSYSDIV if a lower divider was ng used. This lower value is allowed to	
22	USESYSDIV	R/W	0	Enable System Clock Divider		
					as the source for the system clock. The it to be used when the PLL is selected as	
21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.		
20	USEPWMDIV	R/W	0	Enable PWM Clock Divisor		
				Use the PWM clock divider a	s the source for the PWM clock.	

November 29, 2007 71

Bit/Field	Name	Туре	Reset	Description	
19:17	PWMDIV	R/W	0x7	PWM Unit Clock Divisor	
				This field specifies the binary divisor used to predivide the system clock down for use as the timing reference for the PWM module. This clock is only power 2 divide and rising edge is synchronous without phase shift from the system clock.	
				Value Divisor	
				0x0 /2	
				0x1 /4	
				0x2 /8	
				0x3 /16	
				0x4 /32	
				0x5 /64	
				0x6 /64	
				0x7 /64 (default)	
16:14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.	
13	PWRDN	R/W	1	PLL Power Down	
				This bit connects to the PLL PWRDN input. The reset value of 1 powers down the PLL.	
12	reserved	RO	1	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.	
11	BYPASS	R/W	1	PLL Bypass	
				Chooses whether the system clock is derived from the PLL output or the OSC source. If set, the clock that drives the system is the OSC source. Otherwise, the clock that drives the system is the PLL output clock divided by the system divider.	
10	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.	

Bit/Field	Name	Туре	Reset	Description		
9:6	XTAL	R/W	0xB	Crystal Value	e	
					ecifies the crystal value attach this field is provided below.	ned to the main oscillator. The
				Value	Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
				0x0	1.000	reserved
				0x1	1.8432	reserved
				0x2	2.000	reserved
				0x3	2.4576	reserved
Sheet4U.com				0x4	3.5795	545 MHz
				0x5	3.686	64 MHz
				0x6	4	MHz
				0x7	4.09	6 MHz
				8x0	4.915	52 MHz
				0x9	5 1	MHz
				0xA	5.12	2 MHz
				0xB	6 MHz (r	eset value)
				0xC	6.14	4 MHz
				0xD	7.372	28 MHz
				0xE	8 1	MHz
				0xF	8.19	2 MHz
5:4	OSCSRC	R/W	0x1	Oscillator Sc	ource	
				Picks among	g the four input sources for th	e OSC. The values are:
				Value Input	t Source	
				0x0 Main	oscillator (default)	
				0x1 Inter	nal oscillator (default)	
				0x2 Inter	nal oscillator / 4 (this is neces	ssary if used as input to PLL)
				0x3 reser	rved	
3:2	reserved	RO	0x0	compatibility	ould not rely on the value of a with future products, the valu cross a read-modify-write ope	ue of a reserved bit should be
1	IOSCDIS	R/W	0	Internal Osc	illator Disable	
				0: Internal o	scillator (IOSC) is enabled.	
				1: Internal o	scillator is disabled.	
0	MOSCDIS	R/W	1	Main Oscilla	tor Disable	
				0: Main osci	llator is enabled.	
				1: Main osci	llator is disabled (default).	

### Register 9: XTAL to PLL Translation (PLLCFG), offset 0x064

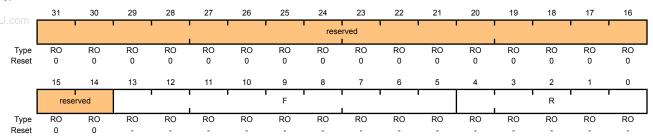
This register provides a means of translating external crystal frequencies into the appropriate PLL settings. This register is initialized during the reset sequence and updated anytime that the XTAL field changes in the **Run-Mode Clock Configuration (RCC)** register (see page 70).

The PLL frequency is calculated using the PLLCFG field values, as follows:

PLLFreq = OSCFreq \* F / (R + 1)

XTAL to PLL Translation (PLLCFG)

Base 0x400F.E000 Offset 0x064 Type RO, reset -



Bit/Field	Name	Type	Reset	Description
31:14	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13:5	F	RO	-	PLL F Value  This field specifies the value supplied to the PLL's F input.
4:0	R	RO	-	PLL R Value

This field specifies the value supplied to the PLL's R input.

www.DataSheet4U.com

### Register 10: Run-Mode Clock Configuration 2 (RCC2), offset 0x070

This register overrides the RCC equivalent register fields when the USERCC2 bit is set. This allows RCC2 to be used to extend the capabilities, while also providing a means to be backward-compatible to previous parts. The fields within the RCC2 register occupy the same bit positions as they do within the RCC register as LSB-justified.

The SYSDIV2 field is wider so that additional larger divisors are possible. This allows a lower system clock frequency for improved Deep Sleep power consumption.

Run-Mode Clock Configuration 2 (RCC2)

Base 0x400F.E000 Offset 0x070

com	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	USERCC2	res	erved			SYS	SDIV2	•					reserved				
Type Reset	R/W 0	RO 0	RO 0	R/W 0	R/W 0	R/W 1	R/W 1	R/W 1	R/W 1	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	reserv	/ed	PWRDN2	reserved	BYPASS2		rese	rved			OSCSRC2		'	rese	rved	•	
Type Reset	RO 0	RO 0	R/W 1	RO 0	R/W 1	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	RO 0	RO 0	RO 0	RO 0	
Bit/Fi	ield		Name		Туре	í	Reset	Descr	iption								
31		l	JSERCC2	2	R/W		0	Use RCC2									
								When set, overrides the <b>RCC</b> register fields.									
30:2	29		reserved		RO		0x0	compa	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.								
28:2	23		SYSDIV2		R/W 0x0F System Clock Divisor												
	.25 STODIVZ R/W (				Specif PLL o		h diviso	r is used	I to gene	erate the	system	clock fr	om th				
								The P	LL VCO	frequer	ncy is 40	0 MHz.					
								addition much the <b>R</b> 0	onal divis lower fre CC regis	sor valu equencion ter sys:	es. This es during DIV enc	permits Deep Soding of	r SYSDIN the syste Sleep mo 1111 pro provides	em clock de. For ovides /1	to be reexample	un at e, whe	
22:′	14		reserved		RO		0x0	compa	atibility w	ith futui	e produ	cts, the v	of a rese alue of a	a reserv			
13	3		PWRDN2	2	R/W		1	Power	r-Down F	PLL							
								When	set, pov	vers do	vn the P	LL.					
12	2		reserved		RO	O Software should not rely on the value of a reserved by compatibility with future products, the value of a reserved across a read-modify-write operation.					a reserv						
11		ı	BYPASS2	2	R/W	1 By			s PLL								
		5117002 1000					When set, bypasses the PLL for the clock source.										

Bit/Field	Name	Туре	Reset	Description
10:7	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6:4	OSCSRC2	R/W	0x0	System Clock Source
				Value Description
				0x0 Main oscillator (MOSC)
				0x1 Internal oscillator (IOSC)
				0x2 Internal oscillator / 4
				0x3 30 kHz internal oscillator
				0x7 32 kHz external oscillator
3:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

### Register 11: Deep Sleep Clock Configuration (DSLPCLKCFG), offset 0x144

This register provides configuration information for the hardware control of Deep Sleep Mode.

Deep Sleep Clock Configuration (DSLPCLKCFG)

Base 0x400F.E000

Offset 0x144 Type R/W, reset 0x0780.0000

_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		reserved			1 I	DSDI\	/ORIDE	· · ·				'	reserved			
Туре	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l.com		•			reserved		1	' '			DSOSCSR	C	'	rese	rved	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/Fi	eld		Name		Туре	I	Reset	Descri	ption							
31:2	29	r	eserved		RO		0x0	compa	itibility w	ith fut	rely on th ure produ read-modi	cts, the	value of a	reserv		
28:2	23	DS	DIVORII	DE	R/W		0x0F	Divide	r Field C	Overrid	le					
								6-bit sy runnin	•	ivider f	field to ove	erride wh	nen Deep	-Sleep	occurs w	ith PLL
22:	7	r	eserved		RO		0x0	compa	itibility w	ith fut	rely on th ure produ read-modi	cts, the	value of a	reserv		
6:4	ļ	DS	SOSCSF	RC	R/W		0x0	Clock	Source							
								When	set, forc	es IO	SC to be	clock sou	urce durir	ng Deep	Sleep r	node.
								Value	Name		Description	1				
								0x0	NOOR	IDE N	lo overrid	e to the	oscillator	clock s	ource is	done
								0x1	IOSC	L	Jse interna	al 12 MF	lz oscillat	or as so	ource	
								0x3	30kHz	L	Jse 30 kH	z interna	ıl oscillato	or		
								0x7	32kHz	L	Jse 32 kH:	z externa	al oscillat	or		
3:0	)	r	eserved		RO		0x0	compa	itibility w	ith fut	rely on th ure produ read-modi	cts, the	value of a	reserv		

## Register 12: Device Identification 1 (DID1), offset 0x004

This register identifies the device family, part number, temperature range, pin count, and package type.

Device Identification 1 (DID1)

Base 0x400F.E000 Offset 0x004 Type RO, reset -

			I ER	I	ı	F	AM	_		1	l	PAR	TNO			ı
Type Reset	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 0	RO 1	RO 0	RO 0	RO 0	RO 1
U.com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ro.com		PINCOUN	T		<u>'</u>	reserved	'	'		TEMP	•	Pł	I (G	ROHS	QL	JAL
Type Reset	RO 0	RO 1	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 0	RO 1	RO 1	RO -	RO -
Bit/F	ield		Name		Туре	1	Reset	Descri	iption							
31:	28		VER		RO		0x1	DID1	Version							
								is num	neric. Th		of the v			ion. The led as fol		
								Value	Descr	iption						
								0x1		evision o class dev		<b>D1</b> regist	ter form	at, indica	ting a S	tellaris
27:	24		FAM		RO		0x0	Family	/							
								Lumin	ary Mic		ct portfo	lio. The		ne device encoded		
								Value	Descr	iption						
								0x0		is family al part n			-	is, all dev 3S.	vices wi	th
23:	16	I	PARTNO	)	RO		0x51	Part N	lumber							
														ce within gs are re		
								Value	Descr	iption						
									LM3S	•						
15:	13	Р	INCOUN	IT	RO		0x2	Packa	ige Pin	Count						
														evice pac		ne value

Value Description

0x2 100-pin package

is encoded as follows (all other encodings are reserved):

	12:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	7:5	TEMP	RO	0x1	Temperature Range
					This field specifies the temperature rating of the device. The value is encoded as follows (all other encodings are reserved):
					Value Description
					0x1 Industrial temperature range (-40°C to 85°C)
www.DataSheet4U.co	<sup>DM</sup> 4:3	PKG	RO	0x1	Package Type
					This field specifies the package type. The value is encoded as follows (all other encodings are reserved):
					Value Description
					0x1 LQFP package
	2	ROHS	RO	1	RoHS-Compliance
					This bit specifies whether the device is RoHS-compliant. A 1 indicates the part is RoHS-compliant.
	1:0	QUAL	RO	-	Qualification Status
					This field specifies the qualification status of the device. The value is encoded as follows (all other encodings are reserved):
					Value Description
					0x0 Engineering Sample (unqualified)
					0x1 Pilot Production (unqualified)
					0x2 Fully Qualified

Bit/Field

Name

Type

Reset

Description

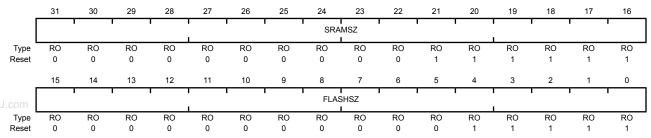
### Register 13: Device Capabilities 0 (DC0), offset 0x008

This register is predefined by the part and can be used to verify features.

Device Capabilities 0 (DC0)

Base 0x400F.E000 Offset 0x008

Type RO, reset 0x003F.001F



Bit/Field	Name	Type	Reset	Description
31:16	SRAMSZ	RO	0x003F	SRAM Size Indicates the size of the on-chip SRAM memory.  Value Description 0x003F 16 KB of SRAM
15:0	FLASHSZ	RO	0x001F	Flash Size

Indicates the size of the on-chip flash memory.

Value Description 0x001F 64 KB of Flash

www.DataSheet4U.com

### Register 14: Device Capabilities 1 (DC1), offset 0x010

This register provides a list of features available in the system. The Stellaris family uses this register format to indicate the availability of the following family features in the specific device: CANs, PWM, ADC, Watchdog timer, Hibernation module, and debug capabilities. This register also indicates the maximum clock frequency and maximum ADC sample rate. The format of this register is consistent with the **RCGC0**, **SCGC0**, and **DCGC0** clock control registers and the **SRCR0** software reset control register.

Device Capabilities 1 (DC1)

Base 0x400F.E000 Offset 0x010

Type RO, reset 0x0110.709F

,																
411.00m	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
4U.com				reserved				CAN0		reserved		PWM		rese	rved	
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 0	RO 0
Reset																-
Г	15	14	13 1	12 I I	11	10	9	8	7	6	5	4	3	2	1	0
		MINS	YSDIV			res	erved		MPU	rese	rved	PLL	WDT	swo	SWD	JTAG
Type Reset	RO 0	RO 1	RO 1	RO 1	RO 0	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 1	RO 1	RO 1	RO 1	RO 1
reset	Ü		•	•	v	Ü	Ü	Ü	•	Ü	Ü	•		•		•
Bit/Fi	iold		Name		Typo		Reset	Dooor	intion							
DIVE	leiu		ivallie		Type		Reset	Descr	iption							
31:2	25	1	reserved		RO		0			uld not re	•					
										vith futur oss a rea					ed bit sh	ould be
								prese	iveu aci	055 a 16	au-moui	ry-write (	operatio	11.		
24	1		CAN0		RO		1	CAN I	Module	0 Preser	nt					
								When	set, ind	icates th	at CAN	unit 0 is	present			
															_	
23:2	21	I	reserved		RO		0			uld not re vith futur						
									•	oss a rea	•				ca bit on	odia be
0.0			D)A/A		D0		4	DIAMA	NA11 -	D						
20	)		PWM		RO		1	PWW	Module	Present						
								When	set, ind	icates th	at the P	WM mod	dule is p	resent.		
19:	16		reserved	ı	RO		0	Softw	are shou	uld not re	elv on th	e value o	of a rese	erved bit.	To prov	ide
								compa	atibility v	vith futur	e produ	cts, the v	alue of	a reserv		
								prese	rved acr	oss a re	ad-modi	fy-write	operatio	n.		
15:	12	М	INSYSD	IV	RO		0x7	Syste	m Clock	Divider						
								Minim	um 4-bi	t divider	value fo	r system	clock.	The rese	t value i	S
										endent.				r how to	change	the
								syster	n clock	divisor u	sing the	SYSDIV	bit.			
								Value	Descri	ption						
								0x7	Specif	ies a 25-	-MHz clo	ock with	a PLL d	ivider of	8.	
									·							
11:	g.		reserved	ı	RO		0	Softw	are shoi	uld not re	alv on th	e value (	of a rece	arved hit	To prov	ide
11.	J		i caci veu	1	NO		U			vith futur						
										oss a rea						

Bit/Field	Name	Туре	Reset	Description
7	MPU	RO	1	MPU Present
				When set, indicates that the Cortex-M3 Memory Protection Unit (MPU) module is present. See the ARM Cortex-M3 Technical Reference Manual for details on the MPU.
6:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	PLL	RO	1	PLL Present
aSheet4U.com				When set, indicates that the on-chip Phase Locked Loop (PLL) is present.
3	WDT	RO	1	Watchdog Timer Present
				When set, indicates that a watchdog timer is present.
2	SWO	RO	1	SWO Trace Port Present
				When set, indicates that the Serial Wire Output (SWO) trace port is present.
1	SWD	RO	1	SWD Present
				When set, indicates that the Serial Wire Debugger (SWD) is present.
0	JTAG	RO	1	JTAG Present
				When set, indicates that the JTAG debugger interface is present.

### Register 15: Device Capabilities 2 (DC2), offset 0x014

This register provides a list of features available in the system. The Stellaris family uses this register format to indicate the availability of the following family features in the specific device: Analog Comparators, General-Purpose Timers, I2Cs, QEIs, SSIs, and UARTs. The format of this register is consistent with the RCGC1, SCGC1, and DCGC1 clock control registers and the SRCR1 software reset control register.

Device Capabilities 2 (DC2)

30

Base 0x400F.E000 Offset 0x014

Type RO, reset 0x0707.1011

_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
Leom			reserved			COMP2	COMP1	COMP0		'	reserved			TIMER2	TIMER1	TIMER0				
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 1	RO 1	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 1	RO 1				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
		reserved		I2C0				reserved		1	'	SSI0		reserved		UART0				
Type Reset	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 0	RO 0	RO 0	RO 1				
Bit/Fi	ield		Name		Type	F	Reset	Descr	iption											
31:2	27	r	eserved		RO		0	compa	atibility v	vith futu	ely on the re produc ad-modif	cts, the v	alue of	a reserv						
26	3	(	COMP2		RO		1	Analo	g Comp	arator 2	Present									
								When	set, ind	icates th	nat analo	g compa	rator 2	is preser	nt.					
25	5	(	COMP1		RO		1	Analo	g Comp	arator 1	Present									
								When set, indicates that analog comparator 1 is present.												
24	1	(	COMP0		RO		1	Analo	g Comp	arator 0	Present									
								When	set, ind	icates th	nat analo	g compa	arator 0	is preser	nt.					
23:	19	r	eserved		RO		0	compa	atibility v	vith futu	ely on the re product ad-modif	cts, the v	alue of	a reserv						
18	3		TIMER2		RO		1	Timer	2 Prese	ent										
								When	set, ind	icates th	nat Gene	ral-Purp	ose Tim	er modu	le 2 is p	resent.				
17	7	7	TIMER1		RO		1	Timer	1 Prese	ent										
								When	set, ind	icates th	nat Gene	ral-Purp	ose Tim	er modu	le 1 is p	resent.				
16	6	7	TIMER0		RO		1	Timer	0 Prese	ent										
								When	set, ind	icates th	nat Gene	ral-Purp	ose Tim	er modu	le 0 is p	resent.				
15:	13	r	eserved		RO		0	compa	atibility v	vith futu	ely on the re product ad-modif	cts, the v	alue of	a reserv						
12	2		I2C0		RO		1	I2C M	odule 0	Present	:									
								When	set, ind	icates th	nat I2C m	odule 0	is prese	ent.						

Bit/Field	Name	Type	Reset	Description
11:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4	SSI0	RO	1	SSI0 Present
				When set, indicates that SSI module 0 is present.
3:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	UART0	RO	1	UART0 Present
eet4U.com				When set, indicates that UART module 0 is present.

www.DataSheet4U.com

### Register 16: Device Capabilities 3 (DC3), offset 0x018

This register provides a list of features available in the system. The Stellaris family uses this register format to indicate the availability of the following family features in the specific device: Analog Comparator I/Os, CCP I/Os, ADC I/Os, and PWM I/Os.

Device Capabilities 3 (DC3)

Base 0x400F.E000

Offset 0x018
Type RO, reset 0x0F00.B7C3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		rese	rved	_	CCP3	CCP2	CCP1	CCP0		'		rese	ved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
J.com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWMFAULT	reserved	C2PLUS	C2MINUS	reserved	C1PLUS	C1MINUS	C0O	C0PLUS	C0MINUS	'	rese	ved		PWM1	PWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	1	1	0	1	1	1	1	1	0	0	0	0	1	1

Bit/Field	Name	Туре	Reset	Description
31:28	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
27	CCP3	RO	1	CCP3 Pin Present  When set, indicates that Capture/Compare/PWM pin 3 is present.
26	CCP2	RO	1	CCP2 Pin Present  When set, indicates that Capture/Compare/PWM pin 2 is present.
25	CCP1	RO	1	CCP1 Pin Present  When set, indicates that Capture/Compare/PWM pin 1 is present.
24	CCP0	RO	1	CCP0 Pin Present  When set, indicates that Capture/Compare/PWM pin 0 is present.
23:16	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	PWMFAULT	RO	1	PWM Fault Pin Present When set, indicates that the PWM Fault pin is present.
14	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
13	C2PLUS	RO	1	C2+ Pin Present  When set, indicates that the analog comparator 2 (+) input pin is present.
12	C2MINUS	RO	1	C2- Pin Present When set, indicates that the analog comparator 2 (-) input pin is present.

Bit/Field	Name	Type	Reset	Description
11	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	C1PLUS	RO	1	C1+ Pin Present
				When set, indicates that the analog comparator 1 (+) input pin is present.
9	C1MINUS	RO	1	C1- Pin Present
				When set, indicates that the analog comparator 1 (-) input pin is present.
8	COO	RO	1	C0o Pin Present
Sheet4U.com				When set, indicates that the analog comparator 0 output pin is present.
7	C0PLUS	RO	1	C0+ Pin Present
				When set, indicates that the analog comparator 0 (+) input pin is present.
6	C0MINUS	RO	1	C0- Pin Present
				When set, indicates that the analog comparator 0 (-) input pin is present.
5:2	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PWM1	RO	1	PWM1 Pin Present
				When set, indicates that the PWM pin 1 is present.
0	PWM0	RO	1	PWM0 Pin Present
				When set, indicates that the PWM pin 0 is present.

### Register 17: Device Capabilities 4 (DC4), offset 0x01C

This register provides a list of features available in the system. The Stellaris family uses this register format to indicate the availability of the following family features in the specific device: Ethernet MAC and PHY, GPIOs, and CCP I/Os. The format of this register is consistent with the RCGC2, SCGC2, and DCGC2 clock control registers and the SRCR2 software reset control register.

Device Capabilities 4 (DC4)

Base 0x400F.E000 Offset 0x01C

0

**GPIOA** 

RO

Offset 0x0 Type RO,		0000.00F	F																
ı	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
		'				•	•	rese	rved										
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0			
ī	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
	·			reser			•	•	GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA			
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 1	RO 1	RO 1	RO 1	RO 1	RO 1	RO 1	RO 1			
Bit/F	ield		Name		Туре		Reset	Descr	iption										
31:	:8	r	reserved		RO		compa	atibility w	ith futur	e produ	cts, the v	of a rese alue of operation	a reserv						
7	•		GPIOH		RO		1	GPIO Port H Present											
	/ GPIUH RU 1								set, indi	cates th	at GPIO	Port H	is presei	nt.					
6	6		GPIOG		RO		1	GPIO	Port G F	Present									
		GPIOG RO						When	set, indi	cates th	at GPIO	Port G	is prese	nt.					
5	5		GPIOF		RO		1	GPIO Port F Present											
								When	set, indi	cates th	at GPIO	Port F i	s preser	nt.					
4	ŀ		GPIOE		RO		1		Port E F										
								When	set, indi	cates th	at GPIO	Port E i	is preser	nt.					
3	3		GPIOD		RO		1		Port D F										
											at GPIO	Port D	is presei	nt.					
2	2		GPIOC		RO		1		Port C F			D 10							
								When set, indicates that GPIO Port C is present.											
1			GPIOB		RO		1		Port B F		ot CDIO	Dort D	ia mraa						
								vvnen	set, indi	cates th	at GPIO	POR B I	is preser	IL.					

GPIO Port A Present

When set, indicates that GPIO Port A is present.

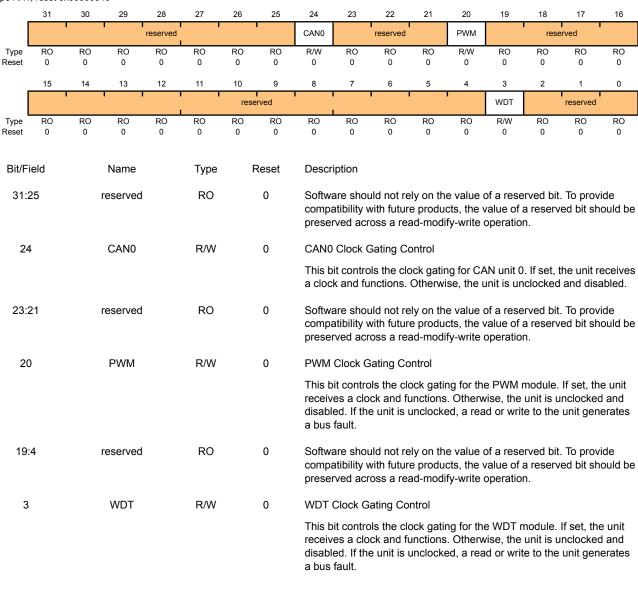
### Register 18: Run Mode Clock Gating Control Register 0 (RCGC0), offset 0x100

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

www.DataSheet4Run Mode Clock Gating Control Register 0 (RCGC0)

Base 0x400F.E000 Offset 0x100

Type R/W, reset 0x00000040



Bit/Field	Name	Туре	Reset	Description
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

ww.DataSheet4U.com

# Register 19: Sleep Mode Clock Gating Control Register 0 (SCGC0), offset 0x110

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

23

22

21

19

17

w.DataSheet4U.com

Sleep Mode Clock Gating Control Register 0 (SCGC0)

29

28

27

26

25

24

Base 0x400F.E000 Offset 0x110 Type R/W, reset 0x00000040

30

ſ		RO RO RO		reserved			1	CAN0		reserved		PWM		res	erved				
Type	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	RO			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
ſ		1	1	_		rese	rved	_		'		•	WDT		reserved				
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	RO			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
Bit/Fi	eld		Name		Туре	F	Reset	Descr	iption										
31:2	25		reserved	l	RO		0	compa	atibility v		e produ	cts, the v	alue of	a reser\	t. To provi ved bit sh				
24			CAN0		R/W		0	CAN0	Clock (	Sating Co	ontrol								
											_	•			the unit re d and disa				
23:2	21		reserved	I	RO		0	compa	atibility v		e produ	cts, the v	alue of	a reser\	t. To provi				
20	)		PWM		R/W		0	PWM	Clock G	ating Co	ntrol								
			PWM								receiv	es a clo ed. If the	ck and fu	ınctions	. Otherw	ise, the	unit is ι	If set, the unclocked unit gene	and
19:	4		reserved	I	RO		0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should b preserved across a read-modify-write operation.											
3			WDT		R/W		0	WDT	Clock G	ating Co	ntrol								
								receiv	es a clo ed. If the	ck and fu	ınctions	. Otherw	ise, the	unit is ι	If set, the unclocked unit gene	and			

Bit/Field	Name	Туре	Reset	Description
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

ww.DataSheet4U.com

## Register 20: Deep Sleep Mode Clock Gating Control Register 0 (DCGC0), offset 0x120

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC0** is the clock configuration register for running operation, **SCGC0** for Sleep operation, and **DCGC0** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

w.DataSheet4U.com

Deep Sleep Mode Clock Gating Control Register 0 (DCGC0)

Base 0x400F.E000 Offset 0x120

Type R/W, reset 0x00000040

_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
		1	' '	reserved			'	CAN0		reserved		PWM		res	erved				
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	RO 0			
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
		1	' '			rese	erved			' '		1	WDT		reserved				
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0			
Bit/Fi	ield		Name		Туре	F	Reset	Descr	iption										
31:2	25	I	reserved		RO		0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.											
24	ŀ		CAN0		R/W		0	CAN0	Clock	Gating Co	ntrol								
								This bit controls the clock gating for CAN unit 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled.											
23:2	21	1	reserved		RO		0	compa	atibility	uld not re with future ross a rea	produ	cts, the v	alue of	a reserv					
20	)		PWM		R/W		0	PWM	Clock C	Gating Co	ntrol								
		This recei								ols the clo ock and fu ie unit is u	nctions	. Otherw	ise, the	unit is ı	unclocked	and			
19:	4	reserved RO 0 Software should not rel compatibility with future preserved across a rea									produ	cts, the v	alue of	a reser					
3			WDT		R/W		0	WDT	Clock G	Sating Cor	ntrol								
								receiv	es a clo ed. If th	ols the clo ock and fu e unit is u	nctions	. Otherw	ise, the	unit is ı	unclocked	and			

Bit/Field	Name	Туре	Reset	Description
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

ww.DataSheet4U.com

#### Register 21: Run Mode Clock Gating Control Register 1 (RCGC1), offset 0x104

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

www.DataSheet4 Run Mode Clock Gating Control Register 1 (RCGC1)

Base 0x400F.E000 Offset 0x104 Type R/W, reset 0x00000000

23:19

			reserved			COMP2	COMP1	COMP0			reserved			TIMER2	TIMER1	TIMER0			
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0			
Reset		-			-		-	-	-	-			-	-	-				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
		reserved		I2C0				reserved				SSI0		reserved		UART0			
Type Reset	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	R/W 0			
110001	ŭ	ŭ	ŭ		· ·	Ū	·	Ü	Ü		ŭ	Ü	Ü	ŭ	Ü	Ü			
Bit/Fi	اماط		Name		Туре		Reset	Descri	ntion										
ווווט	iciu		INAITIC		турс	'	(CSCI	Descri	ption										
31:2	27	re	eserved		RO		0		Software should not rely on the value of a reserved bit. To provide										
								compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.											
								preser	veu acio	)33 a 166	au-moun	iy-wiile (	peralio	11.					
26	6	(	COMP2		R/W		0	Analog Comparator 2 Clock Gating											
		COMP2							t controls										
									es a cloc										
								disabled. If the unit is unclocked, reads or writes to the unit will general a bus fault.								enerate			
	_								_										
25	5	(	COMP1		R/W		0	Analog	g Compa	rator 1	Clock G	ating							
									t controls										
								receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate											
								a bus t				u,			9	0.10.010			
24	ı	(	COMP0		R/W		0	Analog	g Compa	rator 0 (	Clock G	atina							
24	•		JOIVIFU		IX/VV		U					Ū							
									t controls										
									ed. If the										
								a bus t	fault.										

Software should not rely on the value of a reserved bit. To provide

preserved across a read-modify-write operation.

compatibility with future products, the value of a reserved bit should be

RΩ

reserved

0

	Bit/Field	Name	Туре	Reset	Description
	18	TIMER2	R/W	0	Timer 2 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	17	TIMER1	R/W	0	Timer 1 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
v.DataSheet4U	.com 16	TIMER0	R/W	0	Timer 0 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	15:13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	12	I2C0	R/W	0	I2C0 Clock Gating Control
					This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	11:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	4	SSI0	R/W	0	SSI0 Clock Gating Control
					This bit controls the clock gating for SSI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	3:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	0	UART0	R/W	0	UART0 Clock Gating Control
					This bit controls the clock gating for UART module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

www.DataSheet4U.com

## Register 22: Sleep Mode Clock Gating Control Register 1 (SCGC1), offset 0x114

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

w.DataSheet4U.com

Sleep Mode Clock Gating Control Register 1 (SCGC1)

Base 0x400F.E000 Offset 0x114

Type R/W, reset 0x00000000

			reserved			COMP2	COMP1	COMP0			reserved			TIMER2	TIMER1	TIMER0				
Туре	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
		reserved		I2C0			•	reserved				SSI0		reserved		UART0				
Type Reset	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	R/W 0				
Bit/Fi	ield		Name		Туре	F	Reset	Descr	iption											
31:2	27	compatibility with future preserved across a read									e produ	cts, the v	alue of	a reserv						
26	6	C	COMP2	MP2 R/W 0 Analog Comparator								arator 2 Clock Gating								
								receiv	es a clo ed. If the	ck and fu	unctions	. Otherw	ise, the	nparator unit is u s to the u	nclocked	d and				
25	5	C	COMP1		R/W		0	Analo	g Compa	arator 1	or 1 Clock Gating									
								receiv	es a clo ed. If the	ck and fu	unctions	. Otherw	ise, the	nparator unit is u s to the u	nclocked	d and				
24	,	C	OMP0		R/W		0	Analo	g Compa	arator 0	Clock G	ating								
								receiv disabl	This bit controls the clock gating for receives a clock and functions. Other disabled. If the unit is unclocked, read a bus fault.					unit is u	nclocked	d and				
23:	19	re	eserved		RO		0	compa	atibility w		e produ	cts, the v	alue of	erved bit. a reserv n.	•					

Bit/F	ield	Name	Type R	eset	Description
18	3	TIMER2	R/W	0	Timer 2 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
17	7	TIMER1	R/W	0	Timer 1 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
v.DataSheet4U.com 16	6	TIMER0	R/W	0	Timer 0 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
15:	13 r	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
12	2	I2C0	R/W	0	I2C0 Clock Gating Control
					This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
11:	:5 r	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
4		SSI0	R/W	0	SSI0 Clock Gating Control
					This bit controls the clock gating for SSI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
3:	1 r	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0		UART0	R/W	0	UART0 Clock Gating Control
					This bit controls the clock gating for UART module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

November 29, 2007 97

## Register 23: Deep Sleep Mode Clock Gating Control Register 1 (DCGC1), offset 0x124

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC1** is the clock configuration register for running operation, **SCGC1** for Sleep operation, and **DCGC1** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

w.DataSheet4U.com

Deep Sleep Mode Clock Gating Control Register 1 (DCGC1)

Base 0x400F.E000 Offset 0x124 Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		1	reserved			COMP2	COMP1	COMP0			reserved			TIMER2	TIMER1	TIMER0
Туре	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		reserved		I2C0				reserved				SSI0		reserved		UART0
Туре	RO	RO	RO	R/W	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	RO	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:27	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
26	COMP2	R/W	0	Analog Comparator 2 Clock Gating
				This bit controls the clock gating for analog comparator 2. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
25	COMP1	R/W	0	Analog Comparator 1 Clock Gating
				This bit controls the clock gating for analog comparator 1. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
24	COMP0	R/W	0	Analog Comparator 0 Clock Gating
				This bit controls the clock gating for analog comparator 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
23:19	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

	Bit/Field	Name	Туре	Reset	Description
	18	TIMER2	R/W	0	Timer 2 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	17	TIMER1	R/W	0	Timer 1 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
v.DataSheet4U	.com 16	TIMER0	R/W	0	Timer 0 Clock Gating Control
					This bit controls the clock gating for General-Purpose Timer module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	15:13	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	12	I2C0	R/W	0	I2C0 Clock Gating Control
					This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	11:5	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	4	SSI0	R/W	0	SSI0 Clock Gating Control
					This bit controls the clock gating for SSI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	3:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	0	UART0	R/W	0	UART0 Clock Gating Control
					This bit controls the clock gating for UART module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

#### Register 24: Run Mode Clock Gating Control Register 2 (RCGC2), offset 0x108

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. RCGC2 is the clock configuration register for running operation, SCGC2 for Sleep operation, and DCGC2 for Deep-Sleep operation. Setting the ACG bit in the Run-Mode Clock Configuration (RCC) register specifies that the system uses sleep modes.

www.DataSheet4Run Mode Clock Gating Control Register 2 (RCGC2)

Base 0x400F.E000 Offset 0x108 Type R/W, reset 0x00000000

5

**GPIOF** 

**GPIOE** 

R/W

R/W

0

								rese	rvea •							
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				rese	erved				GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Туре	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/F 31:			Name reserved		Type RO	l	Reset 0	Descr Softw		ıld not re	elv on the	e value o	of a rese	erved bit.	. To prov	ride
<b>.</b>							·	compa	atibility v	vith futur oss a re	e produ	cts, the v	alue of	a reserv		
7			GPIOH		R/W		0	Port F	Clock (	Gating C	ontrol					
								clock	and fund	ols the cleations. O	therwise	the un	it is uncl	ocked a	nd disab	led. If
6			GPIOG		R/W		0	Port C	Clock (	Gating C	ontrol					
								clock	and fund	ols the clotions. O	therwise	the un	it is uncl	ocked a	nd disab	led. If

Port F Clock Gating Control

Port E Clock Gating Control

This bit controls the clock gating for Port E. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

This bit controls the clock gating for Port F. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

www.DataSheet4U.com

Bit/	Field	Name	Туре	Reset	Description
	3	GPIOD	R/W	0	Port D Clock Gating Control
					This bit controls the clock gating for Port D. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	2	GPIOC	R/W	0	Port C Clock Gating Control
					This bit controls the clock gating for Port C. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	1	GPIOB	R/W	0	Port B Clock Gating Control
vw.DataSheet4U.com					This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	0	GPIOA	R/W	0	Port A Clock Gating Control
					This bit controls the clock gating for Port A. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

#### Register 25: Sleep Mode Clock Gating Control Register 2 (SCGC2), offset 0x118

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. RCGC2 is the clock configuration register for running operation, SCGC2 for Sleep operation, and DCGC2 for Deep-Sleep operation. Setting the ACG bit in the Run-Mode Clock Configuration (RCC) register specifies that the system uses sleep modes.

Sleep Mode Clock Gating Control Register 2 (SCGC2)

Base 0x400F.E000 Offset 0x118

Type R/W, reset 0x00000000

ypc i d v v	, 10301 0	X000000	,,													
_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'	'	'	1		'	rese	rved							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		'	'	rese	rved •		'	'	GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	RO	RO	RO	RO	RO	RO	RO	RO	R/W							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
INCOCE	U	U	U	U	U	U	U	U	U	U	0	U	U	U	0	0

Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	GPIOH	R/W	0	Port H Clock Gating Control
				This bit controls the clock gating for Port H. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
6	GPIOG	R/W	0	Port G Clock Gating Control
				This bit controls the clock gating for Port G. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
5	GPIOF	R/W	0	Port F Clock Gating Control
				This bit controls the clock gating for Port F. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
4	GPIOE	R/W	0	Port E Clock Gating Control
				This bit controls the clock gating for Port E. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If

the unit is unclocked, reads or writes to the unit will generate a bus fault.

Bit/	Field	Name	Туре	Reset	Description
	3	GPIOD	R/W	0	Port D Clock Gating Control
					This bit controls the clock gating for Port D. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	2	GPIOC	R/W	0	Port C Clock Gating Control
					This bit controls the clock gating for Port C. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	1	GPIOB	R/W	0	Port B Clock Gating Control
ww.DataSheet4U.com					This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	0	GPIOA	R/W	0	Port A Clock Gating Control
					This bit controls the clock gating for Port A. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

# Register 26: Deep Sleep Mode Clock Gating Control Register 2 (DCGC2), offset 0x128

This register controls the clock gating logic. Each bit controls a clock enable for a given interface, function, or unit. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled (saving power). If the unit is unclocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unclocked) unless otherwise noted, so that all functional units are disabled. It is the responsibility of software to enable the ports necessary for the application. Note that these registers may contain more bits than there are interfaces, functions, or units to control. This is to assure reasonable code compatibility with other family and future parts. **RCGC2** is the clock configuration register for running operation, **SCGC2** for Sleep operation, and **DCGC2** for Deep-Sleep operation. Setting the ACG bit in the **Run-Mode Clock Configuration (RCC)** register specifies that the system uses sleep modes.

23

22

21

20

19

18

17

16

w.DataSheet4U.com

Deep Sleep Mode Clock Gating Control Register 2 (DCGC2)

28

27

R/W

R/W

0

0

26

25

24

Base 0x400F.E000 Offset 0x128

5

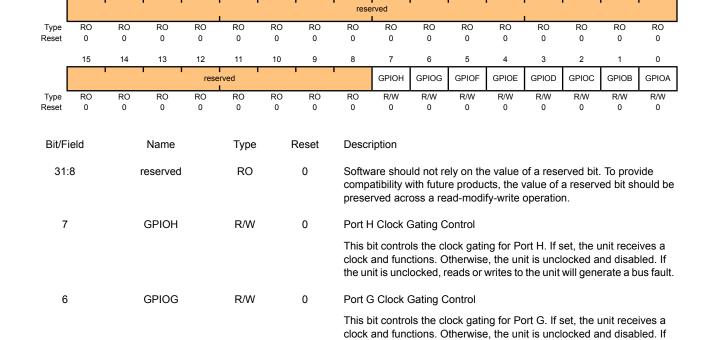
Type R/W, reset 0x00000000

30

29

**GPIOF** 

**GPIOE** 



Port F Clock Gating Control

Port E Clock Gating Control

This bit controls the clock gating for Port E. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

the unit is unclocked, reads or writes to the unit will generate a bus fault.

This bit controls the clock gating for Port F. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

Bit/	Field	Name	Туре	Reset	Description
	3	GPIOD	R/W	0	Port D Clock Gating Control
					This bit controls the clock gating for Port D. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	2	GPIOC	R/W	0	Port C Clock Gating Control
					This bit controls the clock gating for Port C. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	1	GPIOB	R/W	0	Port B Clock Gating Control
ww.DataSheet4U.com					This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.
	0	GPIOA	R/W	0	Port A Clock Gating Control
					This bit controls the clock gating for Port A. If set, the unit receives a clock and functions. Otherwise, the unit is unclocked and disabled. If the unit is unclocked, reads or writes to the unit will generate a bus fault.

## Register 27: Software Reset Control 0 (SRCR0), offset 0x040

Writes to this register are masked by the bits in the **Device Capabilities 1 (DC1)** register.

#### Software Reset Control 0 (SRCR0)

Base 0x400F.E000

Offset 0x040 Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			'	reserved				CAN0		reserved		PWM		rese	rved	
Туре	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	RO	R/W	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U.com			'	_		rese	rved						WDT		reserved	
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0

Bit/Field	Name	Туре	Reset	Description
31:25	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
24	CAN0	R/W	0	CAN0 Reset Control
				Reset control for CAN unit 0.
23:21	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
20	PWM	R/W	0	PWM Reset Control
				Reset control for PWM module.
19:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	WDT	R/W	0	WDT Reset Control
				Reset control for Watchdog unit.
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

## Register 28: Software Reset Control 1 (SRCR1), offset 0x044

Writes to this register are masked by the bits in the **Device Capabilities 2 (DC2)** register.

Software Reset Control 1 (SRCR1)

Base 0x400F.E000

Offset 0x044
Type R/W, reset 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		reserved			COMP2	COMP1	COMP0			reserved			TIMER2	TIMER1	TIMER0
RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	RO	R/W	R/W	R/W
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		I2C0				reserved				SSI0		reserved		UART0
RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	RO 0	RO 0	RO 0	R/W 0
	RO 0 15	RO RO 0 15 14 reserved	reserved	RO   RO   RO   RO   O   O	RO   RO   RO   RO   RO   O   O	RO   RO   RO   RO   RW   RO   RO   RO	reserved         COMP2         COMP1           RO         RO         RO         RO         RW         RW         RW           0	reserved         COMP2         COMP1         COMP0           RO         RO         RO         RO         RW         R/W         R/W         R/W           0	reserved         COMP2         COMP1         COMP0           RO         RO         RO         RO         RW         RW         RW         RW         RO           0	RO	RO         RO<	RO         RO         RO         RO         RO         RO         RW         RW         RW         RW         RO         RO         RO         RO           15         14         13         12         11         10         9         8         7         6         5         4           RO         RO <td< td=""><td>  RO</td><td>  COMP2   COMP1   COMP0   reserved   TIMER2    </td><td>RO         RO         RO&lt;</td></td<>	RO	COMP2   COMP1   COMP0   reserved   TIMER2	RO         RO<

Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit/Fie	ld	Name			Туре	ype Reset			Description								
31:27	7	served		RO	RO 0			Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.									
26		COMP2			R/W		0	Analog Comp 2 Reset Control  Reset control for analog comparator 2.									
25		COMP1			R/W		0	Analog Comp 1 Reset Control  Reset control for analog comparator 1.									
24		С	OMP0		R/W		0	_	Comp 0			arator 0.					
23:19	9	re	served		RO		0	compa		th future	produc	ts, the va	alue of a	reserve	To provided bit sho		
18		TI	IMER2		R/W		0		2 Reset (control fo		al-Purp	ose Time	er modu	le 2.			
17		TIMER1			R/W 0				Timer 1 Reset Control  Reset control for General-Purpose Timer module 1.								
16		TI	IMER0		R/W		0		O Reset (		al-Purp	ose Time	er modu	le 0.			
15:13	3	re	served		RO		0	compa		th future	produc	ts, the va	alue of a	reserve	To provided bit sho		
12			I2C0		R/W		0		teset Cor		nit O.						
11:5		re	served		RO		0	compa		th future	produc	ts, the va	alue of a	reserve	To proviced bit sho		

Bit/Field	Name	Туре	Reset	Description
4	SSI0	R/W	0	SSI0 Reset Control
				Reset control for SSI unit 0.
3:1	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	UART0	R/W	0	UART0 Reset Control
				Reset control for UART unit 0.

ww.DataSheet4U.com

# Register 29: Software Reset Control 2 (SRCR2), offset 0x048

Writes to this register are masked by the bits in the **Device Capabilities 4 (DC4)** register.

## Software Reset Control 2 (SRCR2)

Base 0x400F.E000

Offset 0x048
Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	1/	16
			'	'				rese	rved •							
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U.com			'	rese	rved •				GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Туре	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

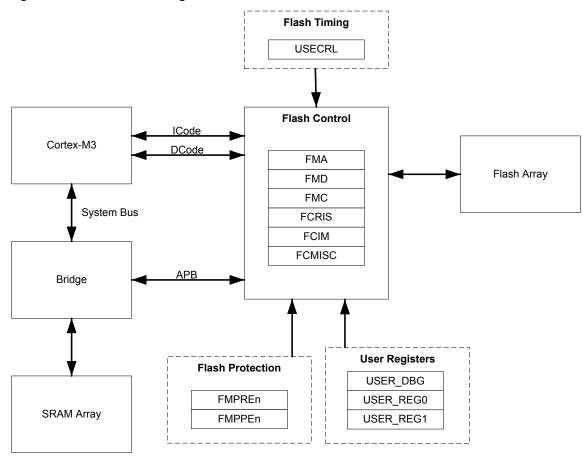
Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	GPIOH	R/W	0	Port H Reset Control
				Reset control for GPIO Port H.
6	GPIOG	R/W	0	Port G Reset Control
				Reset control for GPIO Port G.
5	GPIOF	R/W	0	Port F Reset Control
				Reset control for GPIO Port F.
4	GPIOE	R/W	0	Port E Reset Control
				Reset control for GPIO Port E.
3	GPIOD	R/W	0	Port D Reset Control
				Reset control for GPIO Port D.
2	GPIOC	R/W	0	Port C Reset Control
				Reset control for GPIO Port C.
1	GPIOB	R/W	0	Port B Reset Control
				Reset control for GPIO Port B.
0	GPIOA	R/W	0	Port A Reset Control
				Reset control for GPIO Port A.

# 7 Internal Memory

The LM3S2110 microcontroller comes with 16 KB of bit-banded SRAM and 64 KB of flash memory. The flash controller provides a user-friendly interface, making flash programming a simple task. Flash protection can be applied to the flash memory on a 2-KB block basis.

## 7.1 Block Diagram

Figure 7-1. Flash Block Diagram



# 7.2 Functional Description

This section describes the functionality of both the flash and SRAM memories.

## 7.2.1 SRAM Memory

The internal SRAM of the Stellaris<sup>®</sup> devices is located at address 0x2000.0000 of the device memory map. To reduce the number of time consuming read-modify-write (RMW) operations, ARM has introduced *bit-banding* technology in the Cortex-M3 processor. With a bit-band-enabled processor, certain regions in the memory map (SRAM and peripheral space) can use address aliases to access individual bits in a single, atomic operation.

The bit-band alias is calculated by using the formula:

```
bit-band alias = bit-band base + (byte offset * 32) + (bit number * 4)
```

For example, if bit 3 at address 0x2000.1000 is to be modified, the bit-band alias is calculated as:

```
0x2200.0000 + (0x1000 * 32) + (3 * 4) = 0x2202.000C
```

With the alias address calculated, an instruction performing a read/write to address 0x2202.000C allows direct access to only bit 3 of the byte at address 0x2000.1000.

For details about bit-banding, please refer to Chapter 4, "Memory Map" in the *ARM*® *Cortex*™-*M3 Technical Reference Manual.* 

## 7.2.2 Flash Memory

The flash is organized as a set of 1-KB blocks that can be individually erased. Erasing a block causes the entire contents of the block to be reset to all 1s. An individual 32-bit word can be programmed to change bits that are currently 1 to a 0. These blocks are paired into a set of 2-KB blocks that can be individually protected. The protection allows blocks to be marked as read-only or execute-only, providing different levels of code protection. Read-only blocks cannot be erased or programmed, protecting the contents of those blocks from being modified. Execute-only blocks cannot be erased or programmed, and can only be read by the controller instruction fetch mechanism, protecting the contents of those blocks from being read by either the controller or by a debugger.

See also "Serial Flash Loader" on page 463 for a preprogrammed flash-resident utility used to download code to the flash memory of a device without the use of a debug interface.

## 7.2.2.1 Flash Memory Timing

The timing for the flash is automatically handled by the flash controller. However, in order to do so, it must know the clock rate of the system in order to time its internal signals properly. The number of clock cycles per microsecond must be provided to the flash controller for it to accomplish this timing. It is software's responsibility to keep the flash controller updated with this information via the **USec Reload (USECRL)** register.

On reset, the **USECRL** register is loaded with a value that configures the flash timing so that it works with the maximum clock rate of the part. If software changes the system operating frequency, the new operating frequency minus 1 (in MHz) must be loaded into **USECRL** before any flash modifications are attempted. For example, if the device is operating at a speed of 20 MHz, a value of 0x13 (20-1) must be written to the **USECRL** register.

### 7.2.2.2 Flash Memory Protection

The user is provided two forms of flash protection per 2-KB flash blocks in one pair of 32-bit wide registers. The protection policy for each form is controlled by individual bits (per policy per block) in the **FMPPEn** and **FMPREn** registers.

- Flash Memory Protection Program Enable (FMPPEn): If set, the block may be programmed (written) or erased. If cleared, the block may not be changed.
- Flash Memory Protection Read Enable (FMPREn): If set, the block may be executed or read by software or debuggers. If cleared, the block may only be executed. The contents of the memory block are prohibited from being accessed as data and traversing the DCode bus.

The policies may be combined as shown in Table 7-1 on page 112.

November 29, 2007 111

**Table 7-1. Flash Protection Policy Combinations** 

<b>FMPPE</b> n	FMPREn	Protection
0		Execute-only protection. The block may only be executed and may not be written or erased. This mode is used to protect code.
1	0	The block may be written, erased or executed, but not read. This combination is unlikely to be used.
0		Read-only protection. The block may be read or executed but may not be written or erased. This mode is used to lock the block from further modification while allowing any read or execute access.
1	1	No protection. The block may be written, erased, executed or read.

An access that attempts to program or erase a PE-protected block is prohibited. A controller interrupt may be optionally generated (by setting the AMASK bit in the **FIM** register) to alert software developers of poorly behaving software during the development and debug phases.

An access that attempts to read an RE-protected block is prohibited. Such accesses return data filled with all 0s. A controller interrupt may be optionally generated to alert software developers of poorly behaving software during the development and debug phases.

The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This implements a policy of open access and programmability. The register bits may be changed by writing the specific register bit. The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. Details on programming these bits are discussed in "Nonvolatile Register Programming" on page 113.

# 7.3 Flash Memory Initialization and Configuration

## 7.3.1 Flash Programming

The Stellaris<sup>®</sup> devices provide a user-friendly interface for flash programming. All erase/program operations are handled via three registers: **FMA**, **FMD**, and **FMC**.

# 7.3.1.1 To program a 32-bit word

- Write source data to the FMD register.
- Write the target address to the FMA register.
- Write the flash write key and the WRITE bit (a value of 0xA442.0001) to the FMC register.
- Poll the FMC register until the WRITE bit is cleared.

#### 7.3.1.2 To perform an erase of a 1-KB page

- 1. Write the page address to the **FMA** register.
- Write the flash write key and the ERASE bit (a value of 0xA442.0002) to the FMC register.
- 3. Poll the **FMC** register until the ERASE bit is cleared.

### 7.3.1.3 To perform a mass erase of the flash

- 1. Write the flash write key and the MERASE bit (a value of 0xA442.0004) to the **FMC** register.
- 2. Poll the FMC register until the MERASE bit is cleared.

## 7.3.2 Nonvolatile Register Programming

This section discusses how to update registers that are resident within the flash memory itself. These registers exist in a separate space from the main flash array and are not affected by an ERASE or MASS ERASE operation. These nonvolatile registers are updated by using the COMT bit in the **FMC** register to activate a write operation. For the **USER\_DBG** register, the data to be written must be loaded into the **FMD** register before it is "committed". All other registers are R/W and can have their operation tried before committing them to nonvolatile memory.

Important: These registers can only have bits changed from 1 to 0 by the user and there is no mechanism for the user to erase them back to a 1 value.

In addition, the **USER\_REG0**, **USER\_REG1**, and **USER\_DBG** use bit 31 (NW) of their respective registers to indicate that they are available for user write. These three registers can only be written once whereas the flash protection registers may be written multiple times. Table 7-2 on page 113 provides the FMA address required for commitment of each of the registers and the source of the data to be written when the COMT bit of the **FMC** register is written with a value of 0xA442.0008. After writing the COMT bit, the user may poll the **FMC** register to wait for the commit operation to complete.

Table 7-2. Flash Resident Registers<sup>a</sup>

Register to be Committed	FMA Value	Data Source
FMPRE0	0x0000.0000	FMPRE0
FMPRE1	0x0000.0002	FMPRE1
FMPRE2	0x0000.0004	FMPRE2
FMPRE3	0x0000.0008	FMPRE3
FMPPE0	0x0000.0001	FMPPE0
FMPPE1	0x0000.0003	FMPPE1
FMPPE2	0x0000.0005	FMPPE2
FMPPE3	0x0000.0007	FMPPE3
USER_REG0	0x8000.0000	USER_REG0
USER_REG1	0x8000.0001	USER_REG1
USER_DBG	0x7510.0000	FMD

a. Which FMPREn and FMPPEn registers are available depend on the flash size of your particular Stellaris® device.

# 7.4 Register Map

Table 7-3 on page 113 lists the Flash memory and control registers. The offset listed is a hexadecimal increment to the register's address. The **FMA**, **FMD**, **FMC**, **FCRIS**, **FCIM**, and **FCMISC** registers are relative to the Flash control base address of 0x400F.D000. The **FMPREn**, **FMPPEn**, **USECRL**, **USER\_DBG**, and **USER\_REGn** registers are relative to the System Control base address of 0x400F.E000.

Table 7-3. Flash Register Map

Offset	Name	Туре	Reset	Description	See page
Flash Cor	ntrol Offset				
0x000	FMA	R/W	0x0000.0000	Flash Memory Address	115

 0x000
 FMA
 R/W
 0x0000.0000
 Flash Memory Address
 115

November 29, 2007
Preliminary
Preliminary

Offset	Name	Туре	Reset	Description	See page
0x004	FMD	R/W	0x0000.0000	Flash Memory Data	116
800x0	FMC	R/W	0x0000.0000	Flash Memory Control	117
0x00C	FCRIS	RO	0x0000.0000	Flash Controller Raw Interrupt Status	119
0x010	FCIM	R/W	0x0000.0000	Flash Controller Interrupt Mask	120
0x014	FCMISC	R/W1C	0x0000.0000	Flash Controller Masked Interrupt Status and Clear	121
System C	ontrol Offset				
0x130	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	123
0x200	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	123
0x134	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	124
0x400	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	124
0x140	USECRL	R/W	0x16	USec Reload	122
0x1D0	USER_DBG	R/W	0xFFFF.FFFE	User Debug	125
0x1E0	USER_REG0	R/W	0xFFFF.FFFF	User Register 0	126
0x1E4	USER_REG1	R/W	0xFFFF.FFFF	User Register 1	127
0x204	FMPRE1	R/W	0x0000.0000	Flash Memory Protection Read Enable 1	128
0x208	FMPRE2	R/W	0x0000.0000	Flash Memory Protection Read Enable 2	129
0x20C	FMPRE3	R/W	0x0000.0000	Flash Memory Protection Read Enable 3	130
0x404	FMPPE1	R/W	0x0000.0000	Flash Memory Protection Program Enable 1	131
0x408	FMPPE2	R/W	0x0000.0000	Flash Memory Protection Program Enable 2	132
0x40C	FMPPE3	R/W	0x0000.0000	Flash Memory Protection Program Enable 3	133

# 7.5 Flash Register Descriptions (Flash Control Offset)

The remainder of this section lists and describes the Flash Memory registers, in numerical order by address offset. Registers in this section are relative to the Flash control base address of 0x400F.D000.

## Register 1: Flash Memory Address (FMA), offset 0x000

During a write operation, this register contains a 4-byte-aligned address and specifies where the data is written. During erase operations, this register contains a 1 KB-aligned address and specifies which page is erased. Note that the alignment requirements must be met by software or the results of the operation are unpredictable.

Flash Memory Address (FMA)

Base 0x400F.D000 Offset 0x000

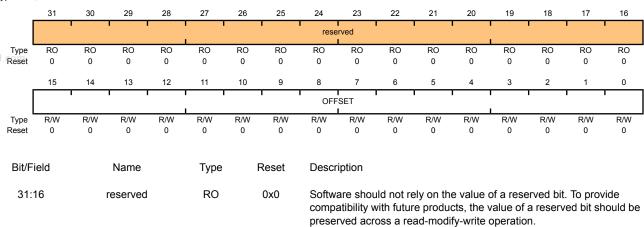
15:0

**OFFSET** 

R/W

0x0

Offset 0x000 Type R/W, reset 0x0000.0000



Address Offset

Address offset in flash where operation is performed, except for nonvolatile registers (see "Nonvolatile Register Programming" on page 113 for details on values for this field).

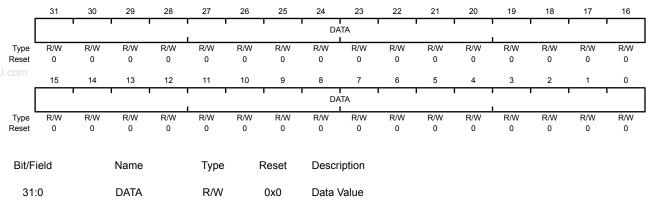
## Register 2: Flash Memory Data (FMD), offset 0x004

This register contains the data to be written during the programming cycle or read during the read cycle. Note that the contents of this register are undefined for a read access of an execute-only block. This register is not used during the erase cycles.

Flash Memory Data (FMD)

Base 0x400F.D000

Offset 0x004 Type R/W, reset 0x0000.0000



Data value for write operation.

## Register 3: Flash Memory Control (FMC), offset 0x008

When this register is written, the flash controller initiates the appropriate access cycle for the location specified by the **Flash Memory Address (FMA)** register (see page 115). If the access is a write access, the data contained in the **Flash Memory Data (FMD)** register (see page 116) is written.

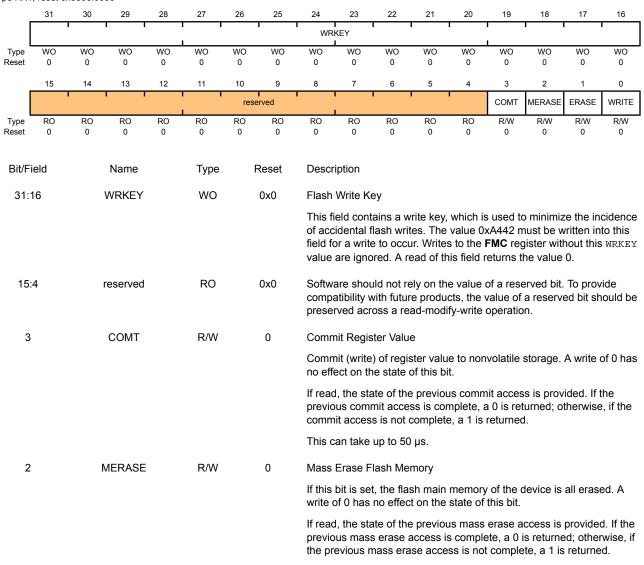
This is the final register written and initiates the memory operation. There are four control bits in the lower byte of this register that, when set, initiate the memory operation. The most used of these register bits are the ERASE and WRITE bits.

It is a programming error to write multiple control bits and the results of such an operation are unpredictable.

#### Flash Memory Control (FMC)

Base 0x400F.D000 Offset 0x008

Type R/W, reset 0x0000.0000



This can take up to 250 ms.

Bit/Field	Name	Туре	Reset	Description
1	ERASE	R/W	0	Erase a Page of Flash Memory
				If this bit is set, the page of flash main memory as specified by the contents of <b>FMA</b> is erased. A write of 0 has no effect on the state of this bit.
				If read, the state of the previous erase access is provided. If the previous erase access is complete, a 0 is returned; otherwise, if the previous erase access is not complete, a 1 is returned.
				This can take up to 25 ms.
0	WRITE	R/W	0	Write a Word into Flash Memory
				If this bit is set, the data stored in <b>FMD</b> is written into the location as specified by the contents of <b>FMA</b> . A write of 0 has no effect on the state of this bit.
				If read, the state of the previous write update is provided. If the previous write access is complete, a 0 is returned; otherwise, if the write access is not complete, a 1 is returned.
				This can take up to 50 µs.

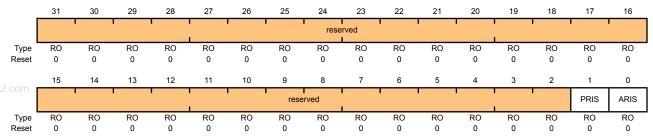
## Register 4: Flash Controller Raw Interrupt Status (FCRIS), offset 0x00C

This register indicates that the flash controller has an interrupt condition. An interrupt is only signaled if the corresponding **FCIM** register bit is set.

Flash Controller Raw Interrupt Status (FCRIS)

Base 0x400F.D000

Offset 0x00C Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PRIS	RO	0	Programming Raw Interrupt Status
				This bit indicates the current state of the programming cycle. If set, the programming cycle completed; if cleared, the programming cycle has not completed. Programming cycles are either write or erase actions generated through the <b>Flash Memory Control (FMC)</b> register bits (see page 117).
0	ARIS	RO	0	Access Raw Interrupt Status

This bit indicates if the flash was improperly accessed. If set, the program tried to access the flash counter to the policy as set in the **Flash Memory Protection Read Enable (FMPREn)** and **Flash Memory Protection Program Enable (FMPPEn)** registers. Otherwise, no access has tried to improperly access the flash.

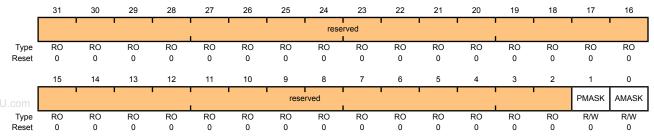
# Register 5: Flash Controller Interrupt Mask (FCIM), offset 0x010

This register controls whether the flash controller generates interrupts to the controller.

Flash Controller Interrupt Mask (FCIM)

Base 0x400F.D000 Offset 0x010

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PMASK	R/W	0	Programming Interrupt Mask
				This bit controls the reporting of the programming raw interrupt status to the controller. If set, a programming-generated interrupt is promoted to the controller. Otherwise, interrupts are recorded but suppressed from the controller.
0	AMASK	R/W	0	Access Interrupt Mask

This bit controls the reporting of the access raw interrupt status to the controller. If set, an access-generated interrupt is promoted to the controller. Otherwise, interrupts are recorded but suppressed from the controller.

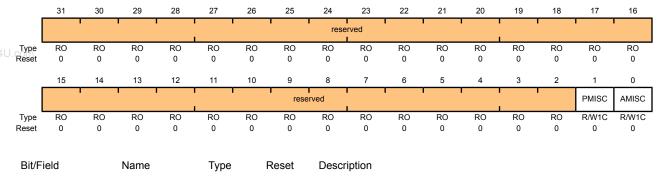
## Register 6: Flash Controller Masked Interrupt Status and Clear (FCMISC), offset 0x014

This register provides two functions. First, it reports the cause of an interrupt by indicating which interrupt source or sources are signalling the interrupt. Second, it serves as the method to clear the interrupt reporting.

Flash Controller Masked Interrupt Status and Clear (FCMISC)

Base 0x400F.D000

Offset 0x014
Type R/W1C, reset 0x0000.0000



31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PMISC	R/W1C	0	Programming Masked Interrupt Status and Clear
				This bit indicates whether an interrupt was signaled because a programming cycle completed and was not masked. This bit is cleared by writing a 1. The PRIS bit in the <b>FCRIS</b> register (see page 119) is also cleared when the PMISC bit is cleared.
0	AMISC	R/W1C	0	Access Masked Interrupt Status and Clear

This bit indicates whether an interrupt was signaled because an improper access was attempted and was not masked. This bit is cleared by writing a 1. The ARIS bit in the FCRIS register is also cleared when the AMISC bit is cleared.

#### 7.6 Flash Register Descriptions (System Control Offset)

The remainder of this section lists and describes the Flash Memory registers, in numerical order by address offset. Registers in this section are relative to the System Control base address of 0x400F.E000.

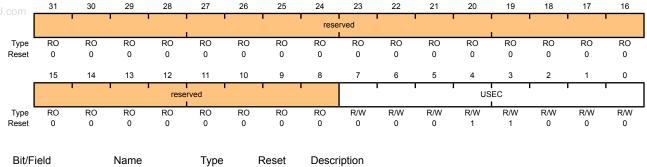
## Register 7: USec Reload (USECRL), offset 0x140

Note: Offset is relative to System Control base address of 0x400F.E000

This register is provided as a means of creating a 1-µs tick divider reload value for the flash controller. The internal flash has specific minimum and maximum requirements on the length of time the high voltage write pulse can be applied. It is required that this register contain the operating frequency (in MHz -1) whenever the flash is being erased or programmed. The user is required to change this value if the clocking conditions are changed for a flash erase/program operation.

USec Reload (USECRL)

Base 0x400F.E000 Offset 0x140 Type R/W, reset 0x16



		• • •		·
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	USEC	R/W	0x18	Microsecond Reload Value

 $\ensuremath{\mathsf{MHz}}$  -1 of the controller clock when the flash is being erased or programmed.

USEC should be set to 0x18 (24 MHz) whenever the flash is being erased or programmed.

# Register 8: Flash Memory Protection Read Enable 0 (FMPRE0), offset 0x130 and 0x200

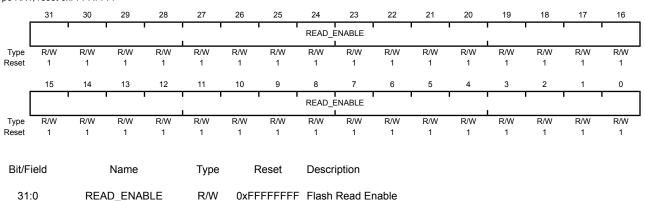
Note: This register is aliased for backwards compatability.

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

#### Flash Memory Protection Read Enable 0 (FMPRE0)

Base 0x400F.D000 Offset 0x130 and 0x200 Type R/W, reset 0xFFFF.FFFF



Enables 2-KB flash blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description

0xFFFFFFF Enables 64 KB of flash.

# Register 9: Flash Memory Protection Program Enable 0 (FMPPE0), offset 0x134 and 0x400

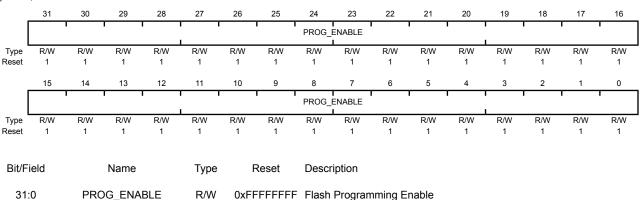
Note: This register is aliased for backwards compatability.

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

#### Flash Memory Protection Program Enable 0 (FMPPE0)

Base 0x400F.D000 Offset 0x134 and 0x400 Type R/W, reset 0xFFFF.FFFF



Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

www.DataSheet4U.com

Value Description

0xFFFFFFF Enables 64 KB of flash.

www.DataSheet4U.com

# Register 10: User Debug (USER\_DBG), offset 0x1D0

**Note:** Offset is relative to System Control base address of 0x400FE000.

This register provides a write-once mechanism to disable external debugger access to the device in addition to 27 additional bits of user-defined data. The DBG0 bit (bit 0) is set to 0 from the factory and the DBG1 bit (bit 1) is set to 1, which enables external debuggers. Changing the DBG1 bit to 0 disables any external debugger access to the device permanently, starting with the next power-up cycle of the device. The NOTWRITTEN bit (bit 31) indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once.

#### User Debug (USER\_DBG)

Base 0x400F.E000 Offset 0x1D0 Type R/W, reset 0xFFF.FFFE

ype K/VV	, reset t	JXFFFF.F	FFE													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NW		1						DATA		1	1		1	1	I
Type Reset	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1
. 10001	15	14	13	12	11	10	9	•	7		•	•	•	2	•	
ī	15	14	1 13	12	<del>''' 1</del>	10	т т	8	, I	6	5 1	4 T	3 I	1	1	0
					ı		DAT	Ά							DBG1	DBG0
Type Reset	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 1	R/W 0
Bit/F	ield		Name		Туре	F	Reset	Descr	ription							
31	1		NW		R/W		1	User	Debug N	lot Writte	en					
								Speci	fies that	this 32-	bit dwor	d has no	t been v	vritten.		
30:	:2		DATA		R/W	0x1F	FFFFFF	User	Data							
									ins the use writter		a value.	This field	d is initia	alized to	all 1s ar	nd can
1			DBG1		R/W		1	Debu	g Contro	l 1						
								The D	BG1 bit r	must be	1 and D	BG0 mus	st be 0 fo	or debug	to be av	/ailable.
0			DBG0		R/W		0	Debu	g Contro	10						
								The D	BG1 bit r	nust be	1 and D	BG0 mus	st be 0 fo	or debug	to be av	/ailable.

## Register 11: User Register 0 (USER\_REG0), offset 0x1E0

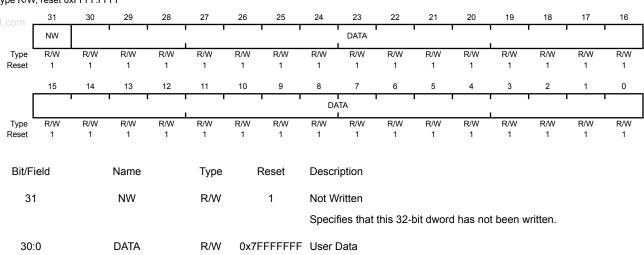
Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be written once. Bit 31 indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device.

User Register 0 (USER\_REG0)

Base 0x400F.E000 Offset 0x1E0

Type R/W, reset 0xFFFF.FFF



Contains the user data value. This field is initialized to all 1s and can only be written once.

## Register 12: User Register 1 (USER REG1), offset 0x1E4

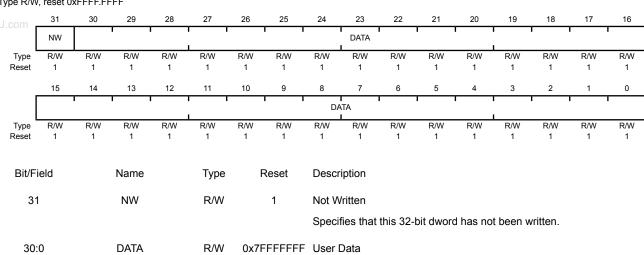
Note: Offset is relative to System Control base address of 0x400FE000.

This register provides 31 bits of user-defined data that is non-volatile and can only be written once. Bit 31 indicates that the register is available to be written and is controlled through hardware to ensure that the register is only written once. The write-once characteristics of this register are useful for keeping static information like communication addresses that need to be unique per part and would otherwise require an external EEPROM or other non-volatile device.

User Register 1 (USER\_REG1)

Base 0x400F.E000 Offset 0x1E4

Type R/W, reset 0xFFFF.FFFF



Contains the user data value. This field is initialized to all 1s and can only be written once.

### Register 13: Flash Memory Protection Read Enable 1 (FMPRE1), offset 0x204

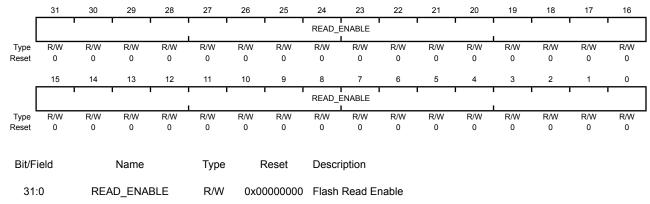
Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Read Enable 1 (FMPRE1)

Base 0x400F.E000 Offset 0x204

Type R/W, reset 0x0000.0000



Enables 2-KB flash blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description

0x00000000 Enables 64 KB of flash.

## Register 14: Flash Memory Protection Read Enable 2 (FMPRE2), offset 0x208

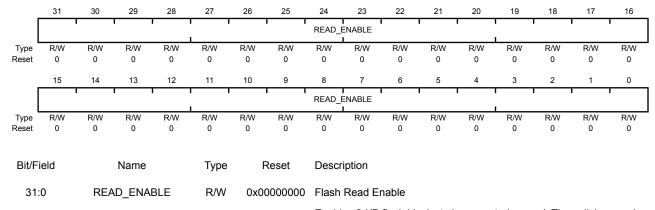
Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Read Enable 2 (FMPRE2)

Base 0x400F.E000 Offset 0x208

Type R/W, reset 0x0000.0000



Enables 2-KB flash blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description

0x00000000 Enables 64 KB of flash.

## Register 15: Flash Memory Protection Read Enable 3 (FMPRE3), offset 0x20C

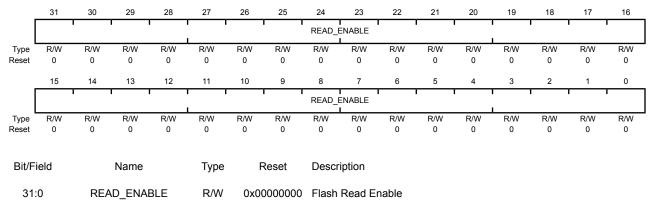
Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the read-only protection bits for each 2-KB flash block (**FMPPEn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Read Enable 3 (FMPRE3)

Base 0x400F.E000 Offset 0x20C

Type R/W, reset 0x0000.0000



Enables 2-KB flash blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description
0x000000000 Enables 64 KB of flash.

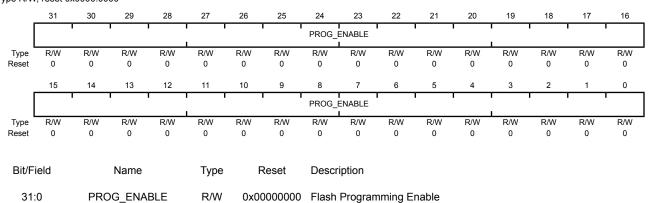
# Register 16: Flash Memory Protection Program Enable 1 (FMPPE1), offset 0x404

**Note:** Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Program Enable 1 (FMPPE1)

Base 0x400F.E000 Offset 0x404 Type R/W, reset 0x0000.0000



Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description

0x00000000 Enables 64 KB of flash.

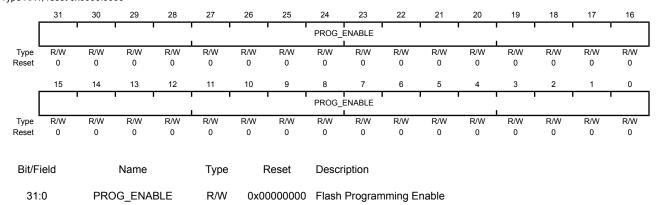
## Register 17: Flash Memory Protection Program Enable 2 (FMPPE2), offset 0x408

Note: Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the FMPREn and FMPPEn registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Program Enable 2 (FMPPE2)

Base 0x400F.E000 Offset 0x408 Type R/W, reset 0x0000.0000



Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value Description

0x00000000 Enables 64 KB of flash.

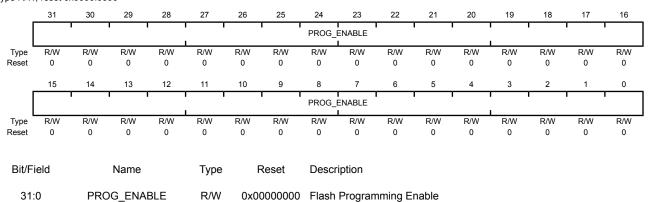
# Register 18: Flash Memory Protection Program Enable 3 (FMPPE3), offset 0x40C

**Note:** Offset is relative to System Control base address of 0x400FE000.

This register stores the execute-only protection bits for each 2-KB flash block (**FMPREn** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPREn** and **FMPPEn** registers are a value of 1 for all implemented banks. This achieves a policy of open access and programmability. The register bits may be changed by writing the specific register bit. However, this register is R/W0; the user can only change the protection bit from a 1 to a 0 (and may NOT change a 0 to a 1). The changes are not permanent until the register is committed (saved), at which point the bit change is permanent. If a bit is changed from a 1 to a 0 and not committed, it may be restored by executing a power-on reset sequence. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Program Enable 3 (FMPPE3)

Base 0x400F.E000 Offset 0x40C Type R/W, reset 0x0000.0000



Configures 2-KB flash blocks to be execute only. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

www.DataSheet4U.com

Value Description

0x00000000 Enables 64 KB of flash.

# 8 General-Purpose Input/Outputs (GPIOs)

The GPIO module is composed of eight physical GPIO blocks, each corresponding to an individual GPIO port (Port A, Port B, Port C, Port D, Port E, Port F, Port G, and Port H). The GPIO module is FiRM-compliant and supports 11-40 programmable input/output pins, depending on the peripherals being used.

The GPIO module has the following features:

- Programmable control for GPIO interrupts
  - Interrupt generation masking
  - Edge-triggered on rising, falling, or both
  - Level-sensitive on High or Low values
- 5-V-tolerant input/outputs
- Bit masking in both read and write operations through address lines
- Programmable control for GPIO pad configuration
  - Weak pull-up or pull-down resistors
  - 2-mA, 4-mA, and 8-mA pad drive
  - Slew rate control for the 8-mA drive
  - Open drain enables
  - Digital input enables

# 8.1 Functional Description

Important: All GPIO pins are tri-stated by default (GPIOAFSEL=0, GPIODEN=0, GPIOPDR=0, and GPIOPUR=0), with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). The JTAG/SWD pins default to their JTAG/SWD functionality (GPIOAFSEL=1, GPIODEN=1 and GPIOPUR=1). A Power-On-Reset (POR) or asserting RST puts both groups of pins back to their default state.

Each GPIO port is a separate hardware instantiation of the same physical block (see Figure 8-1 on page 135). The LM3S2110 microcontroller contains eight ports and thus eight of these physical GPIO blocks.

ww DataSheet4U con

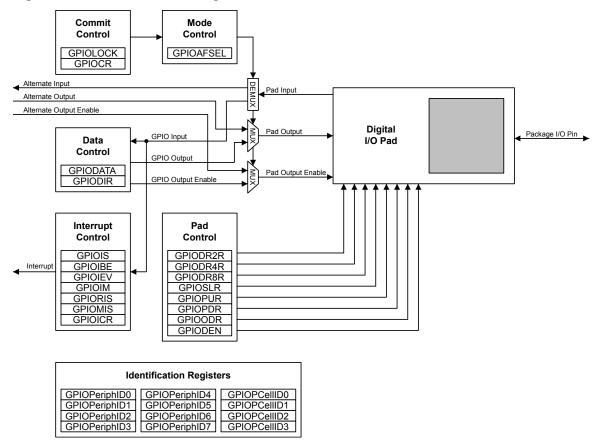


Figure 8-1. GPIO Port Block Diagram

### 8.1.1 Data Control

The data control registers allow software to configure the operational modes of the GPIOs. The data direction register configures the GPIO as an input or an output while the data register either captures incoming data or drives it out to the pads.

## 8.1.1.1 Data Direction Operation

The **GPIO Direction (GPIODIR)** register (see page 142) is used to configure each individual pin as an input or output. When the data direction bit is set to 0, the GPIO is configured as an input and the corresponding data register bit will capture and store the value on the GPIO port. When the data direction bit is set to 1, the GPIO is configured as an output and the corresponding data register bit will be driven out on the GPIO port.

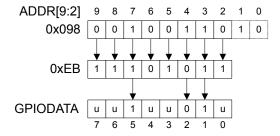
### 8.1.1.2 Data Register Operation

To aid in the efficiency of software, the GPIO ports allow for the modification of individual bits in the **GPIO Data (GPIODATA)** register (see page 141) by using bits [9:2] of the address bus as a mask. This allows software drivers to modify individual GPIO pins in a single instruction, without affecting the state of the other pins. This is in contrast to the "typical" method of doing a read-modify-write operation to set or clear an individual GPIO pin. To accommodate this feature, the **GPIODATA** register covers 256 locations in the memory map.

During a write, if the address bit associated with that data bit is set to 1, the value of the **GPIODATA** register is altered. If it is cleared to 0, it is left unchanged.

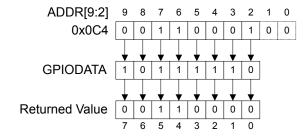
For example, writing a value of 0xEB to the address GPIODATA + 0x098 would yield as shown in Figure 8-2 on page 136, where u is data unchanged by the write.

Figure 8-2. GPIODATA Write Example



During a read, if the address bit associated with the data bit is set to 1, the value is read. If the address bit associated with the data bit is set to 0, it is read as a zero, regardless of its actual value. For example, reading address GPIODATA + 0x0C4 yields as shown in Figure 8-3 on page 136.

Figure 8-3. GPIODATA Read Example



### 8.1.2 Interrupt Control

The interrupt capabilities of each GPIO port are controlled by a set of seven registers. With these registers, it is possible to select the source of the interrupt, its polarity, and the edge properties. When one or more GPIO inputs cause an interrupt, a single interrupt output is sent to the interrupt controller for the entire GPIO port. For edge-triggered interrupts, software must clear the interrupt to enable any further interrupts. For a level-sensitive interrupt, it is assumed that the external source holds the level constant for the interrupt to be recognized by the controller.

Three registers are required to define the edge or sense that causes interrupts:

- **GPIO Interrupt Sense (GPIOIS)** register (see page 143)
- GPIO Interrupt Both Edges (GPIOIBE) register (see page 144)
- GPIO Interrupt Event (GPIOIEV) register (see page 145)

Interrupts are enabled/disabled via the GPIO Interrupt Mask (GPIOIM) register (see page 146).

When an interrupt condition occurs, the state of the interrupt signal can be viewed in two locations: the **GPIO Raw Interrupt Status (GPIORIS)** and **GPIO Masked Interrupt Status (GPIOMIS)** registers (see page 147 and page 148). As the name implies, the **GPIOMIS** register only shows interrupt conditions that are allowed to be passed to the controller. The **GPIORIS** register indicates that a GPIO pin meets the conditions for an interrupt, but has not necessarily been sent to the controller.

Interrupts are cleared by writing a 1 to the GPIO Interrupt Clear (GPIOICR) register (see page 149).

When programming the following interrupt control registers, the interrupts should be masked (**GPIOIM** set to 0). Writing any value to an interrupt control register (**GPIOIS**, **GPIOIBE**, or **GPIOIEV**) can generate a spurious interrupt if the corresponding bits are enabled.

#### 8.1.3 Mode Control

The GPIO pins can be controlled by either hardware or software. When hardware control is enabled via the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 150), the pin state is controlled by its alternate function (that is, the peripheral). Software control corresponds to GPIO mode, where the **GPIODATA** register is used to read/write the corresponding pins.

#### 8.1.4 Commit Control

The commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 150) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 160) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 161) have been set to 1.

#### 8.1.5 Pad Control

The pad control registers allow for GPIO pad configuration by software based on the application requirements. The pad control registers include the **GPIODR2R**, **GPIODR4R**, **GPIODR8R**, **GPIODR8**, **GPIODDR**, **GPIODDR**, **GPIODDR**, and **GPIODDEN** registers.

#### 8.1.6 Identification

The identification registers configured at reset allow software to detect and identify the module as a GPIO block. The identification registers include the **GPIOPeriphID0-GPIOPeriphID7** registers as well as the **GPIOPCeIIID0-GPIOPCeIIID3** registers.

# 8.2 Initialization and Configuration

To use the GPIO, the peripheral clock must be enabled by setting the appropriate GPIO Port bit field (GPIOn) in the RCGC2 register.

On reset, all GPIO pins (except for the five JTAG pins) are configured out of reset to be undriven (tristate): **GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, and **GPIOPUR**=0. Table 8-1 on page 137 shows all possible configurations of the GPIO pads and the control register settings required to achieve them. Table 8-2 on page 138 shows how a rising edge interrupt would be configured for pin 2 of a GPIO port.

**Table 8-1. GPIO Pad Configuration Examples** 

Configuration	GPIO Reg	GPIO Register Bit Value <sup>a</sup>								
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
Digital Input (GPIO)	0	0	0	1	?	?	Х	Х	Х	Х
Digital Output (GPIO)	0	1	0	1	?	?	?	?	?	?
Open Drain Input (GPIO)	0	0	1	1	Х	Х	Х	Х	Х	Х
Open Drain Output (GPIO)	0	1	1	1	Х	Х	?	?	?	?
Open Drain Input/Output (I <sup>2</sup> C)	1	Х	1	1	Х	Х	?	?	?	?

November 29, 2007

Configuration	GPIO Reg	PIO Register Bit Value <sup>a</sup>								
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
Digital Input (Timer CCP)	1	Х	0	1	?	?	Х	Х	Х	Х
Digital Output (PWM)	1	Х	0	1	?	?	?	?	?	?
Digital Output (Timer PWM)	1	Х	0	1	?	?	?	?	?	?
Digital Input/Output (SSI)	1	Х	0	1	?	?	?	?	?	?
Digital Input/Output (UART)	1	Х	0	1	?	?	?	?	?	?
Analog Input (Comparator)	0	0	0	0	0	0	Х	Х	Х	X
Digital Output (Comparator)	1	Х	0	1	?	?	?	?	?	?

a. X=Ignored (don't care bit)

**Table 8-2. GPIO Interrupt Configuration Example** 

Register	Desired Interrupt Event Trigger	Pin 2 Bit Value <sup>a</sup>										
		7	6	5	4	3	2	1	0			
GPIOIS	0=edge 1=level	Х	Х	Х	Х	Х	0	Х	Х			
GPIOIBE	0=single edge 1=both edges	Х	X	X	Х	Х	0	X	Х			
GPIOIEV	0=Low level, or negative edge 1=High level, or positive edge		Х	Х	X	Х	1	X	Х			
GPIOIM	0=masked 1=not masked	0	0	0	0	0	1	0	0			

a. X=Ignored (don't care bit)

# 8.3 Register Map

Table 8-3 on page 139 lists the GPIO registers. The offset listed is a hexadecimal increment to the register's address, relative to that GPIO port's base address:

www.DataSheet4U.com

GPIO Port A: 0x4000.4000

GPIO Port B: 0x4000.5000

GPIO Port C: 0x4000.6000

<sup>?=</sup>Can be either 0 or 1, depending on the configuration

GPIO Port D: 0x4000.7000

GPIO Port E: 0x4002.4000

GPIO Port F: 0x4002.5000

GPIO Port G: 0x4002.6000

GPIO Port H: 0x4002.7000

Important: The GPIO registers in this chapter are duplicated in each GPIO block, however, depending on the block, all eight bits may not be connected to a GPIO pad. In those cases, writing to those unconnected bits has no effect and reading those unconnected bits returns no meaningful data.

Note:

The default reset value for the GPIOAFSEL, GPIOPUR, and GPIODEN registers are 0x0000.0000 for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins default to JTAG/SWD functionality. Because of this, the default reset value of these registers for GPIO Port B is 0x0000.0080 while the default reset value for Port C is 0x0000.000F.

The default register type for the **GPIOCR** register is RO for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins are currently the only GPIOs that are protected by the **GPIOCR** register. Because of this, the register type for GPIO Port B7 and GPIO Port C[3:0] is R/W.

The default reset value for the **GPIOCR** register is 0x0000.00FF for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). To ensure that the JTAG port is not accidentally programmed as a GPIO, these five pins default to non-commitable. Because of this, the default reset value of GPIOCR for GPIO Port B is 0x0000.007F while the default reset value of **GPIOCR** for Port C is 0x0000.00F0.

Table 8-3. GPIO Register Map

Offset	Name	Type	Reset	Description	See page
0x000	GPIODATA	R/W	0x0000.0000	GPIO Data	141
0x400	GPIODIR	R/W	0x0000.0000	GPIO Direction	142
0x404	GPIOIS	R/W	0x0000.0000	GPIO Interrupt Sense	143
0x408	GPIOIBE	R/W	0x0000.0000	GPIO Interrupt Both Edges	144
0x40C	GPIOIEV	R/W	0x0000.0000	GPIO Interrupt Event	145
0x410	GPIOIM	R/W	0x0000.0000	GPIO Interrupt Mask	146
0x414	GPIORIS	RO	0x0000.0000	GPIO Raw Interrupt Status	147
0x418	GPIOMIS	RO	0x0000.0000	GPIO Masked Interrupt Status	148
0x41C	GPIOICR	W1C	0x0000.0000	GPIO Interrupt Clear	149
0x420	GPIOAFSEL	R/W	-	GPIO Alternate Function Select	150
0x500	GPIODR2R	R/W	0x0000.00FF	GPIO 2-mA Drive Select	152
0x504	GPIODR4R	R/W	0x0000.0000	GPIO 4-mA Drive Select	153

Offset	Name	Туре	Reset	Description	See page
0x508	GPIODR8R	R/W	0x0000.0000	GPIO 8-mA Drive Select	154
0x50C	GPIOODR	R/W	0x0000.0000	GPIO Open Drain Select	155
0x510	GPIOPUR	R/W	-	GPIO Pull-Up Select	156
0x514	GPIOPDR	R/W	0x0000.0000	GPIO Pull-Down Select	157
0x518	GPIOSLR	R/W	0x0000.0000	GPIO Slew Rate Control Select	158
0x51C	GPIODEN	R/W	-	GPIO Digital Enable	159
0x520	GPIOLOCK	R/W	0x0000.0001	GPIO Lock	160
0x524	GPIOCR	-	-	GPIO Commit	161
0xFD0	GPIOPeriphID4	RO	0x0000.0000	GPIO Peripheral Identification 4	163
0xFD4	GPIOPeriphID5	RO	0x0000.0000	GPIO Peripheral Identification 5	164
0xFD8	GPIOPeriphID6	RO	0x0000.0000	GPIO Peripheral Identification 6	165
0xFDC	GPIOPeriphID7	RO	0x0000.0000	GPIO Peripheral Identification 7	166
0xFE0	GPIOPeriphID0	RO	0x0000.0061	GPIO Peripheral Identification 0	167
0xFE4	GPIOPeriphID1	RO	0x0000.0000	GPIO Peripheral Identification 1	168
0xFE8	GPIOPeriphID2	RO	0x0000.0018	GPIO Peripheral Identification 2	169
0xFEC	GPIOPeriphID3	RO	0x0000.0001	GPIO Peripheral Identification 3	170
0xFF0	GPIOPCellID0	RO	0x0000.000D	GPIO PrimeCell Identification 0	171
0xFF4	GPIOPCellID1	RO	0x0000.00F0	GPIO PrimeCell Identification 1	172
0xFF8	GPIOPCellID2	RO	0x0000.0005	GPIO PrimeCell Identification 2	173
0xFFC	GPIOPCellID3	RO	0x0000.00B1	GPIO PrimeCell Identification 3	174

8.4 Register Descriptions

The remainder of this section lists and describes the GPIO registers, in numerical order by address offset.

## Register 1: GPIO Data (GPIODATA), offset 0x000

The **GPIODATA** register is the data register. In software control mode, values written in the **GPIODATA** register are transferred onto the GPIO port pins if the respective pins have been configured as outputs through the **GPIO Direction (GPIODIR)** register (see page 142).

In order to write to **GPIODATA**, the corresponding bits in the mask, resulting from the address bus bits [9:2], must be High. Otherwise, the bit values remain unchanged by the write.

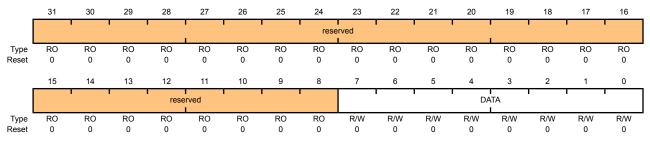
Similarly, the values read from this register are determined for each bit by the mask bit derived from the address used to access the data register, bits [9:2]. Bits that are 1 in the address mask cause the corresponding bits in **GPIODATA** to be read, and bits that are 0 in the address mask cause the corresponding bits in **GPIODATA** to be read as 0, regardless of their value.

A read from **GPIODATA** returns the last bit value written if the respective pins are configured as outputs, or it returns the value on the corresponding input pin when these are configured as inputs. All bits are cleared by a reset.

#### GPIO Data (GPIODATA)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.5000 GPIO Port H base: 0x4002.7000 Offset 0x000

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	R/W	0x00	GPIO Data

This register is virtually mapped to 256 locations in the address space. To facilitate the reading and writing of data to these registers by independent drivers, the data read from and the data written to the registers are masked by the eight address lines  $\mathtt{ipaddr}[9:2]$ . Reads from this register return its current state. Writes to this register only affect bits that are not masked by  $\mathtt{ipaddr}[9:2]$  and are configured as outputs. See "Data Register Operation" on page 135 for examples of reads and writes.

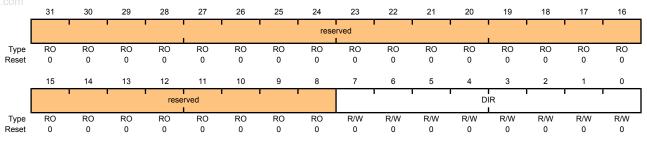
## Register 2: GPIO Direction (GPIODIR), offset 0x400

The **GPIODIR** register is the data direction register. Bits set to 1 in the **GPIODIR** register configure the corresponding pin to be an output, while bits set to 0 configure the pins to be inputs. All bits are cleared by a reset, meaning all GPIO pins are inputs by default.

#### GPIO Direction (GPIODIR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 OFISE 0x400

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DIR	R/W	0x00	GPIO Data Direction

The DIR values are defined as follows:

www.DataSheet4U.com

Value Description

- 0 Pins are inputs.
- Pins are outputs.

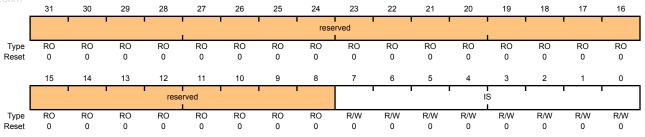
## Register 3: GPIO Interrupt Sense (GPIOIS), offset 0x404

The **GPIOIS** register is the interrupt sense register. Bits set to 1 in **GPIOIS** configure the corresponding pins to detect levels, while bits set to 0 configure the pins to detect edges. All bits are cleared by a reset.

#### GPIO Interrupt Sense (GPIOIS)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x404

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IS	R/W	0x00	GPIO Interrupt Sense

The IS values are defined as follows:

#### Value Description

- 0 Edge on corresponding pin is detected (edge-sensitive).
- 1 Level on corresponding pin is detected (level-sensitive).

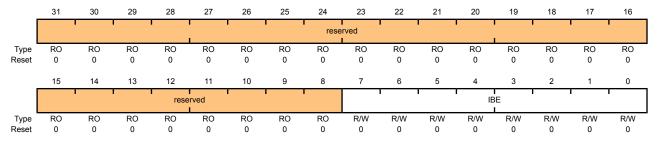
## Register 4: GPIO Interrupt Both Edges (GPIOIBE), offset 0x408

The GPIOIBE register is the interrupt both-edges register. When the corresponding bit in the GPIO Interrupt Sense (GPIOIS) register (see page 143) is set to detect edges, bits set to High in GPIOIBE configure the corresponding pin to detect both rising and falling edges, regardless of the corresponding bit in the GPIO Interrupt Event (GPIOIEV) register (see page 145). Clearing a bit configures the pin to be controlled by **GPIOIEV**. All bits are cleared by a reset.

#### GPIO Interrupt Both Edges (GPIOIBE)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x408

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IBE	R/W	0x00	GPIO Interrupt Both Edges

The IBE values are defined as follows:

#### Value Description

- Interrupt generation is controlled by the GPIO Interrupt Event (GPIOIEV) register (see page 145).
- Both edges on the corresponding pin trigger an interrupt.

Note: Single edge is determined by the corresponding bit in GPIOIEV.

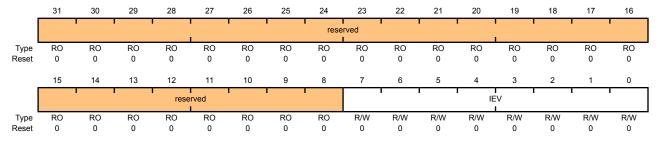
# Register 5: GPIO Interrupt Event (GPIOIEV), offset 0x40C

The **GPIOIEV** register is the interrupt event register. Bits set to High in **GPIOIEV** configure the corresponding pin to detect rising edges or high levels, depending on the corresponding bit value in the GPIO Interrupt Sense (GPIOIS) register (see page 143). Clearing a bit configures the pin to detect falling edges or low levels, depending on the corresponding bit value in GPIOIS. All bits are cleared by a reset.

#### GPIO Interrupt Event (GPIOIEV)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000

Offset 0x40C Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IEV	R/W	0x00	GPIO Interrupt Event

The IEV values are defined as follows:

#### Value Description

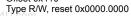
- Falling edge or Low levels on corresponding pins trigger interrupts.
- Rising edge or High levels on corresponding pins trigger interrupts.

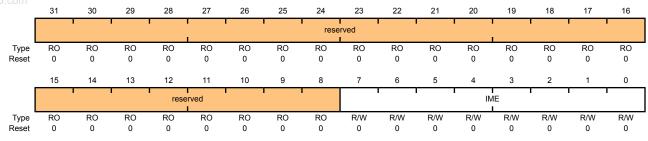
# Register 6: GPIO Interrupt Mask (GPIOIM), offset 0x410

The **GPIOIM** register is the interrupt mask register. Bits set to High in **GPIOIM** allow the corresponding pins to trigger their individual interrupts and the combined GPIOINTR line. Clearing a bit disables interrupt triggering on that pin. All bits are cleared by a reset.

### GPIO Interrupt Mask (GPIOIM)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x410





Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IME	R/W	0x00	GPIO Interrupt Mask Enable

The IME values are defined as follows:

#### Value Description

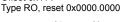
- 0 Corresponding pin interrupt is masked.
- 1 Corresponding pin interrupt is not masked.

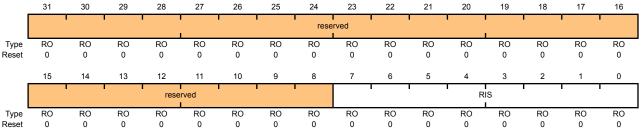
# Register 7: GPIO Raw Interrupt Status (GPIORIS), offset 0x414

The GPIORIS register is the raw interrupt status register. Bits read High in GPIORIS reflect the status of interrupt trigger conditions detected (raw, prior to masking), indicating that all the requirements have been met, before they are finally allowed to trigger by the GPIO Interrupt Mask (GPIOIM) register (see page 146). Bits read as zero indicate that corresponding input pins have not initiated an interrupt. All bits are cleared by a reset.

#### GPIO Raw Interrupt Status (GPIORIS)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x414





Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	RIS	RO	0x00	GPIO Interrupt Raw Status

Reflects the status of interrupt trigger condition detection on pins (raw, prior to masking).

The RIS values are defined as follows:

#### Value Description

- Corresponding pin interrupt requirements not met.
- Corresponding pin interrupt has met requirements.

# Register 8: GPIO Masked Interrupt Status (GPIOMIS), offset 0x418

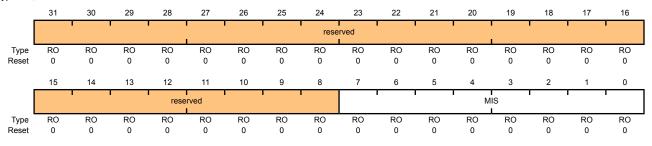
The **GPIOMIS** register is the masked interrupt status register. Bits read High in **GPIOMIS** reflect the status of input lines triggering an interrupt. Bits read as Low indicate that either no interrupt has been generated, or the interrupt is masked.

**GPIOMIS** is the state of the interrupt after masking.

#### GPIO Masked Interrupt Status (GPIOMIS)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x418

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	MIS	RO	0x00	GPIO Masked Interrupt Status

Masked value of interrupt due to corresponding pin.

The MIS values are defined as follows:

#### Value Description

- 0 Corresponding GPIO line interrupt not active.
- 1 Corresponding GPIO line asserting interrupt.

# Register 9: GPIO Interrupt Clear (GPIOICR), offset 0x41C

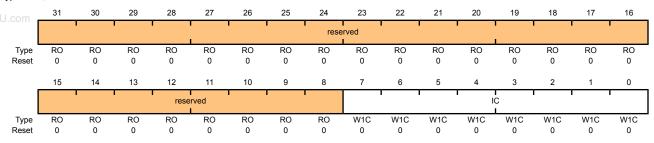
The **GPIOICR** register is the interrupt clear register. Writing a 1 to a bit in this register clears the corresponding interrupt edge detection logic register. Writing a 0 has no effect.

#### GPIO Interrupt Clear (GPIOICR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000

Offset 0x41C

Type W1C, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	IC	W1C	0x00	GPIO Interrupt Clear

The IC values are defined as follows:

#### Value Description

- 0 Corresponding interrupt is unaffected.
- Corresponding interrupt is cleared.

### Register 10: GPIO Alternate Function Select (GPIOAFSEL), offset 0x420

The GPIOAFSEL register is the mode control select register. Writing a 1 to any bit in this register selects the hardware control for the corresponding GPIO line. All bits are cleared by a reset, therefore no GPIO line is set to hardware control by default.

The commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Writes to protected bits of the GPIO Alternate Function Select (GPIOAFSEL) register (see page 150) are not committed to storage unless the GPIO Lock (GPIOLOCK) register (see page 160) has been unlocked and the appropriate bits of the GPIO Commit (GPIOCR) register (see page 161) have been set to 1.

Important: All GPIO pins are tri-stated by default (GPIOAFSEL=0, GPIODEN=0, GPIOPDR=0, and GPIOPUR=0), with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). The JTAG/SWD pins default to their JTAG/SWD functionality (GPIOAFSEL=1. GPIODEN=1 and GPIOPUR=1). A Power-On-Reset (FOR) or asserting RST puts both groups of pins back to their default state.

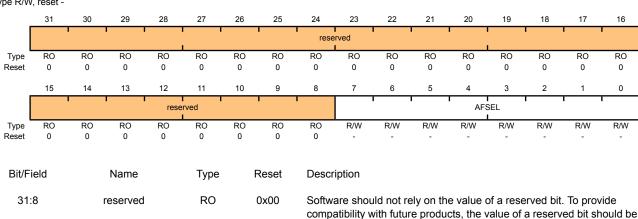
Caution – If the JTAG pins are used as GPIOs in a design, PB7 and PC2 cannot have external pull-down resistors connected to both of them at the same time. If both pins are pulled Low during reset, the controller has unpredictable behavior. If this happens, remove one or both of the pull-down resistors, and apply RST or power-cycle the part.

In addition, it is possible to create a software sequence that prevents the debugger from connecting to the Stellaris® microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. This may lock the debugger out of the part. This can be avoided with a software routine that restores JTAG functionality based on an external or software trigger.

#### GPIO Alternate Function Select (GPIOAFSEL)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000

Offset 0x420 Type R/W, reset



preserved across a read-modify-write operation.

Bit/Field	Name	Type	Reset	Description
7:0	AFSEL	R/W	_	GPIO Alternate Function Select

The AFSEL values are defined as follows:

#### Value Description

- O Software control of corresponding GPIO line (GPIO mode).
- 1 Hardware control of corresponding GPIO line (alternate hardware function).

Note:

The default reset value for the **GPIOAFSEL**, **GPIOPUR**, and **GPIODEN** registers are 0x0000.0000 for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins default to JTAG/SWD functionality. Because of this, the default reset value of these registers for GPIO Port B is 0x0000.0080 while the default reset value for Port C is 0x0000.000F.

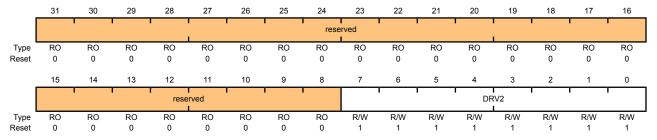
# Register 11: GPIO 2-mA Drive Select (GPIODR2R), offset 0x500

The **GPIODR2R** register is the 2-mA drive control register. It allows for each GPIO signal in the port to be individually configured without affecting the other pads. When writing a DRV2 bit for a GPIO signal, the corresponding DRV4 bit in the **GPIODR4R** register and the DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware.

#### GPIO 2-mA Drive Select (GPIODR2R)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x500

Type R/W, reset 0x0000.00FF



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV2	R/W	0xFF	Output Pad 2-mA Drive Enable

A write of 1 to either **GPIODR4[n]** or **GPIODR8[n]** clears the corresponding 2-mA enable bit. The change is effective on the second clock cycle after the write.

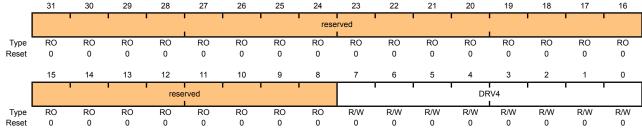
# Register 12: GPIO 4-mA Drive Select (GPIODR4R), offset 0x504

The **GPIODR4R** register is the 4-mA drive control register. It allows for each GPIO signal in the port to be individually configured without affecting the other pads. When writing the DRV4 bit for a GPIO signal, the corresponding DRV2 bit in the **GPIODR2R** register and the DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware.

#### GPIO 4-mA Drive Select (GPIODR4R)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x504

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV4	R/W	0x00	Output Pad 4-mA Drive Enable

A write of 1 to either **GPIODR2[n]** or **GPIODR8[n]** clears the corresponding 4-mA enable bit. The change is effective on the second clock cycle after the write.

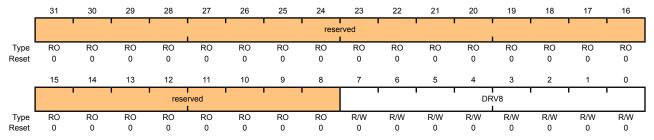
# Register 13: GPIO 8-mA Drive Select (GPIODR8R), offset 0x508

The **GPIODR8R** register is the 8-mA drive control register. It allows for each GPIO signal in the port to be individually configured without affecting the other pads. When writing the DRV8 bit for a GPIO signal, the corresponding DRV2 bit in the **GPIODR2R** register and the DRV4 bit in the **GPIODR4R** register are automatically cleared by hardware.

#### GPIO 8-mA Drive Select (GPIODR8R)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x508

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV8	R/W	0x00	Output Pad 8-mA Drive Enable

A write of 1 to either **GPIODR2[n]** or **GPIODR4[n]** clears the corresponding 8-mA enable bit. The change is effective on the second clock cycle after the write.

# Register 14: GPIO Open Drain Select (GPIOODR), offset 0x50C

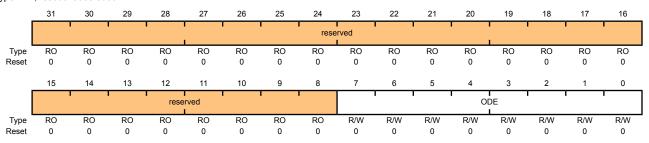
The **GPIOODR** register is the open drain control register. Setting a bit in this register enables the open drain configuration of the corresponding GPIO pad. When open drain mode is enabled, the corresponding bit should also be set in the **GPIO Digital Input Enable (GPIODEN)** register (see page 159). Corresponding bits in the drive strength registers (**GPIODR2R**, **GPIODR4R**, **GPIODR8R**, and **GPIOSLR**) can be set to achieve the desired rise and fall times. The GPIO acts as an open drain input if the corresponding bit in the **GPIODIR** register is set to 0; and as an open drain output when set to 1.

When using the I<sup>2</sup>C module, the **GPIO Alternate Function Select (GPIOAFSEL)** register bit for PB2 and PB3 should be set to 1 (see examples in "Initialization and Configuration" on page 137).

#### GPIO Open Drain Select (GPIOODR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 OFIO Port H base: 0x4002.7000

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	ODE	R/W	0x00	Output Pad Open Drain Enable

The ODE values are defined as follows:

#### Value Description

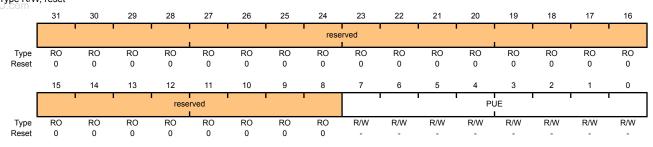
- 0 Open drain configuration is disabled.
- 1 Open drain configuration is enabled.

### Register 15: GPIO Pull-Up Select (GPIOPUR), offset 0x510

The **GPIOPUR** register is the pull-up control register. When a bit is set to 1, it enables a weak pull-up resistor on the corresponding GPIO signal. Setting a bit in **GPIOPUR** automatically clears the corresponding bit in the **GPIO Pull-Down Select (GPIOPDR)** register (see page 157).

### GPIO Pull-Up Select (GPIOPUR)

GPIO Port A base: 0x4000.4000
GPIO Port B base: 0x4000.5000
GPIO Port C base: 0x4000.6000
GPIO Port D base: 0x4002.4000
GPIO Port E base: 0x4002.4000
GPIO Port F base: 0x4002.5000
GPIO Port G base: 0x4002.6000
GPIO Port H base: 0x4002.7000
Offset 0x510
Type R/W, reset -



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PUE	R/W	_	Pad Weak Pull-Up Enable

A write of 1 to **GPIOPDR[n]** clears the corresponding **GPIOPUR[n]** enables. The change is effective on the second clock cycle after the write.

Note:

The default reset value for the **GPIOAFSEL**, **GPIOPUR**, and **GPIODEN** registers are 0x0000.0000 for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins default to JTAG/SWD functionality. Because of this, the default reset value of these registers for GPIO Port B is 0x0000.0080 while the default reset value for Port C is 0x0000.000F.

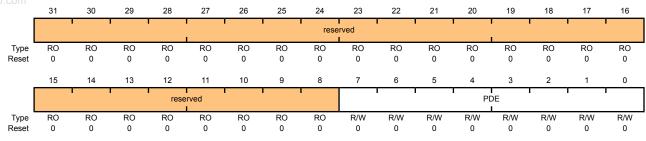
# Register 16: GPIO Pull-Down Select (GPIOPDR), offset 0x514

The **GPIOPDR** register is the pull-down control register. When a bit is set to 1, it enables a weak pull-down resistor on the corresponding GPIO signal. Setting a bit in **GPIOPDR** automatically clears the corresponding bit in the **GPIO Pull-Up Select (GPIOPUR)** register (see page 156).

### GPIO Pull-Down Select (GPIOPDR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 OFIO Port H base: 0x4002.7000

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PDE	R/W	0x00	Pad Weak Pull-Down Enable

A write of 1 to **GPIOPUR[n]** clears the corresponding **GPIOPDR[n]** enables. The change is effective on the second clock cycle after the write.

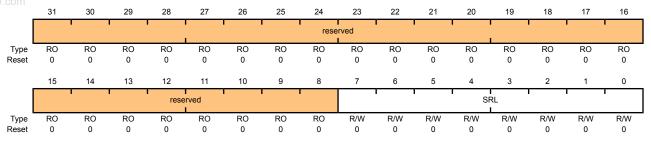
### Register 17: GPIO Slew Rate Control Select (GPIOSLR), offset 0x518

The **GPIOSLR** register is the slew rate control register. Slew rate control is only available when using the 8-mA drive strength option via the **GPIO 8-mA Drive Select (GPIODR8R)** register (see page 154).

### GPIO Slew Rate Control Select (GPIOSLR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x518

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	SRL	R/W	0x00	Slew Rate Limit Enable (8-mA drive only)

The SRL values are defined as follows:

#### Value Description

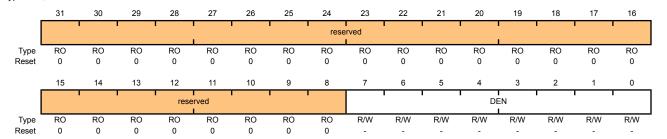
- 0 Slew rate control disabled.
- Slew rate control enabled.

# Register 18: GPIO Digital Enable (GPIODEN), offset 0x51C

The **GPIODEN** register is the digital enable register. By default, with the exception of the GPIO signals used for JTAG/SWD function, all other GPIO signals are configured out of reset to be undriven (tristate). Their digital function is disabled; they do not drive a logic value on the pin and they do not allow the pin voltage into the GPIO receiver. To use the pin in a digital function (either GPIO or alternate function), the corresponding GPIODEN bit must be set.

#### GPIO Digital Enable (GPIODEN)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0x51C Type R/W, reset -



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DEN	R/W	_	Digital Enable

The DEN values are defined as follows:

### Value Description

- Digital functions disabled.
- Digital functions enabled.

The default reset value for the GPIOAFSEL, Note: **GPIOPUR**, and **GPIODEN** registers are 0x0000.0000 for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins default to JTAG/SWD functionality. Because of this, the default reset value of these registers for GPIO Port B is 0x0000.0080 while the default reset value for Port C is 0x0000.000F.

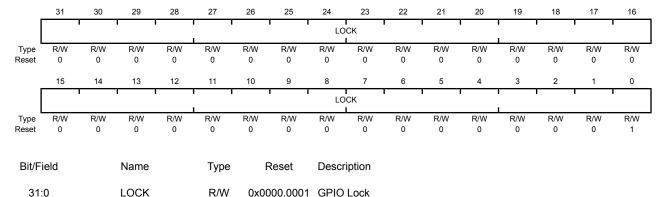
159 November 29, 2007

### Register 19: GPIO Lock (GPIOLOCK), offset 0x520

The **GPIOLOCK** register enables write access to the **GPIOCR** register (see page 161). Writing 0x1ACCE551 to the **GPIOLOCK** register will unlock the **GPIOCR** register. Writing any other value to the **GPIOLOCK** register re-enables the locked state. Reading the **GPIOLOCK** register returns the lock status rather than the 32-bit value that was previously written. Therefore, when write accesses are disabled, or locked, reading the **GPIOLOCK** register returns 0x00000001. When write accesses are enabled, or unlocked, reading the **GPIOLOCK** register returns 0x000000000.

#### GPIO Lock (GPIOLOCK)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.7000 GPIO Port H base: 0x4002.7000 GPIO Port H base: 0x4002.7000 GPIO Port H base: 0x4002.7000 GFISE 0x520 Type R/W, reset 0x0000.0001



A write of the value 0x1ACCE551 unlocks the **GPIO Commit (GPIOCR)** register for write access. A write of any other value reapplies the lock, preventing any register updates. A read of this register returns the following values:

www.DataSheet4U.com

Value Description
0x0000.0001 locked
0x0000.0000 unlocked

### Register 20: GPIO Commit (GPIOCR), offset 0x524

The GPIOCR register is the commit register. The value of the GPIOCR register determines which bits of the GPIOAFSEL register will be committed when a write to the GPIOAFSEL register is performed. If a bit in the GPIOCR register is a zero, the data being written to the corresponding bit in the GPIOAFSEL register will not be committed and will retain its previous value. If a bit in the **GPIOCR** register is a one, the data being written to the corresponding bit of the **GPIOAFSEL** register will be committed to the register and will reflect the new value.

The contents of the **GPIOCR** register can only be modified if the **GPIOLOCK** register is unlocked. Writes to the GPIOCR register will be ignored if the GPIOLOCK register is locked.

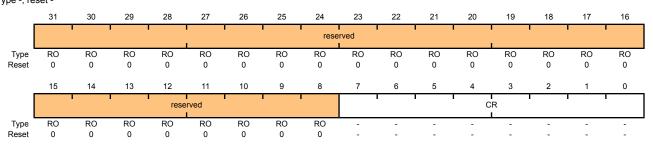
Important: This register is designed to prevent accidental programming of the GPIOAFSEL registers that control connectivity to the JTAG/SWD debug hardware. By initializing the bits of the GPIOCR register to 0 for PB7 and PC[3:0], the JTAG/SWD debug port can only be converted to GPIOs through a deliberate set of writes to the GPIOLOCK, GPIOCR, and **GPIOAFSEL** registers.

> Because this protection is currently only implemented on the JTAG/SWD pins on PB7 and PC[3:0], all of the other bits in the **GPIOCR** registers cannot be written with 0x0. These bits are hardwired to 0x1, ensuring that it is always possible to commit new values to the **GPIOAFSEL** register bits of these other pins.

#### GPIO Commit (GPIOCR)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000 5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000

Offset 0x524 Type -, reset -



Description

Bit/Field Name Type Reset RO 0x00 31.8 reserved

Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

November 29, 2007 161

Bit/Field	Name	Type	Reset	Description
7:0	CR	_	_	GPIO Commit

On a bit-wise basis, any bit set allows the corresponding <code>GPIOAFSEL</code> bit to be set to its alternate function.

Note:

The default register type for the **GPIOCR** register is RO for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). These five pins are currently the only GPIOs that are protected by the **GPIOCR** register. Because of this, the register type for GPIO Port B7 and GPIO Port C[3:0] is R/W.

The default reset value for the **GPIOCR** register is 0x0000.00FF for all GPIO pins, with the exception of the five JTAG/SWD pins (PB7 and PC[3:0]). To ensure that the JTAG port is not accidentally programmed as a GPIO, these five pins default to non-commitable. Because of this, the default reset value of **GPIOCR** for GPIO Port B is 0x0000.007F while the default reset value of **GPIOCR** for Port C is 0x0000.00FO.

# Register 21: GPIO Peripheral Identification 4 (GPIOPeriphID4), offset 0xFD0

The GPIOPeriphID4, GPIOPeriphID5, GPIOPeriphID6, and GPIOPeriphID7 registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 4 (GPIOPeriphID4)

PID4

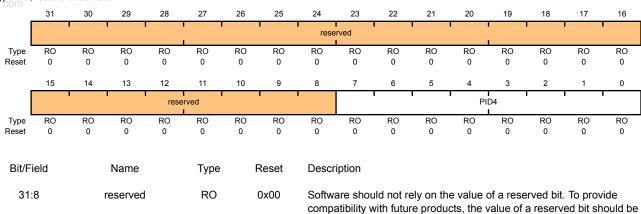
RO

0x00

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFD0

Type RO, reset 0x0000.0000

7:0



preserved across a read-modify-write operation.

GPIO Peripheral ID Register[7:0]

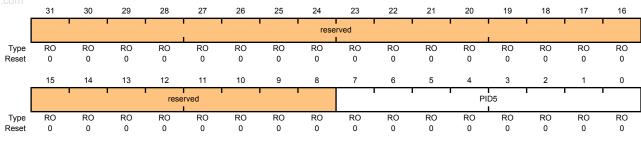
# Register 22: GPIO Peripheral Identification 5 (GPIOPeriphID5), offset 0xFD4

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 5 (GPIOPeriphID5)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFD4

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	GPIO Peripheral ID Register[15:8]

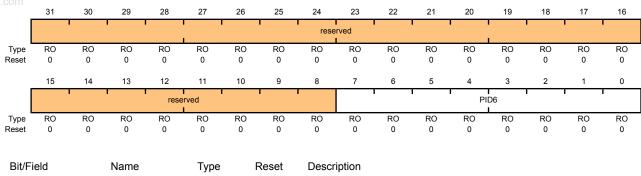
# Register 23: GPIO Peripheral Identification 6 (GPIOPeriphID6), offset 0xFD8

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 6 (GPIOPeriphID6)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFD8

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	GPIO Peripheral ID Register[23:16]

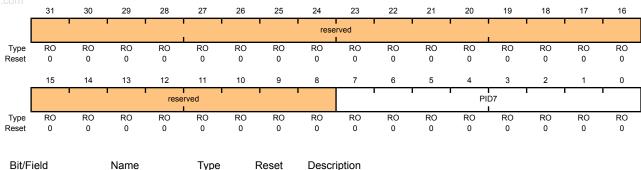
# Register 24: GPIO Peripheral Identification 7 (GPIOPeriphID7), offset 0xFDC

The **GPIOPeriphID4**, **GPIOPeriphID5**, **GPIOPeriphID6**, and **GPIOPeriphID7** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 7 (GPIOPeriphID7)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFDC

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	GPIO Peripheral ID Register[31:24]

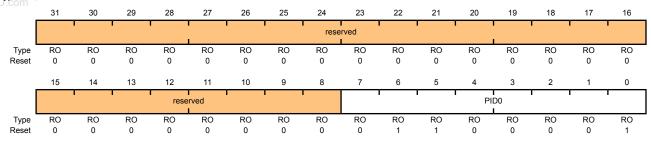
# Register 25: GPIO Peripheral Identification 0 (GPIOPeriphID0), offset 0xFE0

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 0 (GPIOPeriphID0)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFEO

Type RO, reset 0x0000.0061



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x61	GPIO Peripheral ID Register[7:0]

Can be used by software to identify the presence of this peripheral.

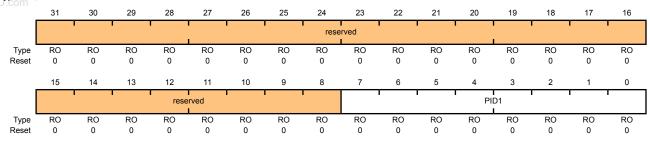
# Register 26: GPIO Peripheral Identification 1 (GPIOPeriphID1), offset 0xFE4

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 1 (GPIOPeriphID1)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFE4

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	GPIO Peripheral ID Register[15:8]

Can be used by software to identify the presence of this peripheral.

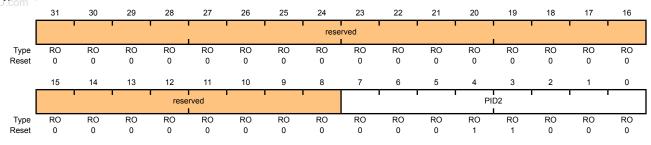
# Register 27: GPIO Peripheral Identification 2 (GPIOPeriphID2), offset 0xFE8

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 2 (GPIOPeriphID2)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 GPIO Port H base: 0x4002.7000

Type RO, reset 0x0000.0018



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	GPIO Peripheral ID Register[23:16]

Can be used by software to identify the presence of this peripheral.

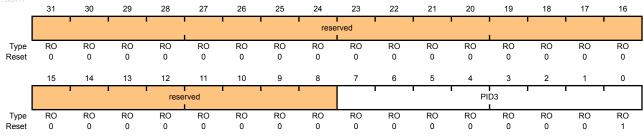
# Register 28: GPIO Peripheral Identification 3 (GPIOPeriphID3), offset 0xFEC

The **GPIOPeriphID0**, **GPIOPeriphID1**, **GPIOPeriphID2**, and **GPIOPeriphID3** registers can conceptually be treated as one 32-bit register; each register contains eight bits of the 32-bit register, used by software to identify the peripheral.

### GPIO Peripheral Identification 3 (GPIOPeriphID3)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 GPIO Port H base: 0x4002.7000

Type RO, reset 0x0000.0001



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	GPIO Peripheral ID Register[31:24]

Can be used by software to identify the presence of this peripheral.

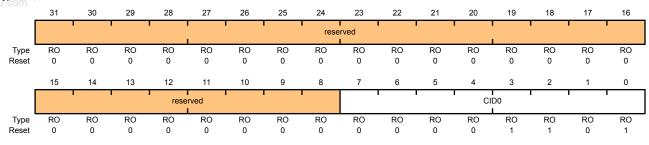
# Register 29: GPIO PrimeCell Identification 0 (GPIOPCellID0), offset 0xFF0

The **GPIOPCeIIID1**, **GPIOPCeIIID1**, and **GPIOPCeIIID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

### GPIO PrimeCell Identification 0 (GPIOPCellID0)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFF0

Type RO, reset 0x0000.000D



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	GPIO PrimeCell ID Register[7:0]

 $\label{provides} \mbox{Provides software a standard cross-peripheral identification system.}$ 

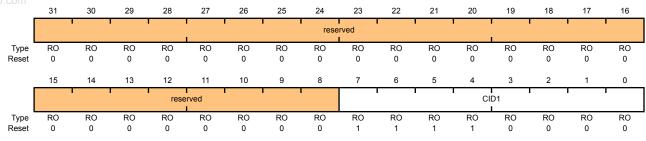
# Register 30: GPIO PrimeCell Identification 1 (GPIOPCellID1), offset 0xFF4

The **GPIOPCeIIID1**, **GPIOPCeIIID1**, and **GPIOPCeIIID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

### GPIO PrimeCell Identification 1 (GPIOPCellID1)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFF4

Type RO, reset 0x0000.00F0



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	GPIO PrimeCell ID Register[15:8]

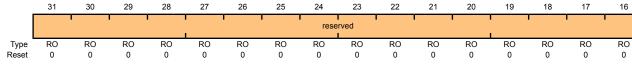
 $\label{provides} \mbox{Provides software a standard cross-peripheral identification system.}$ 

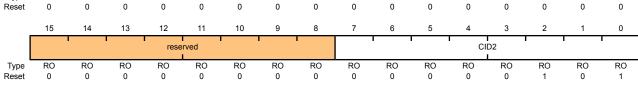
# Register 31: GPIO PrimeCell Identification 2 (GPIOPCellID2), offset 0xFF8

The **GPIOPCeIIID1**, **GPIOPCeIIID1**, and **GPIOPCeIIID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

### GPIO PrimeCell Identification 2 (GPIOPCellID2)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 Offset 0xFF8 Type RO, reset 0x0000.0005





Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	GPIO PrimeCell ID Register[23:16]

Provides software a standard cross-peripheral identification system.

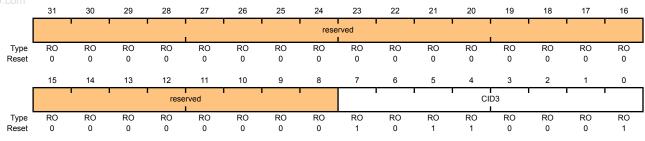
# Register 32: GPIO PrimeCell Identification 3 (GPIOPCellID3), offset 0xFFC

The **GPIOPCeIIID1**, **GPIOPCeIIID1**, and **GPIOPCeIIID3** registers are four 8-bit wide registers, that can conceptually be treated as one 32-bit register. The register is used as a standard cross-peripheral identification system.

### GPIO PrimeCell Identification 3 (GPIOPCellID3)

GPIO Port A base: 0x4000.4000 GPIO Port B base: 0x4000.5000 GPIO Port C base: 0x4000.6000 GPIO Port D base: 0x4000.7000 GPIO Port E base: 0x4002.4000 GPIO Port F base: 0x4002.5000 GPIO Port G base: 0x4002.6000 GPIO Port H base: 0x4002.7000 OFISE 0xFFC

Type RO, reset 0x0000.00B1



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	GPIO PrimeCell ID Register[31:24]

Provides software a standard cross-peripheral identification system.

# 9 General-Purpose Timers

Programmable timers can be used to count or time external events that drive the Timer input pins. The Stellaris<sup>®</sup> General-Purpose Timer Module (GPTM) contains three GPTM blocks (Timer0, Timer1, and Timer 2). Each GPTM block provides two 16-bit timers/counters (referred to as TimerA and TimerB) that can be configured to operate independently as timers or event counters, or configured to operate as one 32-bit timer or one 32-bit Real-Time Clock (RTC).

Note: Timer2 is an internal timer and can only be used to generate internal interrupts.

The General-Purpose Timer Module is one timing resource available on the Stellaris<sup>®</sup> microcontrollers. Other timer resources include the System Timer (SysTick) (see "System Timer (SysTick)" on page 36) and the PWM timer in the PWM module (see "PWM Timer" on page 401).

The following modes are supported:

- 32-bit Timer modes
  - Programmable one-shot timer
  - Programmable periodic timer
  - Real-Time Clock using 32.768-KHz input clock
  - Software-controlled event stalling (excluding RTC mode)
- 16-bit Timer modes
  - General-purpose timer function with an 8-bit prescaler (for one-shot and periodic modes only)
  - Programmable one-shot timer
  - Programmable periodic timer
  - Software-controlled event stalling
- 16-bit Input Capture modes
  - Input edge count capture
  - Input edge time capture
- 16-bit PWM mode
  - Simple PWM mode with software-programmable output inversion of the PWM signal

# 9.1 Block Diagram

Note: In Figure 9-1 on page 176, the specific CCP pins available depend on the Stellaris<sup>®</sup> device. See Table 9-1 on page 176 for the available CCPs.

0x0000 (Down Counter Modes) TimerA Control GPTMTAPMR TA Comparator **GPTMTAPR** Clock / Edge **GPTMTAMATCHR** Detect Interrupt / Confid 32 KHz or **GPTMTAILR** GPTMAR En Even CCP Pin GPTMTAMR **GPTMCFG** TimerA Interrupt **GPTMCTL GPTMIMR** RTC Divider **GPTMRIS** TimerB **GPTMMIS** Interrupt TimerB Control **GPTMICR** GPTMTBR En GPTMTBPMR Clock / Edge **GPTMTBPR** Detect Odd CCP Pin GPTMTBMATCHR TB Comparator GPTMTBILR GPTMTBMR 0x0000 (Down Counter Modes) Člock

Figure 9-1. GPTM Module Block Diagram

Table 9-1. Available CCP Pins

Timer	16-Bit Up/Down Counter	Even CCP Pin	Odd CCP Pin
Timer 0	TimerA	CCP0	-
	TimerB	-	CCP1
Timer 1	TimerA	CCP2	-
	TimerB	-	CCP3
Timer 2	TimerA	-	-
	TimerB	-	-

# 9.2 Functional Description

The main components of each GPTM block are two free-running 16-bit up/down counters (referred to as TimerA and TimerB), two 16-bit match registers, two prescaler match registers, and two 16-bit load/initialization registers and their associated control functions. The exact functionality of each GPTM is controlled by software and configured through the register interface.

Software configures the GPTM using the **GPTM Configuration (GPTMCFG)** register (see page 187), the **GPTM TimerA Mode (GPTMTAMR)** register (see page 188), and the **GPTM TimerB Mode (GPTMTBMR)** register (see page 190). When in one of the 32-bit modes, the timer can only act as a 32-bit timer. However, when configured in 16-bit mode, the GPTM can have its two 16-bit timers configured in any combination of the 16-bit modes.

#### 9.2.1 GPTM Reset Conditions

After reset has been applied to the GPTM module, the module is in an inactive state, and all control registers are cleared and in their default states. Counters TimerA and TimerB are initialized to 0xFFFF, along with their corresponding load registers: the **GPTM TimerA Interval Load** 

176 November 29, 2007

(GPTMTAILR) register (see page 201) and the GPTM TimerB Interval Load (GPTMTBILR) register (see page 202). The prescale counters are initialized to 0x00: the GPTM TimerA Prescale (GPTMTAPR) register (see page 205) and the GPTM TimerB Prescale (GPTMTBPR) register (see page 206).

#### 9.2.2 32-Bit Timer Operating Modes

This section describes the three GPTM 32-bit timer modes (One-Shot, Periodic, and RTC) and their configuration.

The GPTM is placed into 32-bit mode by writing a 0 (One-Shot/Periodic 32-bit timer mode) or a 1 (RTC mode) to the GPTM Configuration (GPTMCFG) register. In both configurations, certain GPTM registers are concatenated to form pseudo 32-bit registers. These registers include:

- GPTM TimerA Interval Load (GPTMTAILR) register [15:0], see page 201
- GPTM TimerB Interval Load (GPTMTBILR) register [15:0], see page 202
- GPTM TimerA (GPTMTAR) register [15:0], see page 209
- GPTM TimerB (GPTMTBR) register [15:0], see page 210

In the 32-bit modes, the GPTM translates a 32-bit write access to GPTMTAILR into a write access to both **GPTMTAILR** and **GPTMTBILR**. The resulting word ordering for such a write operation is:

```
GPTMTBILR[15:0]:GPTMTAILR[15:0]
```

Likewise, a read access to **GPTMTAR** returns the value:

GPTMTBR[15:0]:GPTMTAR[15:0]

#### 9.2.2.1 32-Bit One-Shot/Periodic Timer Mode

In 32-bit one-shot and periodic timer modes, the concatenated versions of the TimerA and TimerB registers are configured as a 32-bit down-counter. The selection of one-shot or periodic mode is determined by the value written to the TAMR field of the GPTM TimerA Mode (GPTMTAMR) register (see page 188), and there is no need to write to the GPTM TimerB Mode (GPTMTBMR) register.

When software writes the TAEN bit in the GPTM Control (GPTMCTL) register (see page 192), the timer begins counting down from its preloaded value. Once the 0x0000.0000 state is reached, the timer reloads its start value from the concatenated GPTMTAILR on the next cycle. If configured to be a one-shot timer, the timer stops counting and clears the TAEN bit in the GPTMCTL register. If configured as a periodic timer, it continues counting.

In addition to reloading the count value, the GPTM generates interrupts and output triggers when it reaches the 0x0000000 state. The GPTM sets the TATORIS bit in the GPTM Raw Interrupt Status (GPTMRIS) register (see page 197), and holds it until it is cleared by writing the GPTM Interrupt Clear (GPTMICR) register (see page 199). If the time-out interrupt is enabled in the GPTM Interrupt Mask (GPTIMR) register (see page 195), the GPTM also sets the TATOMIS bit in the GPTM Masked Interrupt Status (GPTMMIS) register (see page 198).

The output trigger is a one-clock-cycle pulse that is asserted when the counter hits the 0x0000.0000 state, and deasserted on the following clock cycle. It is enabled by setting the TAOTE bit in GPTMCTL.

If software reloads the **GPTMTAILR** register while the counter is running, the counter loads the new

value on the next clock cycle and continues counting from the new value.

November 29, 2007 177 If the TASTALL bit in the **GPTMCTL** register is asserted, the timer freezes counting until the signal is deasserted.

### 9.2.2.2 32-Bit Real-Time Clock Timer Mode

In Real-Time Clock (RTC) mode, the concatenated versions of the TimerA and TimerB registers are configured as a 32-bit up-counter. When RTC mode is selected for the first time, the counter is loaded with a value of 0x0000.0001. All subsequent load values must be written to the **GPTM TimerA Match (GPTMTAMATCHR)** register (see page 203) by the controller.

The input clock on the CCP0, CCP2, or CCP4 pins is required to be 32.768 KHz in RTC mode. The clock signal is then divided down to a 1 Hz rate and is passed along to the input of the 32-bit counter.

When software writes the TAEN bit inthe **GPTMCTL** register, the counter starts counting up from its preloaded value of 0x0000.0001. When the current count value matches the preloaded value in the **GPTMTAMATCHR** register, it rolls over to a value of 0x0000.0000 and continues counting until either a hardware reset, or it is disabled by software (clearing the TAEN bit). When a match occurs, the GPTM asserts the RTCRIS bit in **GPTMRIS**. If the RTC interrupt is enabled in **GPTIMR**, the GPTM also sets the RTCMIS bit in **GPTMISR** and generates a controller interrupt. The status flags are cleared by writing the RTCCINT bit in **GPTMICR**.

If the TASTALL and/or TBSTALL bits in the **GPTMCTL** register are set, the timer does not freeze if the RTCEN bit is set in **GPTMCTL**.

### 9.2.3 16-Bit Timer Operating Modes

The GPTM is placed into global 16-bit mode by writing a value of 0x4 to the **GPTM Configuration (GPTMCFG)** register (see page 187). This section describes each of the GPTM 16-bit modes of operation. TimerA and TimerB have identical modes, so a single description is given using an *n* to reference both.

### 9.2.3.1 16-Bit One-Shot/Periodic Timer Mode

In 16-bit one-shot and periodic timer modes, the timer is configured as a 16-bit down-counter with an optional 8-bit prescaler that effectively extends the counting range of the timer to 24 bits. The selection of one-shot or periodic mode is determined by the value written to the TnMR field of the **GPTMTnMR** register. The optional prescaler is loaded into the **GPTM Timern Prescale (GPTMTnPR)** register.

When software writes the  $\mathtt{TnEN}$  bit in the **GPTMCTL** register, the timer begins counting down from its preloaded value. Once the 0x0000 state is reached, the timer reloads its start value from **GPTMTnILR** and **GPTMTnPR** on the next cycle. If configured to be a one-shot timer, the timer stops counting and clears the  $\mathtt{TnEN}$  bit in the **GPTMCTL** register. If configured as a periodic timer, it continues counting.

In addition to reloading the count value, the timer generates interrupts and output triggers when it reaches the 0x0000 state. The GPTM sets the TnTORIS bit in the **GPTMRIS** register, and holds it until it is cleared by writing the **GPTMICR** register. If the time-out interrupt is enabled in **GPTIMR**, the GPTM also sets the TnTOMIS bit in **GPTMISR** and generates a controller interrupt.

The output trigger is a one-clock-cycle pulse that is asserted when the counter hits the 0x0000 state, and deasserted on the following clock cycle. It is enabled by setting the  $\mathtt{TnOTE}$  bit in the **GPTMCTL** register, and can trigger SoC-level events.

If software reloads the **GPTMTAILR** register while the counter is running, the counter loads the new value on the next clock cycle and continues counting from the new value.

www.DataSheet4U.com

www DataSheet4U com

If the TnSTALL bit in the **GPTMCTL** register is enabled, the timer freezes counting until the signal is deasserted.

The following example shows a variety of configurations for a 16-bit free running timer while using the prescaler. All values assume a 25-MHz clock with Tc=20 ns (clock period).

**Table 9-2. 16-Bit Timer With Prescaler Configurations** 

Prescale	#Clock (T c) <sup>a</sup>	Max Time	Units
00000000	1	2.6214	mS
00000001	2	5.2428	mS
00000010	3	7.8642	mS
11111100	254	665.8458	mS
11111110	255	668.4672	mS
11111111	256	671.0886	mS

a. Tc is the clock period.

### 9.2.3.2 16-Bit Input Edge Count Mode

In Edge Count mode, the timer is configured as a down-counter capable of capturing three types of events: rising edge, falling edge, or both. To place the timer in Edge Count mode, the TnCMR bit of the GPTMTnMR register must be set to 0. The type of edge that the timer counts is determined by the TnEVENT fields of the GPTMCTL register. During initialization, the GPTM Timern Match (GPTMTnMATCHR) register is configured so that the difference between the value in the GPTMTnILR register and the GPTMTnMATCHR register equals the number of edge events that must be counted.

When software writes the TnEN bit in the **GPTM Control (GPTMCTL)** register, the timer is enabled for event capture. Each input event on the CCP pin decrements the counter by 1 until the event count matches **GPTMTnMATCHR**. When the counts match, the GPTM asserts the CnMRIS bit in the **GPTMRIS** register (and the CnMMIS bit, if the interrupt is not masked). The counter is then reloaded using the value in **GPTMTnILR**, and stopped since the GPTM automatically clears the TnEN bit in the **GPTMCTL** register. Once the event count has been reached, all further events are ignored until TnEN is re-enabled by software.

Figure 9-2 on page 180 shows how input edge count mode works. In this case, the timer start value is set to **GPTMnILR** =0x000A and the match value is set to **GPTMnMATCHR** =0x0006 so that four edge events are counted. The counter is configured to detect both edges of the input signal.

Note that the last two edges are not counted since the timer automatically clears the TnEN bit after the current count matches the value in the **GPTMnMR** register.

Count

Timer reload on next cycle Ignored Ignored

0x000A

0x0009

0x0008

0x0007

0x0006

Timer stops, flags asserted

Input Signal

Figure 9-2. 16-Bit Input Edge Count Mode Example

w.DataSheet4U.com

### 9.2.3.3 16-Bit Input Edge Time Mode

**Note:** The prescaler is not available in 16-Bit Input Edge Time mode.

In Edge Time mode, the timer is configured as a free-running down-counter initialized to the value loaded in the **GPTMTnILR** register (or 0xFFFF at reset). This mode allows for event capture of both rising and falling edges. The timer is placed into Edge Time mode by setting the TnCMR bit in the **GPTMTnMR** register, and the type of event that the timer captures is determined by the TnEVENT fields of the **GPTMCnTL** register.

When software writes the TnEN bit in the **GPTMCTL** register, the timer is enabled for event capture. When the selected input event is detected, the current **Tn** counter value is captured in the **GPTMTnR** register and is available to be read by the controller. The GPTM then asserts the CnERIS bit (and the CnEMIS bit, if the interrupt is not masked).

After an event has been captured, the timer does not stop counting. It continues to count until the  ${\tt TnEN}$  bit is cleared. When the timer reaches the 0x0000 state, it is reloaded with the value from the **GPTMnILR** register.

Figure 9-3 on page 181 shows how input edge timing mode works. In the diagram, it is assumed that the start value of the timer is the default value of 0xFFFF, and the timer is configured to capture rising edge events.

Each time a rising edge event is detected, the current count value is loaded into the **GPTMTnR** register, and is held there until another rising edge is detected (at which point the new count value is loaded into **GPTMTnR**).

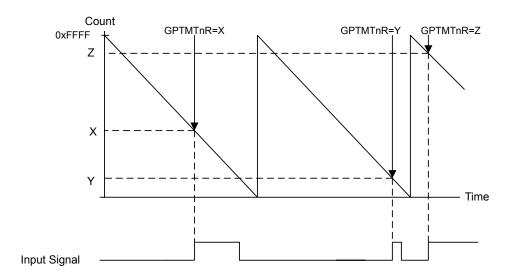


Figure 9-3. 16-Bit Input Edge Time Mode Example

W.DataSHEE14U.COIII

#### 9.2.3.4 16-Bit PWM Mode

The GPTM supports a simple PWM generation mode. In PWM mode, the timer is configured as a down-counter with a start value (and thus period) defined by **GPTMTnILR**. PWM mode is enabled with the **GPTMTnMR** register by setting the TnAMS bit to 0x1, the TnCMR bit to 0x0, and the TnMR field to 0x2.

When software writes the TnEN bit in the **GPTMCTL** register, the counter begins counting down until it reaches the 0x0000 state. On the next counter cycle, the counter reloads its start value from **GPTMTnILR** (and **GPTMTnPR** if using a prescaler) and continues counting until disabled by software clearing the TnEN bit in the **GPTMCTL** register. No interrupts or status bits are asserted in PWM mode.

The output PWM signal asserts when the counter is at the value of the **GPTMTnILR** register (its start state), and is deasserted when the counter value equals the value in the **GPTM Timern Match Register (GPTMnMATCHR)**. Software has the capability of inverting the output PWM signal by setting the TnPWML bit in the **GPTMCTL** register.

Figure 9-4 on page 182 shows how to generate an output PWM with a 1-ms period and a 66% duty cycle assuming a 50-MHz input clock and **TnPWML** =0 (duty cycle would be 33% for the **TnPWML** =1 configuration). For this example, the start value is **GPTMnIRL**=0xC350 and the match value is **GPTMnMR**=0x411A.

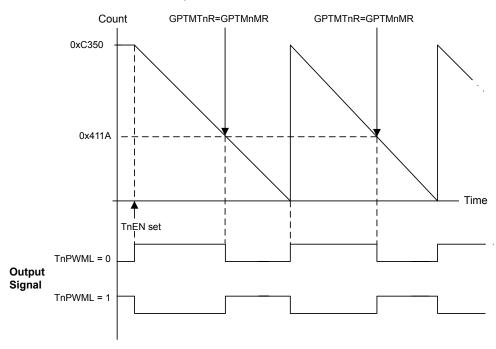


Figure 9-4. 16-Bit PWM Mode Example

ww.DataSHEEt4U.COH

# 9.3 Initialization and Configuration

To use the general-purpose timers, the peripheral clock must be enabled by setting the TIMERO, TIMER1, and TIMER2 bits in the **RCGC1** register.

This section shows module initialization and configuration examples for each of the supported timer modes.

#### 9.3.1 32-Bit One-Shot/Periodic Timer Mode

The GPTM is configured for 32-bit One-Shot and Periodic modes by the following sequence:

- 1. Ensure the timer is disabled (the TAEN bit in the **GPTMCTL** register is cleared) before making any changes.
- 2. Write the **GPTM Configuration Register (GPTMCFG)** with a value of 0x0.
- 3. Set the TAMR field in the GPTM TimerA Mode Register (GPTMTAMR):
  - a. Write a value of 0x1 for One-Shot mode.
  - b. Write a value of 0x2 for Periodic mode.
- 4. Load the start value into the GPTM TimerA Interval Load Register (GPTMTAILR).
- If interrupts are required, set the TATOIM bit in the GPTM Interrupt Mask Register (GPTMIMR).
- 6. Set the TAEN bit in the **GPTMCTL** register to enable the timer and start counting.

7. Poll the TATORIS bit in the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the TATOCINT bit of the **GPTM** Interrupt Clear Register (GPTMICR).

In One-Shot mode, the timer stops counting after step 7 on page 183. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

### 9.3.2 32-Bit Real-Time Clock (RTC) Mode

To use the RTC mode, the timer must have a 32.768-KHz input signal on its CCP0, CCP2, or CCP4 pins. To enable the RTC feature, follow these steps:

- Ensure the timer is disabled (the TAEN bit is cleared) before making any changes.
- Write the GPTM Configuration Register (GPTMCFG) with a value of 0x1.
- Write the desired match value to the GPTM TimerA Match Register (GPTMTAMATCHR).
- Set/clear the RTCEN bit in the GPTM Control Register (GPTMCTL) as desired.
- If interrupts are required, set the RTCIM bit in the GPTM Interrupt Mask Register (GPTMIMR).
- Set the TAEN bit in the GPTMCTL register to enable the timer and start counting.

When the timer count equals the value in the **GPTMTAMATCHR** register, the counter is re-loaded with 0x0000.0000 and begins counting. If an interrupt is enabled, it does not have to be cleared.

### 9.3.3 16-Bit One-Shot/Periodic Timer Mode

A timer is configured for 16-bit One-Shot and Periodic modes by the following sequence:

- 1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
- 2. Write the GPTM Configuration Register (GPTMCFG) with a value of 0x4.
- Set the TnMR field in the GPTM Timer Mode (GPTMTnMR) register:
  - a. Write a value of 0x1 for One-Shot mode.
  - b. Write a value of 0x2 for Periodic mode.
- If a prescaler is to be used, write the prescale value to the GPTM Timern Prescale Register (GPTMTnPR).
- Load the start value into the GPTM Timer Interval Load Register (GPTMTnILR).
- If interrupts are required, set the Thtolm bit in the GPTM Interrupt Mask Register (GPTMIMR).
- 7. Set the TnEN bit in the **GPTM Control Register (GPTMCTL)** to enable the timer and start counting.
- 8. Poll the TnTORIS bit in the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the TnTOCINT bit of the **GPTM** Interrupt Clear Register (GPTMICR).

November 29, 2007 183

In One-Shot mode, the timer stops counting after step 8 on page 183. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

### 9.3.4 16-Bit Input Edge Count Mode

A timer is configured to Input Edge Count mode by the following sequence:

- 1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
- 2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x4.
- 3. In the **GPTM Timer Mode (GPTMTnMR)** register, write the TnCMR field to 0x0 and the TnMR field to 0x3.
- Configure the type of event(s) that the timer captures by writing the Tnevent field of the GPTM Control (GPTMCTL) register.
- 5. Load the timer start value into the **GPTM Timern Interval Load (GPTMTnILR)** register.
- Load the desired event count into the GPTM Timern Match (GPTMTnMATCHR) register.
- 7. If interrupts are required, set the CnMIM bit in the GPTM Interrupt Mask (GPTMIMR) register.
- 8. Set the TREN bit in the GPTMCTL register to enable the timer and begin waiting for edge events.
- 9. Poll the CnMRIS bit in the GPTMRIS register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the CnMCINT bit of the GPTM Interrupt Clear (GPTMICR) register.

In Input Edge Count Mode, the timer stops after the desired number of edge events has been detected. To re-enable the timer, ensure that the  ${\tt TnEN}$  bit is cleared and repeat step 4 on page 184 through step 9 on page 184.

### 9.3.5 16-Bit Input Edge Timing Mode

A timer is configured to Input Edge Timing mode by the following sequence:

- Ensure the timer is disabled (the Then bit is cleared) before making any changes.
- 2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x4.
- In the GPTM Timer Mode (GPTMTnMR) register, write the TnCMR field to 0x1 and the TnMR field to 0x3.
- 4. Configure the type of event that the timer captures by writing the Tnevent field of the **GPTM** Control (GPTMCTL) register.
- Load the timer start value into the GPTM Timern Interval Load (GPTMTnILR) register.
- 6. If interrupts are required, set the Cneim bit in the GPTM Interrupt Mask (GPTMIMR) register.
- Set the Then bit in the GPTM Control (GPTMCTL) register to enable the timer and start counting.
- 8. Poll the Cners bit in the **GPTMRIS** register or wait for the interrupt to be generated (if enabled). In both cases, the status flags are cleared by writing a 1 to the Cnecint bit of the **GPTM**

www.DataShoot411.com

**Interrupt Clear (GPTMICR)** register. The time at which the event happened can be obtained by reading the **GPTM Timern (GPTMTnR)** register.

In Input Edge Timing mode, the timer continues running after an edge event has been detected, but the timer interval can be changed at any time by writing the **GPTMTnILR** register. The change takes effect at the next cycle after the write.

#### 9.3.6 16-Bit PWM Mode

A timer is configured to PWM mode using the following sequence:

- 1. Ensure the timer is disabled (the TnEN bit is cleared) before making any changes.
- 2. Write the **GPTM Configuration (GPTMCFG)** register with a value of 0x4.
- 3. In the **GPTM Timer Mode (GPTMTnMR)** register, set the TnAMS bit to 0x1, the TnCMR bit to 0x0, and the TnMR field to 0x2.
- 4. Configure the output state of the PWM signal (whether or not it is inverted) in the TREVENT field of the GPTM Control (GPTMCTL) register.
- Load the timer start value into the GPTM Timern Interval Load (GPTMTnILR) register.
- 6. Load the GPTM Timern Match (GPTMTnMATCHR) register with the desired value.
- If a prescaler is going to be used, configure the GPTM Timern Prescale (GPTMTnPR) register and the GPTM Timern Prescale Match (GPTMTnPMR) register.
- 8. Set the TnEN bit in the **GPTM Control (GPTMCTL)** register to enable the timer and begin generation of the output PWM signal.

In PWM Timing mode, the timer continues running after the PWM signal has been generated. The PWM period can be adjusted at any time by writing the **GPTMTnILR** register, and the change takes effect at the next cycle after the write.

# 9.4 Register Map

Table 9-3 on page 185 lists the GPTM registers. The offset listed is a hexadecimal increment to the register's address, relative to that timer's base address:

Timer0: 0x4003.0000

Timer1: 0x4003.1000

Timer2: 0x4003.2000

#### Table 9-3. Timers Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	GPTMCFG	R/W	0x0000.0000	GPTM Configuration	187
0x004	GPTMTAMR	R/W	0x0000.0000	GPTM TimerA Mode	188
0x008	GPTMTBMR	R/W	0x0000.0000	GPTM TimerB Mode	190

November 29, 2007 185

Offset	Name	Туре	Reset	Description	See page
0x00C	GPTMCTL	R/W	0x0000.0000	GPTM Control	192
0x018	GPTMIMR	R/W	0x0000.0000	GPTM Interrupt Mask	195
0x01C	GPTMRIS	RO	0x0000.0000	GPTM Raw Interrupt Status	197
0x020	GPTMMIS	RO	0x0000.0000	GPTM Masked Interrupt Status	198
0x024	GPTMICR	W1C	0x0000.0000	GPTM Interrupt Clear	199
<b>0x028</b> 4U.com	GPTMTAILR	R/W	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA Interval Load	201
0x02C	GPTMTBILR	R/W	0x0000.FFFF	GPTM TimerB Interval Load	202
0x030	GPTMTAMATCHR	R/W	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA Match	203
0x034	GPTMTBMATCHR	R/W	0x0000.FFFF	GPTM TimerB Match	204
0x038	GPTMTAPR	R/W	0x0000.0000	GPTM TimerA Prescale	205
0x03C	GPTMTBPR	R/W	0x0000.0000	GPTM TimerB Prescale	206
0x040	GPTMTAPMR	R/W	0x0000.0000	GPTM TimerA Prescale Match	207
0x044	GPTMTBPMR	R/W	0x0000.0000	GPTM TimerB Prescale Match	208
0x048	GPTMTAR	RO	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA	209
0x04C	GPTMTBR	RO	0x0000.FFFF	GPTM TimerB	210

# 9.5 Register Descriptions

The remainder of this section lists and describes the GPTM registers, in numerical order by address offset.

www.DataSheet4U.com

# Register 1: GPTM Configuration (GPTMCFG), offset 0x000

This register configures the global operation of the GPTM module. The value written to this register determines whether the GPTM is in 32- or 16-bit mode.

### GPTM Configuration (GPTMCFG)

Timer0 base: 0x4003.0000
Timer1 base: 0x4003.1000
Timer2 base: 0x4003.2000
Offset 0x000
Type R/W, reset 0x0000.0000

2:0

**GPTMCFG** 

R/W

0x0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'		'				rese	rved	'		'		'		
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset IU.com	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	1	· · ·		reserved		1	1	1	1	l I		GPTMCFG	;
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/F	ield		Name		Туре		Reset	Descr	iption							
31	:3		reserved	I	RO		0x00	compa	atibility v		e produ	cts, the v	alue of	a reserv	To prov ved bit sh	

The GPTMCFG values are defined as follows:

Value Description

**GPTM Configuration** 

0x0 32-bit timer configuration.

0x1 32-bit real-time clock (RTC) counter configuration.

0x2 Reserved. 0x3 Reserved.

0x4-0x7 16-bit timer configuration, function is controlled by bits 1:0 of **GPTMTAMR** and **GPTMTBMR**.

www.DataSheet4U.com

# Register 2: GPTM TimerA Mode (GPTMTAMR), offset 0x004

This register configures the GPTM based on the configuration selected in the **GPTMCFG** register. When in 16-bit PWM mode, set the TAAMS bit to 0x1, the TACMR bit to 0x0, and the TAMR field to 0x2.

#### GPTM TimerA Mode (GPTMTAMR)

**TAAMS** 

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x004

3

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		•				,		rese	rved I				ı	'		
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		•			' '	reserved TAAMS TAC									TAI	MR
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	U	0	0	0	0	0	0	0

Bit/Field	Name	Туре	Reset	Description
31:4	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

GPTM TimerA Alternate Mode Select

The TAAMS values are defined as follows:

Value Description

Capture mode is enabled.

1 PWM mode is enabled.

Note: To enable PWM mode, you must also clear the TACMR bit and set the TAMR field to 0x2.

www.DataSheet4U.com

2 TACMR R/W 0 GPTM TimerA Capture Mode

R/W

The TACMR values are defined as follows:

Value Description

D Edge-Count mode.

Edge-Time mode.

In 16-bit timer configuration,  $\mathtt{TAMR}$  controls the 16-bit timer modes for

In 32-bit timer configuration, this register controls the mode and the

contents of **GPTMTBMR** are ignored.

Bit/Field	Name	Туре	Reset	Description					
1:0	TAMR	R/W	0x0	GPTM TimerA Mode					
				The TAMR values are defined as follows:					
				Value Description					
				0x0 Reserved.					
				0x1 One-Shot Timer mode.					
				0x2 Periodic Timer mode.					
				0x3 Capture mode.					
				The Timer mode is based on the timer configuration defined by bits 2:0 in the <b>GPTMCFG</b> register (16-or 32-bit).					

TimerA.

ww.DataSheet4U.con

# Register 3: GPTM TimerB Mode (GPTMTBMR), offset 0x008

This register configures the GPTM based on the configuration selected in the GPTMCFG register. When in 16-bit PWM mode, set the TBAMS bit to 0x1, the TBCMR bit to 0x0, and the TBMR field to

#### GPTM TimerB Mode (GPTMTBMR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x008

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'	'	'	, , , , , , , , , , , , , , , , , , ,	,		rese	rved				1	,		
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	'		reserved TBAMS TBCMR									TBI	MR
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TBAMS	R/W	0	GPTM TimerB Alternate Mode Select

**GPTM TimerB Alternate Mode Select** 

The TBAMS values are defined as follows:

Value Description

Capture mode is enabled.

PWM mode is enabled.

To enable PWM mode, you must also clear the TBCMR Note: bit and set the TBMR field to 0x2.

www.DataSheet4U.com

**TBCMR** R/W **GPTM TimerB Capture Mode** 

The TBCMR values are defined as follows:

Value Description

Edge-Count mode.

Edge-Time mode.

Bit/Field	Name	Туре	Reset	Description
1:0	TBMR	R/W	0x0	GPTM TimerB Mode
				The TBMR values are defined as follows:
				Value Description

0x0 Reserved.0x1 One-Shot Timer mode.0x2 Periodic Timer mode.0x3 Capture mode.

The timer mode is based on the timer configuration defined by bits 2:0 in the  $\mbox{\bf GPTMCFG}$  register.

In 16-bit timer configuration, these bits control the 16-bit timer modes for  $\mathsf{TimerB}.$ 

In 32-bit timer configuration, this register's contents are ignored and  $\ensuremath{\mathbf{GPTMTAMR}}$  is used.

ww.DataSheet4U.com

# Register 4: GPTM Control (GPTMCTL), offset 0x00C

This register is used alongside the **GPTMCFG** and **GMTMTnMR** registers to fine-tune the timer configuration, and to enable other features such as timer stall.

23

22

21

20

18

www.DataSheet4U.com

### GPTM Control (GPTMCTL)

30

29

28

27

26

25

24

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x00C Type R/W, reset 0x0000.0000

							1 1	rese	rved			<u> </u>			'				
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0			
U.com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
	reserved	TBPWML	TBOTE	reserved	TBEVI		TBSTALL	TBEN	reserved	TAPWML	TAOTE	RTCEN	TAEV		TASTALL	TAEN			
Туре	RO	R/W	R/W	RO	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
D					_			_											
Bit/F	ield		Name		Type	ا	Reset	Descr	iption										
31:	15	r	eserved		RO			compa	atibility v	vith futur	e produ		alue of a	a reserv	To prov ved bit sh				
14	4	Т	BPWML	_	R/W	₹/W 0			GPTM TimerB PWM Output Level										
								The T	BPWML \	/alues ar	e define	d as follo	ows:						
								Volue	Descri	ntion									
								value 0		t is unaff	ected								
								1	•	is inver									
1;	3		ТВОТЕ		R/W		0	GPTM	1 TimerE	Output	Trigger	Enable							
								The T	BOTE <b>va</b>	lues are	defined	as follow	ws:						
								Value	e Descri	ntion									
								0			erB trig	ger is dis	sabled.						
								1		•		ger is en							
12	2	r	eserved		RO		0	compa	atibility v	vith futur	e produ		alue of a	a reserv	To prov ved bit sh				
11:	10	Т	BEVEN <sup>-</sup>	Γ	R/W		0x0	GPTM	1 TimerE	Event N	/lode								
								The T	BEVENT	values a	are defir	ed as fo	llows:						
								\/aluc	e Descri	ntion									
									Positiv										
								0x1		ve edge									
									Reserv	-									
								0x3	Both e	dges.									

E	Bit/Field	Name	Туре	Reset	Description
	9	TBSTALL	R/W	0	GPTM TimerB Stall Enable
					The TBSTALL values are defined as follows:
					Value Description
					0 TimerB stalling is disabled.
					1 TimerB stalling is enabled.
	8	TBEN	R/W	0	GPTM TimerB Enable
					The TBEN values are defined as follows:
DataSheet4U.co					Value Description
					0 TimerB is disabled.
					1 TimerB is enabled and begins counting or the capture logic is enabled based on the GPTMCFG register.
	7	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
	6	TAPWML	R/W	0	GPTM TimerA PWM Output Level
					The TAPWML values are defined as follows:
					Value Description
					0 Output is unaffected.
					1 Output is inverted.
	5	TAOTE	R/W	0	GPTM TimerA Output Trigger Enable
					The TAOTE values are defined as follows:
					Value Description
					0 The output TimerA trigger is disabled.
					1 The output TimerA trigger is enabled.
	4	RTCEN	R/W	0	GPTM RTC Enable
					The RTCEN values are defined as follows:
					Value Description
					0 RTC counting is disabled.
					1 RTC counting is enabled.

Bit/Field	Name	Type	Reset	Description
3:2	TAEVENT	R/W	0x0	GPTM TimerA Event Mode
				The TAEVENT values are defined as follows:
				Value Description
				0x0 Positive edge.
				0x1 Negative edge.
				0x2 Reserved
				0x3 Both edges.
neet4U.com <sup>1</sup>	TASTALL	R/W	0	GPTM TimerA Stall Enable
				The TASTALL values are defined as follows:
				Value Description
				0 TimerA stalling is disabled.
				1 TimerA stalling is enabled.
0	TAEN	R/W	0	GPTM TimerA Enable
				The TAEN values are defined as follows:
				Value Description
				0 TimerA is disabled.

1 TimerA is enabled and begins counting or the capture logic is enabled based on the GPTMCFG register.

### Register 5: GPTM Interrupt Mask (GPTMIMR), offset 0x018

This register allows software to enable/disable GPTM controller-level interrupts. Writing a 1 enables the interrupt, while writing a 0 disables it.

### GPTM Interrupt Mask (GPTMIMR)

Timer0 base: 0x4003.0000
Timer1 base: 0x4003.1000
Timer2 base: 0x4003.2000
Offset 0x018
Type R/W, reset 0x0000.0000

		ı	'					rese	rved			1				•
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
J.com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	10	1	reserved	12		CBEIM	СВМІМ	твтоім	•	rese		1	RTCIM	CAEIM	CAMIM	ТАТОІМ
Type	RO 0	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO 0	RO	R/W	R/W	R/W	R/W
Reset	U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/Fi	ield		Name		Туре	F	Reset	Descri	iption							
31:11 reserved RO 0x00 Software shou compatibility w preserved acro				vith futur	e produ	cts, the	value of	a reserv								
10	)		CBEIM		R/W		0	GPTM	l Captur	eB Even	it Interru	pt Mask	[			
								The C	BEIM <b>va</b>	lues are	defined	l as follo	ws:			
								Value	Descri	ption						
0 In						Interru	pt is disa	abled.								
								1	Interru	pt is ena	ıbled.					
9			СВМІМ		R/W		0	GPTM	l Captur	eB Matc	h Interru	ıpt Mask	<			
								The C	BMIM <b>va</b>	lues are	defined	l as follo	ws:			
								Value	Descri	ption						
								0		pt is disa						
								1	Interru	pt is ena	bled.					
8			ТВТОІМ		R/W		0	GPTM	I TimerE	Time-O	ut Interr	upt Mas	sk			
								The T	BTOIM	/alues ar	e define	ed as foll	lows:			
								Value	Descri	ption						
								0	Interru	pt is disa	abled.					
								1	Interru	pt is ena	ıbled.					
7:4	1		reserved		RO		0	compa	atibility v	vith futur	e produ	cts, the	of a rese value of operation	a reserv		

Bit/Field	Name	Type	Reset	Description
3	RTCIM	R/W	0	GPTM RTC Interrupt Mask  The RTCIM values are defined as follows:
				Value Description  0 Interrupt is disabled.  1 Interrupt is enabled.
2	CAEIM	R/W	0	GPTM CaptureA Event Interrupt Mask The CAEIM values are defined as follows:
				Value Description  0 Interrupt is disabled.  1 Interrupt is enabled.
1	CAMIM	R/W	0	GPTM CaptureA Match Interrupt Mask The CAMIM values are defined as follows:
				Value Description  0 Interrupt is disabled.  1 Interrupt is enabled.
0	TATOIM	R/W	0	GPTM TimerA Time-Out Interrupt Mask The TATOIM values are defined as follows:
				Value Description  0 Interrupt is disabled.  1 Interrupt is enabled.

196 November 29, 2007

### Register 6: GPTM Raw Interrupt Status (GPTMRIS), offset 0x01C

This register shows the state of the GPTM's internal interrupt signal. These bits are set whether or not the interrupt is masked in the **GPTMIMR** register. Each bit can be cleared by writing a 1 to its corresponding bit in **GPTMICR**.

### GPTM Raw Interrupt Status (GPTMRIS)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x01C Type RO, reset 0x0000.0000

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				1 1			'	'	reser	ved							•
www.DataSheet4U	Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				reserved			CBERIS	CBMRIS	TBTORIS		rese	rved		RTCRIS	CAERIS	CAMRIS	TATORIS
	Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	Bit/Fi	ield		Name		Туре	· F	Reset	Descri	ption							
	31:	11		reserved		RO	ı	0x00	compa	tibility w	ith futur	ely on the e produc ad-modit	cts, the	value of	a reserv		
	10	)		CBERIS		RO		0	GPTM	Capture	eB Ever	nt Raw Ir	nterrupt				
									This is	the Cap	otureB E	Event inte	errupt st	atus prid	or to mas	sking.	
	9			CBMRIS		RO		0	GPTM	Capture	eB Mato	h Raw Ir	nterrupt				
									This is	the Cap	ptureB N	/latch int	errupt s	tatus pri	or to ma	sking.	
	8			TBTORIS	;	RO		0	GPTM	TimerB	Time-C	out Raw	Interrup	t			
									This is	the Tim	nerB tim	e-out inte	errupt st	atus prid	or to mas	sking.	
	7:4	4		reserved		RO		0x0	compa	tibility w	ith futur	ely on the e produc ad-modit	cts, the	value of	a reserv		
	3			RTCRIS		RO		0	GPTM	RTC R	aw Inter	rupt					
									This is	the RT	C Event	interrup	t status	prior to 1	masking	•	
	2			CAERIS		RO		0	GPTM	Capture	eA Ever	nt Raw Ir	nterrupt				
									This is	the Cap	ptureA E	Event inte	errupt st	atus prid	or to mas	sking.	
	1			CAMRIS		RO		0	GPTM	Capture	eA Mato	h Raw Ir	nterrupt				
									This is	the Cap	ptureA N	Match int	errupt s	tatus pri	or to ma	sking.	
	0			TATORIS		RO		0	GPTM	TimerA	Time-C	out Raw	Interrup	t			
									This th	e Timer	A time-	out interr	upt stat	us prior	to maski	ng.	

### Register 7: GPTM Masked Interrupt Status (GPTMMIS), offset 0x020

This register show the state of the GPTM's controller-level interrupt. If an interrupt is unmasked in **GPTMIMR**, and there is an event that causes the interrupt to be asserted, the corresponding bit is set in this register. All bits are cleared by writing a 1 to the corresponding bit in **GPTMICR**.

#### GPTM Masked Interrupt Status (GPTMMIS)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x020

Type RO, reset 0x0000.0000

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			'	' '			'	'	rese	rved	'	'	'				
www.DataSheet4U	Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			'	reserved			CBEMIS	CBMMIS	TBTOMIS		rese	erved	'	RTCMIS	CAEMIS	CAMMIS	TATOMIS
	Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	Bit/F	ield		Name		Туре	F	Reset	Descr	iption							
	31:	11		reserved		RO	1	0x00	compa	atibility v	vith futu	re produ	cts, the	of a rese value of operatio	a reserv		
	10	)		CBEMIS		RO		0	GPTM	1 Captur	eB Ever	nt Maske	ed Intern	upt			
									This is	s the Ca	ptureB e	event inte	errupt st	tatus afte	er maskir	ng.	
	9	1		CBMMIS		RO		0	GPTM	1 Captur	eB Mate	ch Maske	ed Interr	rupt			
									This is	s the Ca	ptureB r	natch int	terrupt s	status aft	er maski	ng.	
	8	1		твтоміѕ	3	RO		0	GPTM	1 TimerE	3 Time-C	Out Mask	ced Inter	rrupt			
									This is	s the Tin	nerB tim	e-out int	errupt s	tatus afte	er maskii	ng.	
	7:4	4		reserved		RO		0x0	compa	atibility v	vith futu	re produ	cts, the	of a rese value of operatio	a reserve		
	3	i		RTCMIS		RO		0	GPTM	1 RTC M	lasked I	nterrupt					
									This is	the RT	C event	interrup	t status	after ma	sking.		
	2	!		CAEMIS		RO		0	GPTM	1 Captur	eA Ever	nt Maske	ed Interr	upt			
									This is	s the Ca	ptureA e	event into	errupt st	tatus afte	er maskir	ng.	
	1			CAMMIS		RO		0	GPTM	1 Captur	eA Mato	ch Maske	ed Interr	rupt			
									This is	s the Ca	ptureA r	match int	terrupt s	status aft	er maski	ng.	
	0	١		TATOMIS	;	RO		0	GPTM	1 Timer/	A Time-C	Out Mask	ed Inter	rrupt			
									This is	s the Tin	nerA tim	e-out int	errupt s	tatus afte	er maskir	ng.	

www.DataSheet4U.com

### Register 8: GPTM Interrupt Clear (GPTMICR), offset 0x024

This register is used to clear the status bits in the GPTMRIS and GPTMMIS registers. Writing a 1 to a bit clears the corresponding bit in the **GPTMRIS** and **GPTMMIS** registers.

### GPTM Interrupt Clear (GPTMICR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x024 Type W1C, reset 0x0000.0000

ſ		1	1 1			1	1	rese	rved	•	1	ı	1	1	1	1	
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
		•	reserved			CBECINT	CBMCINT	TBTOCINT		rese	rved	'	RTCCINT	CAECINT	CAMCINT	TATOCINT	
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	W1C 0	W1C 0	W1C 0	RO 0	RO 0	RO 0	RO 0	W1C 0	W1C 0	W1C 0	W1C 0	
Bit/Fi	eld		Name		Туре	Type Reset			Description								
31:11			reserved				0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.									
10 CBECINT W1C 0 GPTM CaptureB E					eB Ever	nt Interru	pt Clea	ır									
								The C	BECINT	values	are defir	ned as f	follows:				
								Value	Descri	ption							
								0	The in	terrupt is	unaffec	cted.					
								1	The in	terrupt is	cleared	d.					
9		(	CBMCINT		W1C		0	GPTM	l Captur	eB Mato	h Interru	upt Clea	ar				
								The C	BMCINT	values	are defir	ned as f	follows:				
								Value	Descri	ption							
								0		terrupt is							
								1	The in	terrupt is	s cleared	1.					
8		Т	BTOCIN	Г	W1C		0	GPTM	I TimerE	3 Time-C	ut Interr	upt Cle	ar				
								The T	BTOCIN	T values	are def	ined as	follows:				
								Value	Descri	ption							
								0	The in	terrupt is	unaffec	cted.					
								1	The in	terrupt is	cleared	d.					
7:4	7:4 reserved RO 0x0 Software should not rely compatibility with future preserved across a read				e produ	cts, the	value of	a reserv									

Bit/Field	Name	Туре	Reset	Description
3	RTCCINT	W1C	0	GPTM RTC Interrupt Clear The RTCCINT values are defined as follows:
				Value Description  On The interrupt is unaffected.  The interrupt is cleared.
2 .com	CAECINT	W1C	0	GPTM CaptureA Event Interrupt Clear The CAECINT values are defined as follows:  Value Description 0 The interrupt is unaffected. 1 The interrupt is cleared.
1	CAMCINT	W1C	0	GPTM CaptureA Match Raw Interrupt This is the CaptureA match interrupt status after masking.
0	TATOCINT	W1C	0	GPTM TimerA Time-Out Raw Interrupt The TATOCINT values are defined as follows:
				Value Description 0 The interrupt is unaffected.

The interrupt is cleared.

200 November 29, 2007

### Register 9: GPTM TimerA Interval Load (GPTMTAILR), offset 0x028

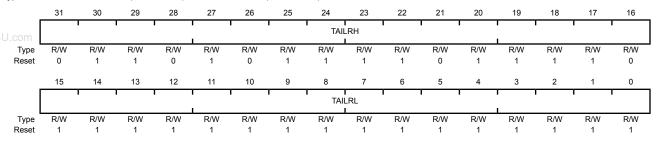
This register is used to load the starting count value into the timer. When GPTM is configured to one of the 32-bit modes, **GPTMTAILR** appears as a 32-bit register (the upper 16-bits correspond to the contents of the **GPTM TimerB Interval Load (GPTMTBILR)** register). In 16-bit mode, the upper 16 bits of this register read as 0s and have no effect on the state of **GPTMTBILR**.

#### GPTM TimerA Interval Load (GPTMTAILR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x028

Type R/W, reset 0x0000.FFFF (16-bit mode) and 0xFFFF.FFFF (32-bit mode)



Bit/Field	Name	Type	Reset	Description
31:16	TAILRH	R/W	0xFFFF (32-bit mode) 0x0000 (16-bit mode)	GPTM TimerA Interval Load Register High  When configured for 32-bit mode via the GPTMCFG register, the GPTM  TimerB Interval Load (GPTMTBILR) register loads this value on a write. A read returns the current value of GPTMTBILR.
				In 16-bit mode, this field reads as 0 and does not have an effect on the state of <b>GPTMTBILR</b> .
15:0	TAILRL	R/W	0xFFFF	GPTM TimerA Interval Load Register Low

For both 16- and 32-bit modes, writing this field loads the counter for TimerA. A read returns the current value of  ${\bf GPTMTAILR}.$ 

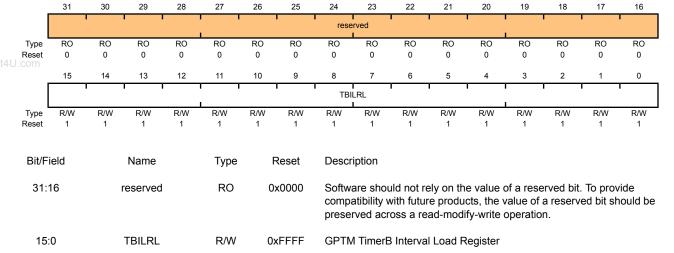
# Register 10: GPTM TimerB Interval Load (GPTMTBILR), offset 0x02C

This register is used to load the starting count value into TimerB. When the GPTM is configured to a 32-bit mode, **GPTMTBILR** returns the current value of TimerB and ignores writes.

#### GPTM TimerB Interval Load (GPTMTBILR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x02C

Type R/W, reset 0x0000.FFFF



When the GPTM is not configured as a 32-bit timer, a write to this field updates **GPTMTBILR**. In 32-bit mode, writes are ignored, and reads return the current value of **GPTMTBILR**.

# Register 11: GPTM TimerA Match (GPTMTAMATCHR), offset 0x030

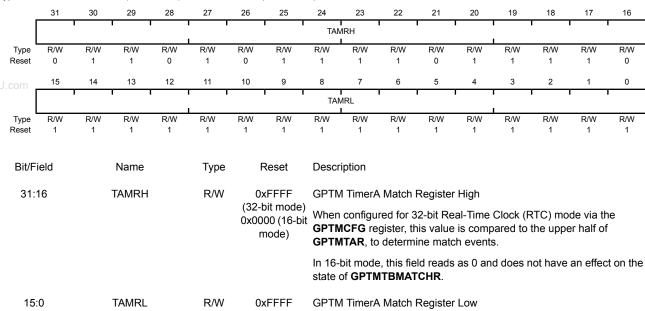
This register is used in 32-bit Real-Time Clock mode and 16-bit PWM and Input Edge Count modes.

#### GPTM TimerA Match (GPTMTAMATCHR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x030

Type R/W, reset 0x0000.FFFF (16-bit mode) and 0xFFFF.FFFF (32-bit mode)



When configured for 32-bit Real-Time Clock (RTC) mode via the **GPTMCFG** register, this value is compared to the lower half of **GPTMTAR**, to determine match events.

When configured for PWM mode, this value along with **GPTMTAILR**, determines the duty cycle of the output PWM signal.

When configured for Edge Count mode, this value along with **GPTMTAILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTMTAILR** minus this value.

### Register 12: GPTM TimerB Match (GPTMTBMATCHR), offset 0x034

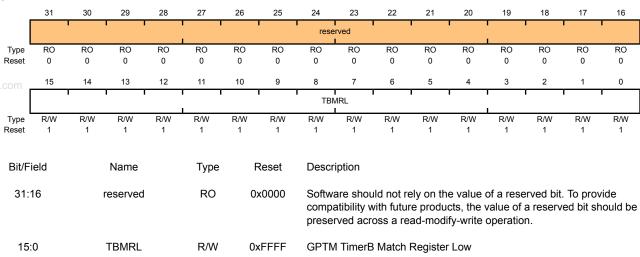
This register is used in 32-bit Real-Time Clock mode and 16-bit PWM and Input Edge Count modes.

#### GPTM TimerB Match (GPTMTBMATCHR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x034

Type R/W, reset 0x0000.FFFF



When configured for PWM mode, this value along with **GPTMTBILR**, determines the duty cycle of the output PWM signal.

When configured for Edge Count mode, this value along with **GPTMTBILR**, determines how many edge events are counted. The total number of edge events counted is equal to the value in **GPTMTBILR** minus this value.

# Register 13: GPTM TimerA Prescale (GPTMTAPR), offset 0x038

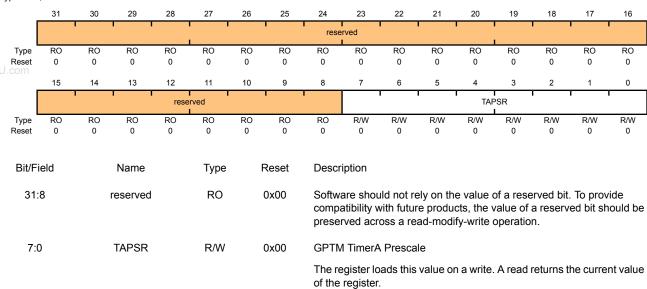
This register allows software to extend the range of the 16-bit timers when operating in one-shot or periodic mode.

#### GPTM TimerA Prescale (GPTMTAPR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x038

Type R/W, reset 0x0000.0000



Refer to Table 9-2 on page 179 for more details and an example.

# Register 14: GPTM TimerB Prescale (GPTMTBPR), offset 0x03C

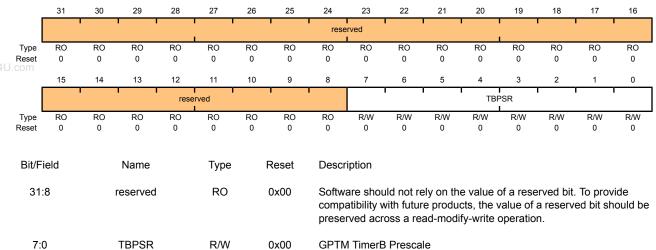
This register allows software to extend the range of the 16-bit timers when operating in one-shot or periodic mode.

#### GPTM TimerB Prescale (GPTMTBPR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x03C

Type R/W, reset 0x0000.0000

7:0



**GPTM TimerB Prescale** 

The register loads this value on a write. A read returns the current value of this register.

www.DataSheet4U.com

Refer to Table 9-2 on page 179 for more details and an example.

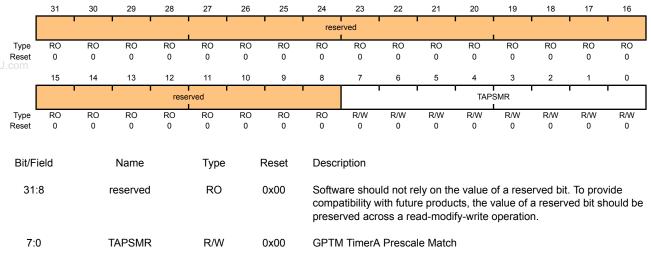
# Register 15: GPTM TimerA Prescale Match (GPTMTAPMR), offset 0x040

This register effectively extends the range of **GPTMTAMATCHR** to 24 bits when operating in 16-bit one-shot or periodic mode.

#### GPTM TimerA Prescale Match (GPTMTAPMR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x040

Type R/W, reset 0x0000.0000



This value is used alongside **GPTMTAMATCHR** to detect timer match events while using a prescaler.

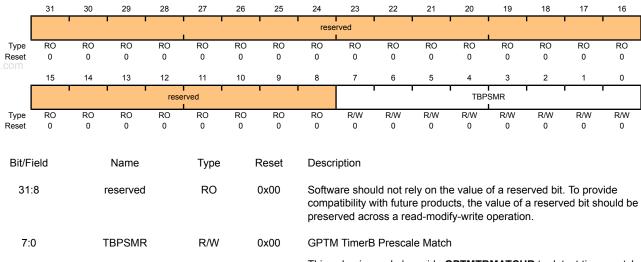
# Register 16: GPTM TimerB Prescale Match (GPTMTBPMR), offset 0x044

This register effectively extends the range of GPTMTBMATCHR to 24 bits when operating in 16-bit one-shot or periodic mode.

#### GPTM TimerB Prescale Match (GPTMTBPMR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x044

Type R/W, reset 0x0000.0000



This value is used alongside **GPTMTBMATCHR** to detect timer match events while using a prescaler.

www.DataSheet4U.com

### Register 17: GPTM TimerA (GPTMTAR), offset 0x048

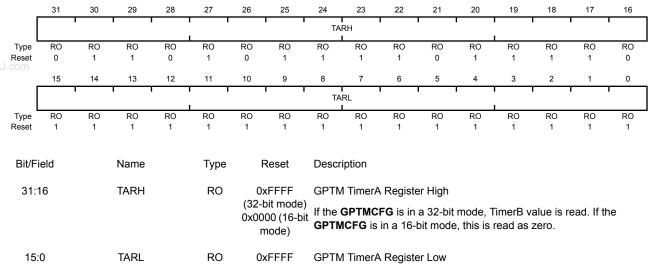
This register shows the current value of the TimerA counter in all cases except for Input Edge Count mode. When in this mode, this register contains the time at which the last edge event took place.

#### **GPTM TimerA (GPTMTAR)**

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000

Offset 0x048

Type RO, reset 0x0000.FFFF (16-bit mode) and 0xFFFF.FFFF (32-bit mode)



A read returns the current value of the **GPTM TimerA Count Register**, except in Input Edge Count mode, when it returns the timestamp from the last edge event.

www.DataSheet4U.com

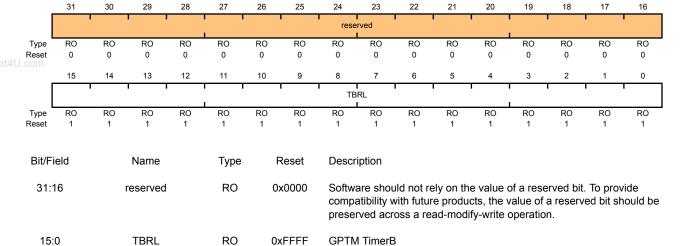
# Register 18: GPTM TimerB (GPTMTBR), offset 0x04C

This register shows the current value of the TimerB counter in all cases except for Input Edge Count mode. When in this mode, this register contains the time at which the last edge event took place.

#### GPTM TimerB (GPTMTBR)

Timer0 base: 0x4003.0000 Timer1 base: 0x4003.1000 Timer2 base: 0x4003.2000 Offset 0x04C

Type RO, reset 0x0000.FFFF



A read returns the current value of the **GPTM TimerB Count Register**, except in Input Edge Count mode, when it returns the timestamp from the last edge event.

# 10 Watchdog Timer

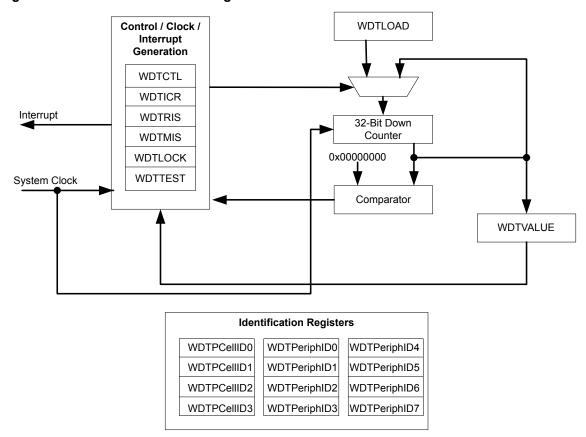
A watchdog timer can generate nonmaskable interrupts (NMIs) or a reset when a time-out value is reached. The watchdog timer is used to regain control when a system has failed due to a software error or due to the failure of an external device to respond in the expected way.

The Stellaris<sup>®</sup> Watchdog Timer module consists of a 32-bit down counter, a programmable load register, interrupt generation logic, a locking register, and user-enabled stalling.

The Watchdog Timer can be configured to generate an interrupt to the controller on its first time-out, and to generate a reset signal on its second time-out. Once the Watchdog Timer has been configured, the lock register can be written to prevent the timer configuration from being inadvertently altered.

# www.DataSheet410:1 Block Diagram

Figure 10-1. WDT Module Block Diagram



# 10.2 Functional Description

The Watchdog Timer module generates the first time-out signal when the 32-bit counter reaches the zero state after being enabled; enabling the counter also enables the watchdog timer interrupt. After the first time-out event, the 32-bit counter is re-loaded with the value of the **Watchdog Timer Load (WDTLOAD)** register, and the timer resumes counting down from that value. Once the

Watchdog Timer has been configured, the **Watchdog Timer Lock (WDTLOCK)** register is written, which prevents the timer configuration from being inadvertently altered by software.

If the timer counts down to its zero state again before the first time-out interrupt is cleared, and the reset signal has been enabled (via the WatchdogResetEnable function), the Watchdog timer asserts its reset signal to the system. If the interrupt is cleared before the 32-bit counter reaches its second time-out, the 32-bit counter is loaded with the value in the **WDTLOAD** register, and counting resumes from that value.

If **WDTLOAD** is written with a new value while the Watchdog Timer counter is counting, then the counter is loaded with the new value and continues counting.

Writing to **WDTLOAD** does not clear an active interrupt. An interrupt must be specifically cleared by writing to the **Watchdog Interrupt Clear (WDTICR)** register.

The Watchdog module interrupt and reset generation can be enabled or disabled as required. When the interrupt is re-enabled, the 32-bit counter is preloaded with the load register value and not its last state.

### 10.3 Initialization and Configuration

To use the WDT, its peripheral clock must be enabled by setting the WDT bit in the **RCGC0** register. The Watchdog Timer is configured using the following sequence:

- 1. Load the **WDTLOAD** register with the desired timer load value.
- If the Watchdog is configured to trigger system resets, set the RESEN bit in the WDTCTL register.
- Set the INTEN bit in the WDTCTL register to enable the Watchdog and lock the control register.

If software requires that all of the watchdog registers are locked, the Watchdog Timer module can be fully locked by writing any value to the **WDTLOCK** register. To unlock the Watchdog Timer, write a value of 0x1ACC.E551.

# 10.4 Register Map

Table 10-1 on page 212 lists the Watchdog registers. The offset listed is a hexadecimal increment to the register's address, relative to the Watchdog Timer base address of 0x4000.0000.

Table 10-1. Watchdog Timer Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	WDTLOAD	R/W	0xFFFF.FFFF	Watchdog Load	214
0x004	WDTVALUE	RO	0xFFFF.FFFF	Watchdog Value	215
800x0	WDTCTL	R/W	0x0000.0000	Watchdog Control	216
0x00C	WDTICR	WO	-	Watchdog Interrupt Clear	217
0x010	WDTRIS	RO	0x0000.0000	Watchdog Raw Interrupt Status	218
0x014	WDTMIS	RO	0x0000.0000	Watchdog Masked Interrupt Status	219
0x418	WDTTEST	R/W	0x0000.0000	Watchdog Test	220
0xC00	WDTLOCK	R/W	0x0000.0000	Watchdog Lock	221

212 November 29, 2007
Preliminary

www.DataSheet4U.com

Offset	Name	Туре	Reset	Description	See page
0xFD0	WDTPeriphID4	RO	0x0000.0000	Watchdog Peripheral Identification 4	222
0xFD4	WDTPeriphID5	RO	0x0000.0000	Watchdog Peripheral Identification 5	223
0xFD8	WDTPeriphID6	RO	0x0000.0000	Watchdog Peripheral Identification 6	224
0xFDC	WDTPeriphID7	RO	0x0000.0000	Watchdog Peripheral Identification 7	225
0xFE0	WDTPeriphID0	RO	0x0000.0005	Watchdog Peripheral Identification 0	226
0xFE4	WDTPeriphID1	RO	0x0000.0018	Watchdog Peripheral Identification 1	227
0xFE8	WDTPeriphID2	RO	0x0000.0018	Watchdog Peripheral Identification 2	228
0xFEC	WDTPeriphID3	RO	0x0000.0001	Watchdog Peripheral Identification 3	229
0xFF0	WDTPCellID0	RO	0x0000.000D	Watchdog PrimeCell Identification 0	230
0xFF4	WDTPCellID1	RO	0x0000.00F0	Watchdog PrimeCell Identification 1	231
0xFF8	WDTPCellID2	RO	0x0000.0005	Watchdog PrimeCell Identification 2	232
0xFFC	WDTPCellID3	RO	0x0000.00B1	Watchdog PrimeCell Identification 3	233

10.5 Register Descriptions

The remainder of this section lists and describes the WDT registers, in numerical order by address offset.

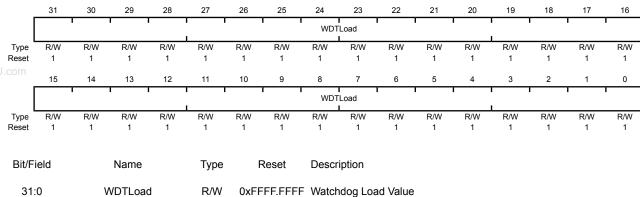
### Register 1: Watchdog Load (WDTLOAD), offset 0x000

This register is the 32-bit interval value used by the 32-bit counter. When this register is written, the value is immediately loaded and the counter restarts counting down from the new value. If the **WDTLOAD** register is loaded with 0x0000.0000, an interrupt is immediately generated.

Watchdog Load (WDTLOAD)

Base 0x4000.0000

Offset 0x000 Type R/W, reset 0xFFFF.FFF



### Register 2: Watchdog Value (WDTVALUE), offset 0x004

This register contains the current count value of the timer.

RO

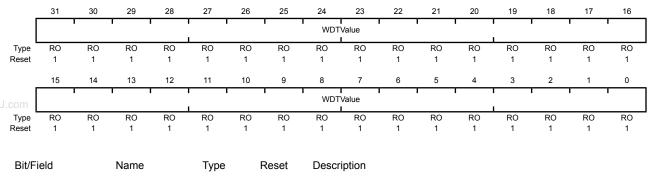
Watchdog Value (WDTVALUE)

Base 0x4000.0000 Offset 0x004

31:0

WDTValue

Type RO, reset 0xFFFF.FFF



0xFFFF.FFFF Watchdog Value

Current value of the 32-bit down counter.

### Register 3: Watchdog Control (WDTCTL), offset 0x008

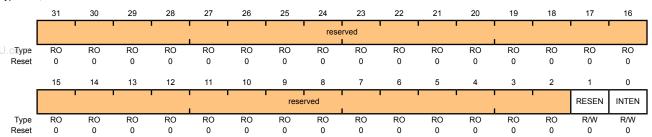
This register is the watchdog control register. The watchdog timer can be configured to generate a reset signal (on second time-out) or an interrupt on time-out.

When the watchdog interrupt has been enabled, all subsequent writes to the control register are ignored. The only mechanism that can re-enable writes is a hardware reset.

### Watchdog Control (WDTCTL)

Base 0x4000.0000 Offset 0x008

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	RESEN	R/W	0	Watchdog Reset Enable
				The RESEN values are defined as follows:
				Value Description
				0 Disabled.
				1 Enable the Watchdog module reset output.
0	INTEN	R/W	0	Watchdog Interrupt Enable
				The INTEN values are defined as follows:

#### Value Description

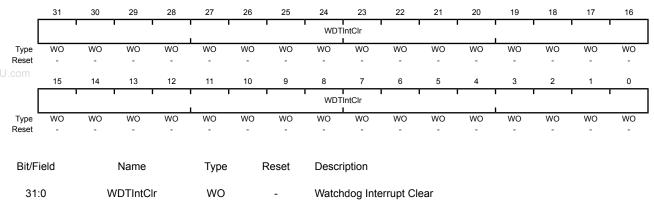
- 0 Interrupt event disabled (once this bit is set, it can only be cleared by a hardware reset).
- 1 Interrupt event enabled. Once enabled, all writes are ignored.

#### Register 4: Watchdog Interrupt Clear (WDTICR), offset 0x00C

This register is the interrupt clear register. A write of any value to this register clears the Watchdog interrupt and reloads the 32-bit counter from the **WDTLOAD** register. Value for a read or reset is indeterminate.

Watchdog Interrupt Clear (WDTICR)

Base 0x4000.0000 Offset 0x00C Type WO, reset -

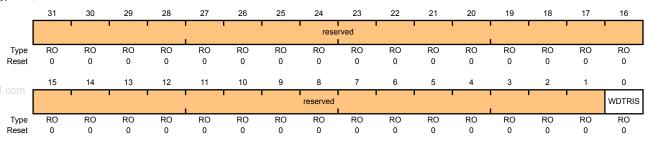


### Register 5: Watchdog Raw Interrupt Status (WDTRIS), offset 0x010

This register is the raw interrupt status register. Watchdog interrupt events can be monitored via this register if the controller interrupt is masked.

Watchdog Raw Interrupt Status (WDTRIS)

Base 0x4000.0000 Offset 0x010 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	WDTRIS	RO	0	Watchdog Raw Interrupt Status

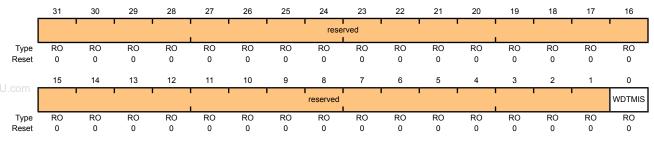
Gives the raw interrupt state (prior to masking) of WDTINTR.

#### Register 6: Watchdog Masked Interrupt Status (WDTMIS), offset 0x014

This register is the masked interrupt status register. The value of this register is the logical AND of the raw interrupt bit and the Watchdog interrupt enable bit.

Watchdog Masked Interrupt Status (WDTMIS)

Base 0x4000.0000 Offset 0x014 Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	WDTMIS	RO	0	Watchdog Masked Interrupt Status

Gives the masked interrupt state (after masking) of the WDTINTR interrupt.

#### Register 7: Watchdog Test (WDTTEST), offset 0x418

RO

reserved

0x00

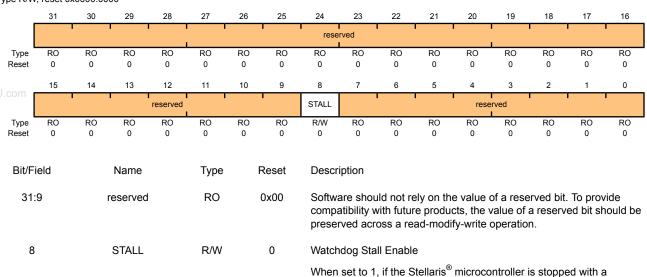
This register provides user-enabled stalling when the microcontroller asserts the CPU halt flag during debug.

#### Watchdog Test (WDTTEST)

Base 0x4000.0000

7:0

Offset 0x418 Type R/W, reset 0x0000.0000



Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be

debugger, the watchdog timer stops counting. Once the microcontroller

preserved across a read-modify-write operation.

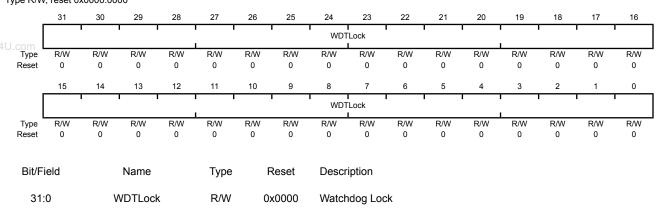
is restarted, the watchdog timer resumes counting.

#### Register 8: Watchdog Lock (WDTLOCK), offset 0xC00

Writing 0x1ACC.E551 to the **WDTLOCK** register enables write access to all other registers. Writing any other value to the **WDTLOCK** register re-enables the locked state for register writes to all the other registers. Reading the **WDTLOCK** register returns the lock status rather than the 32-bit value written. Therefore, when write accesses are disabled, reading the **WDTLOCK** register returns 0x0000.0001 (when locked; otherwise, the returned value is 0x0000.0000 (unlocked)).

#### Watchdog Lock (WDTLOCK)

Base 0x4000.0000 Offset 0xC00 Type R/W, reset 0x0000.0000



A write of the value 0x1ACC.E551 unlocks the watchdog registers for write access. A write of any other value reapplies the lock, preventing any register updates.

A read of this register returns the following values:

Value Description
0x0000.0001 Locked
0x0000.0000 Unlocked

#### Register 9: Watchdog Peripheral Identification 4 (WDTPeriphID4), offset 0xFD0

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 4 (WDTPeriphID4)

PID4

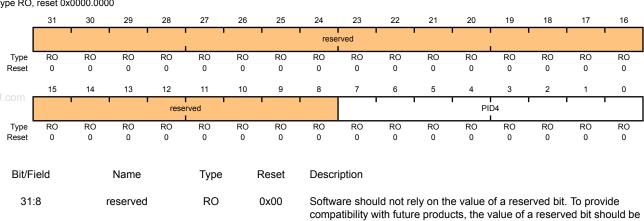
RO

0x00

Base 0x4000.0000

7:0

Offset 0xFD0
Type RO, reset 0x0000.0000



preserved across a read-modify-write operation.

WDT Peripheral ID Register[7:0]

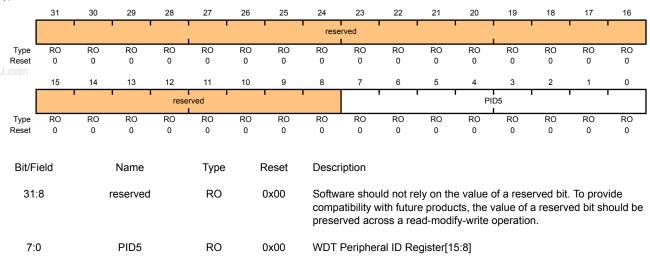
# Register 10: Watchdog Peripheral Identification 5 (WDTPeriphID5), offset 0xFD4

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 5 (WDTPeriphID5)

Base 0x4000.0000

Offset 0xFD4
Type RO, reset 0x0000.0000



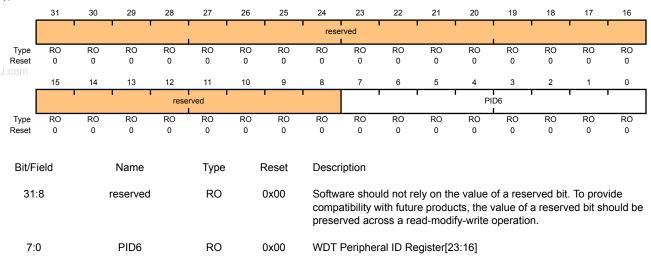
#### Register 11: Watchdog Peripheral Identification 6 (WDTPeriphID6), offset 0xFD8

The WDTPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 6 (WDTPeriphID6)

Base 0x4000.0000

Offset 0xFD8
Type RO, reset 0x0000.0000



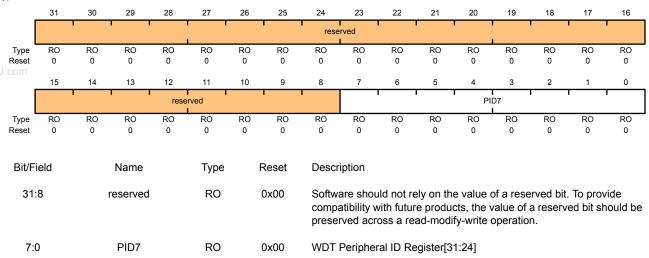
# Register 12: Watchdog Peripheral Identification 7 (WDTPeriphID7), offset 0xFDC

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 7 (WDTPeriphID7)

Base 0x4000.0000

Offset 0xFDC Type RO, reset 0x0000.0000



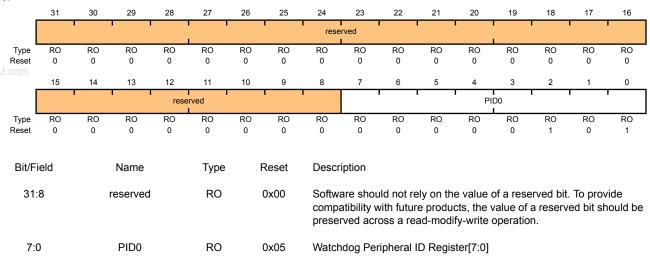
#### Register 13: Watchdog Peripheral Identification 0 (WDTPeriphID0), offset 0xFE0

The WDTPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 0 (WDTPeriphID0)

Base 0x4000.0000

Offset 0xFE0
Type RO, reset 0x0000.0005



www.DataSheet4U.com

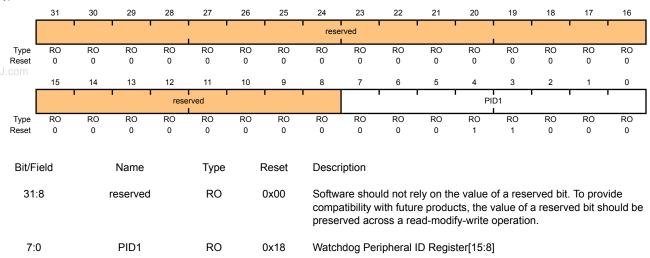
# Register 14: Watchdog Peripheral Identification 1 (WDTPeriphID1), offset 0xFE4

The **WDTPeriphIDn** registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 1 (WDTPeriphID1)

Base 0x4000.0000

Offset 0xFE4
Type RO, reset 0x0000.0018



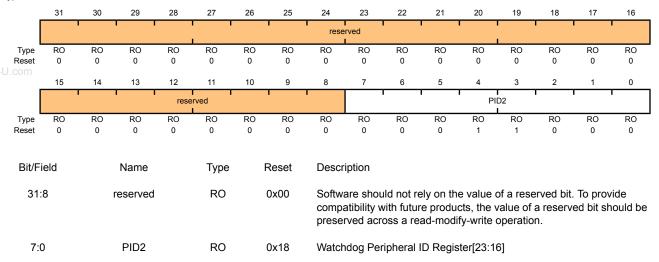
#### Register 15: Watchdog Peripheral Identification 2 (WDTPeriphID2), offset 0xFE8

The WDTPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 2 (WDTPeriphID2)

Base 0x4000.0000

Offset 0xFE8
Type RO, reset 0x0000.0018



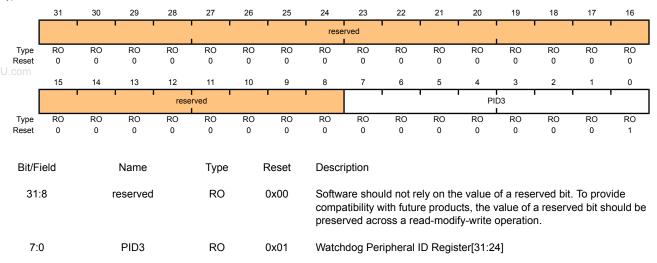
#### Register 16: Watchdog Peripheral Identification 3 (WDTPeriphID3), offset 0xFEC

The WDTPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog Peripheral Identification 3 (WDTPeriphID3)

Base 0x4000.0000

Offset 0xFEC
Type RO, reset 0x0000.0001



#### Register 17: Watchdog PrimeCell Identification 0 (WDTPCellID0), offset 0xFF0

The WDTPCellIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 0 (WDTPCellID0)

reserved

CID0

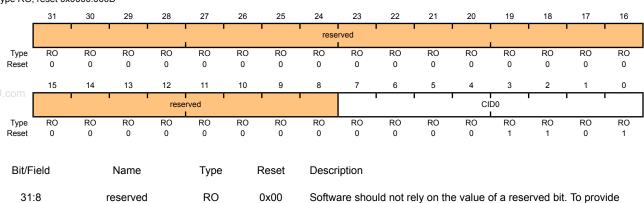
RO

0x0D

Base 0x4000.0000

7:0

Offset 0xFF0
Type RO, reset 0x0000.000D



compatibility with future products, the value of a reserved bit should be

preserved across a read-modify-write operation.

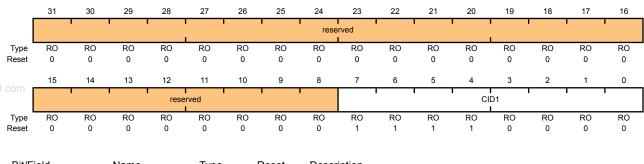
Watchdog PrimeCell ID Register[7:0]

#### Register 18: Watchdog PrimeCell Identification 1 (WDTPCellID1), offset 0xFF4

The WDTPCellIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 1 (WDTPCellID1)

Base 0x4000.0000 Offset 0xFF4 Type RO, reset 0x0000.00F0



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	Watchdog PrimeCell ID Register[15:8]

#### Register 19: Watchdog PrimeCell Identification 2 (WDTPCellID2), offset 0xFF8

The WDTPCellIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 2 (WDTPCellID2)

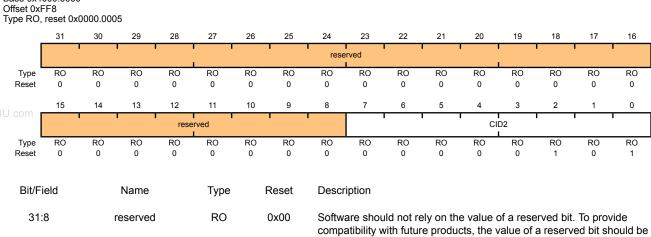
CID2

RO

0x05

Base 0x4000.0000

7:0



preserved across a read-modify-write operation.

Watchdog PrimeCell ID Register[23:16]

www.DataSheet4U.com

#### Register 20: Watchdog PrimeCell Identification 3 (WDTPCellID3), offset 0xFFC

The WDTPCellIDn registers are hard-coded and the fields within the register determine the reset value.

Watchdog PrimeCell Identification 3 (WDTPCellID3)

reserved

CID3

RO

RO

0x00

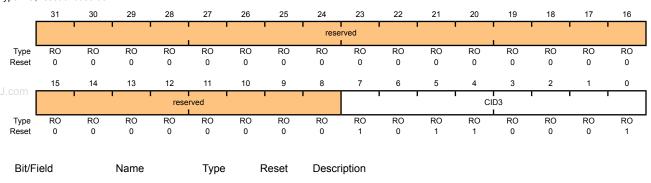
0xB1

Base 0x4000.0000

31:8

7:0

Offset 0xFFC Type RO, reset 0x0000.00B1



Software should not rely on the value of a reserved bit. To provide

preserved across a read-modify-write operation.

Watchdog PrimeCell ID Register[31:24]

compatibility with future products, the value of a reserved bit should be

# 11 Universal Asynchronous Receivers/Transmitters (UARTs)

The Stellaris<sup>®</sup> Universal Asynchronous Receiver/Transmitter (UART) provides fully programmable, 16C550-type serial interface characteristics. The LM3S2110 controller is equipped with one UART module.

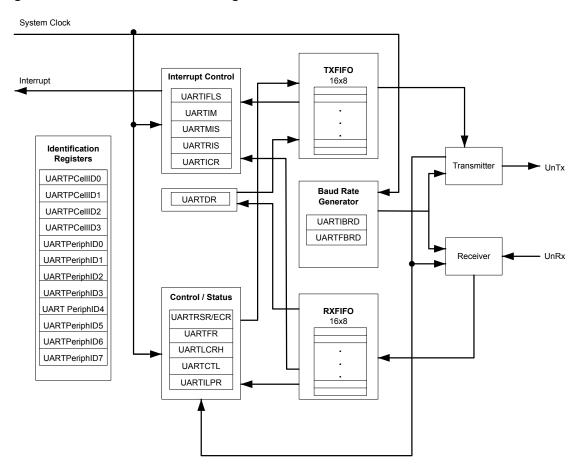
The UART has the following features:

- Separate transmit and receive FIFOs
- Programmable FIFO length, including 1-byte deep operation providing conventional double-buffered interface
- FIFO trigger levels of 1/8, 1/4, 1/2, 3/4, and 7/8
- Programmable baud-rate generator allowing rates up to 1.5625 Mbps
- Standard asynchronous communication bits for start, stop, and parity
- False start bit detection
- Line-break generation and detection
- Fully programmable serial interface characteristics:
  - 5, 6, 7, or 8 data bits
  - Even, odd, stick, or no-parity bit generation/detection
  - 1 or 2 stop bit generation
- IrDA serial-IR (SIR) encoder/decoder providing:
  - Programmable use of IrDA Serial InfraRed (SIR) or UART input/output
  - Support of IrDA SIR encoder/decoder functions for data rates up to 115.2 Kbps half-duplex
  - Support of normal 3/16 and low-power (1.41-2.23 µs) bit durations
  - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration

ww.DataSheet4U.com

#### 11.1 Block Diagram

Figure 11-1. UART Module Block Diagram



## 11.2 Functional Description

Each Stellaris<sup>®</sup> UART performs the functions of parallel-to-serial and serial-to-parallel conversions. It is similar in functionality to a 16C550 UART, but is not register compatible.

The UART is configured for transmit and/or receive via the TXE and RXE bits of the **UART Control** (**UARTCTL**) register (see page 253). Transmit and receive are both enabled out of reset. Before any control registers are programmed, the UART must be disabled by clearing the UARTEN bit in **UARTCTL**. If the UART is disabled during a TX or RX operation, the current transaction is completed prior to the UART stopping.

The UART peripheral also includes a serial IR (SIR) encoder/decoder block that can be connected to an infrared transceiver to implement an IrDA SIR physical layer. The SIR function is programmed using the UARTCTL register.

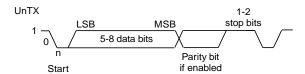
#### 11.2.1 Transmit/Receive Logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. The control logic outputs the serial bit stream beginning with a start bit, and followed by the data

bits (LSB first), parity bit, and the stop bits according to the programmed configuration in the control registers. See Figure 11-2 on page 236 for details.

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Overrun, parity, frame error checking, and line-break detection are also performed, and their status accompanies the data that is written to the receive FIFO.

Figure 11-2. UART Character Frame



001-0.00111

#### 11.2.2 Baud-Rate Generation

The baud-rate divisor is a 22-bit number consisting of a 16-bit integer and a 6-bit fractional part. The number formed by these two values is used by the baud-rate generator to determine the bit period. Having a fractional baud-rate divider allows the UART to generate all the standard baud rates.

The 16-bit integer is loaded through the **UART Integer Baud-Rate Divisor (UARTIBRD)** register (see page 249) and the 6-bit fractional part is loaded with the **UART Fractional Baud-Rate Divisor (UARTFBRD)** register (see page 250). The baud-rate divisor (BRD) has the following relationship to the system clock (where *BRDI* is the integer part of the BRD and *BRDF* is the fractional part, separated by a decimal place.):

```
BRD = BRDI + BRDF = SysClk / (16 * Baud Rate)
```

The 6-bit fractional number (that is to be loaded into the DIVFRAC bit field in the **UARTFBRD** register) can be calculated by taking the fractional part of the baud-rate divisor, multiplying it by 64, and adding 0.5 to account for rounding errors:

```
UARTFBRD[DIVFRAC] = integer(BRDF * 64 + 0.5)
```

The UART generates an internal baud-rate reference clock at 16x the baud-rate (referred to as Baud16). This reference clock is divided by 16 to generate the transmit clock, and is used for error detection during receive operations.

Along with the **UART Line Control**, **High Byte (UARTLCRH)** register (see page 251), the **UARTIBRD** and **UARTFBRD** registers form an internal 30-bit register. This internal register is only updated when a write operation to **UARTLCRH** is performed, so any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register for the changes to take effect.

To update the baud-rate registers, there are four possible sequences:

- UARTIBRD write, UARTFBRD write, and UARTLCRH write
- UARTFBRD write, UARTIBRD write, and UARTLCRH write
- UARTIBRD write and UARTLCRH write
- UARTFBRD write and UARTLCRH write

#### 11.2.3 Data Transmission

Data received or transmitted is stored in two 16-byte FIFOs, though the receive FIFO has an extra four bits per character for status information. For transmission, data is written into the transmit FIFO. If the UART is enabled, it causes a data frame to start transmitting with the parameters indicated in the **UARTLCRH** register. Data continues to be transmitted until there is no data left in the transmit FIFO. The BUSY bit in the **UART Flag (UARTFR)** register (see page 246) is asserted as soon as data is written to the transmit FIFO (that is, if the FIFO is non-empty) and remains asserted while data is being transmitted. The BUSY bit is negated only when the transmit FIFO is empty, and the last character has been transmitted from the shift register, including the stop bits. The UART can indicate that it is busy even though the UART may no longer be enabled.

When the receiver is idle (the UnRx is continuously 1) and the data input goes Low (a start bit has been received), the receive counter begins running and data is sampled on the eighth cycle of Baud16 (described in "Transmit/Receive Logic" on page 235).

The start bit is valid if UnRx is still low on the eighth cycle of Baud16, otherwise a false start bit is detected and it is ignored. Start bit errors can be viewed in the **UART Receive Status (UARTRSR)** register (see page 244). If the start bit was valid, successive data bits are sampled on every 16th cycle of Baud16 (that is, one bit period later) according to the programmed length of the data characters. The parity bit is then checked if parity mode was enabled. Data length and parity are defined in the **UARTLCRH** register.

Lastly, a valid stop bit is confirmed if UnRx is High, otherwise a framing error has occurred. When a full word is received, the data is stored in the receive FIFO, with any error bits associated with that word.

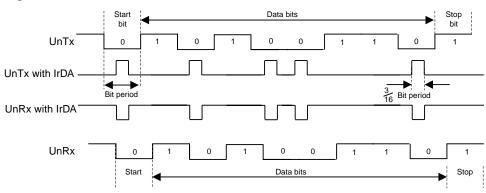
#### 11.2.4 Serial IR (SIR)

The UART peripheral includes an IrDA serial-IR (SIR) encoder/decoder block. The IrDA SIR block provides functionality that converts between an asynchronous UART data stream, and half-duplex serial SIR interface. No analog processing is performed on-chip. The role of the SIR block is to provide a digital encoded output, and decoded input to the UART. The UART signal pins can be connected to an infrared transceiver to implement an IrDA SIR physical layer link. The SIR block has two modes of operation:

- In normal IrDA mode, a zero logic level is transmitted as high pulse of 3/16th duration of the selected baud rate bit period on the output pin, while logic one levels are transmitted as a static LOW signal. These levels control the driver of an infrared transmitter, sending a pulse of light for each zero. On the reception side, the incoming light pulses energize the photo transistor base of the receiver, pulling its output LOW. This drives the UART input pin LOW.
- In low-power IrDA mode, the width of the transmitted infrared pulse is set to three times the period of the internally generated IrLPBaud16 signal (1.63 μs, assuming a nominal 1.8432 MHz frequency) by changing the appropriate bit in the UARTCR register.

Figure 11-3 on page 238 shows the UART transmit and receive signals, with and without IrDA modulation.

Figure 11-3. IrDA Data Modulation



ww.DataSheet4U.com

In both normal and low-power IrDA modes:

- During transmission, the UART data bit is used as the base for encoding
- During reception, the decoded bits are transferred to the UART receive logic

The IrDA SIR physical layer specifies a half-duplex communication link, with a minimum 10 ms delay between transmission and reception. This delay must be generated by software because it is not automatically supported by the UART. The delay is required because the infrared receiver electronics might become biased, or even saturated from the optical power coupled from the adjacent transmitter LED. This delay is known as latency, or receiver setup time.

#### 11.2.5 FIFO Operation

The UART has two 16-entry FIFOs; one for transmit and one for receive. Both FIFOs are accessed via the **UART Data (UARTDR)** register (see page 242). Read operations of the **UARTDR** register return a 12-bit value consisting of 8 data bits and 4 error flags while write operations place 8-bit data in the transmit FIFO.

Out of reset, both FIFOs are disabled and act as 1-byte-deep holding registers. The FIFOs are enabled by setting the FEN bit in **UARTLCRH** (page 251).

FIFO status can be monitored via the **UART Flag (UARTFR)** register (see page 246) and the **UART Receive Status (UARTRSR)** register. Hardware monitors empty, full and overrun conditions. The **UARTFR** register contains empty and full flags (TXFE, TXFF, RXFE, and RXFF bits) and the **UARTRSR** register shows overrun status via the OE bit.

The trigger points at which the FIFOs generate interrupts is controlled via the **UART Interrupt FIFO Level Select (UARTIFLS)** register (see page 255). Both FIFOs can be individually configured to trigger interrupts at different levels. Available configurations include 1/8, ½, ½, ¾, and 7/8. For example, if the ¼ option is selected for the receive FIFO, the UART generates a receive interrupt after 4 data bytes are received. Out of reset, both FIFOs are configured to trigger an interrupt at the ½ mark.

#### 11.2.6 Interrupts

The UART can generate interrupts when the following conditions are observed:

- Overrun Error
- Break Error

- Parity Error
- Framing Error
- Receive Timeout
- Transmit (when condition defined in the TXIFLSEL bit in the UARTIFLS register is met)
- Receive (when condition defined in the RXIFLSEL bit in the UARTIFLS register is met)

All of the interrupt events are ORed together before being sent to the interrupt controller, so the UART can only generate a single interrupt request to the controller at any given time. Software can service multiple interrupt events in a single interrupt service routine by reading the **UART Masked Interrupt Status (UARTMIS)** register (see page 260).

The interrupt events that can trigger a controller-level interrupt are defined in the **UART Interrupt Mask (UARTIM**) register (see page 257) by setting the corresponding IM bit to 1. If interrupts are not used, the raw interrupt status is always visible via the **UART Raw Interrupt Status (UARTRIS)** register (see page 259).

Interrupts are always cleared (for both the **UARTMIS** and **UARTRIS** registers) by setting the corresponding bit in the **UART Interrupt Clear (UARTICR)** register (see page 261).

The receive timeout interrupt is asserted when the receive FIFO is not empty, and no further data is received over a 32-bit period. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when a 1 is written to the corresponding bit in the **UARTICR** register.

#### 11.2.7 Loopback Operation

The UART can be placed into an internal loopback mode for diagnostic or debug work. This is accomplished by setting the LBE bit in the **UARTCTL** register (see page 253). In loopback mode, data transmitted on UnTx is received on the UnRx input.

#### 11.2.8 IrDA SIR block

The IrDA SIR block contains an IrDA serial IR (SIR) protocol encoder/decoder. When enabled, the SIR block uses the  $\mathtt{UnTx}$  and  $\mathtt{UnRx}$  pins for the SIR protocol, which should be connected to an IR transceiver.

The SIR block can receive and transmit, but it is only half-duplex so it cannot do both at the same time. Transmission must be stopped before data can be received. The IrDA SIR physical layer specifies a minimum 10-ms delay between transmission and reception.

#### 11.3 Initialization and Configuration

To use the UART, the peripheral clock must be enabled by setting the UARTO bit in the **RCGC1** register.

This section discusses the steps that are required for using a UART module. For this example, the system clock is assumed to be 20 MHz and the desired UART configuration is:

- 115200 baud rate
- Data length of 8 bits
- One stop bit

November 29, 2007 239

- No parity
- FIFOs disabled
- No interrupts

The first thing to consider when programming the UART is the baud-rate divisor (BRD), since the **UARTIBRD** and **UARTFBRD** registers must be written before the **UARTLCRH** register. Using the equation described in "Baud-Rate Generation" on page 236, the BRD can be calculated:

```
BRD = 20,000,000 / (16 * 115,200) = 10.8507
```

which means that the DIVINT field of the **UARTIBRD** register (see page 249) should be set to 10. The value to be loaded into the **UARTFBRD** register (see page 250) is calculated by the equation:

```
UARTFBRD[DIVFRAC] = integer(0.8507 * 64 + 0.5) = 54
```

With the BRD values in hand, the UART configuration is written to the module in the following order:

- 1. Disable the UART by clearing the UARTEN bit in the **UARTCTL** register.
- Write the integer portion of the BRD to the **UARTIBRD** register.
- 3. Write the fractional portion of the BRD to the **UARTFBRD** register.
- Write the desired serial parameters to the **UARTLCRH** register (in this case, a value of 0x0000.0060).
- 5. Enable the UART by setting the UARTEN bit in the **UARTCTL** register.

#### 11.4 Register Map

Table 11-1 on page 240 lists the UART registers. The offset listed is a hexadecimal increment to the register's address, relative to that UART's base address:

UART0: 0x4000.C000

Note: The UART must be disabled (see the UARTEN bit in the **UARTCTL** register on page 253) before any of the control registers are reprogrammed. When the UART is disabled during a TX or RX operation, the current transaction is completed prior to the UART stopping.

Table 11-1. UART Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	UARTDR	R/W	0x0000.0000	UART Data	242
0x004	UARTRSR/UARTECR	R/W	0x0000.0000	UART Receive Status/Error Clear	244
0x018	UARTFR	RO	0x0000.0090	UART Flag	246
0x020	UARTILPR	R/W	0x0000.0000	UART IrDA Low-Power Register	248
0x024	UARTIBRD	R/W	0x0000.0000	UART Integer Baud-Rate Divisor	249
0x028	UARTFBRD	R/W	0x0000.0000	UART Fractional Baud-Rate Divisor	250

240 November 29, 2007
Preliminary

Offset	Name	Туре	Reset	Description	See page
0x02C	UARTLCRH	R/W	0x0000.0000	UART Line Control	251
0x030	UARTCTL	R/W	0x0000.0300	UART Control	253
0x034	UARTIFLS	R/W	0x0000.0012	UART Interrupt FIFO Level Select	255
0x038	UARTIM	R/W	0x0000.0000	UART Interrupt Mask	257
0x03C	UARTRIS	RO	0x0000.000F	UART Raw Interrupt Status	259
0x040	UARTMIS	RO	0x0000.0000	UART Masked Interrupt Status	260
0x044	UARTICR	W1C	0x0000.0000	UART Interrupt Clear	261
0xFD0	UARTPeriphID4	RO	0x0000.0000	UART Peripheral Identification 4	263
0xFD4	UARTPeriphID5	RO	0x0000.0000	UART Peripheral Identification 5	264
0xFD8	UARTPeriphID6	RO	0x0000.0000	UART Peripheral Identification 6	265
0xFDC	UARTPeriphID7	RO	0x0000.0000	UART Peripheral Identification 7	266
0xFE0	UARTPeriphID0	RO	0x0000.0011	UART Peripheral Identification 0	267
0xFE4	UARTPeriphID1	RO	0x0000.0000	UART Peripheral Identification 1	268
0xFE8	UARTPeriphID2	RO	0x0000.0018	UART Peripheral Identification 2	269
0xFEC	UARTPeriphID3	RO	0x0000.0001	UART Peripheral Identification 3	270
0xFF0	UARTPCellID0	RO	0x0000.000D	UART PrimeCell Identification 0	271
0xFF4	UARTPCellID1	RO	0x0000.00F0	UART PrimeCell Identification 1	272
0xFF8	UARTPCellID2	RO	0x0000.0005	UART PrimeCell Identification 2	273
0xFFC	UARTPCellID3	RO	0x0000.00B1	UART PrimeCell Identification 3	274

## 11.5 Register Descriptions

The remainder of this section lists and describes the UART registers, in numerical order by address offset.

November 29, 2007

#### Register 1: UART Data (UARTDR), offset 0x000

This register is the data register (the interface to the FIFOs).

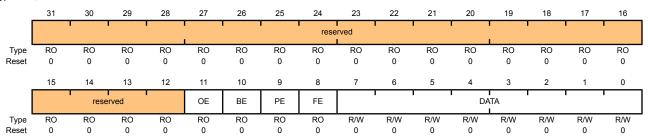
When FIFOs are enabled, data written to this location is pushed onto the transmit FIFO. If FIFOs are disabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO). A write to this register initiates a transmission from the UART.

For received data, if the FIFO is enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO. If FIFOs are disabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO). The received data can be retrieved by reading this register.

#### **UART Data (UARTDR)**

UART0 base: 0x4000.C000 Offset 0x000 Type R/W, reset 0x0000.0000

9



Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11	OE	RO	0	UART Overrun Error  The OE values are defined as follows:
				Value Description

- There has been no data loss due to a FIFO overrun.
- New data was received when the FIFO was full, resulting in data loss.

10	BE	RO	0	UART Break Error
				This bit is set to 1 when a break condition is detected, indicating that the receive data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits).
				In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the

ne goes to a 1 (marking state) and the next valid start bit is received.

#### PΕ RO 0 **UART Parity Error**

This bit is set to 1 when the parity of the received data character does not match the parity defined by bits 2 and 7 of the **UARTLCRH** register.

In FIFO mode, this error is associated with the character at the top of the FIFO.

Bit/Field	Name	Туре	Reset	Description
8	FE	RO	0	UART Framing Error
				This bit is set to 1 when the received character does not have a valid stop bit (a valid stop bit is 1).
7:0	DATA	R/W	0	Data Transmitted or Received
				When written, the data that is to be transmitted via the UART. When read, the data that was received by the UART.

## Register 2: UART Receive Status/Error Clear (UARTRSR/UARTECR), offset 0x004

The **UARTRSR/UARTECR** register is the receive status register/error clear register.

In addition to the **UARTDR** register, receive status can also be read from the **UARTRSR** register. If the status is read from this register, then the status information corresponds to the entry read from **UARTDR** prior to reading **UARTRSR**. The status information for overrun is set immediately when an overrun condition occurs.

The **UARTRSR** register cannot be written.

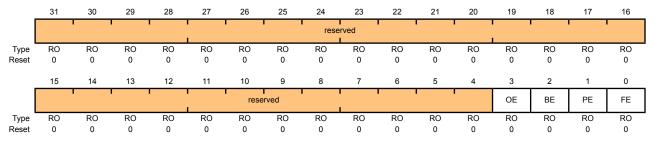
A write of any value to the **UARTECR** register clears the framing, parity, break, and overrun errors. All the bits are cleared to 0 on reset.

#### Read-Only Receive Status (UARTRSR) Register

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000 Offset 0x004

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	OE	RO	0	UART Overrun Error
				When this bit is set to 1, data is received and the FIFO is already full. This bit is cleared to 0 by a write to <b>UARTECR</b> .
				The FIFO contents remain valid since no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The CPU must now read the data in order to empty the FIFO.
2	BE	RO	0	UART Break Error

This bit is set to 1 when a break condition is detected, indicating that the received data input was held Low for longer than a full-word transmission time (defined as start, data, parity, and stop bits).

This bit is cleared to 0 by a write to **UARTECR**.

In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state) and the next valid start bit is received.

Bit/Field	Name	Туре	Reset	Description
1	PE	RO	0	UART Parity Error
				This bit is set to 1 when the parity of the received data character does not match the parity defined by bits 2 and 7 of the <b>UARTLCRH</b> register.
				This bit is cleared to 0 by a write to <b>UARTECR</b> .
0	FE	RO	0	UART Framing Error
				This bit is set to 1 when the received character does not have a valid stop bit (a valid stop bit is 1).

This bit is cleared to 0 by a write to **UARTECR**.

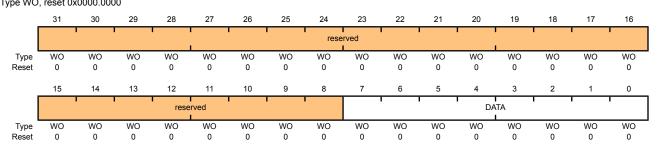
In FIFO mode, this error is associated with the character at the top of the FIFO.

JalaSHEEL4U.COIII

#### Write-Only Error Clear (UARTECR) Register

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000 Offset 0x004 Type WO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	WO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	WO	0	Error Clear

A write to this register of any data clears the framing, parity, break, and overrun flags.

#### Register 3: UART Flag (UARTFR), offset 0x018

The UARTFR register is the flag register. After reset, the TXFF, RXFF, and BUSY bits are 0, and TXFE and RXFE bits are 1.

#### UART Flag (UARTFR)

UART0 base: 0x4000.C000 Offset 0x018 Type RO, reset 0x0000.0090

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								rese	rved •						' '	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
III com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ro.com				rese	rved I				TXFE	RXFF	TXFF	RXFE	BUSY		reserved	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	TXFE	RO	1	UART Transmit FIFO Empty
				The meaning of this bit depends on the state of the ${\tt FEN}$ bit in the ${\tt UARTLCRH}$ register.
				If the FIFO is disabled (FEN is 0), this bit is set when the transmit holding register is empty.
				If the FIFO is enabled (FEN is 1), this bit is set when the transmit FIFO is empty.
6	RXFF	RO	0	UART Receive FIFO Full
				The meaning of this bit depends on the state of the ${\tt FEN}$ bit in the ${\tt UARTLCRH}$ register.
				If the FIFO is disabled, this bit is set when the receive holding register is full.
				If the FIFO is enabled, this bit is set when the receive FIFO is full.
5	TXFF	RO	0	UART Transmit FIFO Full
				The meaning of this bit depends on the state of the ${\tt FEN}$ bit in the ${\tt UARTLCRH}$ register.
				If the FIFO is disabled, this bit is set when the transmit holding register is full.
				If the FIFO is enabled, this bit is set when the transmit FIFO is full.
4	RXFE	RO	1	UART Receive FIFO Empty
				The meaning of this bit depends on the state of the ${\tt FEN}$ bit in the ${\tt UARTLCRH}$ register.
				If the FIFO is disabled, this bit is set when the receive holding register is empty.
				If the FIFO is enabled, this bit is set when the receive FIFO is empty.

Bit/Field	Name	Туре	Reset	Description
3	BUSY	RO	0	UART Busy
				When this bit is 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all stop bits, has been sent from the shift register.
				This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether UART is enabled).
2:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

#### Register 4: UART IrDA Low-Power Register (UARTILPR), offset 0x020

The **UARTILPR** register is an 8-bit read/write register that stores the low-power counter divisor value used to generate the <code>IrlPBaud16</code> signal by dividing down the system clock (SysClk). All the bits are cleared to 0 when reset.

The IrlpBaud16 internal signal is generated by dividing down the UARTCLK signal according to the low-power divisor value written to **UARTILPR**. The low-power divisor value is calculated as follows:

 $ILPDVSR = SysClk / F_{IrLPBaud16}$ 

where  $F_{\text{IrlPBaud16}}$  is nominally 1.8432 MHz.

IrlPBaud16 is an internal signal used for SIR pulse generation when low-power mode is used. You must choose the divisor so that  $1.42\,\mathrm{MHz} < \mathrm{F}_{\mathrm{IrlPBaud16}} < 2.12\,\mathrm{MHz}$ , which results in a low-power pulse duration of  $1.41-2.11\,\mu\mathrm{s}$  (three times the period of IrlPBaud16). The minimum frequency of IrlPBaud16 ensures that pulses less than one period of IrlPBaud16 are rejected, but that pulses greater than  $1.4\,\mu\mathrm{s}$  are accepted as valid pulses.

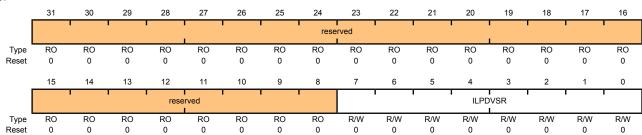
Note: Zero is an illegal value. Programming a zero value results in no IrlpBaud16 pulses being generated.

#### UART IrDA Low-Power Register (UARTILPR)

UART0 base: 0x4000.C000

Offset 0x020

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	ILPDVSR	R/W	0x00	IrDA Low-Power Divisor

This is an 8-bit low-power divisor value.

www.DataSheet4U.com

248 November 29, 2007
Preliminary

ww.Da

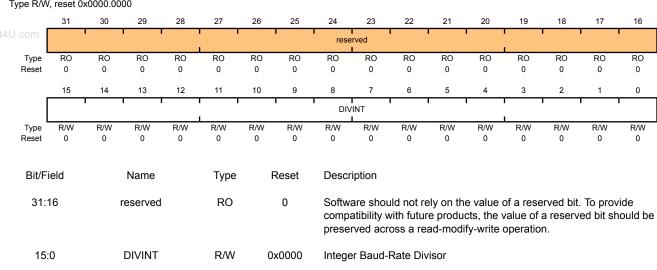
#### Register 5: UART Integer Baud-Rate Divisor (UARTIBRD), offset 0x024

The **UARTIBRD** register is the integer part of the baud-rate divisor value. All the bits are cleared on reset. The minimum possible divide ratio is 1 (when UARTIBRD=0), in which case the UARTFBRD register is ignored. When changing the **UARTIBRD** register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register. See "Baud-Rate Generation" on page 236 for configuration details.

#### UART Integer Baud-Rate Divisor (UARTIBRD)

UART0 base: 0x4000.C000 Offset 0x024

Type R/W, reset 0x0000.0000



#### Register 6: UART Fractional Baud-Rate Divisor (UARTFBRD), offset 0x028

The **UARTFBRD** register is the fractional part of the baud-rate divisor value. All the bits are cleared on reset. When changing the **UARTFBRD** register, the new value does not take effect until transmission/reception of the current character is complete. Any changes to the baud-rate divisor must be followed by a write to the **UARTLCRH** register. See "Baud-Rate Generation" on page 236 for configuration details.

#### UART Fractional Baud-Rate Divisor (UARTFBRD)

UART0 base: 0x4000.C000 Offset 0x028 Type R/W reset 0x0000.0000

Type R/W	/, reset	0x0000.00	00													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U.com		1	1	1	,		1	rese	rved I				·	1		•
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	•	reser	ved	'	'				•	I DIVF I	RAC	I	'
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/F	ield		Name		Туре	I	Reset	Descr	ription							
31:	:6		reserved	I	RO		0x00	comp	are shou atibility w rved acre	vith futur	e produ	cts, the v	alue of	a reserv		
5:0	0	Γ	DIVFRAC		R/W	(	0x000	Fracti	onal Bau	ıd-Rate	Divisor					

#### Register 7: UART Line Control (UARTLCRH), offset 0x02C

The **UARTLCRH** register is the line control register. Serial parameters such as data length, parity, and stop bit selection are implemented in this register.

When updating the baud-rate divisor (**UARTIBRD** and/or **UARTIFRD**), the **UARTLCRH** register must also be written. The write strobe for the baud-rate divisor registers is tied to the **UARTLCRH** register.

#### UART Line Control (UARTLCRH)

UART0 base: 0x4000.C000 Offset 0x02C Type R/W, reset 0x0000.0000

31 30 29 28 26 25 reserved Туре RO Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 15 14 13 12 11 10 9 2 SPS WLEN FEN EPS PEN BRK reserved R/W R/W R/W R/W R/W R/W R/W R/W Type RO RO RO RO RO RO RO RO Reset 0 0 0 n 0 0 0 0 0 0 0 0 0 0 O 0

Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	SPS	R/W	0	UART Stick Parity Select
				When bits 1, 2, and 7 of <b>UARTLCRH</b> are set, the parity bit is transmitted and checked as a 0. When bits 1 and 7 are set and 2 is cleared, the parity bit is transmitted and checked as a 1.
				When this bit is cleared, stick parity is disabled.
6:5	WLEN	R/W	0	UART Word Length
				The bits indicate the number of data bits transmitted or received in a frame as follows:
				Value Description
				0x3 8 bits
				0x2 7 bits
				0x1 6 bits
				0x0 5 bits (default)
4	FEN	R/W	0	UART Enable FIFOs
				If this bit is set to 1, transmit and receive FIFO buffers are enabled (FIFO mode).
				When cleared to 0, FIFOs are disabled (Character mode). The FIFOs become 1-byte-deep holding registers.
3	STP2	R/W	0	UART Two Stop Bits Select
				If this bit is set to 1, two stop bits are transmitted at the end of a frame.

The receive logic does not check for two stop bits being received.

Bit/	Field	Name	Туре	Reset	Description
	2	EPS	R/W	0	UART Even Parity Select
					If this bit is set to 1, even parity generation and checking is performed during transmission and reception, which checks for an even number of 1s in data and parity bits.
					When cleared to 0, then odd parity is performed, which checks for an odd number of 1s.
					This bit has no effect when parity is disabled by the $\mathtt{PEN}$ bit.
	1	PEN	R/W	0	UART Parity Enable
w.DataSheet4U.com					If this bit is set to 1, parity checking and generation is enabled; otherwise, parity is disabled and no parity bit is added to the data frame.
	0	BRK	R/W	0	UART Send Break
					If this bit is set to 1, a Low level is continually output on the ${\tt UnTX}$ output, after completing transmission of the current character. For the proper execution of the break command, the software must set this bit for at least two frames (character periods). For normal use, this bit must be cleared to 0.

# Register 8: UART Control (UARTCTL), offset 0x030

The **UARTCTL** register is the control register. All the bits are cleared on reset except for the Transmit Enable (TXE) and Receive Enable (RXE) bits, which are set to 1.

To enable the UART module, the UARTEN bit must be set to 1. If software requires a configuration change in the module, the UARTEN bit must be cleared before the configuration changes are written. If the UART is disabled during a transmit or receive operation, the current transaction is completed prior to the UART stopping.

## UART Control (UARTCTL)

UART0 base: 0x4000.C000

Offset 0x0 Type R/W		x0000.0	300													
4U.com .	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'	'	'			'	rese	rved •					'		
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	rese	erved			RXE	TXE	LBE		rese	rved		SIRLP	SIREN	UARTEN
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 1	R/W 1	R/W 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0
Bit/F	ield		Name		Туре	ı	Reset	Descr	iption							
31:	10		reserved	d	RO		0	compa		vith futur	e produ	cts, the v	alue of			vide nould be
9 RXE		R/W		1	UART	Receive	e Enable	)								
								the U		sabled i	n the mid			UART is , it comp		d. When current
								Note:	То е	enable re	eception	, the UAI	RTEN bit	must als	so be se	et.
8			TXE		R/W		1	UART Transmit Enable								
								the U		isabled	in the mi	ddle of a		UART is nission, i		
								Note:	То е	enable tr	ansmiss	ion, the	UARTEN	bit mus	t also be	e set.
7			LBE		R/W		0	UART	Loop B	ack Ena	ble					
								If this	bit is set	t to 1, th	e UnTX	oath is fe	ed throu	gh the ਹ	nRX <b>pat</b>	h.
6:	3		reserved	d	RO		0	compa		vith futur	e produ	cts, the v	alue of			vide nould be

Ві	it/Field	Name	Туре	Reset	Description
	2	SIRLP	R/W	0	UART SIR Low Power Mode
					This bit selects the IrDA encoding mode. If this bit is cleared to 0, low-level bits are transmitted as an active High pulse with a width of 3/16th of the bit period. If this bit is set to 1, low-level bits are transmitted with a pulse width which is 3 times the period of the IrLPBaud16 input signal, regardless of the selected bit rate. Setting this bit uses less power, but might reduce transmission distances. See page 248 for more information.
	1	SIREN	R/W	0	UART SIR Enable
w.DataSheet4U.cor					If this bit is set to 1, the IrDA SIR block is enabled, and the UART will transmit and receive data using SIR protocol.
	0	JARTEN	R/W	0	UART Enable
					If this bit is set to 1, the UART is enabled. When the UART is disabled in the middle of transmission or reception, it completes the current character before stopping.

# Register 9: UART Interrupt FIFO Level Select (UARTIFLS), offset 0x034

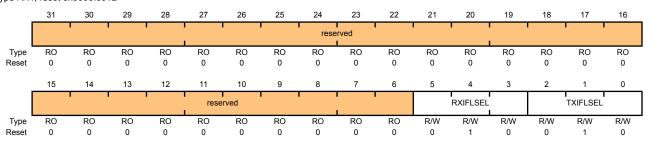
The **UARTIFLS** register is the interrupt FIFO level select register. You can use this register to define the FIFO level at which the TXRIS and RXRIS bits in the **UARTRIS** register are triggered.

The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level. For example, if the receive trigger level is set to the half-way mark, the interrupt is triggered as the module is receiving the 9th character.

Out of reset, the TXIFLSEL and RXIFLSEL bits are configured so that the FIFOs trigger an interrupt at the half-way mark.

## UART Interrupt FIFO Level Select (UARTIFLS)

UART0 base: 0x4000.C000 Offset 0x034 Type R/W, reset 0x0000.0012



Bit/Field	Name	Type	Reset	Description
31:6	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:3	RXIFLSEL	R/W	0x2	UART Receive Interrupt FIFO Level Select

The trigger points for the receive interrupt are as follows:

Value	Description
0x0	RX FIFO ≥ 1/8 full
0x1	RX FIFO ≥ ¼ full
0x2	RX FIFO ≥ ½ full (default)
0x3	RX FIFO ≥ ¾ full
0x4	RX FIFO ≥ 7/8 full
0x5-0x7	Reserved

Bit/Field	Name	Туре	Reset	Description
2:0	TXIFLSEL	R/W	0x2	UART Transmit Interrupt FIFO Level Select
				The trigger points for the transmit interrupt are as follows:
				Value Description
				0x0 TX FIFO ≤ 1/8 full
				0x1 TX FIFO ≤ ¼ full
				0x2 TX FIFO ≤ ½ full (default)
				0x3 TX FIFO ≤ ¾ full
				0x4 TX FIFO ≤ 7/8 full
				0x5-0x7 Reserved

# Register 10: UART Interrupt Mask (UARTIM), offset 0x038

25

RO

0

24

RO

0

reserved

RO

0

The **UARTIM** register is the interrupt mask set/clear register.

26

RO

0

On a read, this register gives the current value of the mask on the relevant interrupt. Writing a 1 to a bit allows the corresponding raw interrupt signal to be routed to the interrupt controller. Writing a 0 prevents the raw interrupt signal from being sent to the interrupt controller.

22

RO

0

RO

0

20

RO

0

19

RO

0

18

RO

0

17

RO

0

16

RO

0

## UART Interrupt Mask (UARTIM)

30

RO

0

29

RO

0

28

RO

0

27

RO

0

UART0 base: 0x4000.C000

RO

0

Offset 0x038

Reset

www.DataSheet4U.cType

Type R/W, reset 0x0000.0000 31

_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	'		reserved			OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM		rese	rved	•
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	RO 0	RO 0	RO 0	RO 0
Bit/Fi	eld		Name		Туре	F	Reset	Descr	iption							
31:1	11		reserved		RO	(	0x00	compa	atibility v	vith futur	e produ	cts, the v	of a resevalue of operatio	a reserv		
10	)		OEIM		R/W		0	UART	Overru	n Error li	nterrupt	Mask				
								On a	ead, the	current	mask fo	or the OE	и inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	е ОЕІМ іі	nterrupt	to the int	errupt co	ontroller.
9			BEIM		R/W		0	UART	Break E	Error Inte	errupt Ma	ask				
								On a	ead, the	current	mask fo	or the BE	и inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	е веім і	nterrupt	to the int	errupt co	ontroller.
8			PEIM		R/W		0	UART	Parity E	Error Inte	errupt Ma	ask				
								On a	ead, the	current	mask fo	or the ₽E	и inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	е РЕІМ і	nterrupt	to the int	errupt co	ontroller.
7			FEIM		R/W		0	UART	Framin	g Error I	nterrupt	Mask				
								On a	ead, the	current	mask fo	or the FE	и inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	e FEIM i	nterrupt	to the int	errupt co	ontroller.
6			RTIM		R/W		0	UART	Receive	e Time-C	Out Inter	rupt Mas	sk			
								On a	ead, the	current	mask fo	or the RI	ти inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	e RTIM i	nterrupt	to the int	errupt co	ontroller.
5			TXIM		R/W		0	UART	Transm	it Interru	ıpt Mask					
								On a	ead, the	current	mask fo	or the TX	им inter	rupt is re	eturned.	
								Settin	g this bit	to 1 pror	notes the	е тхім іі	nterrupt	to the int	errupt co	ontroller.

Bit/Field	Name	Type	Reset	Description
4	RXIM	R/W	0	UART Receive Interrupt Mask
				On a read, the current mask for the ${\tt RXIM}$ interrupt is returned.
				Setting this bit to 1 promotes the ${\tt RXIM}$ interrupt to the interrupt controller.
3:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

# Register 11: UART Raw Interrupt Status (UARTRIS), offset 0x03C

The **UARTRIS** register is the raw interrupt status register. On a read, this register gives the current raw status value of the corresponding interrupt. A write has no effect.

## UART Raw Interrupt Status (UARTRIS)

UART0 base: 0x4000.C000 Offset 0x03C Type RO, reset 0x0000.000F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
						1	)	rese	rved					1		1
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Loom	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
J.COIII			reserved			OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS		rese	rved	'
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

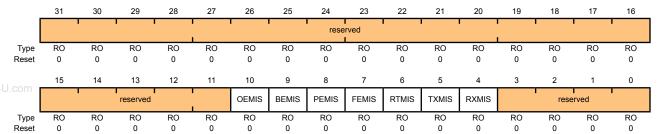
Bit/Field	Name	Туре	Reset	Description
31:11	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OERIS	RO	0	UART Overrun Error Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
9	BERIS	RO	0	UART Break Error Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
8	PERIS	RO	0	UART Parity Error Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
7	FERIS	RO	0	UART Framing Error Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
6	RTRIS	RO	0	UART Receive Time-Out Raw Interrupt Status Gives the raw interrupt state (prior to masking) of this interrupt.
5	TXRIS	RO	0	UART Transmit Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
4	RXRIS	RO	0	UART Receive Raw Interrupt Status  Gives the raw interrupt state (prior to masking) of this interrupt.
3:0	reserved	RO	0xF	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

# Register 12: UART Masked Interrupt Status (UARTMIS), offset 0x040

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

**UART Masked Interrupt Status (UARTMIS)** 

UART0 base: 0x4000.C000 Offset 0x040 Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:11	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OEMIS	RO	0	UART Overrun Error Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
9	BEMIS	RO	0	UART Break Error Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
8	PEMIS	RO	0	UART Parity Error Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
7	FEMIS	RO	0	UART Framing Error Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
6	RTMIS	RO	0	UART Receive Time-Out Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
5	TXMIS	RO	0	UART Transmit Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
4	RXMIS	RO	0	UART Receive Masked Interrupt Status
				Gives the masked interrupt state of this interrupt.
3:0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

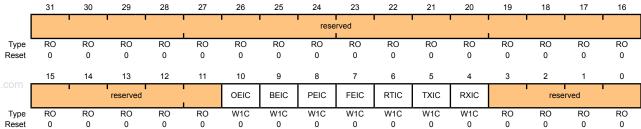
# Register 13: UART Interrupt Clear (UARTICR), offset 0x044

The **UARTICR** register is the interrupt clear register. On a write of 1, the corresponding interrupt (both raw interrupt and masked interrupt, if enabled) is cleared. A write of 0 has no effect.

UART Interrupt Clear (UARTICR)

UART0 base: 0x4000.C000

Offset 0x044 Type W1C, reset 0x0000.0000



Reset 0	0 0 0	0	0 0	
Bit/Field	Name	Туре	Reset	Description
31:11	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	OEIC	W1C	0	Overrun Error Interrupt Clear
				The OEIC values are defined as follows:
				Value Description
				0 No effect on the interrupt.
				1 Clears interrupt.
9	BEIC	W1C	0	Break Error Interrupt Clear
				The BEIC values are defined as follows:
				Value Description
				0 No effect on the interrupt.
				1 Clears interrupt.
8	PEIC	W1C	0	Parity Error Interrupt Clear
				The PEIC values are defined as follows:
				Value Description
				0 No effect on the interrupt.
				1 Clears interrupt.
7	FEIC	W1C	0	Framing Error Interrupt Clear
				The FEIC values are defined as follows:
				Value Description
				0 No effect on the interrupt.

Clears interrupt.

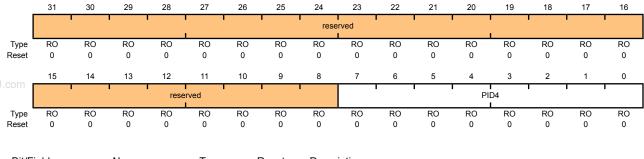
Bit/Field	Name	Туре	Reset	Description
6	RTIC	W1C	0	Receive Time-Out Interrupt Clear The RTIC values are defined as follows:  Value Description  0 No effect on the interrupt.  1 Clears interrupt.
5 et4U.com	TXIC	W1C	0	Transmit Interrupt Clear The TXIC values are defined as follows:  Value Description  0 No effect on the interrupt.  1 Clears interrupt.
4	RXIC	W1C	0	Receive Interrupt Clear The RXIC values are defined as follows:  Value Description  0 No effect on the interrupt.  1 Clears interrupt.
3:0	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

## Register 14: UART Peripheral Identification 4 (UARTPeriphID4), offset 0xFD0

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 4 (UARTPeriphID4)

UART0 base: 0x4000.C000 Offset 0xFD0 Type RO, reset 0x0000.0000



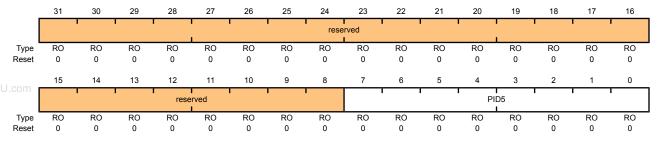
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x0000	UART Peripheral ID Register[7:0]

## Register 15: UART Peripheral Identification 5 (UARTPeriphID5), offset 0xFD4

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 5 (UARTPeriphID5)

UART0 base: 0x4000.C000 Offset 0xFD4 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x0000	UART Peripheral ID Register[15:8]

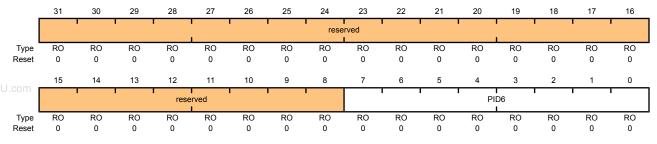
Can be used by software to identify the presence of this peripheral.

# Register 16: UART Peripheral Identification 6 (UARTPeriphID6), offset 0xFD8

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 6 (UARTPeriphID6)

UART0 base: 0x4000.C000 Offset 0xFD8 Type RO, reset 0x0000.0000



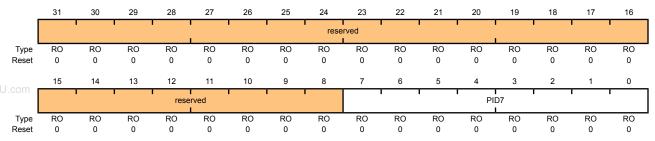
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x0000	UART Peripheral ID Register[23:16]

# Register 17: UART Peripheral Identification 7 (UARTPeriphID7), offset 0xFDC

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 7 (UARTPeriphID7)

UART0 base: 0x4000.C000 Offset 0xFDC Type RO, reset 0x0000.0000



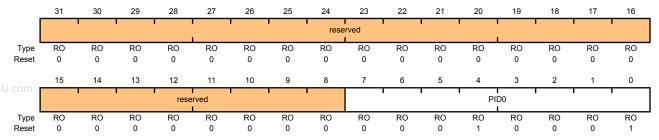
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x0000	UART Peripheral ID Register[31:24]

# Register 18: UART Peripheral Identification 0 (UARTPeriphID0), offset 0xFE0

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 0 (UARTPeriphID0)

UART0 base: 0x4000.C000 Offset 0xFE0 Type RO, reset 0x0000.0011



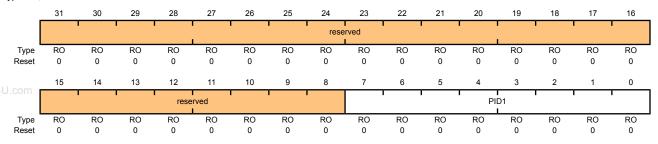
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x11	UART Peripheral ID Register[7:0]

## Register 19: UART Peripheral Identification 1 (UARTPeriphID1), offset 0xFE4

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 1 (UARTPeriphID1)

UART0 base: 0x4000.C000 Offset 0xFE4 Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	UART Peripheral ID Register[15:8]

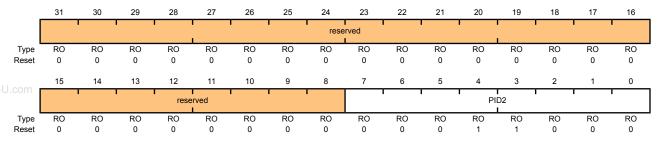
Can be used by software to identify the presence of this peripheral.

# Register 20: UART Peripheral Identification 2 (UARTPeriphID2), offset 0xFE8

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 2 (UARTPeriphID2)

UART0 base: 0x4000.C000 Offset 0xFE8 Type RO, reset 0x0000.0018



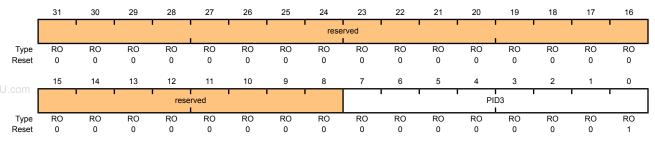
Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	UART Peripheral ID Register[23:16]

# Register 21: UART Peripheral Identification 3 (UARTPeriphID3), offset 0xFEC

The **UARTPeriphIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART Peripheral Identification 3 (UARTPeriphID3)

UART0 base: 0x4000.C000 Offset 0xFEC Type RO, reset 0x0000.0001



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	UART Peripheral ID Register[31:24]

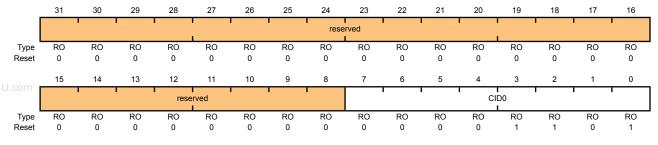
www.DataSheet4U.com

# Register 22: UART PrimeCell Identification 0 (UARTPCellID0), offset 0xFF0

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 0 (UARTPCellID0)

UART0 base: 0x4000.C000 Offset 0xFF0 Type RO, reset 0x0000.000D



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	UART PrimeCell ID Register[7:0]

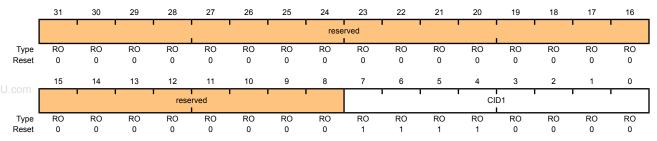
Provides software a standard cross-peripheral identification system.

# Register 23: UART PrimeCell Identification 1 (UARTPCellID1), offset 0xFF4

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 1 (UARTPCellID1)

UART0 base: 0x4000.C000 Offset 0xFF4 Type RO, reset 0x0000.00F0



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	UART PrimeCell ID Register[15:8]

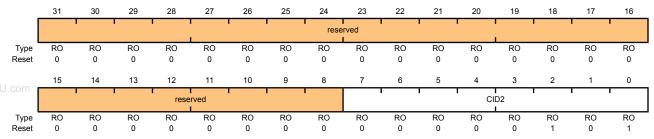
Provides software a standard cross-peripheral identification system.

# Register 24: UART PrimeCell Identification 2 (UARTPCellID2), offset 0xFF8

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

## UART PrimeCell Identification 2 (UARTPCellID2)

UART0 base: 0x4000.C000 Offset 0xFF8 Type RO, reset 0x0000.0005



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	UART PrimeCell ID Register[23:16]

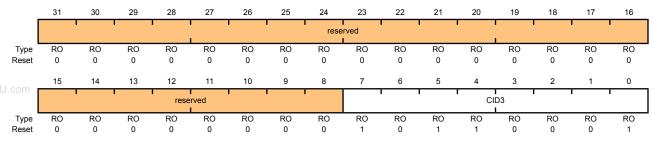
Provides software a standard cross-peripheral identification system.

# Register 25: UART PrimeCell Identification 3 (UARTPCellID3), offset 0xFFC

The **UARTPCellIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 3 (UARTPCellID3)

UART0 base: 0x4000.C000 Offset 0xFFC Type RO, reset 0x0000.00B1



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	UART PrimeCell ID Register[31:24]

Provides software a standard cross-peripheral identification system.

# 12 Synchronous Serial Interface (SSI)

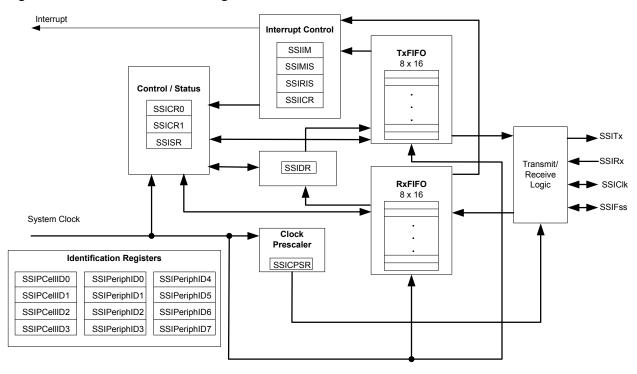
The Stellaris<sup>®</sup> Synchronous Serial Interface (SSI) is a master or slave interface for synchronous serial communication with peripheral devices that have either Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces.

The Stellaris® SSI module has the following features:

- Master or slave operation
- Programmable clock bit rate and prescale
- Separate transmit and receive FIFOs, 16 bits wide, 8 locations deep
- Programmable interface operation for Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces
- Programmable data frame size from 4 to 16 bits
- Internal loopback test mode for diagnostic/debug testing

# 12.1 Block Diagram

Figure 12-1. SSI Module Block Diagram



# 12.2 Functional Description

The SSI performs serial-to-parallel conversion on data received from a peripheral device. The CPU accesses data, control, and status information. The transmit and receive paths are buffered with

internal FIFO memories allowing up to eight 16-bit values to be stored independently in both transmit and receive modes.

## 12.2.1 Bit Rate Generation

The SSI includes a programmable bit rate clock divider and prescaler to generate the serial output clock. Bit rates are supported to 2 MHz and higher, although maximum bit rate is determined by peripheral devices.

The serial bit rate is derived by dividing down the 25-MHz input clock. The clock is first divided by an even prescale value CPSDVSR from 2 to 254, which is programmed in the **SSI Clock Prescale** (**SSICPSR**) register (see page 294). The clock is further divided by a value from 1 to 256, which is 1 + SCR, where SCR is the value programmed in the **SSI Control0** (**SSICR0**) register (see page 287).

The frequency of the output clock SSIClk is defined by:

```
FSSIClk = FSysClk / (CPSDVSR * (1 + SCR))
```

Note that although the SSIC1k transmit clock can theoretically be 12.5 MHz, the module may not be able to operate at that speed. For master mode, the system clock must be at least two times faster than the SSIC1k. For slave mode, the system clock must be at least 12 times faster than the SSIC1k.

See "Synchronous Serial Interface (SSI)" on page 455 to view SSI timing parameters.

## 12.2.2 FIFO Operation

## 12.2.2.1 Transmit FIFO

The common transmit FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. The CPU writes data to the FIFO by writing the **SSI Data (SSIDR)** register (see page 291), and data is stored in the FIFO until it is read out by the transmission logic.

When configured as a master or a slave, parallel data is written into the transmit FIFO prior to serial conversion and transmission to the attached slave or master, respectively, through the SSITX pin.

## 12.2.2.2 Receive FIFO

The common receive FIFO is a 16-bit wide, 8-locations deep, first-in, first-out memory buffer. Received data from the serial interface is stored in the buffer until read out by the CPU, which accesses the read FIFO by reading the **SSIDR** register.

When configured as a master or slave, serial data received through the SSIRx pin is registered prior to parallel loading into the attached slave or master receive FIFO, respectively.

## 12.2.3 Interrupts

The SSI can generate interrupts when the following conditions are observed:

- Transmit FIFO service
- Receive FIFO service
- Receive FIFO time-out
- Receive FIFO overrun

276 November 29, 2007
Preliminary

All of the interrupt events are ORed together before being sent to the interrupt controller, so the SSI can only generate a single interrupt request to the controller at any given time. You can mask each of the four individual maskable interrupts by setting the appropriate bits in the SSI Interrupt Mask (SSIIM) register (see page 295). Setting the appropriate mask bit to 1 enables the interrupt.

Provision of the individual outputs, as well as a combined interrupt output, allows use of either a global interrupt service routine, or modular device drivers to handle interrupts. The transmit and receive dynamic dataflow interrupts have been separated from the status interrupts so that data can be read or written in response to the FIFO trigger levels. The status of the individual interrupt sources can be read from the SSI Raw Interrupt Status (SSIRIS) and SSI Masked Interrupt Status (SSIMIS) registers (see page 297 and page 298, respectively).

#### 12.2.4 Frame Formats

Each data frame is between 4 and 16 bits long, depending on the size of data programmed, and is transmitted starting with the MSB. There are three basic frame types that can be selected:

- Texas Instruments synchronous serial
- Freescale SPI
- **MICROWIRE**

For all three formats, the serial clock (SSIC1k) is held inactive while the SSI is idle, and SSIC1k transitions at the programmed frequency only during active transmission or reception of data. The idle state of SSIC1k is utilized to provide a receive timeout indication that occurs when the receive FIFO still contains data after a timeout period.

For Freescale SPI and MICROWIRE frame formats, the serial frame (SSIFss) pin is active Low. and is asserted (pulled down) during the entire transmission of the frame.

For Texas Instruments synchronous serial frame format, the SSIFss pin is pulsed for one serial clock period starting at its rising edge, prior to the transmission of each frame. For this frame format, both the SSI and the off-chip slave device drive their output data on the rising edge of SSIC1k, and latch data from the other device on the falling edge.

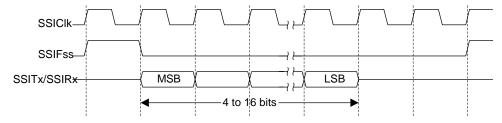
Unlike the full-duplex transmission of the other two frame formats, the MICROWIRE format uses a special master-slave messaging technique, which operates at half-duplex. In this mode, when a frame begins, an 8-bit control message is transmitted to the off-chip slave. During this transmit, no incoming data is received by the SSI. After the message has been sent, the off-chip slave decodes it and, after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

#### 12.2.4.1 **Texas Instruments Synchronous Serial Frame Format**

Figure 12-2 on page 278 shows the Texas Instruments synchronous serial frame format for a single transmitted frame.

November 29, 2007 277 **Preliminary** 

Figure 12-2. TI Synchronous Serial Frame Format (Single Transfer)

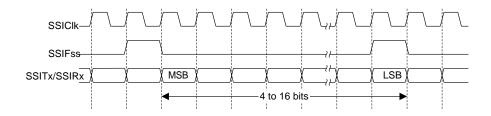


In this mode, <code>SSIClk</code> and <code>SSIFss</code> are forced Low, and the transmit data line <code>SSITx</code> is tristated whenever the SSI is idle. Once the bottom entry of the transmit FIFO contains data, <code>SSIFss</code> is pulsed High for one <code>SSIClk</code> period. The value to be transmitted is also transferred from the transmit FIFO to the serial shift register of the transmit logic. On the next rising edge of <code>SSIClk</code>, the MSB of the 4 to 16-bit data frame is shifted out on the <code>SSITx</code> pin. Likewise, the MSB of the received data is shifted onto the <code>SSIRx</code> pin by the off-chip serial slave device.

Both the SSI and the off-chip serial slave device then clock each data bit into their serial shifter on the falling edge of each SSIClk. The received data is transferred from the serial shifter to the receive FIFO on the first rising edge of SSIClk after the LSB has been latched.

Figure 12-3 on page 278 shows the Texas Instruments synchronous serial frame format when back-to-back frames are transmitted.

Figure 12-3. TI Synchronous Serial Frame Format (Continuous Transfer)



## 12.2.4.2 Freescale SPI Frame Format

The Freescale SPI interface is a four-wire interface where the SSIFss signal behaves as a slave select. The main feature of the Freescale SPI format is that the inactive state and phase of the SSIClk signal are programmable through the SPO and SPH bits within the **SSISCR0** control register.

## SPO Clock Polarity Bit

When the SPO clock polarity control bit is Low, it produces a steady state Low value on the SSIClk pin. If the SPO bit is High, a steady state High value is placed on the SSIClk pin when data is not being transferred.

#### SPH Phase Control Bit

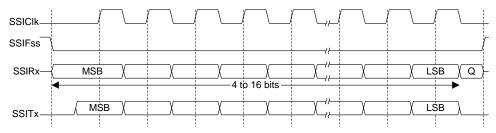
The SPH phase control bit selects the clock edge that captures data and allows it to change state. It has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. When the SPH phase control bit is Low, data is captured on the first clock edge transition. If the SPH bit is High, data is captured on the second clock edge transition.

www.DataSheet4U.com

## 12.2.4.3 Freescale SPI Frame Format with SPO=0 and SPH=0

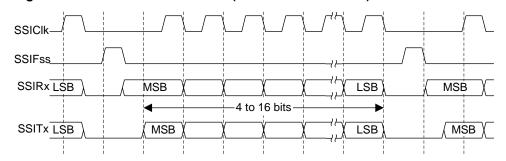
Single and continuous transmission signal sequences for Freescale SPI format with SPO=0 and SPH=0 are shown in Figure 12-4 on page 279 and Figure 12-5 on page 279.

Figure 12-4. Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0



Note: Q is undefined.

Figure 12-5. Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0



In this configuration, during idle periods:

- SSIC1k is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low. This causes slave data to be enabled onto the SSIRx input line of the master. The master SSITx output pad is enabled.

One half SSIC1k period later, valid master data is transferred to the SSITx pin. Now that both the master and slave data have been set, the SSIC1k master clock pin goes High after one further half SSIC1k period.

The data is now captured on the rising and propagated on the falling edges of the SSIClk signal.

In the case of a single word transmission, after all bits of the data word have been transferred, the SSIFss line is returned to its idle High state one SSIC1k period after the last bit has been captured.

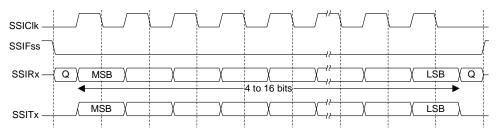
However, in the case of continuous back-to-back transmissions, the  ${\tt SSIFss}$  signal must be pulsed High between each data word transfer. This is because the slave select pin freezes the data in its

serial peripheral register and does not allow it to be altered if the SPH bit is logic zero. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

## 12.2.4.4 Freescale SPI Frame Format with SPO=0 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=0 and SPH=1 is shown in Figure 12-6 on page 280, which covers both single and continuous transfers.

Figure 12-6. Freescale SPI Frame Format with SPO=0 and SPH=1



Note: Q is undefined.

In this configuration, during idle periods:

- SSIC1k is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low. The master SSITx output is enabled. After a further one half SSIClk period, both master and slave valid data is enabled onto their respective transmission lines. At the same time, the SSIClk is enabled with a rising edge transition.

Data is then captured on the falling edges and propagated on the rising edges of the SSIC1k signal.

In the case of a single word transfer, after all bits have been transferred, the SSIFss line is returned to its idle High state one SSIClk period after the last bit has been captured.

For continuous back-to-back transfers, the SSIFss pin is held Low between successive data words and termination is the same as that of the single word transfer.

## 12.2.4.5 Freescale SPI Frame Format with SPO=1 and SPH=0

Single and continuous transmission signal sequences for Freescale SPI format with SPO=1 and SPH=0 are shown in Figure 12-7 on page 281 and Figure 12-8 on page 281.

ww.DataSheet4U.com

LSB

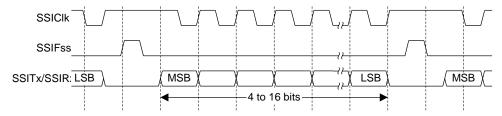
Figure 12-7. Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0

Note: Q is undefined.

SSITx

MSB





In this configuration, during idle periods:

- SSIC1k is forced High
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low, which causes slave data to be immediately transferred onto the SSIRx line of the master. The master SSITx output pad is enabled.

One half period later, valid master data is transferred to the  $\mathtt{SSITx}$  line. Now that both the master and slave data have been set, the  $\mathtt{SSIClk}$  master clock pin becomes Low after one further half  $\mathtt{SSIClk}$  period. This means that data is captured on the falling edges and propagated on the rising edges of the  $\mathtt{SSIClk}$  signal.

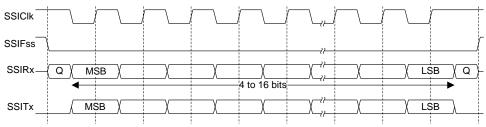
In the case of a single word transmission, after all bits of the data word are transferred, the SSIFss line is returned to its idle High state one SSIClk period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSIFss signal must be pulsed High between each data word transfer. This is because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the SPH bit is logic zero. Therefore, the master device must raise the SSIFss pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSIFss pin is returned to its idle state one SSIClk period after the last bit has been captured.

## 12.2.4.6 Freescale SPI Frame Format with SPO=1 and SPH=1

The transfer signal sequence for Freescale SPI format with SPO=1 and SPH=1 is shown in Figure 12-9 on page 282, which covers both single and continuous transfers.

Figure 12-9. Freescale SPI Frame Format with SPO=1 and SPH=1



Note: Q is undefined.

In this configuration, during idle periods:

- SSIC1k is forced High
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low
- When the SSI is configured as a master, it enables the SSIClk pad
- When the SSI is configured as a slave, it disables the SSIClk pad

If the SSI is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSIFss master signal being driven Low. The master SSITx output pad is enabled. After a further one-half SSIClk period, both master and slave data are enabled onto their respective transmission lines. At the same time, SSIClk is enabled with a falling edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SSIClk signal.

After all bits have been transferred, in the case of a single word transmission, the SSIFss line is returned to its idle high state one SSIClk period after the last bit has been captured.

For continuous back-to-back transmissions, the SSIFss pin remains in its active Low state, until the final bit of the last word has been captured, and then returns to its idle state as described above.

For continuous back-to-back transfers, the SSIFss pin is held Low between successive data words and termination is the same as that of the single word transfer.

## 12.2.4.7 MICROWIRE Frame Format

Figure 12-10 on page 283 shows the MICROWIRE frame format, again for a single frame. Figure 12-11 on page 284 shows the same format when back-to-back frames are transmitted.

SSICIK
SSIFss
SSITX
MSB

(MSB)

Figure 12-10. MICROWIRE Frame Format (Single Frame)

MICROWIRE format is very similar to SPI format, except that transmission is half-duplex instead of full-duplex, using a master-slave message passing technique. Each serial transmission begins with an 8-bit control word that is transmitted from the SSI to the off-chip slave device. During this transmission, no incoming data is received by the SSI. After the message has been sent, the off-chip slave decodes it and, after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the required data. The returned data is 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

In this configuration, during idle periods:

- SSIC1k is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low

A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of SSIFss causes the value contained in the bottom entry of the transmit FIFO to be transferred to the serial shift register of the transmit logic, and the MSB of the 8-bit control frame to be shifted out onto the SSITxpin. SSIFss remains Low for the duration of the frame transmission. The SSIRxpin pin remains tristated during this transmission.

The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SSIClk. After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSI. Each bit is driven onto the SSIRx line on the falling edge of SSIClk. The SSI in turn latches each bit on the rising edge of SSIClk. At the end of the frame, for single transfers, the SSIFss signal is pulled High one clock period after the last bit has been latched in the receive serial shifter, which causes the data to be transferred to the receive FIFO.

Note: The off-chip slave device can tristate the receive line either on the falling edge of SSIC1k after the LSB has been latched by the receive shifter, or when the SSIFss pin goes High.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the SSIFss line is continuously asserted (held Low) and transmission of data occurs back-to-back. The control byte of the next frame follows directly after the LSB of the received data from the current frame. Each of the received values is transferred from the receive shifter on the falling edge of SSIClk, after the LSB of the frame has been latched into the SSI.

SSICIK

SSIFss

SSITX

VISB

MSB

VIX VISB

SSIRX

O MSB

VIX VISB

A to 16 bits
output data
output data

Figure 12-11. MICROWIRE Frame Format (Continuous Transfer)

In the MICROWIRE mode, the SSI slave samples the first bit of receive data on the rising edge of SSIClk after SSIFss has gone Low. Masters that drive a free-running SSIClk must ensure that the SSIFss signal has sufficient setup and hold margins with respect to the rising edge of SSIClk.

Figure 12-12 on page 284 illustrates these setup and hold time requirements. With respect to the <code>SSIClk</code> rising edge on which the first bit of receive data is to be sampled by the SSI slave, <code>SSIFss</code> must have a setup of at least two times the period of <code>SSIClk</code> on which the SSI operates. With respect to the <code>SSIClk</code> rising edge previous to this edge, <code>SSIFss</code> must have a hold of at least one <code>SSIClk</code> period.

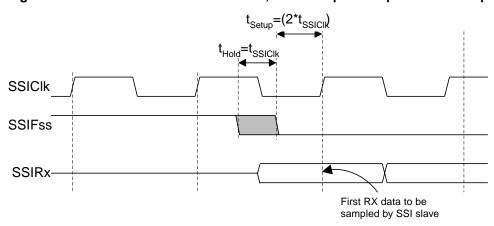


Figure 12-12. MICROWIRE Frame Format, SSIFss Input Setup and Hold Requirements

# 12.3 Initialization and Configuration

To use the SSI, its peripheral clock must be enabled by setting the SSI bit in the **RCGC1** register. For each of the frame formats, the SSI is configured using the following steps:

- 1. Ensure that the SSE bit in the **SSICR1** register is disabled before making any configuration changes.
- 2. Select whether the SSI is a master or slave:
  - **a.** For master operations, set the **SSICR1** register to 0x0000.0000.
  - **b.** For slave mode (output enabled), set the **SSICR1** register to 0x0000.0004.
  - c. For slave mode (output disabled), set the SSICR1 register to 0x0000.000C.
- 3. Configure the clock prescale divisor by writing the **SSICPSR** register.

www.DataSheet/III.com

- 4. Write the **SSICR0** register with the following configuration:
  - Serial clock rate (SCR)
  - Desired clock phase/polarity, if using Freescale SPI mode (SPH and SPO)
  - The protocol mode: Freescale SPI, TI SSF, MICROWIRE (FRF)
  - The data size (DSS)
- 5. Enable the SSI by setting the SSE bit in the SSICR1 register.

As an example, assume the SSI must be configured to operate with the following parameters:

- Master operation
- Freescale SPI mode (SPO=1, SPH=1)
- 1 Mbps bit rate
- 8 data bits

Assuming the system clock is 20 MHz, the bit rate calculation would be:

```
FSSIClk = FSysClk / (CPSDVSR * (1 + SCR))
1x106 = 20x106 / (CPSDVSR * (1 + SCR))
```

In this case, if CPSDVSR=2, SCR must be 9.

The configuration sequence would be as follows:

- 1. Ensure that the SSE bit in the SSICR1 register is disabled.
- 2. Write the **SSICR1** register with a value of 0x0000.0000.
- 3. Write the **SSICPSR** register with a value of 0x0000.0002.
- Write the SSICR0 register with a value of 0x0000.09C7.
- 5. The SSI is then enabled by setting the SSE bit in the **SSICR1** register to 1.

# 12.4 Register Map

Table 12-1 on page 285 lists the SSI registers. The offset listed is a hexadecimal increment to the register's address, relative to that SSI module's base address:

SSI0: 0x4000.8000

Note: The SSI must be disabled (see the SSE bit in the **SSICR1** register) before any of the control registers are reprogrammed.

## Table 12-1. SSI Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	SSICR0	R/W	0x0000.0000	SSI Control 0	287

November 29, 2007 285

C	Offset	Name	Туре	Reset	Description	See page
0	0x004	SSICR1	R/W	0x0000.0000	SSI Control 1	289
C	800x0	SSIDR	R/W	0x0000.0000	SSI Data	291
C	0x00C	SSISR	RO	0x0000.0003	SSI Status	292
C	0x010	SSICPSR	R/W	0x0000.0000	SSI Clock Prescale	294
C	0x014	SSIIM	R/W	0x0000.0000	SSI Interrupt Mask	295
C	0x018	SSIRIS	RO	0x0000.0008	SSI Raw Interrupt Status	297
C	)x01C	SSIMIS	RO	0x0000.0000	SSI Masked Interrupt Status	298
4U.c	0x020	SSIICR	W1C	0x0000.0000	SSI Interrupt Clear	299
0	xFD0	SSIPeriphID4	RO	0x0000.0000	SSI Peripheral Identification 4	300
0	xFD4	SSIPeriphID5	RO	0x0000.0000	SSI Peripheral Identification 5	301
0	xFD8	SSIPeriphID6	RO	0x0000.0000	SSI Peripheral Identification 6	302
0	xFDC	SSIPeriphID7	RO	0x0000.0000	SSI Peripheral Identification 7	303
C	xFE0	SSIPeriphID0	RO	0x0000.0022	SSI Peripheral Identification 0	304
C	xFE4	SSIPeriphID1	RO	0x0000.0000	SSI Peripheral Identification 1	305
C	xFE8	SSIPeriphID2	RO	0x0000.0018	SSI Peripheral Identification 2	306
0	xFEC	SSIPeriphID3	RO	0x0000.0001	SSI Peripheral Identification 3	307
C	xFF0	SSIPCellID0	RO	0x0000.000D	SSI PrimeCell Identification 0	308
C	xFF4	SSIPCellID1	RO	0x0000.00F0	SSI PrimeCell Identification 1	309
C	xFF8	SSIPCellID2	RO	0x0000.0005	SSI PrimeCell Identification 2	310
0	xFFC	SSIPCelIID3	RO	0x0000.00B1	SSI PrimeCell Identification 3	311

12.5 Register Descriptions

The remainder of this section lists and describes the SSI registers, in numerical order by address offset.

## Register 1: SSI Control 0 (SSICR0), offset 0x000

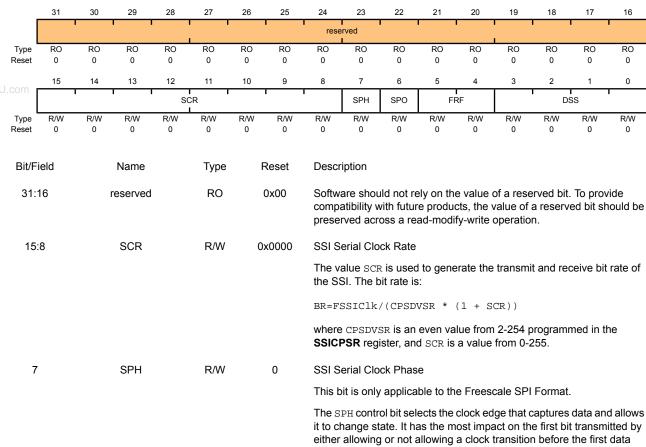
SSICR0 is control register 0 and contains bit fields that control various functions within the SSI module. Functionality such as protocol mode, clock rate, and data size are configured in this register.

## SSI Control 0 (SSICR0)

SSI0 base: 0x4000.8000

6

Offset 0x000 Type R/W, reset 0x0000.0000



capture edge.

SSI Serial Clock Polarity

R/W

0

SPO

This bit is only applicable to the Freescale SPI Format.

When the SPO bit is 0, it produces a steady state Low value on the SSIC1k pin. If SPO is 1, a steady state High value is placed on the SSIC1k pin when data is not being transferred.

When the SPH bit is 0, data is captured on the first clock edge transition. If SPH is 1, data is captured on the second clock edge transition.

Bit/Field	Name	Туре	Reset	Description
5:4	FRF	R/W	0x0	SSI Frame Format Select
				The FRF values are defined as follows:
				Value Frame Format  0x0 Freescale SPI Frame Format  0x1 Texas Intruments Synchronous Serial Frame Format  0x2 MICROWIRE Frame Format  0x3 Reserved
w.DataSheet4U.com <sup>3:0</sup>	DSS	R/W	0x00	SSI Data Size Select
				The DSS values are defined as follows:
				Value Data Size
				0x0-0x2 Reserved
				0x3 4-bit data
				0x4 5-bit data
				0x5 6-bit data
				0x6 7-bit data
				0x7 8-bit data
				0x8 9-bit data
				0x9 10-bit data
				0xA 11-bit data
				0xB 12-bit data
				0xC 13-bit data
				0xD 14-bit data
				0xE 15-bit data

0xF 16-bit data

## Register 2: SSI Control 1 (SSICR1), offset 0x004

SSICR1 is control register 1 and contains bit fields that control various functions within the SSI module. Master and slave mode functionality is controlled by this register.

SSI Control 1 (SSICR1)

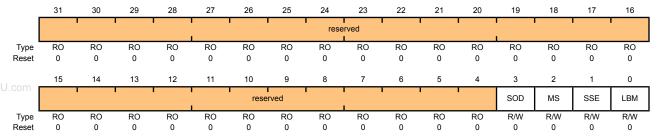
SSI0 base: 0x4000.8000

Dit/Eiold

3

SOD

Offset 0x004 Type R/W, reset 0x0000.0000



Divrieiu	Name	rype	Reset	Description
31:4	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

Docorintion

Dooot

0

This bit is relevant only in the Slave mode (MS=1). In multiple-slave systems, it is possible for the SSI master to broadcast a message to all slaves in the system while ensuring that only one slave drives data onto the serial output line. In such systems, the TXD lines from multiple slaves could be tied together. To operate in such a system, the SOD bit can be configured so that the SSI slave does not drive the SSITx pin.

The SOD values are defined as follows:

SSI Slave Mode Output Disable

#### Value Description

- SSI can drive SSITx output in Slave Output mode.
- SSI must not drive the  ${\tt SSITx}$  output in Slave mode.

2 MS R/W 0 SSI Master/Slave Select

R/W

This bit selects Master or Slave mode and can be modified only when SSI is disabled (SSE=0).

The MS values are defined as follows:

#### Value Description

- Device configured as a master.
- Device configured as a slave.

Bit/Field	Name	Type	Reset	Description
1	SSE	R/W	0	SSI Synchronous Serial Port Enable Setting this bit enables SSI operation. The SSE values are defined as follows:  Value Description 0 SSI operation disabled. 1 SSI operation enabled.
				Note: This bit must be set to 0 before any control registers are reprogrammed.
4U.com 0	LBM	R/W	0	SSI Loopback Mode Setting this bit enables Loopback Test mode.

Value Description

0 Normal serial port operation enabled.

The LBM values are defined as follows:

Output of the transmit serial shift register is connected internally to the input of the receive serial shift register.

#### Register 3: SSI Data (SSIDR), offset 0x008

**SSIDR** is the data register and is 16-bits wide. When **SSIDR** is read, the entry in the receive FIFO (pointed to by the current FIFO read pointer) is accessed. As data values are removed by the SSI receive logic from the incoming data frame, they are placed into the entry in the receive FIFO (pointed to by the current FIFO write pointer).

When **SSIDR** is written to, the entry in the transmit FIFO (pointed to by the write pointer) is written to. Data values are removed from the transmit FIFO one value at a time by the transmit logic. It is loaded into the transmit serial shifter, then serially shifted out onto the SSITX pin at the programmed bit rate.

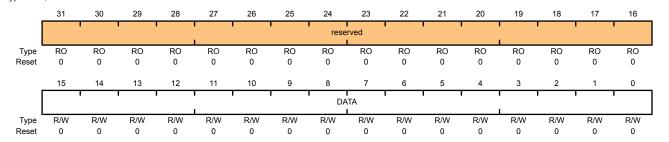
When a data size of less than 16 bits is selected, the user must right-justify data written to the transmit FIFO. The transmit logic ignores the unused bits. Received data less than 16 bits is automatically right-justified in the receive buffer.

When the SSI is programmed for MICROWIRE frame format, the default size for transmit data is eight bits (the most significant byte is ignored). The receive data size is controlled by the programmer. The transmit FIFO and the receive FIFO are not cleared even when the SSE bit in the **SSICR1** register is set to zero. This allows the software to fill the transmit FIFO before enabling the SSI.

#### SSI Data (SSIDR)

D:4/E: -1-4

SSI0 base: 0x4000.8000 Offset 0x008 Type R/W, reset 0x0000.0000



Bit/Field	Name	туре	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	DATA	R/W	0x0000	SSI Receive/Transmit Data

A read operation reads the receive FIFO. A write operation writes the transmit FIFO.

Software must right-justify data when the SSI is programmed for a data size that is less than 16 bits. Unused bits at the top are ignored by the transmit logic. The receive logic automatically right-justifies the data.

November 29, 2007 291

## Register 4: SSI Status (SSISR), offset 0x00C

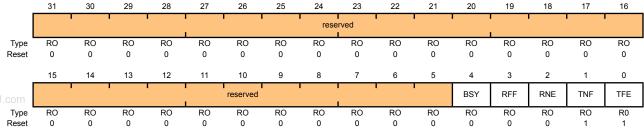
**SSISR** is a status register that contains bits that indicate the FIFO fill status and the SSI busy status.

#### SSI Status (SSISR)

SSI0 base: 0x4000.8000

Offset 0x00C

Type RO, reset 0x0000.0003



Reset	0	0 0	0	0	0	0	0	0	0	0	0	0	0	1	1
Bit/Fie	eld	Name		Туре		Reset	Descri	ption							
31:5	i	reserved		RO		0x00	compa	are shoul atibility wi ved acro	ith future	e produc	cts, the v	alue of	a reserve		
4		BSY		RO		0	SSI Bu	usy Bit							
							The B	SY value	s are de	fined as	follows:				
							Value	Descrip	otion						
							0	SSI is i	dle.						
							1		currently t FIFO is		itting and npty.	d/or rece	eiving a f	rame, or	the
3		RFF		RO		0	SSI Re	eceive Fl	IFO Full						
							The RI	FF value	s are de	fined as	follows:				
							Value	Descrip	otion						
							0	Receive	e FIFO i	s not ful	l.				
							1	Receive	e FIFO is	s full.					
2		RNE		RO		0	SSI Re	eceive Fl	IFO Not	Empty					
							The RI	NE value	s are de	fined as	follows:				
							Value	Descrip	otion						
							0	Receive	e FIFO i	s empty					
							1	Receive	e FIFO is	s not en	npty.				
1		TNF		RO		1	SSI Tr	ansmit F	IFO Not	Full					
							The Ti	NF value	s are de	fined as	follows:				
							Value	Descrip	otion						

Transmit FIFO is full.

Transmit FIFO is not full.

Bit/Field	Name	Туре	Reset	Description
0	TFE	R0	1	SSI Transmit FIFO Empty
				The TFE values are defined as follows:
				Value Description
				0 Transmit FIFO is not empty.
				1 Transmit FIFO is empty.

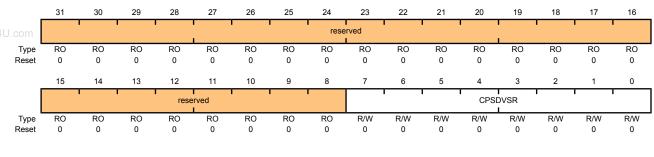
## Register 5: SSI Clock Prescale (SSICPSR), offset 0x010

SSICPSR is the clock prescale register and specifies the division factor by which the system clock must be internally divided before further use.

The value programmed into this register must be an even number between 2 and 254. The least-significant bit of the programmed number is hard-coded to zero. If an odd number is written to this register, data read back from this register has the least-significant bit as zero.

#### SSI Clock Prescale (SSICPSR)

SSI0 base: 0x4000.8000 Offset 0x010 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CPSDVSR	R/W	0x00	SSI Clock Prescale Divisor

This value must be an even number from 2 to 254, depending on the frequency of SSIC1k. The LSB always returns 0 on reads.

## Register 6: SSI Interrupt Mask (SSIIM), offset 0x014

The SSIIM register is the interrupt mask set or clear register. It is a read/write register and all bits are cleared to 0 on reset.

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

#### SSI Interrupt Mask (SSIIM)

SSI0 base: 0x4000.8000 Offset 0x014

Type R/W		0x0000.0	000													
	31	30	29	28	27	26	25	24	23	22	21	20	19 I	18	17	16
U.com								rese	rved							
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		•	•			res	erved			•	•	'	TXIM	RXIM	RTIM	RORIM
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0
Bit/Fi	ield		Name		Туре		Reset	Descri	iption							
31:	4		reserved		RO		0x00	compa	atibility v	vith futu	e produ	cts, the	of a rese value of operation	a reserv		
3			TXIM		R/W	R/W		SSI Tr	SSI Transmit FIFO Interrupt Mask							
								The T	хім <b>val</b> ı	ues are	defined	as follov	vs:			
								Value 0 1		· FO half-f			ion inter	•		ed.
2			RXIM		R/W		0	SSI R	eceive F	FIFO Inte	errupt M	ask				
								The R	хім <b>val</b> ı	ues are	defined	as follov	/s:			
									Descri							
								0					lition inte	•		
								1	RX FII	-O half-i	ull or m	ore cond	lition inte	errupt is	not mas	ked.
1			RTIM		R/W		0	SSI R	eceive 1	Γime-Ou	t Interru	pt Mask				
								The R	TIM <b>val</b>	ues are	defined	as follov	/s:			
								Value	Descri	ption						
								0	RX FI	O time-	out inte	rrupt is r	nasked.			

RX FIFO time-out interrupt is not masked.

Bit/Field	Name	Туре	Reset	Description
0	RORIM	R/W	0	SSI Receive Overrun Interrupt Mask
				The RORIM values are defined as follows:
				Value Description
				0 RX FIFO overrun interrupt is masked.
				1 RX FIFO overrun interrupt is not masked.

w.DataSheet4U.com

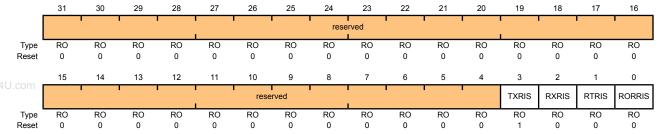
www.DataSheet4U.com

## Register 7: SSI Raw Interrupt Status (SSIRIS), offset 0x018

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

SSI Raw Interrupt Status (SSIRIS)

SSI0 base: 0x4000.8000 Offset 0x018 Type RO, reset 0x0000.0008



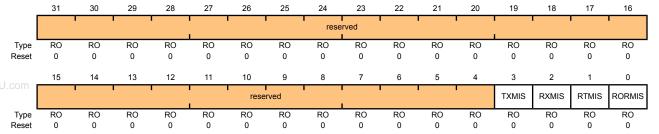
Bit/Field	Name	Туре	Reset	Description
31:4	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TXRIS	RO	1	SSI Transmit FIFO Raw Interrupt Status Indicates that the transmit FIFO is half full or less, when set.
2	RXRIS	RO	0	SSI Receive FIFO Raw Interrupt Status Indicates that the receive FIFO is half full or more, when set.
1	RTRIS	RO	0	SSI Receive Time-Out Raw Interrupt Status Indicates that the receive time-out has occurred, when set.
0	RORRIS	RO	0	SSI Receive Overrun Raw Interrupt Status Indicates that the receive FIFO has overflowed, when set.

## Register 8: SSI Masked Interrupt Status (SSIMIS), offset 0x01C

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

SSI Masked Interrupt Status (SSIMIS)

SSI0 base: 0x4000.8000 Offset 0x01C Type RO, reset 0x0000.0000



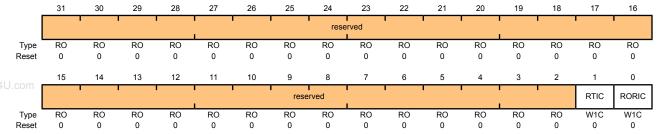
Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	TXMIS	RO	0	SSI Transmit FIFO Masked Interrupt Status Indicates that the transmit FIFO is half full or less, when set.
2	RXMIS	RO	0	SSI Receive FIFO Masked Interrupt Status Indicates that the receive FIFO is half full or more, when set.
1	RTMIS	RO	0	SSI Receive Time-Out Masked Interrupt Status Indicates that the receive time-out has occurred, when set.
0	RORMIS	RO	0	SSI Receive Overrun Masked Interrupt Status Indicates that the receive FIFO has overflowed, when set.

## Register 9: SSI Interrupt Clear (SSIICR), offset 0x020

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

SSI Interrupt Clear (SSIICR)

SSI0 base: 0x4000.8000 Offset 0x020 Type W1C, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	RTIC	W1C	0	SSI Receive Time-Out Interrupt Clear The RTIC values are defined as follows:  Value Description  0 No effect on interrupt.  1 Clears interrupt.
0	RORIC	W1C	0	SSI Receive Overrun Interrupt Clear

Value Description

No effect on interrupt.

The RORIC values are defined as follows:

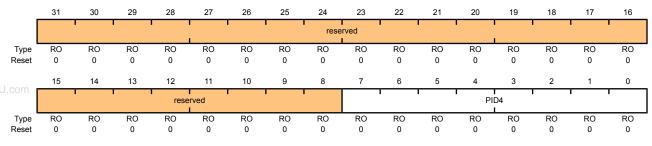
Clears interrupt.

## Register 10: SSI Peripheral Identification 4 (SSIPeriphID4), offset 0xFD0

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 4 (SSIPeriphID4)

SSI0 base: 0x4000.8000 Offset 0xFD0 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	SSI Peripheral ID Register[7:0]

Can be used by software to identify the presence of this peripheral.

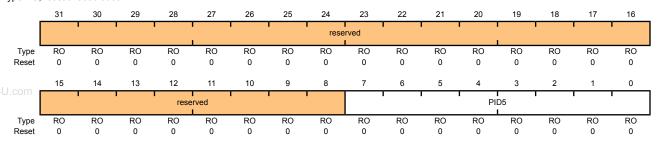
www.DataSheet4U.com

## Register 11: SSI Peripheral Identification 5 (SSIPeriphID5), offset 0xFD4

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 5 (SSIPeriphID5)

SSI0 base: 0x4000.8000 Offset 0xFD4 Type RO, reset 0x0000.0000



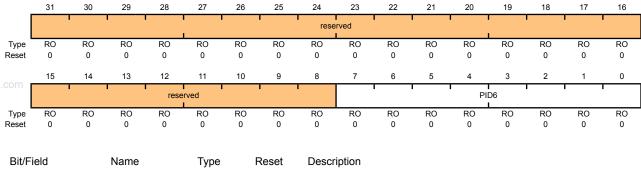
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x00	SSI Peripheral ID Register[15:8]

## Register 12: SSI Peripheral Identification 6 (SSIPeriphID6), offset 0xFD8

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

#### SSI Peripheral Identification 6 (SSIPeriphID6)

SSI0 base: 0x4000.8000 Offset 0xFD8 Type RO, reset 0x0000.0000



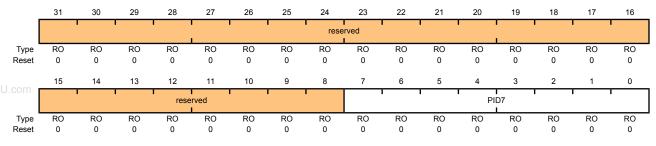
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	SSI Peripheral ID Register[23:16]

## Register 13: SSI Peripheral Identification 7 (SSIPeriphID7), offset 0xFDC

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 7 (SSIPeriphID7)

SSI0 base: 0x4000.8000 Offset 0xFDC Type RO, reset 0x0000.0000



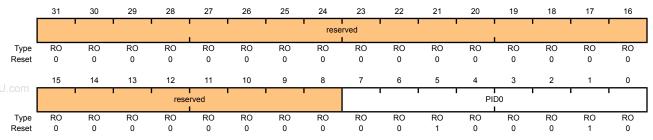
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x00	SSI Peripheral ID Register[31:24]

## Register 14: SSI Peripheral Identification 0 (SSIPeriphID0), offset 0xFE0

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 0 (SSIPeriphID0)

SSI0 base: 0x4000.8000 Offset 0xFE0 Type RO, reset 0x0000.0022



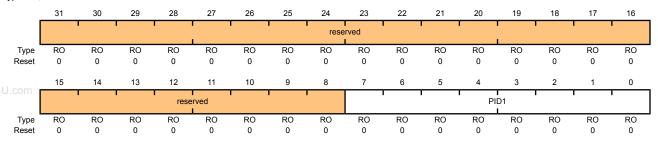
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x22	SSI Peripheral ID Register[7:0]

## Register 15: SSI Peripheral Identification 1 (SSIPeriphID1), offset 0xFE4

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 1 (SSIPeriphID1)

SSI0 base: 0x4000.8000 Offset 0xFE4 Type RO, reset 0x0000.0000



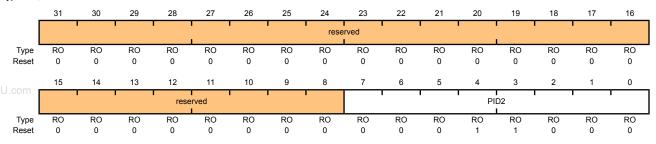
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID1	RO	0x00	SSI Peripheral ID Register [15:8]

## Register 16: SSI Peripheral Identification 2 (SSIPeriphID2), offset 0xFE8

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 2 (SSIPeriphID2)

SSI0 base: 0x4000.8000 Offset 0xFE8 Type RO, reset 0x0000.0018



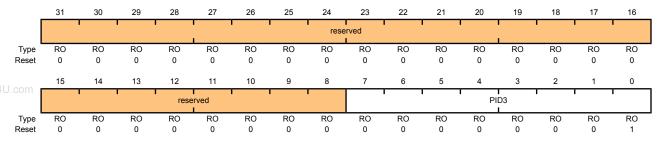
Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	SSI Peripheral ID Register [23:16]

## Register 17: SSI Peripheral Identification 3 (SSIPeriphID3), offset 0xFEC

The SSIPeriphIDn registers are hard-coded and the fields within the register determine the reset value.

SSI Peripheral Identification 3 (SSIPeriphID3)

SSI0 base: 0x4000.8000 Offset 0xFEC Type RO, reset 0x0000.0001



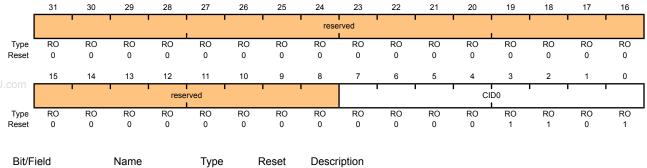
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	SSI Peripheral ID Register [31:24]

## Register 18: SSI PrimeCell Identification 0 (SSIPCellID0), offset 0xFF0

The SSIPCeIIIDn registers are hard-coded and the fields within the register determine the reset value.

#### SSI PrimeCell Identification 0 (SSIPCellID0)

SSI0 base: 0x4000.8000 Offset 0xFF0 Type RO, reset 0x0000.000D



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	SSI PrimeCell ID Register [7:0]

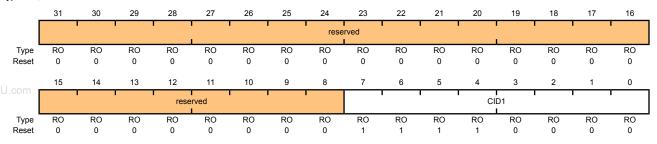
Provides software a standard cross-peripheral identification system.

## Register 19: SSI PrimeCell Identification 1 (SSIPCellID1), offset 0xFF4

The SSIPCeIIIDn registers are hard-coded and the fields within the register determine the reset value.

SSI PrimeCell Identification 1 (SSIPCellID1)

SSI0 base: 0x4000.8000 Offset 0xFF4 Type RO, reset 0x0000.00F0



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	SSI PrimeCell ID Register [15:8]

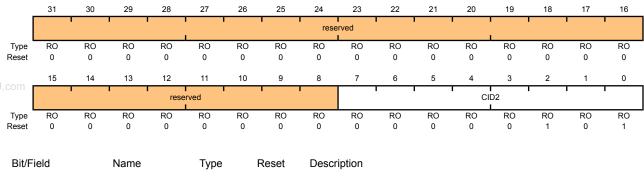
Provides software a standard cross-peripheral identification system.

## Register 20: SSI PrimeCell Identification 2 (SSIPCellID2), offset 0xFF8

The SSIPCellIDn registers are hard-coded and the fields within the register determine the reset value.

#### SSI PrimeCell Identification 2 (SSIPCelIID2)

SSI0 base: 0x4000.8000 Offset 0xFF8 Type RO, reset 0x0000.0005



31:8 RO 0x00 Software should not rely on the value of a reserved bit. To provide reserved compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. 7:0 CID2 RO 0x05 SSI PrimeCell ID Register [23:16]

Provides software a standard cross-peripheral identification system.

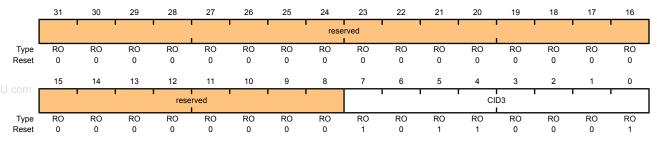
www.DataSheet4U.com

## Register 21: SSI PrimeCell Identification 3 (SSIPCelIID3), offset 0xFFC

The SSIPCeIIIDn registers are hard-coded and the fields within the register determine the reset value.

SSI PrimeCell Identification 3 (SSIPCellID3)

SSI0 base: 0x4000.8000 Offset 0xFFC Type RO, reset 0x0000.00B1



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID3	RO	0xB1	SSI PrimeCell ID Register [31:24]

Provides software a standard cross-peripheral identification system.

# 13 Inter-Integrated Circuit (I<sup>2</sup>C) Interface

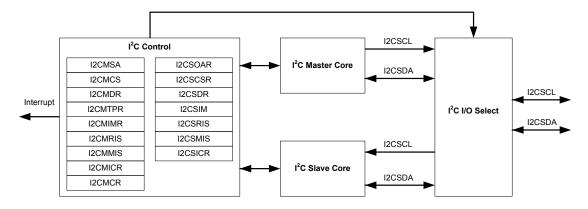
The Inter-Integrated Circuit ( $I^2C$ ) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL), and interfaces to external  $I^2C$  devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The  $I^2C$  bus may also be used for system testing and diagnostic purposes in product development and manufacture. The LM3S2110 microcontroller includes one  $I^2C$  module, providing the ability to interact (both send and receive) with other  $I^2C$  devices on the bus.

Devices on the I<sup>2</sup>C bus can be designated as either a master or a slave. The Stellaris<sup>®</sup> I<sup>2</sup>C module supports both sending and receiving data as either a master or a slave, and also supports the simultaneous operation as both a master and a slave. There are a total of four I<sup>2</sup>C modes: Master Transmit, Master Receive, Slave Transmit, and Slave Receive. The Stellaris<sup>®</sup> I<sup>2</sup>C module can operate at two speeds: Standard (100 Kbps) and Fast (400 Kbps).

Both the I<sup>2</sup>C master and slave can generate interrupts; the I<sup>2</sup>C master generates interrupts when a transmit or receive operation completes (or aborts due to an error) and the I<sup>2</sup>C slave generates interrupts when data has been sent or requested by a master.

## 13.1 Block Diagram

Figure 13-1. I<sup>2</sup>C Block Diagram



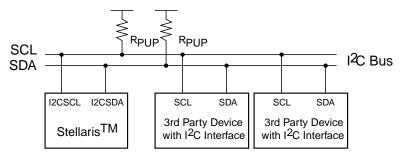
## 13.2 Functional Description

The I<sup>2</sup>C module is comprised of both master and slave functions which are implemented as separate peripherals. For proper operation, the SDA and SCL pins must be connected to bi-directional open-drain pads. A typical I<sup>2</sup>C bus configuration is shown in Figure 13-2 on page 313.

See "I<sup>2</sup>C" on page 454 for I<sup>2</sup>C timing diagrams.

ww.DataSneet4U.cor

Figure 13-2. I<sup>2</sup>C Bus Configuration



#### I<sup>2</sup>C Bus Functional Overview 13.2.1

The I<sup>2</sup>C bus uses only two signals: SDA and SCL, named I2CSDA and I2CSCL on Stellaris® microcontrollers. SDA is the bi-directional serial data line and SCL is the bi-directional serial clock line. The bus is considered idle when both lines are high.

Every transaction on the I<sup>2</sup>C bus is nine bits long, consisting of eight data bits and a single acknowledge bit. The number of bytes per transfer (defined as the time between a valid START and STOP condition, described in "START and STOP Conditions" on page 313) is unrestricted, but each byte has to be followed by an acknowledge bit, and data must be transferred MSB first. When a receiver cannot receive another complete byte, it can hold the clock line SCL Low and force the transmitter into a wait state. The data transfer continues when the receiver releases the clock SCL.

#### 13.2.1.1 START and STOP Conditions

The protocol of the I<sup>2</sup>C bus defines two states to begin and end a transaction: START and STOP. A high-to-low transition on the SDA line while the SCL is high is defined as a START condition, and a low-to-high transition on the SDA line while SCL is high is defined as a STOP condition. The bus is considered busy after a START condition and free after a STOP condition. See Figure 13-3 on page 313.

Figure 13-3. START and STOP Conditions

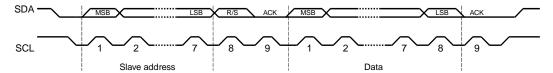


#### 13.2.1.2 Data Format with 7-Bit Address

Data transfers follow the format shown in Figure 13-4 on page 314. After the START condition, a slave address is sent. This address is 7-bits long followed by an eighth bit, which is a data direction bit (R/S bit in the I2CMSA register). A zero indicates a transmit operation (send), and a one indicates a request for data (receive). A data transfer is always terminated by a STOP condition generated by the master, however, a master can initiate communications with another device on the bus by generating a repeated START condition and addressing another slave without first generating a STOP condition. Various combinations of receive/send formats are then possible within a single transfer.

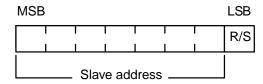
November 29, 2007 313 **Preliminary** 

Figure 13-4. Complete Data Transfer with a 7-Bit Address



The first seven bits of the first byte make up the slave address (see Figure 13-5 on page 314). The eighth bit determines the direction of the message. A zero in the R/S position of the first byte means that the master will write (send) data to the selected slave, and a one in this position means that the master will receive data from the slave.

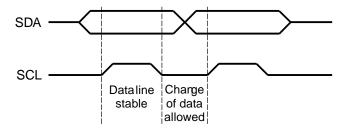
Figure 13-5. R/S Bit in First Byte



#### 13.2.1.3 Data Validity

The data on the SDA line must be stable during the high period of the clock, and the data line can only change when SCL is low (see Figure 13-6 on page 314).

Figure 13-6. Data Validity During Bit Transfer on the I<sup>2</sup>C Bus



#### 13.2.1.4 Acknowledge

All bus transactions have a required acknowledge clock cycle that is generated by the master. During the acknowledge cycle, the transmitter (which can be the master or slave) releases the SDA line. To acknowledge the transaction, the receiver must pull down SDA during the acknowledge clock cycle. The data sent out by the receiver during the acknowledge cycle must comply with the data validity requirements described in "Data Validity" on page 314.

When a slave receiver does not acknowledge the slave address, SDA must be left high by the slave so that the master can generate a STOP condition and abort the current transfer. If the master device is acting as a receiver during a transfer, it is responsible for acknowledging each transfer made by the slave. Since the master controls the number of bytes in the transfer, it signals the end of data to the slave transmitter by not generating an acknowledge on the last data byte. The slave transmitter must then release SDA to allow the master to generate the STOP or a repeated START condition.

#### 13.2.1.5 Arbitration

A master may start a transfer only if the bus is idle. It's possible for two or more masters to generate a START condition within minimum hold time of the START condition. In these situations, an arbitration scheme takes place on the SDA line, while SCL is high. During arbitration, the first of the competing master devices to place a '1' (high) on SDA while another master transmits a '0' (low) will switch off its data output stage and retire until the bus is idle again.

Arbitration can take place over several bits. Its first stage is a comparison of address bits, and if both masters are trying to address the same device, arbitration continues on to the comparison of data bits.

### 13.2.2 Available Speed Modes

The I<sup>2</sup>C clock rate is determined by the parameters: CLK\_PRD, TIMER\_PRD, SCL\_LP, and SCL\_HP.

#### where:

CLK\_PRD is the system clock period

SCL\_LP is the low phase of SCL (fixed at 6)

SCL\_HP is the high phase of SCL (fixed at 4)

TIMER\_PRD is the programmed value in the I<sup>2</sup>C Master Timer Period (I2CMTPR) register (see page 332).

The I<sup>2</sup>C clock period is calculated as follows:

```
SCL PERIOD = 2*(1 + TIMER PRD)*(SCL LP + SCL HP)*CLK PRD
```

#### For example:

```
CLK_PRD = 50 ns
TIMER_PRD = 2
SCL_LP=6
SCL_HP=4
```

yields a SCL frequency of:

```
1/T = 333 \text{ Khz}
```

Table 13-1 on page 315 gives examples of timer period, system clock, and speed mode (Standard or Fast).

Table 13-1. Examples of I<sup>2</sup>C Master Timer Period versus Speed Mode

System Clock	Timer Period	Standard Mode	Timer Period	Fast Mode
4 Mhz	0x01	100 Kbps	-	-
6 Mhz	0x02	100 Kbps	-	-
12.5 Mhz	0x06	89 Kbps	0x01	312 Kbps
16.7 Mhz	0x08	93 Kbps	0x02	278 Kbps
20 Mhz	0x09	100 Kbps	0x02	333 Kbps
25 Mhz	0x0C	96.2 Kbps	0x03	312 Kbps

#### 13.2.3 Interrupts

The I<sup>2</sup>C can generate interrupts when the following conditions are observed:

- Master transaction completed
- Master transaction error
- Slave transaction received
- Slave transaction requested

There is a separate interrupt signal for the I<sup>2</sup>C master and I<sup>2</sup>C modules. While both modules can generate interrupts for multiple conditions, only a single interrupt signal is sent to the interrupt controller.

### 13.2.3.1 I<sup>2</sup>C Master Interrupts

The I<sup>2</sup>C master module generates an interrupt when a transaction completes (either transmit or receive), or when an error occurs during a transaction. To enable the I<sup>2</sup>C master interrupt, software must write a '1' to the I<sup>2</sup>C Master Interrupt Mask (I2CMIMR) register. When an interrupt condition is met, software must check the ERROR bit in the I<sup>2</sup>C Master Control/Status (I2CMCS) register to verify that an error didn't occur during the last transaction. An error condition is asserted if the last transaction wasn't acknowledge by the slave or if the master was forced to give up ownership of the bus due to a lost arbitration round with another master. If an error is not detected, the application can proceed with the transfer. The interrupt is cleared by writing a '1' to the I<sup>2</sup>C Master Interrupt Clear (I2CMICR) register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the  $I^2C$  Master Raw Interrupt Status (I2CMRIS) register.

## 13.2.3.2 I<sup>2</sup>C Slave Interrupts

The slave module generates interrupts as it receives requests from an  $I^2C$  master. To enable the  $I^2C$  slave interrupt, write a '1' to the  $I^2C$  Slave Interrupt Mask (I2CSIMR) register. Software determines whether the module should write (transmit) or read (receive) data from the  $I^2C$  Slave Data (I2CSDR) register, by checking the RREQ and TREQ bits of the  $I^2C$  Slave Control/Status (I2CSCSR) register. If the slave module is in receive mode and the first byte of a transfer is received, the FBR bit is set along with the RREQ bit. The interrupt is cleared by writing a '1' to the  $I^2C$  Slave Interrupt Clear (I2CSICR) register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the  $I^2C$  Slave Raw Interrupt Status (I2CSRIS) register.

### 13.2.4 Loopback Operation

The  $I^2C$  modules can be placed into an internal loopback mode for diagnostic or debug work. This is accomplished by setting the LPBK bit in the  $I^2C$  Master Configuration (I2CMCR) register. In loopback mode, the SDA and SCL signals from the master and slave modules are tied together.

### 13.2.5 Command Sequence Flow Charts

This section details the steps required to perform the various I<sup>2</sup>C transfer types in both master and slave mode.

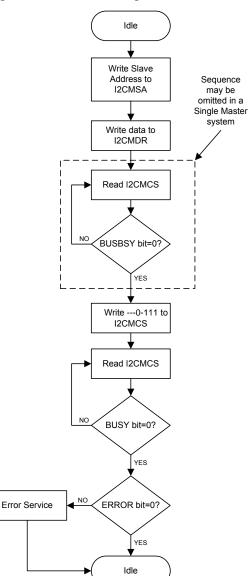
www.DataSheet4U.com

D 1 01 1411

## 13.2.5.1 I<sup>2</sup>C Master Command Sequences

The figures that follow show the command sequences available for the I<sup>2</sup>C master.

Figure 13-7. Master Single SEND



/ww.DataSheet4U.coi

Idle Sequence may be omitted in a Single Write Slave Master system Address to I2CMSA Read I2CMCS BUSBSY bit=0? YES Write ---00111 to I2CMCS Read I2CMCS BUSY bit=0? YES Error Service ERROR bit=0? YES Read data from I2CMDR Idle

Figure 13-8. Master Single RECEIVE

MANAY Data Shoot All Loo

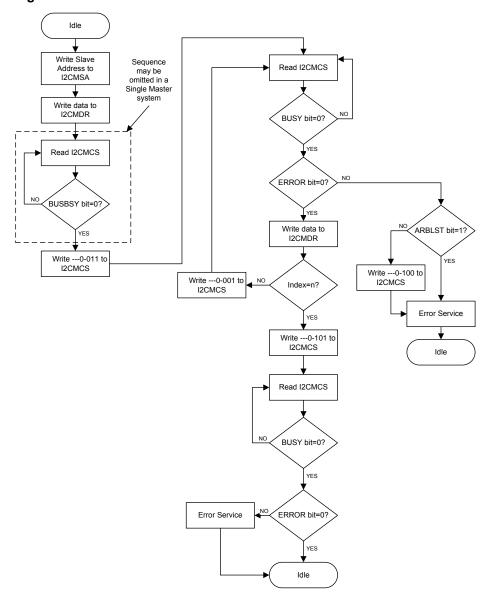


Figure 13-9. Master Burst SEND

Sequence may be omitted in a Single Master Write Slave Address to I2CMSA Read I2CMCS BUSY bit=0? Read I2CMCS YES BUSBSY bit=0 ERROR bit=0? YES ARBLST bit=1? Write ---01011 to I2CMCS Read data from I2CMDR YES Write ---0-100 to I2CMCS Write ---01001 to I2CMCS Index=m-1? Error Service YES Write ---00101 to Idle BUSY bit=0? YES ERROR bit=0° YES Read data from I2CMDR Error Service

Figure 13-10. Master Burst RECEIVE

MAN DataShootAll cor

Idle Master operates in Master Transmit mode STOP condition is not generated Write Slave Address to **I2CMSA** Write ---01011 to **I2CMCS** Repeated START condition is generated with changing data Master operates in direction Master Receive mode Idle

Figure 13-11. Master Burst RECEIVE after Burst SEND

Idle Master operates in Master Receive mode STOP condition is not generated Write Slave Address to **I2CMSA** Write ---0-011 to **I2CMCS** Repeated START condition is generated with changing data Master operates in direction Master Transmit mode Idle

Figure 13-12. Master Burst SEND after Burst RECEIVE

# 13.2.5.2 I<sup>2</sup>C Slave Command Sequences

Figure 13-13 on page 323 presents the command sequence available for the  $I^2C$  slave.

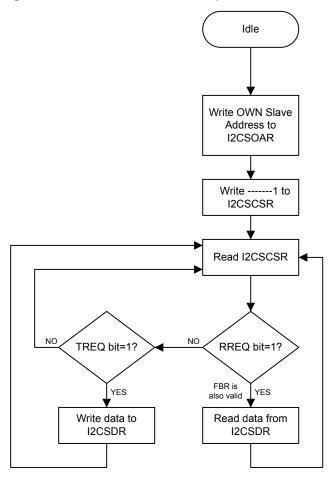


Figure 13-13. Slave Command Sequence

## 13.3 Initialization and Configuration

The following example shows how to configure the  $I^2C$  module to send a single byte as a master. This assumes the system clock is 20 MHz.

- Enable the I<sup>2</sup>C clock by writing a value of 0x0000.1000 to the RCGC1 register in the System Control module.
- Enable the clock to the appropriate GPIO module via the RCGC2 register in the System Control module.
- 3. In the GPIO module, enable the appropriate pins for their alternate function using the **GPIOAFSEL** register. Also, be sure to enable the same pins for Open Drain operation.
- 4. Initialize the I<sup>2</sup>C Master by writing the **I2CMCR** register with a value of 0x0000.0020.
- 5. Set the desired SCL clock speed of 100 Kbps by writing the **I2CMTPR** register with the correct value. The value written to the **I2CMTPR** register represents the number of system clock periods in one SCL clock period. The TPR value is determined by the following equation:

November 29, 2007 323

```
TPR = (System Clock / (2 * (SCL_LP + SCL_HP) * SCL_CLK)) - 1;

TPR = (20MHz / (2 * (6 + 4) * 100000)) - 1;

TPR = 9
```

Write the **I2CMTPR** register with the value of 0x0000.0009.

- 6. Specify the slave address of the master and that the next operation will be a Send by writing the **I2CMSA** register with a value of 0x0000.0076. This sets the slave address to 0x3B.
- Place data (byte) to be sent in the data register by writing the I2CMDR register with the desired data.
- 8. Initiate a single byte send of the data from Master to Slave by writing the **I2CMCS** register with a value of 0x0000.0007 (STOP, START, RUN).
- 9. Wait until the transmission completes by polling the I2CMCS register's BUSBSY bit until it has been cleared.

## 13.4 I<sup>2</sup>C Register Map

Table 13-2 on page 324 lists the  $I^2C$  registers. All addresses given are relative to the  $I^2C$  base addresses for the master and slave:

- I<sup>2</sup>C Master 0: 0x4002.0000
- I<sup>2</sup>C Slave 0: 0x4002.0800

Table 13-2. Inter-Integrated Circuit (I<sup>2</sup>C) Interface Register Map

Offset	Name	Type	Reset	Description	See page
I <sup>2</sup> C Maste	r				,
0x000	I2CMSA	R/W	0x0000.0000	I2C Master Slave Address	326
0x004	I2CMCS	R/W	0x0000.0000	I2C Master Control/Status	327
0x008	I2CMDR	R/W	0x0000.0000	I2C Master Data	331
0x00C	I2CMTPR	R/W	0x0000.0001	I2C Master Timer Period	332
0x010	I2CMIMR	R/W	0x0000.0000	I2C Master Interrupt Mask	333
0x014	I2CMRIS	RO	0x0000.0000	I2C Master Raw Interrupt Status	334
0x018	I2CMMIS	RO	0x0000.0000	I2C Master Masked Interrupt Status	335
0x01C	I2CMICR	WO	0x0000.0000	I2C Master Interrupt Clear	336
0x020	I2CMCR	R/W	0x0000.0000	I2C Master Configuration	337
I <sup>2</sup> C Slave					1
0x000	I2CSOAR	R/W	0x0000.0000	I2C Slave Own Address	339
0x004	I2CSCSR	RO	0x0000.0000	I2C Slave Control/Status	340
0x008	I2CSDR	R/W	0x0000.0000	I2C Slave Data	342
0x00C	I2CSIMR	R/W	0x0000.0000	I2C Slave Interrupt Mask	343

324 November 29, 2007

Offset	Name	Туре	Reset	Description	See page
0x010	I2CSRIS	RO	0x0000.0000	I2C Slave Raw Interrupt Status	344
0x014	I2CSMIS	RO	0x0000.0000	I2C Slave Masked Interrupt Status	345
0x018	I2CSICR	WO	0x0000.0000	I2C Slave Interrupt Clear	346

# 13.5 Register Descriptions (I<sup>2</sup>C Master)

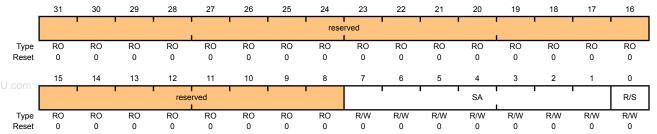
The remainder of this section lists and describes the  $I^2C$  master registers, in numerical order by address offset. See also "Register Descriptions (I2C Slave)" on page 338.

# Register 1: I<sup>2</sup>C Master Slave Address (I2CMSA), offset 0x000

This register consists of eight bits: seven address bits (A6-A0), and a Receive/Send bit, which determines if the next operation is a Receive (High), or Send (Low).

#### I2C Master Slave Address (I2CMSA)

I2C Master 0 base: 0x4002.0000 Offset 0x000 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:1	SA	R/W	0	I <sup>2</sup> C Slave Address
				This field specifies bits A6 through A0 of the slave address.
0	R/S	R/W	0	Receive/Send

The R/S bit specifies if the next operation is a Receive (High) or Send (Low).

www.DataSheet4U.com

Value Description

Send.

Receive.

## Register 2: I<sup>2</sup>C Master Control/Status (I2CMCS), offset 0x004

This register accesses four control bits when written, and accesses seven status bits when read.

The status register consists of seven bits, which when read determine the state of the I<sup>2</sup>C bus controller.

The control register consists of four bits: the RUN, START, STOP, and ACK bits. The START bit causes the generation of the START, or REPEATED START condition.

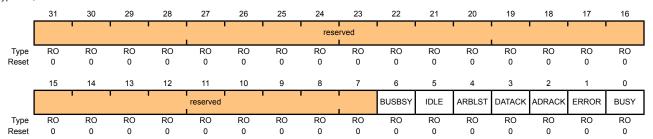
The STOP bit determines if the cycle stops at the end of the data cycle, or continues on to a burst. To generate a single send cycle, the  $I^2C$  Master Slave Address (I2CMSA) register is written with the desired address, the R/S bit is set to 0, and the Control register is written with ACK=X (0 or 1), STOP=1, START=1, and RUN=1 to perform the operation and stop. When the operation is completed (or aborted due an error), the interrupt pin becomes active and the data may be read from the I2CMDR register. When the  $I^2C$  module operates in Master receiver mode, the ACK bit must be set normally to logic 1. This causes the  $I^2C$  bus controller to send an acknowledge automatically after each byte. This bit must be reset when the  $I^2C$  bus controller requires no further data to be sent from the slave transmitter.

#### **Read-Only Status Register**

I2C Master Control/Status (I2CMCS)

I2C Master 0 base: 0x4002.0000 Offset 0x004

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:7	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	BUSBSY	RO	0	Bus Busy
				This bit specifies the state of the $I^2C$ bus. If set, the bus is busy; otherwise, the bus is idle. The bit changes based on the START and STOP conditions.
5	IDLE	RO	0	I <sup>2</sup> C Idle
				This bit specifies the $I^2C$ controller state. If set, the controller is idle; otherwise the controller is not idle.
4	ARBLST	RO	0	Arbitration Lost
				This bit specifies the result of bus arbitration. If set, the controller lost

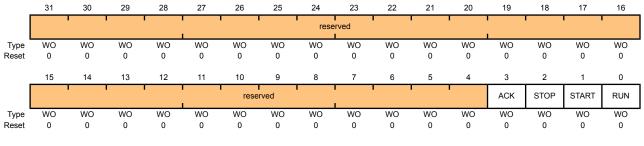
arbitration; otherwise, the controller won arbitration.

Bit/Field	Name	Туре	Reset	Description
3	DATACK	RO	0	Acknowledge Data
				This bit specifies the result of the last data operation. If set, the transmitted data was not acknowledged; otherwise, the data was acknowledged.
2	ADRACK	RO	0	Acknowledge Address
				This bit specifies the result of the last address operation. If set, the transmitted address was not acknowledged; otherwise, the address was acknowledged.
1	ERROR	RO	0	Error
				This bit specifies the result of the last bus operation. If set, an error occurred on the last operation; otherwise, no error was detected. The error can be from the slave address not being acknowledged, the transmit data not being acknowledged, or because the controller lost arbitration.
0	BUSY	RO	0	I <sup>2</sup> C Busy
				This bit specifies the state of the controller. If set, the controller is busy;

### **Write-Only Control Register**

I2C Master Control/Status (I2CMCS)

I2C Master 0 base: 0x4002.0000 Offset 0x004 Type WO, reset 0x0000.0000



bits are not valid.

otherwise, the controller is idle. When the BUSY bit is set, the other status

Bit/Field	Name	Туре	Reset	Description
31:4	reserved	WO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	ACK	WO	0	Data Acknowledge Enable  When set, causes received data byte to be acknowledged automatically by the master. See field decoding in Table 13-3 on page 329.
2	STOP	WO	0	Generate STOP When set, causes the generation of the STOP condition. See field

decoding in Table 13-3 on page 329.

www.DataSheet4U.com

Bit/Field	Name	Type	Reset	Description
1	START	WO	0	Generate START
				When set, causes the generation of a START or repeated START condition. See field decoding in Table 13-3 on page 329.
0	RUN	WO	0	I <sup>2</sup> C Master Enable

When set, allows the master to send or receive data. See field decoding in Table 13-3 on page 329.

Table 13-3. Write Field Decoding for I2CMCS[3:0] Field (Sheet 1 of 3)

Current	I2CMSA[0]		I2CMCS[3:0]			Description
State	R/S	ACK	STOP	START	RUN	
Idle	0	X <sup>a</sup>	0	1	1	START condition followed by SEND (master goes to the Master Transmit state).
	0	Х	1	1	1	START condition followed by a SEND and STOP condition (master remains in Idle state).
	1	0	0	1	1	START condition followed by RECEIVE operation with negative ACK (master goes to the Master Receive state)
	1	0	1	1	1	START condition followed by RECEIVE and STOP condition (master remains in Idle state).
	1	1	0	1	1	START condition followed by RECEIVE (master goes to the Master Receive state).
	1	1	1	1	1	Illegal.
	All other co	mbinations	s not listed	are non-o	perations.	NOP.
Master Transmit	Х	Х	0	0	1	SEND operation (master remains in Master Transmit state).
	Х	Х	1	0	0	STOP condition (master goes to Idle state).
	Х	Х	1	0	1	SEND followed by STOP condition (master goes to Idle state).
	0	Х	0	1	1	Repeated START condition followed by a SEND (master remains in Master Transmit state).
	0	Х	1	1	1	Repeated START condition followed by SEND and STOP condition (master goes to Idle state).
	1	0	0	1	1	Repeated START condition followed by a RECEIVE operation with a negative ACK (master goes to Master Receive state).
	1	0	1	1	1	Repeated START condition followed by a SEND and STOP condition (master goes to Idle state).
	1	1	0	1	1	Repeated START condition followed by RECEIVE (master goes to Master Receive state).
	1	1	1	1	1	Illegal.
	All other co	mbinations	s not listed	are non-o	perations.	NOP.

www DataSheet4LL com

Current	I2CMSA[0]		I2CMC	CS[3:0]		Description
State	R/S	ACK	STOP	START	RUN	
Master Receive	Х	0	0	0	1	RECEIVE operation with negative ACK (master remains in Master Receive state).
	Х	Х	1	0	0	STOP condition (master goes to Idle state). <sup>b</sup>
	Х	0	1	0	1	RECEIVE followed by STOP condition (master goes to Idle state).
	Х	1	0	0	1	RECEIVE operation (master remains in Master Receive state).
	Х	1	1	0	1	Illegal.
	1	0	0	1	1	Repeated START condition followed by RECEIVE operation with a negative ACK (master remains in Master Receive state).
	1	0	1	1	1	Repeated START condition followed by RECEIVE and STOP condition (master goes to Idle state).
	1	1	0	1	1	Repeated START condition followed by RECEIVE (master remains in Master Receive state).
	0	Х	0	1	1	Repeated START condition followed by SEND (master goes to Master Transmit state).
	0	Х	1	1	1	Repeated START condition followed by SEND and STOP condition (master goes to Idle state).
	All other co	mbinations	s not listed	are non-op	erations.	NOP.

a. An X in a table cell indicates the bit can be 0 or 1.

b. In Master Receive mode, a STOP condition should be generated only after a Data Negative Acknowledge executed by the master or an Address Negative Acknowledge executed by the slave.

# Register 3: I<sup>2</sup>C Master Data (I2CMDR), offset 0x008

This register contains the data to be transmitted when in the Master Transmit state, and the data received when in the Master Receive state.

#### I2C Master Data (I2CMDR)

I2C Master 0 base: 0x4002.0000

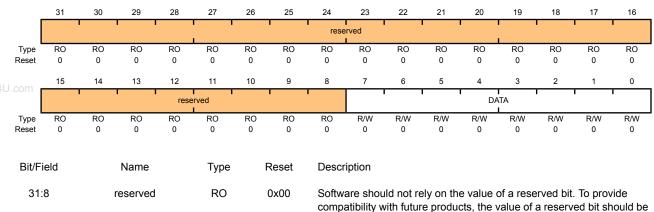
7:0

DATA

R/W

0x00

Offset 0x008
Type R/W, reset 0x0000.0000



Data transferred during transaction.

Data Transferred

preserved across a read-modify-write operation.

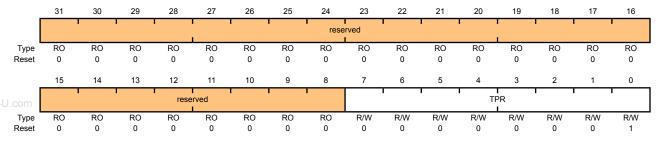
# Register 4: I<sup>2</sup>C Master Timer Period (I2CMTPR), offset 0x00C

This register specifies the period of the SCL clock.

#### I2C Master Timer Period (I2CMTPR)

I2C Master 0 base: 0x4002.0000 Offset 0x00C

Type R/W, reset 0x0000.0001



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TPR	R/W	0x1	SCL Clock Period

This field specifies the period of the SCL clock.

 $SCL\_PRD = 2*(1 + TPR)*(SCL\_LP + SCL\_HP)*CLK\_PRD$ 

www.DataSheet4U.com

where:

SCL\_PRD is the SCL line period (I<sup>2</sup>C clock).

 $\ensuremath{\mathtt{TPR}}$  is the Timer Period register value (range of 1 to 255).

 ${\tt SCL\_LP}$  is the SCL Low period (fixed at 6).

SCL\_HP is the SCL High period (fixed at 4).

# Register 5: I<sup>2</sup>C Master Interrupt Mask (I2CMIMR), offset 0x010

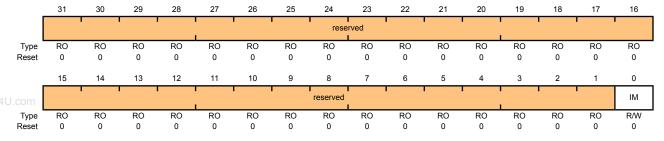
This register controls whether a raw interrupt is promoted to a controller interrupt.

#### I2C Master Interrupt Mask (I2CMIMR)

I2C Master 0 base: 0x4002.0000

Offset 0x010

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IM	R/W	0	Interrunt Mask

This bit controls whether a raw interrupt is promoted to a controller interrupt. If set, the interrupt is not masked and the interrupt is promoted; otherwise, the interrupt is masked.

# Register 6: I<sup>2</sup>C Master Raw Interrupt Status (I2CMRIS), offset 0x014

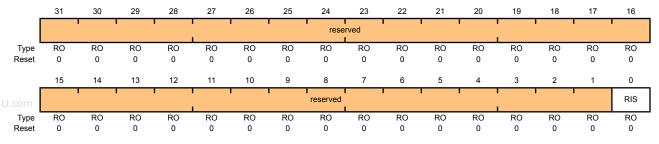
This register specifies whether an interrupt is pending.

#### I2C Master Raw Interrupt Status (I2CMRIS)

I2C Master 0 base: 0x4002.0000

Offset 0x014

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	RIS	RO	0	Raw Interrupt Status

This bit specifies the raw interrupt state (prior to masking) of the  $I^2C$  master block. If set, an interrupt is pending; otherwise, an interrupt is not pending.

# Register 7: I<sup>2</sup>C Master Masked Interrupt Status (I2CMMIS), offset 0x018

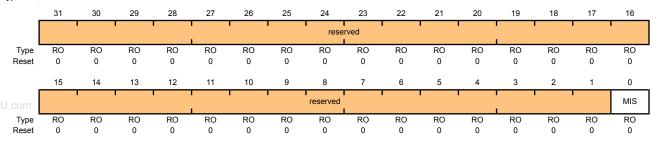
This register specifies whether an interrupt was signaled.

### I2C Master Masked Interrupt Status (I2CMMIS)

I2C Master 0 base: 0x4002.0000

Offset 0x018

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	MIS	RO	0	Masked Interrupt Status

This bit specifies the raw interrupt state (after masking) of the  $l^2C$  master block. If set, an interrupt was signaled; otherwise, an interrupt has not been generated since the bit was last cleared.

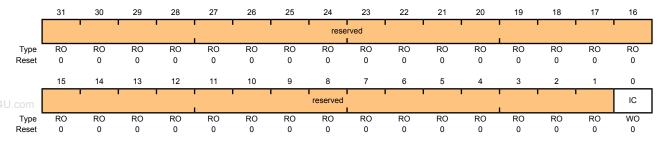
# Register 8: I<sup>2</sup>C Master Interrupt Clear (I2CMICR), offset 0x01C

This register clears the raw interrupt.

I2C Master Interrupt Clear (I2CMICR)

I2C Master 0 base: 0x4002.0000 Offset 0x01C

Type WO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IC	WO	0	Interrupt Clear

This bit controls the clearing of the raw interrupt. A write of 1 clears the interrupt; otherwise, a write of 0 has no affect on the interrupt state. A read of this register returns no meaningful data.

# Register 9: I<sup>2</sup>C Master Configuration (I2CMCR), offset 0x020

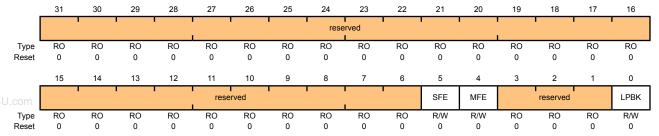
This register configures the mode (Master or Slave) and sets the interface for test mode loopback.

#### I2C Master Configuration (I2CMCR)

I2C Master 0 base: 0x4002.0000

Offset 0x020

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:6	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5	SFE	R/W	0	I <sup>2</sup> C Slave Function Enable
				This bit specifies whether the interface may operate in Slave mode. If set, Slave mode is enabled; otherwise, Slave mode is disabled.
4	MFE	R/W	0	I <sup>2</sup> C Master Function Enable
				This bit specifies whether the interface may operate in Master mode. If set, Master mode is enabled; otherwise, Master mode is disabled and the interface clock is disabled.
3:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	LPBK	R/W	0	I <sup>2</sup> C Loopback

This bit specifies whether the interface is operating normally or in Loopback mode. If set, the device is put in a test mode loopback configuration; otherwise, the device operates normally.

# 13.6 Register Descriptions (I2C Slave)

The remainder of this section lists and describes the  $I^2C$  slave registers, in numerical order by address offset. See also "Register Descriptions ( $I^2C$  Master)" on page 325.

www DataSheet411 com

# Register 10: I<sup>2</sup>C Slave Own Address (I2CSOAR), offset 0x000

This register consists of seven address bits that identify the Stellaris<sup>®</sup> I<sup>2</sup>C device on the I<sup>2</sup>C bus.

I2C Slave Own Address (I2CSOAR)

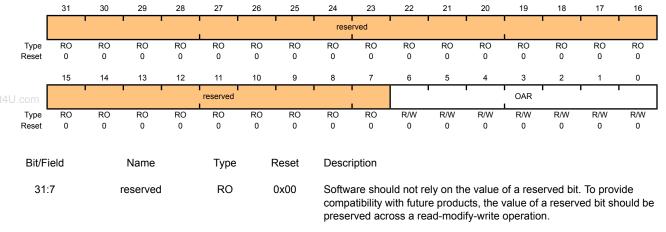
OAR

R/W

0x00

I2C Slave 0 base: 0x4002.0800 Offset 0x000 Type R/W, reset 0x0000.0000

6:0



I<sup>2</sup>C Slave Own Address

This field specifies bits A6 through A0 of the slave address.

## Register 11: I<sup>2</sup>C Slave Control/Status (I2CSCSR), offset 0x004

This register accesses one control bit when written, and three status bits when read.

The read-only Status register consists of three bits: the FBR, RREQ, and TREQ bits. The First Byte Received (FBR) bit is set only after the Stellaris® device detects its own slave address and receives the first data byte from the  $I^2C$  master. The Receive Request (RREQ) bit indicates that the Stellaris® I2C device has received a data byte from an I2C master. Read one data byte from the I2C Slave Data (I2CSDR) register to clear the RREO bit. The Transmit Request (TREO) bit indicates that the Stellaris<sup>®</sup> I<sup>2</sup>C device is addressed as a Slave Transmitter. Write one data byte into the I<sup>2</sup>C Slave Data (I2CSDR) register to clear the TREQ bit.

The write-only Control register consists of one bit: the DA bit. The DA bit enables and disables the Stellaris<sup>®</sup> I<sup>2</sup>C slave operation.

#### Read-Only Status Register

I2C Slave Control/Status (I2CSCSR)

Name

**RREQ** 

Type

RO

Reset

0

I2C Slave 0 base: 0x4002.0800 Offset 0x004 Type RO, reset 0x0000.0000

Bit/Field

0

26 RO RO RO RO RO RO RO RO Type Reset

	•	•	•	·	•	•	•	•	•	•	•	•	•	•	•	·	
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
		•	'	'		'	reserved	'			1		_	FBR	TREQ	RREQ	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Description

OAA	
31:3 reserved RO 0x00 Software should not rely on the value of a reserved bit. To preserved across a read-modify-write operation.	should be
2 FBR RO 0 First Byte Received	
Indicates that the first byte following the slave's own address. This bit is only valid when the RREQ bit is set, and is automatic when data has been read from the I2CSDR register.	
Note: This bit is not used for slave transmit operations.	

TRFO RO 0 Transmit Request

> This bit specifies the state of the I<sup>2</sup>C slave with regards to outstanding transmit requests. If set, the I<sup>2</sup>C unit has been addressed as a slave transmitter and uses clock stretching to delay the master until data has been written to the I2CSDR register. Otherwise, there is no outstanding transmit request.

This bit specifies the status of the I<sup>2</sup>C slave with regards to outstanding receive requests. If set, the I<sup>2</sup>C unit has outstanding receive data from the I<sup>2</sup>C master and uses clock stretching to delay the master until the data has been read from the I2CSDR register. Otherwise, no receive

www.DataSheet4U.com

data is outstanding.

340 November 29, 2007 **Preliminary** 

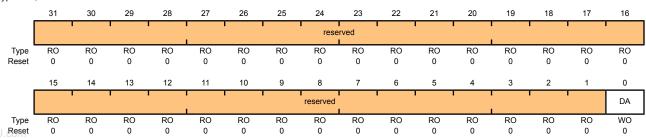
Receive Request

### **Write-Only Control Register**

#### I2C Slave Control/Status (I2CSCSR)

I2C Slave 0 base: 0x4002.0800 Offset 0x004

Type WO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	DA	WO	0	Device Active

#### Value Description

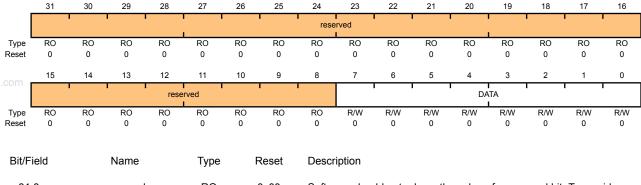
- Disables the I<sup>2</sup>C slave operation.
- Enables the I<sup>2</sup>C slave operation.

# Register 12: I<sup>2</sup>C Slave Data (I2CSDR), offset 0x008

This register contains the data to be transmitted when in the Slave Transmit state, and the data received when in the Slave Receive state.

#### I2C Slave Data (I2CSDR)

I2C Slave 0 base: 0x4002.0800 Offset 0x008
Type R/W, reset 0x0000.0000



31:8 reserved RO 0x00 Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. DATA R/W **Data for Transfer** 

7:0 0x0

> This field contains the data for transfer during a slave receive or transmit operation.

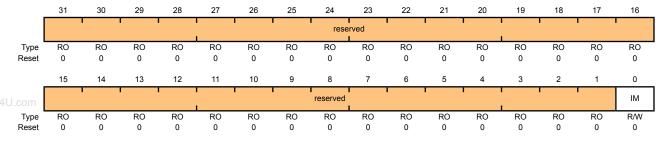
# Register 13: I<sup>2</sup>C Slave Interrupt Mask (I2CSIMR), offset 0x00C

This register controls whether a raw interrupt is promoted to a controller interrupt.

#### I2C Slave Interrupt Mask (I2CSIMR)

I2C Slave 0 base: 0x4002.0800 Offset 0x00C

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IM	R/W	0	Interrunt Mask

This bit controls whether a raw interrupt is promoted to a controller interrupt. If set, the interrupt is not masked and the interrupt is promoted; otherwise, the interrupt is masked.

# Register 14: I<sup>2</sup>C Slave Raw Interrupt Status (I2CSRIS), offset 0x010

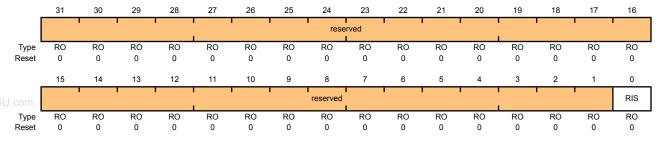
This register specifies whether an interrupt is pending.

#### I2C Slave Raw Interrupt Status (I2CSRIS)

I2C Slave 0 base: 0x4002.0800

Offset 0x010

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	RIS	RO	0	Raw Interrupt Status

This bit specifies the raw interrupt state (prior to masking) of the  $I^2C$  slave block. If set, an interrupt is pending; otherwise, an interrupt is not pending.

www.DataSheet4U.com

# Register 15: I<sup>2</sup>C Slave Masked Interrupt Status (I2CSMIS), offset 0x014

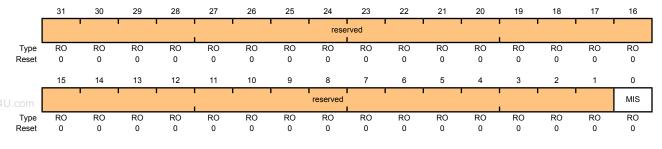
This register specifies whether an interrupt was signaled.

#### I2C Slave Masked Interrupt Status (I2CSMIS)

I2C Slave 0 base: 0x4002.0800

Offset 0x014

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	MIS	RO	0	Masked Interrupt Status

This bit specifies the raw interrupt state (after masking) of the  $I^2C$  slave block. If set, an interrupt was signaled; otherwise, an interrupt has not been generated since the bit was last cleared.

# Register 16: I<sup>2</sup>C Slave Interrupt Clear (I2CSICR), offset 0x018

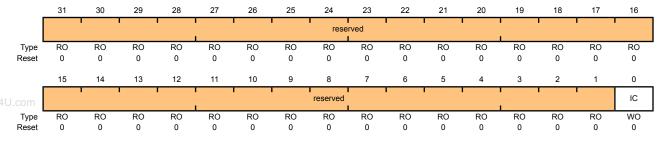
This register clears the raw interrupt.

I2C Slave Interrupt Clear (I2CSICR)

I2C Slave 0 base: 0x4002.0800

Offset 0x018

Type WO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IC	WO	0	Clear Interrupt

This bit controls the clearing of the raw interrupt. A write of 1 clears the interrupt; otherwise a write of 0 has no affect on the interrupt state. A read of this register returns no meaningful data.

# 14 Controller Area Network (CAN) Module

### 14.1 Controller Area Network Overview

Controller Area Network (CAN) is a multicast shared serial bus standard for connecting electronic control units (ECUs). CAN was specifically designed to be robust in electromagnetically noisy environments and can utilize a differential balanced line like RS-485 or a more robust twisted-pair wire. Originally created for automotive purposes, it is also used in many embedded control applications (such as industrial and medical). Bit rates up to 1 Mbps are possible at network lengths below 40 meters. Decreased bit rates allow longer network distances (for example, 125 Kbps at 500 m).

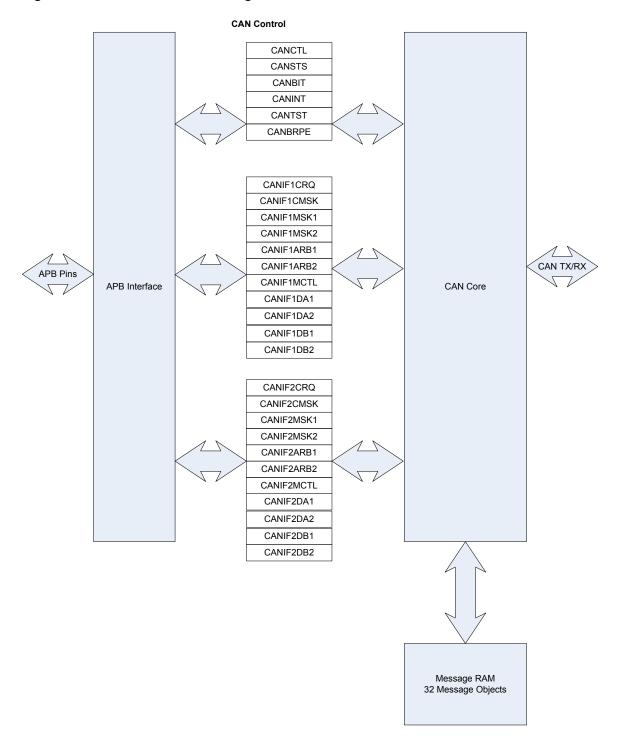
#### www.DataSheet414.2 Controller Area Network Features

The Stellaris<sup>®</sup> CAN module supports the following features:

- CAN protocol version 2.0 part A/B
- Bit rates up to 1 Mbps
- 32 message objects
- Each message object has its own identifier mask
- Maskable interrupt
- Disable Automatic Retransmission mode for Time Triggered CAN (TTCAN) applications
- Programmable Loopback mode for self-test operation
- Programmable FIFO mode
- Gluelessly attach to an external CAN PHY through the CANOTx and CANORx pins

# 14.3 Controller Area Network Block Diagram

Figure 14-1. CAN Module Block Diagram



348 November 29, 2007

# 14.4 Controller Area Network Functional Description

The CAN module conforms to the CAN protocol version 2.0 (parts A and B). Message transfers that include data, remote, error, and overload frames with an 11-bit identifier (standard) or a 29-bit identifier (extended) are supported. Transfer rates can be programmed up to 1 Mbps.

The CAN module consists of three major parts:

- CAN protocol controller and message handler
- Message memory
- CAN register interface

The protocol controller transfers and receives the serial data from the CAN bus and passes the data on to the message handler. The message handler then loads this information into the appropriate message object based on the current filtering and identifiers in the message object memory. The message handler is also responsible for generating interrupts based on events on the CAN bus.

The message object memory is a set of 32 identical memory blocks that hold the current configuration, status, and actual data for each message object. These are accessed via the CAN message object register interface. The message memory is not directly accessable in the Stellaris<sup>®</sup> memory map, so the Stellaris<sup>®</sup> CAN controller provides an interface to communicate with the message memory.

The CAN message object register interface provides two register sets for communicating with the message objects. Since there is no direct access to the message object memory, these two interfaces must be used to read or write to each message object. The two message object interfaces allow parallel access to the CAN controller message objects when multiple objects may have new information that needs to be processed.

#### 14.4.1 Initialization

The software initialization is started by setting the INIT bit in the **CAN Control (CANCTL)** register, with software or by a hardware reset, or by going bus-off, which occurs when the transmitter's error counter exceeds a count of 255. While INIT is set, all message transfers to and from the CAN bus are stopped and the status of the CAN transmit output is recessive (High). Entering the initialization state does not change the configuration of the CAN controller, the message objects, or the error counters. However, some configuration registers are only accessible when in the initialization state.

To initialize the CAN controller, set the **CAN Bit Timing (CANBIT)** register and configure each message object. If a message object is not needed, it is sufficient to set it as not valid by clearing the MsgVal bit in the **CANIFnARB2** register. Otherwise, the whole message object has to be initialized, as the fields of the message object may not have valid information causing unexpected results. Access to the **CAN Bit Timing (CANBIT)** register and to the **CAN Baud Rate Prescalar Extension (CANBRPE)** register to configure the bit timing are enabled when both the INIT and CCE bits in the **CANCTL** register are set. To leave the initialization state, the INIT bit must be cleared. Afterwards, the internal Bit Stream Processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus Idle) before it takes part in bus activities and starts message transfers. The initialization of the message objects is independent of being in the initialization state and can be done on the fly, but message objects should all be configured to particular identifiers or set to not valid before the BSP starts the message transfer. To change the configuration of a message object during normal operation, set the MsgVal bit in the **CANIFnARB2** register to 0 (not valid). When the configuration is completed, MsgVal is set to 1 again (valid).

#### 14.4.2 Operation

Once the CAN module is initialized and the INIT bit in the **CANCTL** register is reset to 0, the CAN module synchronizes itself to the CAN bus and starts the message transfer. As messages are received, they are stored in their appropriate message objects if they pass the message handler's filtering. The whole message (including all arbitration bits, data-length code, and eight data bytes) is stored in the message object. If the Identifier Mask (the Msk bits in the **CANIFnMSKn** registers) is used, the arbitration bits which are masked to "don't care" may be overwritten in the message object.

The CPU may read or write each message any time via the CAN Interface Registers (CANIFnCRQ, CANIFnCMSK, CANIFnMSKn, CANIFnARBn, CANIFnMCTL, CANIFnDAn, and CANIFnDBn). The message handler guarantees data consistency in case of concurrent accesses.

The transmission of message objects are under the control of the software that is managing the CAN hardware. These can be message objects used for one-time data transfers, or permanent message objects used to respond in a more periodic manner. Permanent message objects have all arbitration and control set up, and only the data bytes are updated. To start the transmission, the  $\mathtt{TxRqst}$  bit in the **CANTXRQn** register and the  $\mathtt{NewDat}$  bit in the **CANNWDAn** register are set. If several transmit messages are assigned to the same message object (when the number of message objects is not sufficient), the whole message object has to be configured before the transmission of this message is requested.

The transmission of any number of message objects may be requested at the same time; they are transmitted according to their internal priority, which is based on the message identifier for the message object. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data is discarded when a message is updated before its pending transmission has started. Depending on the configuration of the message object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

There are two sets of CAN Interface Registers (**CANIF1x** and **CANIF2x**), which are used to access the Message Objects in the Message RAM. The CAN controller coordinates transfers to and from the Message RAM to and from the registers. The function of the two sets are independent and identical and can be used to gueue transactions.

### 14.4.3 Transmitting Message Objects

If the internal transmit shift register of the CAN module is ready for loading, and if there is no data transfer between the CAN Interface Registers and message RAM, the valid message object with the highest priority and that has a pending transmission request is loaded into the transmit shift register by the message handler and the transmission is started. The message object's NewDat bit is reset and can be viewed in the CANNWDAn register. After a successful transmission, and if no new data was written to the message object since the start of the transmission, the TxRqst bit in the CANIFnCMSK register is reset. If the TxIE bit in the CANIFnMCTL register is set, the IntPnd bit in the CANIFnMCTL register is set after a successful transmission. If the CAN module has lost the arbitration or if an error occurred during the transmission, the message is re-transmitted as soon as the CAN bus is free again. If, meanwhile, the transmission of a message with higher priority has been requested, the messages are transmitted in the order of their priority.

### 14.4.4 Configuring a Transmit Message Object

Table 14-1 on page 351 specifies the bit settings for a transmit message object.

uny Data Shoot / LL con

Table 14-1. Transmit Message Object Bit Settings

Register	CANIFnARB2	CAI	CANIFnCMSK		CANIFnMCTL	CANIFnARB2		CANIFnMCTL					
Bit	MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
Value	1	appl	appl	appl	1	1	0	0	0	appl	0	appl	0

The Xtd and ID bit fields in the **CANIFnARBn** registers are set by an application. They define the identifier and type of the outgoing message. If an 11-bit Identifier (Standard Frame) is used, it is programmed to bits [28:18] of **CANIFnARB1**, as bits 17:0 of **CANIFnARBn** are not used by the CAN controller for 11-bit identifiers.

If the  $\mathtt{TxIE}$  bit is set, the  $\mathtt{IntPnd}$  bit is set after a successful transmission of the message object.

If the  $\mathtt{RmtEn}$  bit is set, a matching received Remote Frame causes the  $\mathtt{TxRqst}$  bit to be set and the Remote Frame is autonomously answered by a Data Frame with the data from the message object.

The DLC bit in the **CANIFNMCTL** register is set by an application. TxRqst and RmtEn may not be set before the data is valid.

The CAN mask registers (Msk bits in **CANIFnMSKn**, UMask bit in **CANIFnMCTL** register, and MXtd and MDir bits in **CANIFnMSK2** register) may be used (UMask=1) to allow groups of Remote Frames with similar identifiers to set the TxRqst bit. The Dir bit should not be masked.

### 14.4.5 Updating a Transmit Message Object

The CPU may update the data bytes of a Transmit Message Object any time via the CAN Interface Registers and neither the MsgVal nor the TxRqst bits have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding **CANIFnDAn** or **CANIFnDBn** register have to be valid before the content of that register is transferred to the message object. Either the CPU has to write all four bytes into the **CANIFnDAn** or **CANIFnDBn** register or the message object is transferred to the **CANIFnDAn** or **CANIFnDBn** register before the CPU writes the new data bytes.

In order to only update the data in a message object, the WR, NewDat, DataA, and DataB bits are written to the CAN IFn Command Mask (CANIFnMSKn) register, followed by writing the CAN IFn Data registers, and then the number of the message object is written to the CAN IFn Command Request (CANIFnCRQ) register, to update the data bytes and the TxRqst bit at the same time.

To prevent the reset of  $\mathtt{TxRqst}$  at the end of a transmission that may already be in progress while the data is updated,  $\mathtt{NewDat}$  has to be set together with  $\mathtt{TxRqst}$ . When  $\mathtt{NewDat}$  is set together with  $\mathtt{TxRqst}$ ,  $\mathtt{NewDat}$  is reset as soon as the new transmission has started.

### 14.4.6 Accepting Received Message Objects

When the arbitration and control field (ID + Xtd + RmtEn + DLC) of an incoming message is completely shifted into the CAN module, the message handling capability of the module starts scanning the message RAM for a matching valid message object. To scan the message RAM for a matching message object, the Acceptance Filtering unit is loaded with the arbitration bits from the core. Then the arbitration and mask fields (including MsgVal, UMask, NewDat, and EoB) of message object 1 are loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following message object until a matching message object is found or until the end of the message RAM is reached. If a match occurs, the scanning is stopped and the message handler proceeds depending on the type of frame received.

ww DataSheet4U con

### 14.4.7 Receiving a Data Frame

The message handler stores the message from the CAN module receive shift register into the respective message object in the message RAM. It stores the data bytes, all arbitration bits, and the Data Length Code into the corresponding message object. This is implemented to keep the data bytes connected with the identifier even if arbitration mask registers are used. The CANIFNMCTL.NewDat bit is set to indicate that new data has been received. The CPU should reset CANIFNMCTL.NewDat when it reads the message object to indicate to the controller that the message has been received and the buffer is free to receive more messages. If the CAN controller receives a message and the CANIFNMCTL.NewDat bit was already set, the MsgLst bit is set to indicate that the previous data was lost. If the CANIFNMCTL.RxIE bit is set, the CANIFNMCTL.IntPnd bit is set, causing the CANIFNMCTL.RxIE bit opinit to the message object that just received a message. The CANIFNMCTL.TxRqst bit of this message object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

### 14.4.8 Receiving a Remote Frame

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

- Dir = 1 (direction = transmit), RmtEn = 1, UMask = 1 or 0
  - At the reception of a matching Remote Frame, the TxRqst bit of this message object is set. The rest of the message object remains unchanged.
- Dir = 1 (direction = transmit), RmtEn = 0, UMask = 0
  - At the reception of a matching Remote Frame, the TxRqst bit of this message object remains unchanged; the Remote Frame is ignored. This remote frame is disabled and will not automatically respond or indicate that the remote frame ever happened.
- Dir = 1 (direction = transmit), RmtEn = 0, UMask = 1

At the reception of a matching Remote Frame, the TxRqst bit of this message object is reset. The arbitration and control field (ID + Xtd + RmtEn + DLC) from the shift register is stored into the message object in the message RAM and the NewDat bit of this message object is set. The data field of the message object remains unchanged; the Remote Frame is treated similar to a received Data Frame. This is useful for a remote data request from another CAN device for which the Stellaris® controller does not have readily available data. The software must fill the data and answer the frame manually.

### 14.4.9 Receive/Transmit Priority

The receive/transmit priority for the message objects is controlled by the message number. Message object 1 has the highest priority, while message object 32 has the lowest priority. If more than one transmission request is pending, the message objects are transmitted in order based on the message object with the lowest message number. This should not be confused with the message identifier as that priority is enforced by the CAN bus. This means that if message object 1 and message object 2 both have valid messages that need to be transmitted, message object 1 will always be transmitted first regardless of the message identifier in the message object itself.

### 14.4.10 Configuring a Receive Message Object

Table 14-2 on page 353 specifies the bit settings for a transmit message object.

Table 14-2. Receive Message Object Bit Settings

Register	CANIFnARB2	CA	CANIFnCMSK		CANIFnMCTL	CANIFnARB2	CANIFnMCTL						
Bit	MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
Value	1	appl	appl	appl	1	0	0	0	appl	0	0	0	0

The Xtd and ID bit fields in the **CANIFnARBn** registers are set by an application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (Standard Frame) is used, it is programmed to bits [28:18] of **CANIFnARB1**, and bits [17:0] are ignored by the CAN controller. When a Data Frame with an 11-bit Identifier is received, bits [17:0] are set to 0.

If the RxIE bit is set, the IntPnd bit is set when a received Data Frame is accepted and stored in the message object.

When the message handler stores a Data Frame in the message object, it stores the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the message object are overwritten by nonspecified values.

The CAN mask registers (Msk bits in **CANIFnMSKn**, UMask bit in **CANIFnMCTL** register, and MXtd and MDir bits in **CANIFnMSK2** register) may be used (UMask=1) to allow groups of Data Frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications.

### 14.4.11 Handling of Received Message Objects

The CPU may read a received message any time via the CAN Interface registers because the data consistency is guaranteed by the message handler state machine.

Typically, the CPU first writes 0x007F to the CAN IFn Command Mask (CANIFnCMSK) register and then writes the number of the message object to the CAN IFn Command Request (CANIFnCRQ) register. That combination transfers the whole received message from the message RAM into the Message Buffer registers (CANIFnMSKn, CANIFnARBn, and CANIFnMCTL). Additionally, the NewDat and IntPnd bits are cleared in the message RAM, acknowledging that the message has been read and clearing the pending interrupt being generated by this message object.

If the message object uses masks for acceptance filtering, the arbitration bits show which of the matching messages has been received.

The actual value of NewDat shows whether a new message has been received since the last time this message object was read. The actual value of MsgLst shows whether more than one message has been received since the last time this message object was read. MsgLst is not automatically reset.

Using a Remote Frame, the CPU may request new data from another CAN node on the CAN bus. Setting the  $\mathtt{TxRqst}$  bit of a receive object causes the transmission of a Remote Frame with the receive object's identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the  $\mathtt{TxRqst}$  bit is automatically reset. This prevents the possible loss of data when the other device on the CAN bus has already transmitted the data, slightly earlier than expected.

### 14.4.12 Handling of Interrupts

If several interrupts are pending, the **CAN Interrupt (CANINT)** register points to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it.

ny DataShoot/III oon

The Status Interrupt has the highest priority. Among the message interrupts, the message object's interrupt priority decreases with increasing message number. A message interrupt is cleared by clearing the message object's IntPnd bit. The Status Interrupt is cleared by reading the **CAN Status** (**CANSTS**) register.

The interrupt identifier IntId in the **CANINT** register indicates the cause of the interrupt. When no interrupt is pending, the register holds the value to 0. If the value of **CANINT** is different from 0, then there is an interrupt pending. If the IE bit is set in the **CANCTL** register, the interrupt line to the CPU is active. The interrupt line remains active until **CANINT** is 0, all interrupt sources have been cleared, (the cause of the interrupt is reset), or until IE is reset, which disables interrupts from the CAN controller.

The value 0x8000 in the **CANINT** register indicates that an interrupt is pending because the CAN module has updated, but not necessarily changed, the **CANSTS** register (Error Interrupt or Status Interrupt). This indicates that there is either a new Error Interrupt or a new Status Interrupt. A write access can clear the RxOK, TxOK, and LEC flags in the **CANSTS** register, however, only a read access to the **CANSTS** register will clear the source of the status interrupt.

IntId points to the pending message interrupt with the highest interrupt priority. The SIE bit in the **CANCTL** register controls whether a change of the status register may cause an interrupt. The EIE bit in the **CANCTL** register controls whether any interrupt from the CAN controller actually generates an interrupt to the microcontroller's interrupt controller. The **CANINT** interrupt register is updated even when the IE bit is set to zero.

There are two possibilities when handling the source of a message interrupt. The first is to read the IntId bit in the **CANINT** interrupt register to determine the highest priority interrupt that is pending, and the second is to read the **CAN Message Interrupt Pending (CANMSGnINT)** register to see all of the message objects that have pending interrupts.

An interrupt service routine reading the message that is the source of the interrupt may read the message and reset the message object's IntPnd at the same time by setting the ClrIntPnd bit in the CAN IFn Command Mask (CANIFnCMSK) register. When the IntPnd bit is cleared, the CANINT register will contain the message number for the next message object with a pending interrupt.

### 14.4.13 Bit Timing Configuration Error Considerations

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly. In many cases, the CAN bit synchronization amends a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration, however, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive. The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and of the CAN nodes' interaction on the CAN bus.

#### 14.4.14 Bit Time and Bit Rate

The CAN system supports bit rates in the range of lower than 1 Kbps up to 1000 Kbps. Each member of the CAN network has its own clock generator. The timing parameter of the bit time can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes' oscillator periods may be different.

Because of small variations in frequency caused by changes in temperature or voltage and by deteriorating components, these oscillators are not absolutely stable. As long as the variations

www.DataSheet4U.com

remain inside a specific oscillator's tolerance range, the CAN nodes are able to compensate for the different bit rates by periodically resynchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see Figure 14-2 on page 355): the Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1, and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (see Table 14-3 on page 355). The length of the time quantum (tq), which is the basic time unit of the bit time, is defined by the CAN controller's system clock (fsys) and the Baud Rate Prescaler (grap):

The CAN module's system clock fsys is the frequency of its CAN module clock (CAN\_CLK) input.

The Synchronization Segment Sync\_Seg is that part of the bit time where edges of the CAN bus level are expected to occur; the distance between an edge that occurs outside of Sync\_Seg and the Sync\_Seg is called the *phase error* of that edge.

The Propagation Time Segment Prop\_Seg is intended to compensate for the physical delay times within the CAN network.

The Phase Buffer Segments Phase\_Seg1 and Phase\_Seg2 surround the Sample Point.

The (Re-)Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

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

Figure 14-2. CAN Bit Time

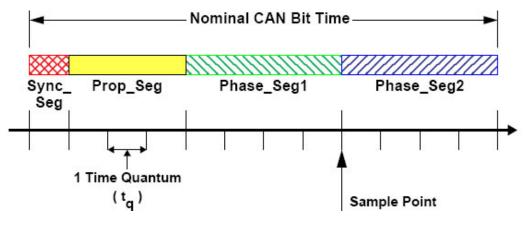


Table 14-3. CAN Protocol Ranges<sup>a</sup>

Parameter	Range	Remark
BRP	[1 32]	Defines the length of the time quantum t <sub>q</sub>
Sync_Seg	1 t <sub>q</sub>	Fixed length, synchronization of bus input to system clock
Prop_Seg	[1 8] t <sub>q</sub>	Compensates for the physical delay times
Phase_Seg1	[1 8] t <sub>q</sub>	May be lengthened temporarily by synchronization
Phase_Seg2	[1 8] t <sub>q</sub>	May be shortened temporarily by synchronization

Parameter	Range	Remark
SJW	[1 4] t <sub>q</sub>	May not be longer than either Phase Buffer Segment

a. This table describes the minimum programmable ranges required by the CAN protocol.

The bit timing configuration is programmed in two register bytes in the **CANBIT** register. The sum of Prop\_Seg and Phase\_Seg1 (as TSEG1) is combined with Phase\_Seg2 (as TSEG2) in one byte, and SJW and BRP are combined in the other byte.

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

```
[TSEG1 + TSEG2 + 3] tq
or (functional values):
[Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq
```

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

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the CAN controller and are evaluated once per time quantum.

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

The IPT is application-specific but may not be longer than 2 tq; the CAN's IPT is 0 tq. Its length is the lower limit of the programmed length of  $Phase\_Seg2$ . In case of synchronization,  $Phase\_Seg2$  may be shortened to a value less than IPT, which does not affect bus timing.

### 14.4.15 Calculating the Bit Timing Parameters

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

The bit time may consist of 4 to 25 time quanta. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

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

The  $Sync\_Seg$  is 1 tq long (fixed), which leaves (bit time -  $Prop\_Seg$  - 1) tq for the two Phase Buffer Segments. If the number of remaining tq is even, the Phase Buffer Segments have the same length, that is,  $Phase\_Seg2$  =  $Phase\_Seg1$ , else  $Phase\_Seg2$  =  $Phase\_Seg1$  + 1.

www.DataSheet4U.com

NANA Data Shoot All Loor

The minimum nominal length of Phase\_Seg2 has to be regarded as well. Phase\_Seg2 may not be shorter than the CAN controller's Information Processing Time, which is, depending on the actual implementation, in the range of [0..2] tq.

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase Seq1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formula given below:

```
(1 - df) \times fnom <= fosc <= (1 + df) \times fnom
```

#### where:

- df = maximum tolerance of oscillator frequency
- fosc = actual oscillator frequency
- fnom = nominal oscillator frequency

Maximum frequency tolerance must take into account the following formulas:

```
df <= (Phase_Seg1,Phase_Seg2)min/ 2 x (13 x tbit - Phase_Seg2)
dfmax = 2 x df x fnom</pre>
```

#### where:

- Phase\_Seg1 and Phase\_Seg2 are from Table 14-3 on page 355
- tbit = Bit Time
- dfmax = maximum difference between two oscillators

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

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

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

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

The resulting configuration is written into the CAN Bit Timing (CANBIT) register :

```
(Phase_Seg2-1)&(Phase_Seg1+Prop_Seg-1)&(SynchronizationJumpWidth-1)&(Prescaler-1)
```

#### 14.4.15.1 Example for Bit Timing at High Baud Rate

In this example, the frequency of CAN\_CLK is 10 MHz, BRP is 0, and the bit rate is 1 Mbps.

```
tq 100 ns = tCAN_CLK
delay of bus driver 50 ns
delay of receiver circuit 30 ns
delay of bus line (40m) 220 ns
```

November 29, 2007

Preliminary

www.DataSheet4U.com

```
tProp 600 ns = 6 × tq
tSJW 100 ns = 1 × tq
tTSeg1 700 ns = tProp + tSJW
tTSeg2 200 ns = Information Processing Time + 1 × tq
tSync-Seg 100 ns = 1 × tq
bit time 1000 ns = tSync-Seg + tTSeg1 + tTSeg2
tolerance for CAN_CLK 0.39 % =
    min(PB1,PB2)/ 2 × (13 x bit time - PB2) =
    0.1us/ 2 x (13x lus - 2us)
```

In the above example, the concatenated bit time parameters are (2-1)3&(7-1)4&(1-1)2&(1-1)6, and **CANBIT** is programmed to 0x1600.

### www.DataSheet414.4.15.2 Example for Bit Timing at Low Baud Rate

In this example, the frequency of CAN\_CLK is 2 MHz, BRP is 1, and the bit rate is 100 Kbps.

```
tq 1 ms = 2 x tCAN_CLK
delay of bus driver 200 ns
delay of receiver circuit 80 ns
delay of bus line (40m) 220 ns
tProp 1 ms = 1 x tq
tSJW 4 ms = 4 x tq
tTSeg1 5 ms = tProp + tSJW
tTSeg2 4 ms = Information Processing Time + 3 x tq
tSync-Seg 1 ms = 1 x tq
bit time 10 ms = tSync-Seg + tTSeg1 + tTSeg2
tolerance for CAN_CLK 1.58 % =
   min(PB1,PB2)/ 2 x (13 x bit time - PB2) =
   4us/ 2 x (13 x 10us - 4us)
```

In this example, the concatenated bit time parameters are (4-1)3&(5-1)4&(4-1)2&(2-1)6, and **CANBIT** is programmed to 0x34C1.

# 14.5 Controller Area Network Register Map

Table 14-4 on page 358 lists the registers. All addresses given are relative to the CAN base address of:

CAN0: 0x4004.0000

All accesses are on word (32-bit) boundaries.

Table 14-4. CAN Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	CANCTL	R/W	0x0000.0001	CAN Control	361
0x004	CANSTS	R/W	0x0000.0000	CAN Status	363
0x008	CANERR	RO	0x0000.0000	CAN Error Counter	366
0x00C	CANBIT	R/W	0x0000.2301	CAN Bit Timing	367
0x010	CANINT	RO	0x0000.0000	CAN Interrupt	369

Offset	Name	Туре	Reset	Description	See page
0x014	CANTST	R/W	0x0000.0000	CAN Test	370
0x018	CANBRPE	R/W	0x0000.0000	CAN Baud Rate Prescalar Extension	372
0x020	CANIF1CRQ	R/W	0x0000.0001	CAN IF1 Command Request	373
0x024	CANIF1CMSK	R/W	0x0000.0000	CAN IF1 Command Mask	374
0x028	CANIF1MSK1	R/W	0x0000.FFFF	CAN IF1 Mask 1	377
0x02C	CANIF1MSK2	R/W	0x0000.FFFF	CAN IF1 Mask 2	378
0x030	CANIF1ARB1	R/W	0x0000.0000	CAN IF1 Arbitration 1	379
0x034	CANIF1ARB2	R/W	0x0000.0000	CAN IF1 Arbitration 2	380
0x038	CANIF1MCTL	R/W	0x0000.0000	CAN IF1 Message Control	381
0x03C	CANIF1DA1	R/W	0x0000.0000	CAN IF1 Data A1	383
0x040	CANIF1DA2	R/W	0x0000.0000	CAN IF1 Data A2	383
0x044	CANIF1DB1	R/W	0x0000.0000	CAN IF1 Data B1	383
0x048	CANIF1DB2	R/W	0x0000.0000	CAN IF1 Data B2	383
0x080	CANIF2CRQ	R/W	0x0000.0001	CAN IF2 Command Request	373
0x084	CANIF2CMSK	R/W	0x0000.0000	CAN IF2 Command Mask	374
0x088	CANIF2MSK1	R/W	0x0000.FFFF	CAN IF2 Mask 1	377
0x08C	CANIF2MSK2	R/W	0x0000.FFFF	CAN IF2 Mask 2	378
0x090	CANIF2ARB1	R/W	0x0000.0000	CAN IF2 Arbitration 1	379
0x094	CANIF2ARB2	R/W	0x0000.0000	CAN IF2 Arbitration 2	380
0x098	CANIF2MCTL	R/W	0x0000.0000	CAN IF2 Message Control	381
0x09C	CANIF2DA1	R/W	0x0000.0000	CAN IF2 Data A1	383
0x0A0	CANIF2DA2	R/W	0x0000.0000	CAN IF2 Data A2	383
0x0A4	CANIF2DB1	R/W	0x0000.0000	CAN IF2 Data B1	383
0x0A8	CANIF2DB2	R/W	0x0000.0000	CAN IF2 Data B2	383
0x100	CANTXRQ1	RO	0x0000.0000	CAN Transmission Request 1	384
0x104	CANTXRQ2	RO	0x0000.0000	CAN Transmission Request 2	384
0x120	CANNWDA1	RO	0x0000.0000	CAN New Data 1	385
0x124	CANNWDA2	RO	0x0000.0000	CAN New Data 2	385
0x140	CANMSG1INT	RO	0x0000.0000	CAN Message 1 Interrupt Pending	386
0x144	CANMSG2INT	RO	0x0000.0000	CAN Message 2 Interrupt Pending	386
0x160	CANMSG1VAL	RO	0x0000.0000	CAN Message 1 Valid	387
0x164	CANMSG2VAL	RO	0x0000.0000	CAN Message 2 Valid	387

November 29, 2007 359

# 14.6 Register Descriptions

The remainder of this section lists and describes the CAN registers, in numerical order by address offset. There are two sets of Interface Registers which are used to access the Message Objects in the Message RAM: **CANIF1x** and **CANIF2x**. The function of the two sets are identical and are used to queue transactions.

www.DataSheet4LL.com

www.DataSheet4U.com

### Register 1: CAN Control (CANCTL), offset 0x000

This control register initializes the module and enables test mode and interrupts.

The bus-off recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting INIT. If the device goes bus-off, it sets INIT, stopping all bus activities. Once INIT has been cleared by the CPU, the device then waits for 129 occurrences of Bus Idle (129 \* 11 consecutive High bits) before resuming normal operations. At the end of the bus-off recovery sequence, the Error Management Counters are reset.

During the waiting time after INIT is reset, each time a sequence of 11 High bits has been monitored, a BitOError code is written to the **CANSTS** status register, enabling the CPU to readily check whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus-off recovery sequence.

ww.DataSheet4U.com

#### CAN Control (CANCTL)

CAN0 base: 0x4004.0000 Offset 0x000 Type R/W, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			•	1				rese	rved							•
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
_	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		'	'	rese	erved	1	,	1	Test	CCE	DAR	reserved	EIE	SIE	IE	INIT
Туре	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	Test	R/W	0	Test Mode Enable 0: Normal Operation 1: Test Mode
6	CCE	R/W	0	Configuration Change Enable  0: Do not allow write access to the <b>CANBIT</b> register.  1: Allow write access to the <b>CANBIT</b> register if the INIT bit is 1.
5	DAR	R/W	0	Disable Automatic Retransmission  0: Auto retransmission of disturbed messages is enabled.  1: Auto retransmission is disabled.
4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	EIE	R/W	0	Error Interrupt Enable  0: Disabled. No Error Status interrupt is generated.  1: Enabled. A change in the Boff or Ewarn bits in the CANSTS register

generates an interrupt.

	Bit/Field	Name	Туре	Reset	Description
	2	SIE	R/W	0	Status Change Interrupt Enable
					0: Disabled. No Status Change interrupt is generated.
					1: Enabled. An interrupt is generated when a message has successfully been transmitted or received, or a CAN bus error has been detected. A change in the ${\tt TxOk}$ or ${\tt RxOk}$ bits in the <code>CANSTS</code> register generates an interrupt.
	1	IE	R/W	0	CAN Interrupt Enable
					0: Interrupt disabled.
					1: Interrupt enabled.
w.DataSheet4U.	com 0	INIT	R/W	1	Initialization
					0: Normal operation.
					1: Initialization started.

### Register 2: CAN Status (CANSTS), offset 0x004

The status register contains information for interrupt servicing such as Bus-Off, error count threshold, and error types.

The LEC field holds the code that indicates the type of the last error to occur on the CAN bus. This field is cleared to 0 when a message has been transferred (reception or transmission) without error. The unused error code 7 may be written by the CPU to check for updates.

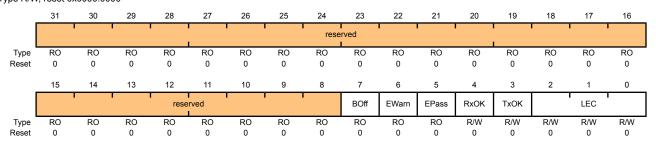
An Error Interrupt is generated by the BOff and EWarn bits and a Status Change Interrupt is generated by the RxOk, TxOk, and LEC bits, assuming that the corresponding enable bits in the **CAN Control (CANCTL)** register are set. A change of the EPass bit or a write to the RxOk, TxOk, or LEC bits does not generate an interrupt.

Reading the **CAN Status (CANSTS)** register clears the **CAN Interrupt (CANINT)** register, if it is pending.

00110100111

#### CAN Status (CANSTS)

CAN0 base: 0x4004.0000 Offset 0x004 Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:8	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7	BOff	RO	0	Bus-Off Status
				0: Module is not in bus-off state.
				1: Module is in bus-off state.
6	EWarn	RO	0	Warning Status
				0: Both error counters are below the error warning limit of 96.
				1: At least one of the error counters has reached the error warning limit of 96.
5	EPass	RO	0	Error Passive

0: The CAN module is in the Error Active state, that is, the receive or transmit error count is less than or equal to 127.

1: The CAN module is in the Error Passive state, that is, the receive or transmit error count is greater than 127.

Bit/Field	Name	Type	Reset	Description
4	RxOK	R/W	0	Received a Message Successfully
				0: Since this bit was last reset to 0, no message has been successfully received.
				1: Since this bit was last reset to 0, a message has been successfully received, independent of the result of the acceptance filtering.
				This bit is never reset by the CAN module.
3	TxOK	R/W	0	Transmitted a Message Successfully
				0: Since this bit was last reset to 0, no message has been successfully transmitted. $ \\$
				1: Since this bit was last reset to 0, a message has been successfully transmitted error-free and acknowledged by at least one other node.
				This bit is never reset by the CAN module.

Bit/Field	Name	Туре	Reset	Description
2:0	LEC	R/W	0x0	Last Error Code
				This is the type of the last error to occur on the CAN bus.

Value Definition 0x0 No Error 0x1 Stuff Error

More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.

0x2 Form Erro

A fixed format part of the received frame has the wrong format.

0x3 ACK Error

The message transmitted was not acknowledged by another node.

0x4 Bit 1 Error

When a message is transmitted, the CAN controller monitors the data lines to detect any conflicts. When the arbitration field is transmitted, data conflicts are a part of the arbitration protocol. When other frame fields are transmitted, data conflicts are considered errors.

A Bit 1 Error indicates that the device wanted to send a High level (logical 1) but the monitored bus value was Low (logical 0)

0x5 Bit 0 Error

A Bit 0 Error indicates that the device wanted to send a Low level (logical 0) but the monitored bus value was High (logical 1)

During bus-off recovery, this status is set each time a sequence of 11 High bits has been monitored. This enables the CPU to monitor the proceeding of the bus-off recovery sequence without any disturbances to the bus.

0x6 CRC Error

The CRC checksum was incorrect in the received message, indicating that the calculated value received did not match the calculated CRC of the data.

0x7 Unused

When the LEC bit shows this value, no CAN bus event was detected since the CPU wrote this value to LEC.

November 29, 2007 365

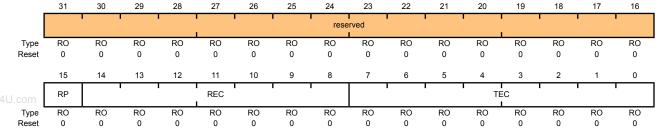
### Register 3: CAN Error Counter (CANERR), offset 0x008

This register contains the error counter values, which can be used to analyze the cause of an error.

#### CAN Error Counter (CANERR)

CAN0 base: 0x4004.0000 Offset 0x008

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	RP	RO	0	Received Error Passive
				0: The Receive Error counter is below the Error Passive level (127 or less).
				1: The Receive Error counter has reached the Error Passive level (128 or greater).
14:8	REC	RO	0x0	Receive Error Counter
				State of the receiver error counter (0 to 127).
7:0	TEC	RO	0x0	Transmit Error Counter
				State of the transmit error counter (0 to 255).

### Register 4: CAN Bit Timing (CANBIT), offset 0x00C

This register is used to program the bit width and bit quantum. Values are to be programmed to the system clock frequency. This register is write-enabled by the CCE and INIT bits in the **CANCTL** register.

With a CAN module clock (CAN\_CLK) of 8 MHz, the register reset value of 0x230 configures the CAN for a bit rate of 500 Kbps.

#### CAN Bit Timing (CANBIT)

CAN0 base: 0x4004.0000 Offset 0x00C Type R/W, reset 0x0000.2301

30 29 25 16 28 26 23 22 reserved RO Type Reset 0 0 0 0 0 0 0 0 0 0 0 0 15 13 12 10 TSea2 BRP SJW eserved TSeq1 Туре RO R/W 0 0 0 0 0 0 0 0 0 0 0 0 Reset Bit/Field Description Name Type Reset 31:15 reserved RO 0x0000 Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation. 14:12 TSeg2 R/W 0x2 Time Segment after Sample Point 0x00-0x07: The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. So, for example, a reset value of 0x2 defines that there is 3(2+1) bit time quanta defined for Phase\_Seg2 (see Figure 14-2 on page 355). The bit time quanta is defined by BRP. Time Segment Before Sample Point 11:8 TSeg1 R/W 0x3 0x00-0x0F: The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. So, for example, the reset value of 0x3 defines that there is 4(3+1) bit time quanta defined for Phase\_Seg1 (see Figure 14-2 on page 355). The bit time quanta is define by BRP. 7:6 SJW R/W 0x0 (Re)Synchronization Jump Width

0x00-0x03: The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

During the start of frame (SOF), if the CAN controller detects a phase error (misalignment), it can adjust the length of  ${\tt TSeg2}$  or  ${\tt TSeg1}$  by the value in  ${\tt SJW}$ . So the reset value of 0 adjusts the length by 1 bit time quanta.

Bit/Field	Name	Type	Reset	Description
5:0	BRP	R/W	0x1	Baud Rate Prescalar
				0x00-0x03F: The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quantum. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
				$\tt BRP$ defines the number of CAN clock periods that make up 1 bit time quanta, so the reset value is 2 bit time quanta (1+1).
				The <b>BRPRE</b> register can be used to further divide the bit time.

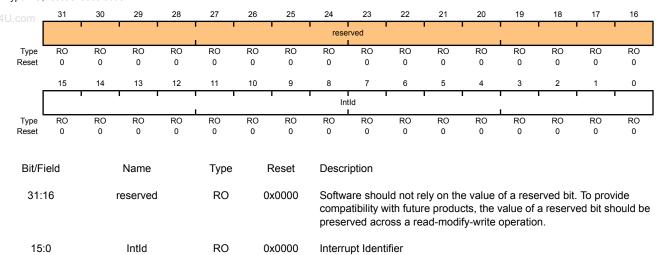
### Register 5: CAN Interrupt (CANINT), offset 0x010

This register indicates the source of the interrupt.

If several interrupts are pending, the **CAN Interrupt (CANINT)** register points to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it. If the IntId bit is not 0x0000 (the default) and the IE bit in the **CANCTL** register is set, the interrupt is active. The interrupt line remains active until the IntId bit is set back to 0x0000 when the cause of all interrupts are reset or until IE is reset.

#### CAN Interrupt (CANINT)

CAN0 base: 0x4004.0000 Offset 0x010 Type RO, reset 0x0000.0000



The number in this field indicates the source of the interrupt.

Value Definition

0x0000 No interrupt pending

0x0001-0x0020 Number of the message object that caused the

interrupt

0x0021-0x7FFF Unused

0x8000 Status Interrupt

0x8001-0xFFFF Unused

### Register 6: CAN Test (CANTST), offset 0x014

26

RO

RO

This is the test mode register for self-test and external pin access. It is write-enabled by the Test bit in the **CANCTL** register. Different test functions may be combined but when the  $\mathtt{Tx}$  bit is not equal to 0x0, it disturbs message transmits.

reserved

RO

RO

RO

RO

RO

RO

18

RO

RO

RO

www.DataSheet4U.com

#### CAN Test (CANTST)

CAN0 base: 0x4004.0000

Offset 0x014
Type R/W, reset 0x0000.0000

RO

30

RO

RO

RO

RO

Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
.com	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1		reser	ved		•	'	Rx	Т	x	LBack	Silent	Basic	rese	rved
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	RO 0	RO 0
Bit/Fi	eld		Name		Туре	ı	Reset	Descr	iption							
31:	8		reserved		RO	0	0000x	O000 Software should not compatibility with fur preserved across a			e produ	cts, the v	alue of	a reserv		
7			Rx		RO		0	Recei	ve Obse	rvation						
								Displa	ys the v	alue on	the CAN	nRx <b>pin</b> .				
6:5	5		Tx		R/W		0x0	Trans	mit Cont	rol						
								Overri	des con	trol of th	<b>e</b> CANn1	x pin.				
								Value	Descri	ption						
								00	CAN_	TX is cor	ntrolled	by the C	AN mod	ule (defa	ault)	
								01	Sampl	e Point s	signal dı	iven on	the CAN	_TX pin		
								10	CAN_	TX drives	s a Low	value				
								11	CAN_	TX drive:	s a High	value				
4			LBack		R/W		0	Loopb	ack Mo	de						
								0: Dis	abled.							
								1: Ena	abled.							
3			Silent		R/W		0	Silent	Mode							
								Do no	t transm	it data; m	nonitor th	ne bus. A	lso knov	vn as Bu	s Monito	r mode.
								0: Dis	abled.							
								1: Ena	abled.							
2			Basic		R/W		0	Basic	Mode							
								0: Dis	abled.							
									e CANIF eive buf	•	ers as tr	ansmit b	uffer, an	d use C	ANIF2 re	egisters

Bit/Field	Name	Type	Reset	Description
1:0	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

www DataSheet411 com

### Register 7: CAN Baud Rate Prescalar Extension (CANBRPE), offset 0x018

This register is used to further divide the bit time set with the BRP bit in the CANBIT register. It is write-enabled with the CCE bit in the CANCTL register.

#### CAN Baud Rate Prescalar Extension (CANBRPE)

**BRPE** 

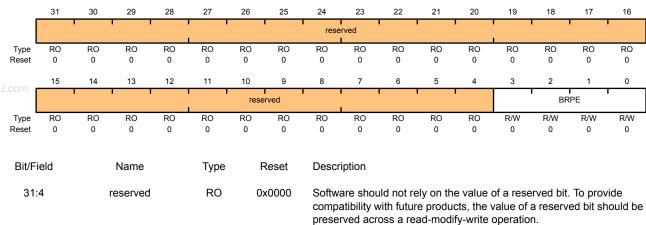
R/W

0x0

CAN0 base: 0x4004.0000

3:0

Offset 0x018 Type R/W, reset 0x0000.0000



Baud Rate Prescalar Extension.

0x00-0x0F: Extend the BRP bit to values up to 1023. The actual interpretation by the hardware is one more than the value programmed by BRPE (MSBs) and BRP (LSBs) are used.

# Register 8: CAN IF1 Command Request (CANIF1CRQ), offset 0x020 Register 9: CAN IF2 Command Request (CANIF2CRQ), offset 0x080

This register is used to start a transfer when its MNUM bit field is updated. Its Busy bit indicates that the information is transferring from the CAN Interface Registers to the internal message RAM.

A message transfer is started as soon as there is a write of the message object number with the MNUM bit. With this write operation, the Busy bit is automatically set to 1 to indicate that a transfer is in progress. After a wait time of 3 to 6 CAN\_CLK periods, the transfer between the interface register and the message RAM completes, which then sets the Busy bit back to 0.

#### CAN IF1 Command Request (CANIF1CRQ)

CAN0 base: 0x4004.0000 Offset 0x020 Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								rese	rved							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Busy			'	1	reserved							MN	UM		
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15	Busy	RO	0x0	Busy Flag  0: Reset when read/write action has finished.  1: Set when a write occurs to the message number in this register.
14:6	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
5:0	MNUM	R/W	0x01	Message Number

Selects one of the 32 message objects in the message RAM for data transfer. The message objects are numbered from 1 to 32.

Value Description

0x00 0 is not a valid message number; it is interpreted as 0x20, or object 32.

0x01-0x20 Indicates specified message object 1 to 32.

0x21-0x3F Not a valid message number; values are shifted and it is interpreted as 0x01-0x1F.

November 29, 2007

# Register 10: CAN IF1 Command Mask (CANIF1CMSK), offset 0x024 Register 11: CAN IF2 Command Mask (CANIF2CMSK), offset 0x084

The Command Mask registers specify the transfer direction and select which buffer registers are the source or target of the data transfer.

#### CAN IF1 Command Mask (CANIF1CMSK)

CAN0 base: 0x4004.0000

Offset 0x024

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		l	! !	'	•		!	rese	rved L				! !			
Type 4 Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			' '	rese	rved		'	'	WRNRD	Mask	Arb	Control	ClrIntPnd	TxRqstNewDat	DataA	DataB
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0
Bit/F	ield		Name		Туре	F	Reset	Descr	ription							
31:8 reserve		reserved		RO	0	x0000	comp		vith futur	e produ	cts, the v	alue of	erved bit. a reserve n.			
7		,	WRNRD		R/W		0	Write,	Not Rea	ad						
								Comr registe CANI	mand Re	quest (0 IIFnMSK CANIFn	CANIFn (1, CANI	CRQ) red IFnMSK	gister to 2, CANII	specifie the CAN FnARB1 NIFnDB1	messag , <b>CANIF</b>	e buffer
									te. Trans t addres:			_		registers egister.	to the m	essage
6	;		Mask		R/W		0x0	Acces	ss Mask	Bits						
								When	WRNRD=	=1 (write:	s):					
								0: Ma	sk bits u	nchange	ed.					
								1: Tra	nsfer ID	Mask +	Dir+M	xtd <b>to</b> n	nessage	object.		
								When	WRNRD=	=0 (reads	s):					
								0: Ma	sk bits u	nchange	ed.					
									nsfer ID ace Regi		Dir+M	xtd <b>of t</b> l	he mess	age obje	ect into t	he
5	i		Arb		R/W		0x0	Acces	ss Arbitra	ation Bits	;					
								When	WRNRD=	=1 (write:	s):					
								0: Arb	itration I	oits unch	anged.					
								1: Tra	nsfer ID	+Dir+	- Xtd + 1	MsgVal	to mess	age obje	ct.	
								When	WRNRD=	=0 (reads	s):					
								0: Arb	itration I	oits unch	anged.					
									_							

1: Transfer ID + Dir + Xtd + MsgVal to Message Buffer Register.

Bit/Field	Name	Туре	Reset	Description
4	Control	R/W	0x0	Access Control Bits
				When wrnrd=1 (writes):
				0: Control bits unchanged.
				1: Transfer control bits to message object.
				When wrnrd=0 (reads):
				0: Control bits unchanged.
				1: Transfer control bits to Message Buffer Register.
3	ClrIntPnd	R/W	0x0	Clear Interrupt Pending Bit
vw.DataSheet4U.com				Note: This bit is not used when in write (WRNRD=1).
				0: IntPnd bit in CANIFnMCTL register remains unchanged.
				1: Clear IntPnd bit in the CANIFnMCTL register in the message object.
2	TxRqst/NewDat	R/W	0x0	Access Transmission Request or New Data
				When wrnrd=1 (writes):
				Access Transmission Request Bit
				0: TxRqst bit unchanged.
				1: Set TxRqst bit
				Note: If a transmission is requested by programming this TxRqst bit, the parallel TxRqst in the CANIFnMCTL register is ignored.
				When wrnrd=0 (reads):
				Access New Data Bit
				0: NewDat bit unchanged.
				1: Clear NewDat bit in the message object.
				Note: A read access to a message object can be combined with the reset of the control bits IntPdn and NewDat. The values of these bits that are transferred to the CANIFnMCTL register always reflect the status before resetting these bits.
1	DataA	R/W	0x0	Access Data Byte 0 to 3
				When wrnrd=1 (writes):
				0: Data bytes 0-3 are unchanged.
				1: Transfer data bytes 0-3 (CANIFnDA1 and CANIFnDA2) to message object.
				When wrnrd=0 (reads):
				0: Data bytes 0-3 are unchanged.
				1: Transfer data bytes 0-3 in message object to <b>CANIFnDA1</b> and <b>CANIFnDA2</b> .

Bit/Field	Name	Туре	Reset	Description
0	DataB	R/W	0x0	Access Data Byte 4 to 7
				When wrnrd=1 (writes):
				0: Data bytes 4-7 unchanged.
				1: Transfer data bytes 4-7 ( <b>CANIFnDB1</b> and <b>CANIFnDB2</b> ) to message object.
				When wrnrd=0 (reads):
				0: Data bytes 4-7 unchanged.
				1: Transfer data bytes 4-7 in message object to <b>CANIFnDB1</b> and <b>CANIFnDB2</b> .

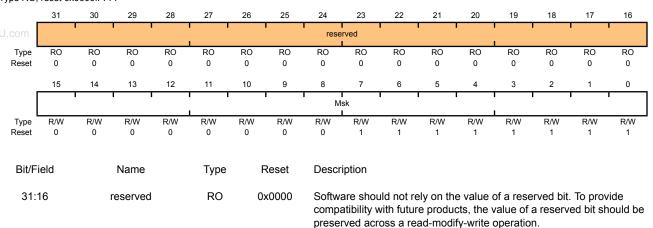
# Register 12: CAN IF1 Mask 1 (CANIF1MSK1), offset 0x028 Register 13: CAN IF2 Mask 1 (CANIF2MSK1), offset 0x088

The mask information provided in this register accompanies the data (CANIFnDAn), arbitration information (CANIFnARBn), and control information (CANIFnMCTL) to the message object in the message RAM. The mask is used with the ID bit in the CANIFnARBn register for acceptance filtering. Additional mask information is contained in the CANIFnMSK2 register.

#### CAN IF1 Mask 1 (CANIF1MSK1)

CAN0 base: 0x4004.0000 Offset 0x028 Type RO, reset 0x0000.FFFF

15:0



Identifier Mask

R/W

Msk

0xFF

0: The corresponding identifier bit (ID) in the message object cannot inhibit the match in acceptance filtering.

<sup>1:</sup> The corresponding identifier bit (ID) is used for acceptance filtering.

## Register 14: CAN IF1 Mask 2 (CANIF1MSK2), offset 0x02C Register 15: CAN IF2 Mask 2 (CANIF2MSK2), offset 0x08C

RO

26

RO

This register holds extended mask information that accompanies the CANIFnMSK1 register.

reserved

RO

RO

RO

RO

#### CAN IF1 Mask 2 (CANIF1MSK2)

30

RO

29

RO

reserved

Msk

RO

R/W

0x1

0xFF

28

CAN0 base: 0x4004.0000 Offset 0x02C

Type RO, reset 0x0000.FFFF 31

RO

13

12:0

Reset	0	0	0	0	0	0	0		0	0	0	0		0	0	0		0	0
J.com	15	14	13	12	11	10	9		8	7	6	5		4	3	2		1	0
	MXtd	MDir	reserved	1			1	ì	1	ı	Msk		Ì	1			ı	,	
Туре	R/W	R/W	RO	R/W	R/W	R/W	R/W		R/W	R/W	R/W	R/W	R	2/W	R/W	R/W	R	/W	R/W
Reset	1	1	1	0	0	0	0		0	1	1	1		1	1	1		1	1
Bit/F	ield		Name		Туре		Reset		Descri	iption									
31:	16		reserved		RO	C	)x0000		compa	are shoul atibility w ved acro	ith futur	e produ	ucts,	the v	alue of	a resei		•	
15	5		MXtd		R/W		0x1		Mask	Extende	d Identif	ier							
										extende ect on the			•		e CANI	FnARI	<b>32</b> re	gister	) has
									1: The	extende	ed identi	fier bit	Xtd	is use	ed for a	ccepta	nce fi	Iterin	g.
14	1		MDir		R/W		0x1		Mask	Message	e Directi	on							
								messag			`	in the	e CANI	FnARI	<b>32</b> re(	gister	) has		
									1: The	messag	e direct	ion bit	Dir	is use	ed for a	ccepta	nce fi	Iterin	<b>g</b> .

Identifier Mask

0: The corresponding identifier bit (ID) in the message object cannot inhibit the match in acceptance filtering.

Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be

preserved across a read-modify-write operation.

18

RO

19

16

RO

www.DataSheet4U.com

1: The corresponding identifier bit (ID) is used for acceptance filtering.

www.DataSheet4U.com

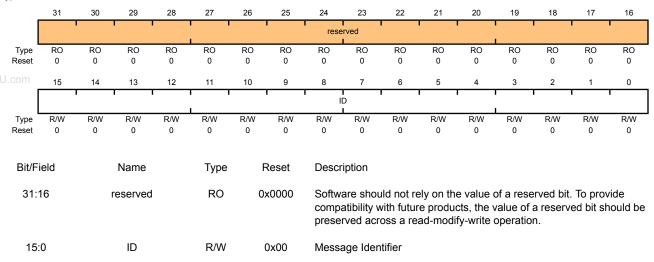
# Register 16: CAN IF1 Arbitration 1 (CANIF1ARB1), offset 0x030 Register 17: CAN IF2 Arbitration 1 (CANIF2ARB1), offset 0x090

These registers hold the identifiers for acceptance filtering.

CAN IF1 Arbitration 1 (CANIF1ARB1)

CAN0 base: 0x4004.0000 Offset 0x030

Type RO, reset 0x0000.0000



This bit field is used with the ID field in the **CANIFnARB2** register to create the message identifier. ID[28:0] is the Extended Frame and ID[28:18] is the Standard Frame.

# Register 18: CAN IF1 Arbitration 2 (CANIF1ARB2), offset 0x034 Register 19: CAN IF2 Arbitration 2 (CANIF2ARB2), offset 0x094

These registers hold information for acceptance filtering.

#### CAN IF1 Arbitration 2 (CANIF1ARB2)

CAN0 base: 0x4004.0000

Offset 0x034
Type RO, reset 0x0000.0000
31 30

							•	rese	erved						•	
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U.com	15	14	13	12	11 1 I	10	9	8	7 1	6 1 1	5	4	3	2	1 1	0
	MsgVal	Xtd	Dir		ı					ID						
Type Reset	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0
Bit/F	ield		Name		Type	F	Reset	Desc	ription							
21.	16		raaam (ad		DO	0	·0000	Coffu	ara ahau	ıld not ro	lu on th	م برماییم د	of a rasa	muad bit	To prov	ido
31:	10		reserved		RO	U.	x0000			uld not re vith futur	-					
									,	oss a rea	•					
15	5		MsgVal		R/W		0x0	Mess	age Valid	d						
								0: Th	e messa	ge objec	t is igno	red by th	ne messa	age han	dler.	
										ge objec dler withi		•		e consid	dered by	the
								initiali The № are m fields	zation and signal to the contract of the contr	essage of nd before pit must a or if the m ANIFnAI 2 register	e clearir also be nessage RBn reg	ig the Ir cleared to cobject in gisters, the	nit bit in pefore ar s no long ne Xtd a	the <b>CA</b> ny of the ger requ and Dir	NCTL re following followin	egister. ng bits e ID bit ne
14	4		Xtd		R/W		0x0	Exten	ded Ider	ntifier						
								0: Th	e 11-bit 9	Standard	Identifie	er will be	used fo	r this m	essage o	object.
								1: Th	e 29-bit l	Extended	l Identifi	er will be	e used fo	or this m	nessage	object.
1;	3		Dir		R/W		0x0	Mess	age Dire	ection						
								mess	age obje	n TxRqs ect is tran itifier, tha	smitted	. On rece	eption of	a Data	Frame v	with
								as a l	Data Fra	On TxRqs me. On r qst bit o	eceptio	n of a Re	emote Fr	ame wi	th match	ing
12	:0		ID		R/W		0x0	Mess	age Ider	ntifier						
									fier. ID[2	ID bit in t 8:0] is the			Ū			•

# Register 20: CAN IF1 Message Control (CANIF1MCTL), offset 0x038 Register 21: CAN IF2 Message Control (CANIF2MCTL), offset 0x098

This register holds the control information associated with the message object to be sent to the Message RAM.

#### CAN IF1 Message Control (CANIF1MCTL)

CAN0 base: 0x4004.0000

Offset 0x038

12

Type RO, reset 0x0000.0000

								rese	rvea							
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
4U Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB		reserved		·	DL	.c	
Type	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	R/W 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0
Reset	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
Bit/F	ield		Name		Туре	F	Reset	Descri	iption							
31.	31:16 reserved		RO	0	x0000	Softwa	are shou	ld not r	elv on the	e value d	of a rese	rved bit	To prov	ide		
10001100		compa	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.													
15 NewDat			R/W		0x0	New E	Data									
							by the n		een writte e handler					•		
										•	ller or the ge object.		as writtei	n new da	ata into t	he data
14	1		MsgLst		R/W		0x0	Messa	age Lost							
								0 : No CPU.	messag	je was l	ost since	the last	time thi	s bit was	s reset b	y the
									•	•	ller stored CPU has		Ū		is object	when
									,		or messager set to 0			h <b>e</b> Dir l	oit in the	
13	3		IntPnd		R/W		0x0	Interru	ıpt Pend	ing						
								0: This	s messa	ge obje	ct is not t	he sour	ce of an	interrup	t.	
								identif	ier in the	CANI	ct is the s nterrupt re is not a	(CANIN	IT) regis	ter will p	oint to th	nis

Use Acceptance Mask

1: Use mask (Msk, MXtd, and MDir) for acceptance filtering.

0: Mask ignored.

R/W

**UMask** 

0x0

Bit/Field	Name	Туре	Reset	Description
11	TxIE	R/W	0x0	Transmit Interrupt Enable
				0: The IntPnd bit in the <b>CANIFnMCTL</b> register is unchanged after a successful transmission of a frame.
				1: The IntPnd bit in the <b>CANIFnMCTL</b> register is set after a successful transmission of a frame.
10	RxIE	R/W	0x0	Receive Interrupt Enable
				0: The IntPnd bit in the <b>CANIFnMCTL</b> register is unchanged after a successful reception of a frame.
				1: The IntPnd bit in the <b>CANIFnMCTL</b> register is set after a successful reception of a frame.
9	RmtEn	R/W	0x0	Remote Enable
				0: At the reception of a Remote Frame, the TxRqst bit in the CANIFnMCTL register is left unchanged.
				1: At the reception of a Remote Frame, the TxRqst bit in the CANIFnMCTL register is set.
8	TxRqst	R/W	0x0	Transmit Request
				0: This message object is not waiting for transmission.
				1: The transmission of this message object is requested and is not yet done.
7	EoB	R/W	0x0	End of Buffer
				0: Message object belongs to a FIFO Buffer and is not the last message object of that FIFO Buffer.
				1: Single message object or last message object of a FIFO Buffer.
				This bit is used to concatenate two or more message objects (up to 32) to build a FIFO buffer. For a single message object (thus not belonging to a FIFO buffer), this bit must be set to 1.
6:4	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3:0	DLC	R/W	0x0	Data Length Code
				Value Description
				0x0-0x8 Specifies the number of bytes in the Data Frame.
				0x9-0xF Defaults to a Data Frame with 8 bytes.
				The DLC bit in the <b>CANIFNMCTL</b> register of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message bandler stores a data frame

382 November 29, 2007

identifier at other nodes. When the message handler stores a data frame,

it writes  ${\tt DLC}$  to the value given by the received message.

Register 22: CAN IF1 Data A1 (CANIF1DA1), offset 0x03C

Register 23: CAN IF1 Data A2 (CANIF1DA2), offset 0x040

Register 24: CAN IF1 Data B1 (CANIF1DB1), offset 0x044

Register 25: CAN IF1 Data B2 (CANIF1DB2), offset 0x048

Register 26: CAN IF2 Data A1 (CANIF2DA1), offset 0x09C

Register 27: CAN IF2 Data A2 (CANIF2DA2), offset 0x0A0

Register 28: CAN IF2 Data B1 (CANIF2DB1), offset 0x0A4

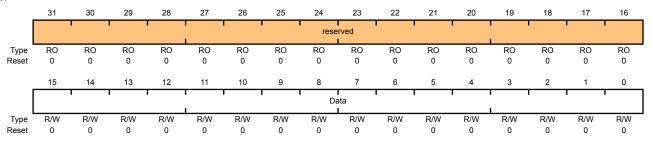
Register 29: CAN IF2 Data B2 (CANIF2DB2), offset 0x0A8

These registers contain the data to be sent or that has been received. In a CAN Data Frame, data byte 0 is the first byte to be transmitted or received and data byte 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte is transmitted first.

#### CAN IF1 Data A1 (CANIF1DA1)

CAN0 base: 0x4004.0000 Offset 0x03C

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	Data	R/W	0x00	Data

The **CANIFnDA1** registers contain data bytes 1 and 0; **CANIFnDA2** data bytes 3 and 2; **CANIFnDB1** data bytes 5 and 4; and **CANIFnDB2** data bytes 7 and 6.

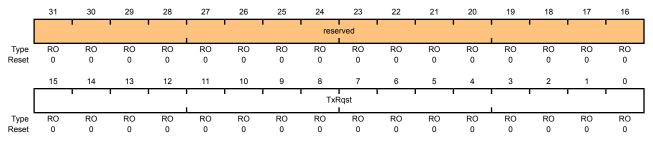
# Register 30: CAN Transmission Request 1 (CANTXRQ1), offset 0x100 Register 31: CAN Transmission Request 2 (CANTXRQ2), offset 0x104

The **CANTXRQ1** and **CANTXRQ2** registers hold the  $\mathtt{TxRqst}$  bits of the 32 message objects. By reading out these bits, the CPU can check which message object has a transmission request pending. The  $\mathtt{TxRqst}$  bit of a specific message object can be changed by three sources: (1) the CPU via the **CAN IFn Message Control (CANIFnMCTL)** register, (2) the message handler state machine after the reception of a Remote Frame, or (3) the message handler state machine after a successful transmission.

The **CANTXRQ1** register contains the TxRqst bit of the first 16 message objects in the message RAM; the **CANTXRQ2** register contains the TxRqst bit of the second 16 message objects.

#### ww.DataSheet4 CANnTransmission Request 1 (CANTXRQ1)

CAN0 base: 0x4004.0000 Offset 0x100 Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	TxRqst	RO	0x00	Transmission Request Bits

(of all message objects)

0: The message object is not waiting for transmission.

<sup>1:</sup> The transmission of the message object is requested and is not yet done.

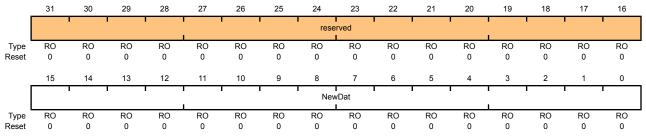
# Register 32: CAN New Data 1 (CANNWDA1), offset 0x120 Register 33: CAN New Data 2 (CANNWDA2), offset 0x124

The **CANNWDA1** and **CANNWDA2** registers hold the <code>NewDat</code> bits of the 32 message objects. By reading these bits, the CPU can check which message object has its data portion updated. The <code>NewDat</code> bit of a specific message object can be changed by three sources: (1) the CPU via the **CAN IFn Message Control (CANIFnMCTL)** register, (2) the message handler state machine after the reception of a Data Frame, or (3) the message handler state machine after a successful transmission.

The **CANNWDA1** register contains the NewDat bit of the first 16 message objects in the message RAM; the **CANNWDA2** register contains the NewDat bit of the second 16 message objects.

#### ww.DataSheet4CAN New Data 1 (CANNWDA1)

CAN0 base: 0x4004.0000 Offset 0x120 Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	NewDat	RO	0x00	New Data Bits

(of all message objects)

0: No new data has been written into the data portion of this message object by the message handler since the last time this flag was cleared by the CPU.

1: The message handler or the CPU has written new data into the data portion of this message object.

# Register 34: CAN Message 1 Interrupt Pending (CANMSG1INT), offset 0x140 Register 35: CAN Message 2 Interrupt Pending (CANMSG2INT), offset 0x144

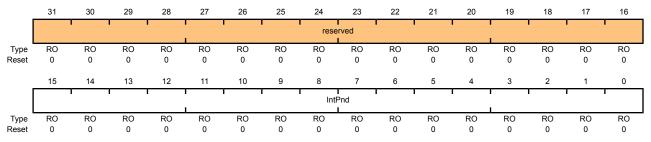
The **CANMSG1INT** and **CANMSG2INT** registers hold the IntPnd bits of the 32 message objects. By reading these bits, the CPU can check which message object has an interrupt pending. The IntPnd bit of a specific message object can be changed through two sources: (1) the CPU via the **CAN IFN Message Control (CANIFNMCTL)** register, or (2) the message handler state machine after the reception or transmission of a frame.

This field is also encoded in the CAN Interrupt (CANINT) register.

The **CANMSG1INT** register contains the IntPnd bit of the first 16 message objects in the message RAM; the **CANMSG2INT** register contains the IntPnd bit of the second 16 message objects.

CAN Message 1 Interrupt Pending (CANMSG1INT)

CAN0 base: 0x4004.0000 Offset 0x140 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	IntPnd	RO	0x00	Interrupt Pending Bits

(of all message objects)

0: This message object is not the source of an interrupt.

www.DataSheet4U.com

1: This message object is the source of an interrupt.

# Register 36: CAN Message 1 Valid (CANMSG1VAL), offset 0x160 Register 37: CAN Message 2 Valid (CANMSG2VAL), offset 0x164

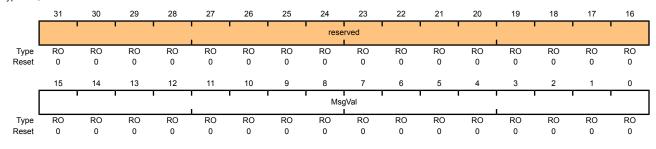
The **CANMSG1VAL** and **CANMSG2VAL** registers hold the MsgVal bits of the 32 message objects. By reading these bits, the CPU can check which message object is valid. The message value of a specific message object can be changed with the **CAN IFn Message Control (CANIFnMCTL)** register.

The **CANMSG1VAL** register contains the MsgVal bit of the first 16 message objects in the message RAM; the **CANMSG2VAL** register contains the MsgVal bit of the second 16 message objects in the message RAM.

#### CAN Message 1 Valid (CANMSG1VAL)

DataSheet4 CAN0 base: 0x4004.0000

Offset 0x160 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	MsgVal	RO	0x00	Message Valid Bits

(of all message objects)

- $\ensuremath{\mathsf{0}}$  : This message object is not configured and is ignored by the message handler.
- 1: This message object is configured and should be considered by the message handler.

# 15 Analog Comparators

An analog comparator is a peripheral that compares two analog voltages, and provides a logical output that signals the comparison result.

The LM3S2110 controller provides three independent integrated analog comparators that can be configured to drive an output or generate an interrupt.

Note: Not all comparators have the option to drive an output pin. See the Comparator Operating Mode tables for more information.

A comparator can compare a test voltage against any one of these voltages:

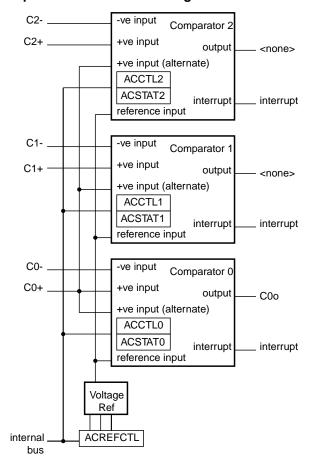
- An individual external reference voltage
- A shared single external reference voltage
- A shared internal reference voltage

The comparator can provide its output to a device pin, acting as a replacement for an analog comparator on the board, or it can be used to signal the application via interrupts to cause it to start capturing a sample sequence.

ww.DataSheet4U.com

# 15.1 Block Diagram

Figure 15-1. Analog Comparator Module Block Diagram



# 15.2 Functional Description

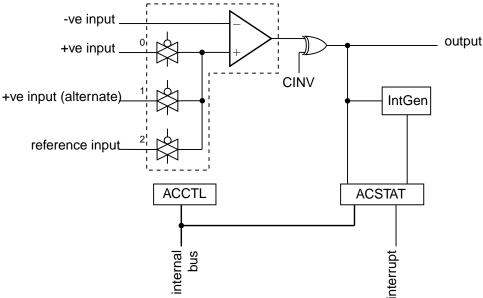
Important: It is recommended that the Digital-Input enable (the GPIODEN bit in the GPIO module) for the analog input pin be disabled to prevent excessive current draw from the I/O pads.

The comparator compares the VIN- and VIN+ inputs to produce an output, VOUT.

```
VIN- < VIN+, VOUT = 1
VIN- > VIN+, VOUT = 0
```

As shown in Figure 15-2 on page 390, the input source for VIN- is an external input. In addition to an external input, input sources for VIN+ can be the +ve input of comparator 0 or an internal reference.

Figure 15-2. Structure of Comparator Unit



A comparator is configured through two status/control registers (ACCTL and ACSTAT). The internal reference is configured through one control register (ACREFCTL). Interrupt status and control is configured through three registers (ACMIS, ACRIS, and ACINTEN). The operating modes of the comparators are shown in the Comparator Operating Mode tables.

Typically, the comparator output is used internally to generate controller interrupts. It may also be used to drive an external pin.

Important: Certain register bit values must be set before using the analog comparators. The proper pad configuration for the comparator input and output pins are described in the Comparator Operating Mode tables.

Table 15-1. Comparator 0 Operating Modes

ACCNTL0	Comparator 0				
ASRCP	VIN-	VIN+	Output	Interrupt	
00	C0-	C0+	C0o/C1+	yes	
01	C0-	C0+	C0o/C1+	yes	
10	C0-	Vref	C0o/C1+	yes	
11	C0-	reserved	C0o/C1+	yes	

Table 15-2. Comparator 1 Operating Modes

ACCNTL1	Comparator 1				
ASRCP	VIN-	VIN+	Output	Interrupt	
00	C1-	C0o/C1+ <sup>a</sup>	n/a	yes	
01	C1-	C0+	n/a	yes	
10	C1-	Vref	n/a	yes	
11	C1-	reserved	n/a	yes	

a. C0o and C1+ signals share a single  $\operatorname{pin}$  and  $\operatorname{may}$  only be used as one or the other.

**Table 15-3. Comparator 2 Operating Modes** 

ACCNTL2	Comparator 2				
ASRCP	VIN-	VIN+	Output	Interrupt	
00	C2-	C2+	n/a	yes	
01	C2-	C0+	n/a	yes	
10	C2-	Vref	n/a	yes	
11	C2-	reserved	n/a	yes	

### 15.2.1 Internal Reference Programming

The structure of the internal reference is shown in Figure 15-3 on page 391. This is controlled by a single configuration register (**ACREFCTL**). Table 15-4 on page 391 shows the programming options to develop specific internal reference values, to compare an external voltage against a particular voltage generated internally.

Figure 15-3. Comparator Internal Reference Structure

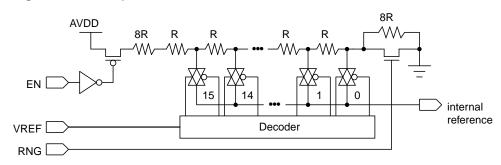


Table 15-4. Internal Reference Voltage and ACREFCTL Field Values

ACREFCTL Register		Output Reference Voltage Based on VREF Field Value
EN Bit Value	RNG Bit Value	
EN=0		0 V (GND) for any value of VREF; however, it is recommended that RNG=1 and VREF=0 for the least noisy ground reference.

November 29, 2007 391

ACREFCTL Register		Output Reference Voltage Based on VREF Field Value			
EN Bit Value	RNG Bit Value				
EN=1	RNG=0	Total resistance in ladder is 32 R.			
		$V_{REF} = AV_{DD} \times \frac{R_{VREF}}{R_{T}}$			
		$V_{REF} = AV_{DD} \times \frac{(VREF + 8)}{32}$			
		$V_{REF} = 0.825 + 0.103 \text{ VREF}$			
		The range of internal reference in this mode is 0.825-2.37 V.			
	RNG=1	Total resistance in ladder is 24 R.			
		$V_{REF} = AV_{DD} \times \frac{R_{VREF}}{R_{T}}$			
		$V_{REF} = AV_{DD} \times \frac{(VREF)}{24}$			
		$V_{REF} = 0.1375 \times V_{REF}$			
		The range of internal reference for this mode is 0.0-2.0625 V.			

## 15.3 Initialization and Configuration

The following example shows how to configure an analog comparator to read back its output value from an internal register.

- 1. Enable the analog comparator 0 clock by writing a value of 0x0010.0000 to the **RCGC1** register in the System Control module.
- 2. In the GPIO module, enable the GPIO port/pin associated with co- as a GPIO input.
- Configure the internal voltage reference to 1.65 V by writing the ACREFCTL register with the value 0x0000.030C.
- 4. Configure comparator 0 to use the internal voltage reference and to *not* invert the output on the C00 pin by writing the **ACCTL0** register with the value of 0x0000.040C.
- Delay for some time.
- 6. Read the comparator output value by reading the **ACSTAT0** register's OVAL value.

Change the level of the signal input on  ${\tt CO-}$  to see the  ${\tt OVAL}$  value change.

## 15.4 Register Map

Table 15-5 on page 393 lists the comparator registers. The offset listed is a hexadecimal increment to the register's address, relative to the Analog Comparator base address of 0x4003.C000.

**Table 15-5. Analog Comparators Register Map** 

Offset	Name	Туре	Reset	Description	See page
0x00	ACMIS	R/W1C	0x0000.0000	Analog Comparator Masked Interrupt Status	394
0x04	ACRIS	RO	0x0000.0000	Analog Comparator Raw Interrupt Status	395
0x08	ACINTEN	R/W	0x0000.0000	Analog Comparator Interrupt Enable	396
0x10	ACREFCTL	R/W	0x0000.0000	Analog Comparator Reference Voltage Control	397
0x20	ACSTAT0	RO	0x0000.0000	Analog Comparator Status 0	398
0x24	ACCTL0	R/W	0x0000.0000	Analog Comparator Control 0	399
0x40	ACSTAT1	RO	0x0000.0000	Analog Comparator Status 1	398
0x44	ACCTL1	R/W	0x0000.0000	Analog Comparator Control 1	399
0x60	ACSTAT2	RO	0x0000.0000	Analog Comparator Status 2	398
0x64	ACCTL2	R/W	0x0000.0000	Analog Comparator Control 2	399

#### 15.5 **Register Descriptions**

The remainder of this section lists and describes the Analog Comparator registers, in numerical order by address offset.

November 29, 2007 393

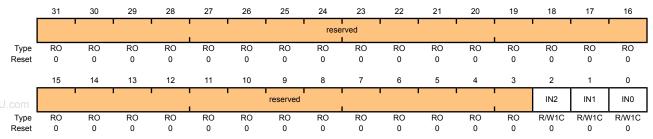
## Register 1: Analog Comparator Masked Interrupt Status (ACMIS), offset 0x00

This register provides a summary of the interrupt status (masked) of the comparator.

Analog Comparator Masked Interrupt Status (ACMIS)

Base 0x4003.C000

Offset 0x00 Type R/W1C, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:3	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	IN2	R/W1C	0	Comparator 2 Masked Interrupt Status
				Gives the masked interrupt state of this interrupt. Write 1 to this bit to clear the pending interrupt.
1	IN1	R/W1C	0	Comparator 1 Masked Interrupt Status
				Gives the masked interrupt state of this interrupt. Write 1 to this bit to clear the pending interrupt.
0	IN0	R/W1C	0	Comparator 0 Masked Interrupt Status
				Gives the masked interrupt state of this interrupt. Write 1 to this bit to clear the pending interrupt.

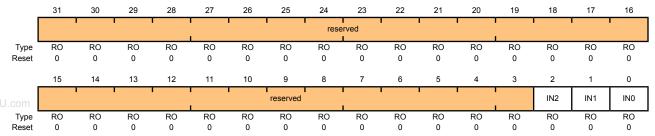
## Register 2: Analog Comparator Raw Interrupt Status (ACRIS), offset 0x04

This register provides a summary of the interrupt status (raw) of the comparator.

Analog Comparator Raw Interrupt Status (ACRIS)

Base 0x4003.C000 Offset 0x04

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:3	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	IN2	RO	0	Comparator 2 Interrupt Status
				When set, indicates that an interrupt has been generated by comparator 2.
1	IN1	RO	0	Comparator 1 Interrupt Status
				When set, indicates that an interrupt has been generated by comparator 1.
0	IN0	RO	0	Comparator 0 Interrupt Status
				When set, indicates that an interrupt has been generated by comparator 0.

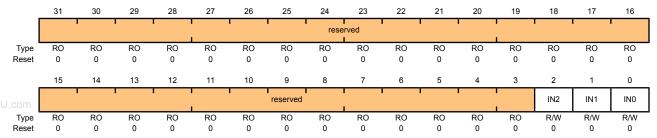
## Register 3: Analog Comparator Interrupt Enable (ACINTEN), offset 0x08

This register provides the interrupt enable for the comparator.

Analog Comparator Interrupt Enable (ACINTEN)

Base 0x4003.C000

Offset 0x08
Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:3	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
2	IN2	R/W	0	Comparator 2 Interrupt Enable  When set, enables the controller interrupt from the comparator 2 output
1	IN1	R/W	0	Comparator 1 Interrupt Enable  When set, enables the controller interrupt from the comparator 1 output.
0	IN0	R/W	0	Comparator 0 Interrupt Enable  When set, enables the controller interrupt from the comparator 0 output.

18

RO

0

RO

16

RO

### Register 4: Analog Comparator Reference Voltage Control (ACREFCTL), offset 0x10

23

RO

0

reserved

RO

0

22

RO

RO

0

RO

RO

0

This register specifies whether the resistor ladder is powered on as well as the range and tap.

Analog Comparator Reference Voltage Control (ACREFCTL)

28

RO

0

27

RO

0

26

RO

RO

0

Base 0x4003.C000

Туре

Reset

Offset 0x10
Type R/W, reset 0x0000.0000

RΩ

0

30

RΩ

0

RO

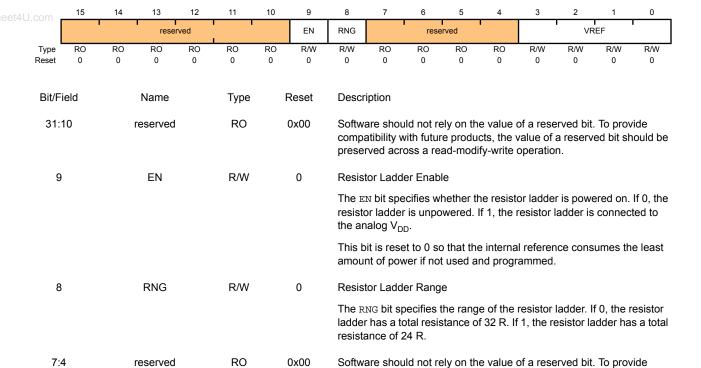
0

**VREF** 

3:0

R/W

0x00



The  $\mathtt{VREF}$  bit field specifies the resistor ladder tap that is passed through an analog multiplexer. The voltage corresponding to the tap position is the internal reference voltage available for comparison. See Table 15-4 on page 391 for some output reference voltage examples.

compatibility with future products, the value of a reserved bit should be

preserved across a read-modify-write operation.

Resistor Ladder Voltage Ref

Register 5: Analog Comparator Status 0 (ACSTAT0), offset 0x20

Register 6: Analog Comparator Status 1 (ACSTAT1), offset 0x40

Register 7: Analog Comparator Status 2 (ACSTAT2), offset 0x60

These registers specify the current output value of the comparator.

### Analog Comparator Status 0 (ACSTAT0)

Base 0x4003.C000

Offset 0x20
Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								rese	rved				1			'
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							rese	rved	 				! !		OVAL	reserved
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	OVAL	RO	0	Comparator Output Value
				The OVAL bit specifies the current output value of the comparator.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

www.DataSheet4U.com

# Register 8: Analog Comparator Control 0 (ACCTL0), offset 0x24 Register 9: Analog Comparator Control 1 (ACCTL1), offset 0x44 Register 10: Analog Comparator Control 2 (ACCTL2), offset 0x64

These registers configure the comparator's input and output.

Analog Comparator Control 0 (ACCTL0)

Base 0x4003.C000

Offset 0x24
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		'	'		·		1	rese	rved			'		1	'	
U.cType	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		'	reserved			AS	RCP		rese	erved		ISLVAL	IS	EN	CINV	reserved
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	RO 0	RO 0	RO 0	RO 0	R/W 0	R/W 0	R/W 0	R/W 0	RO 0
Nosci	Ü	O	U	Ü	U	Ü	Ü	O	Ü	O	Ü	Ü	Ü	O	O	Ü
Bit/F	ield		Name		Туре	ı	Reset	Descr	iption							
31:	11		reserved		RO		0x00	Softwa	are shoi	uld not re	dy on th	e value d	of a rese	arved hit	To prov	vide
01.			10001700		110		OXOO	compa	atibility v	with futur	e produ	cts, the v	alue of	a reserv		
								presei	rved acr	oss a rea	ad-modi	ify-write o	operatio	n.		
10:	:9		ASRCP		R/W		0x00	Analo	g Sourc	e Positiv	е					
										ld specifi			•	•		terminal
								of the	compar	ator. The	encodi	ings for t	his field	are as f	ollows:	
								Value	Functi	on						
								0x0	Pin va	lue						
								0x1	Pin va	lue of C0	)+					
								0x2	Interna	al voltage	e refere	nce				
								0x3	Reser	ved						
8:	5		reserved		RO		0	Softwa	are shoi	uld not re	ely on th	e value d	of a rese	arved hit	To prov	vide
0.	5		reserveu		NO		Ü	compa	atibility v	with futur	e produ	cts, the v	alue of	a reserv	•	
								presei	rved acr	oss a rea	ad-modi	ity-write o	operatio	n.		
4			ISLVAL		R/W		0	Interru	ıpt Sens	se Level	Value					
										bit specif						
										in Level : utput is L						

comparator output is High.

Bit/Field	Name	Туре	Reset	Description
3:2	ISEN	R/W	0x0	Interrupt Sense
				The ISEN field specifies the sense of the comparator output that generates an interrupt. The sense conditioning is as follows:
				Value Function
				0x0 Level sense, see ISLVAL
				0x1 Falling edge
				0x2 Rising edge
				0x3 Either edge
taSheet4U.com 1	CINV	R/W	0	Comparator Output Invert
				The CINV bit conditionally inverts the output of the comparator. If 0, the output of the comparator is unchanged. If 1, the output of the comparator is inverted prior to being processed by hardware.
0	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

# 16 Pulse Width Modulator (PWM)

Pulse width modulation (PWM) is a powerful technique for digitally encoding analog signal levels. High-resolution counters are used to generate a square wave, and the duty cycle of the square wave is modulated to encode an analog signal. Typical applications include switching power supplies and motor control.

The Stellaris<sup>®</sup> PWM module consists of one PWM generator block and a control block. The PWM generator block contains one timer (16-bit down or up/down counter), two PWM comparators, a PWM signal generator, a dead-band generator, and an interrupt selector. The control block determines the polarity of the PWM signals, and which signals are passed through to the pins.

The PWM generator block produces two PWM signals that can either be independent signals (other than being based on the same timer and therefore having the same frequency) or a single pair of complementary signals with dead-band delays inserted. The output of the PWM generation block is managed by the output control block before being passed to the device pins.

The Stellaris<sup>®</sup> PWM module provides a great deal of flexibility. It can generate simple PWM signals, such as those required by a simple charge pump. It can also generate paired PWM signals with dead-band delays, such as those required by a half-H bridge driver.

# 16.1 Block Diagram

Figure 16-1 on page 401 provides a block diagram of a Stellaris<sup>®</sup> PWM module. The LM3S2110 controller contains one generator block (PWM0) and generates two independent PWM signals or one paired PWM signal with dead-band delays inserted.

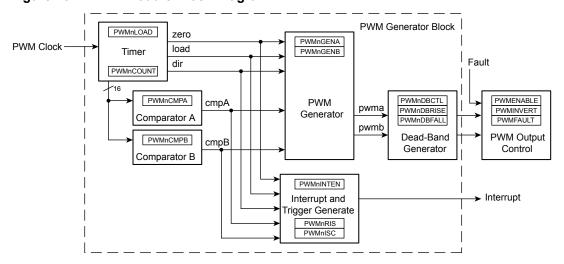


Figure 16-1. PWM Module Block Diagram

# 16.2 Functional Description

### **16.2.1 PWM Timer**

The timer runs in one of two modes: Count-Down mode or Count-Up/Down mode. In Count-Down mode, the timer counts from the load value to zero, goes back to the load value, and continues counting down. In Count-Up/Down mode, the timer counts from zero up to the load value, back down to zero, back up to the load value, and so on. Generally, Count-Down mode is used for

ww.DataSheet4U.cor

generating left- or right-aligned PWM signals, while the Count-Up/Down mode is used for generating center-aligned PWM signals.

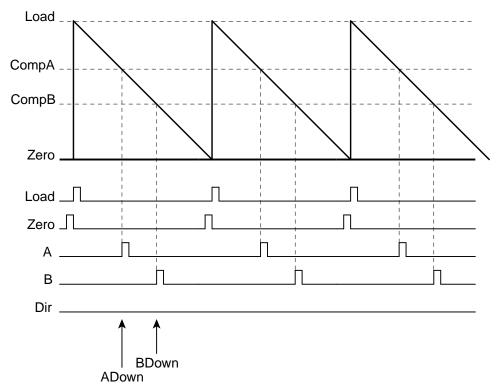
The timers output three signals that are used in the PWM generation process: the direction signal (this is always Low in Count-Down mode, but alternates between Low and High in Count-Up/Down mode), a single-clock-cycle-width High pulse when the counter is zero, and a single-clock-cycle-width High pulse when the counter is equal to the load value. Note that in Count-Down mode, the zero pulse is immediately followed by the load pulse.

#### 16.2.2 PWM Comparators

There are two comparators in the PWM generator that monitor the value of the counter; when either match the counter, they output a single-clock-cycle-width High pulse. When in Count-Up/Down mode, these comparators match both when counting up and when counting down; they are therefore qualified by the counter direction signal. These qualified pulses are used in the PWM generation process. If either comparator match value is greater than the counter load value, then that comparator never outputs a High pulse.

Figure 16-2 on page 402 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Down mode. Figure 16-3 on page 403 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Up/Down mode.





ww.DataSheet4U.com

www.DataSheet4U.com

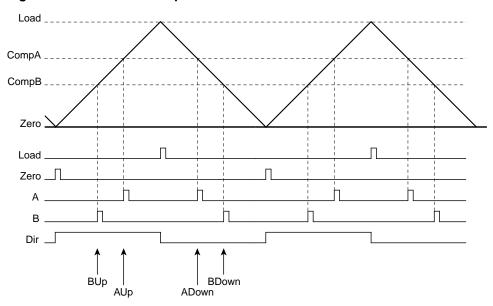


Figure 16-3. PWM Count-Up/Down Mode

/w.DataSheet4U.com

### 16.2.3 PWM Signal Generator

The PWM generator takes these pulses (qualified by the direction signal), and generates two PWM signals. In Count-Down mode, there are four events that can affect the PWM signal: zero, load, match A down, and match B down. In Count-Up/Down mode, there are six events that can affect the PWM signal: zero, load, match A down, match A up, match B down, and match B up. The match A or match B events are ignored when they coincide with the zero or load events. If the match A and match B events coincide, the first signal, PWMA, is generated based only on the match A event, and the second signal, PWMB, is generated based only on the match B event.

For each event, the effect on each output PWM signal is programmable: it can be left alone (ignoring the event), it can be toggled, it can be driven Low, or it can be driven High. These actions can be used to generate a pair of PWM signals of various positions and duty cycles, which do or do not overlap. Figure 16-4 on page 403 shows the use of Count-Up/Down mode to generate a pair of center-aligned, overlapped PWM signals that have different duty cycles.

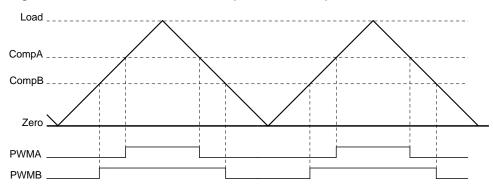


Figure 16-4. PWM Generation Example In Count-Up/Down Mode

In this example, the first generator is set to drive High on match A up, drive Low on match A down, and ignore the other four events. The second generator is set to drive High on match B up, drive Low on match B down, and ignore the other four events. Changing the value of comparator A

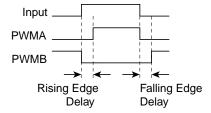
changes the duty cycle of the PWMA signal, and changing the value of comparator B changes the duty cycle of the PWMB signal.

#### 16.2.4 Dead-Band Generator

The two PWM signals produced by the PWM generator are passed to the dead-band generator. If disabled, the PWM signals simply pass through unmodified. If enabled, the second PWM signal is lost and two PWM signals are generated based on the first PWM signal. The first output PWM signal is the input signal with the rising edge delayed by a programmable amount. The second output PWM signal is the inversion of the input signal with a programmable delay added between the falling edge of the input signal and the rising edge of this new signal.

This is therefore a pair of active High signals where one is always High, except for a programmable amount of time at transitions where both are Low. These signals are therefore suitable for driving a half-H bridge, with the dead-band delays preventing shoot-through current from damaging the power electronics. Figure 16-5 on page 404 shows the effect of the dead-band generator on an input PWM signal.

Figure 16-5. PWM Dead-Band Generator



## 16.2.5 Interrupt Selector

The PWM generator also takes the same four (or six) counter events and uses them to generate an interrupt. Any of these events or a set of these events can be selected as a source for an interrupt; when any of the selected events occur, an interrupt is generated. The selection of events allows the interrupt to occur at a specific position within the PWM signal. Note that interrupts are based on the raw events; delays in the PWM signal edges caused by the dead-band generator are not taken into account.

### 16.2.6 Synchronization Methods

There is a global reset capability that can reset the counter of the PWM generator.

The counter load values and comparator match values of the PWM generator can be updated in two ways. The first is immediate update mode, where a new value is used as soon as the counter reaches zero. By waiting for the counter to reach zero, a guaranteed behavior is defined, and overly short or overly long output PWM pulses are prevented.

The other update method is synchronous, where the new value is not used until a global synchronized update signal is asserted, at which point the new value is used as soon as the counter reaches zero. This second mode allows multiple items to be updated simultaneously without odd effects during the update; everything runs from the old values until a point at which they all run from the new values.

#### 16.2.7 Fault Conditions

There are two external conditions that affect the PWM block; the signal input on the Fault pin and the stalling of the controller by a debugger. There are two mechanisms available to handle such

ww.DataSheet4U.com

conditions: the output signals can be forced into an inactive state and/or the PWM timers can be stopped.

Each output signal has a fault bit. If set, a fault input signal causes the corresponding output signal to go into the inactive state. If the inactive state is a safe condition for the signal to be in for an extended period of time, this keeps the output signal from driving the outside world in a dangerous manner during the fault condition. A fault condition can also generate a controller interrupt.

Each PWM generator can also be configured to stop counting during a stall condition. The user can select for the counters to run until they reach zero then stop, or to continue counting and reloading. A stall condition does not generate a controller interrupt.

### 16.2.8 Output Control Block

With the PWM generator block producing two raw PWM signals, the output control block takes care of the final conditioning of the PWM signals before they go to the pins. Via a single register, the set of PWM signals that are actually enabled to the pins can be modified; this can be used, for example, to perform commutation of a brushless DC motor with a single register write (and without modifying the individual PWM generators, which are modified by the feedback control loop). Similarly, fault control can disable any of the PWM signals as well. A final inversion can be applied to any of the PWM signals, making them active Low instead of the default active High.

# 16.3 Initialization and Configuration

The following example shows how to initialize the PWM Generator 0 with a 25-KHz frequency, and with a 25% duty cycle on the PWM0 pin and a 75% duty cycle on the PWM1 pin. This example assumes the system clock is 20 MHz.

- 1. Enable the PWM clock by writing a value of 0x0010.0000 to the **RCGC0** register in the System Control module.
- Enable the clock to the appropriate GPIO module via the RCGC2 register in the System Control module.
- In the GPIO module, enable the appropriate pins for their alternate function using the GPIOAFSEL register.
- 4. Configure the Run-Mode Clock Configuration (RCC) register in the System Control module to use the PWM divide (USEPWMDIV) and set the divider (PWMDIV) to divide by 2 (000).
- Configure the PWM generator for countdown mode with immediate updates to the parameters.
  - Write the PWM0CTL register with a value of 0x0000.0000.
  - Write the **PWM0GENA** register with a value of 0x0000.008C.
  - Write the PWM0GENB register with a value of 0x0000.080C.
- 6. Set the period. For a 25-KHz frequency, the period = 1/25,000, or 40 microseconds. The PWM clock source is 10 MHz; the system clock divided by 2. This translates to 400 clock ticks per period. Use this value to set the PWM0LOAD register. In Count-Down mode, set the Load field in the PWM0LOAD register to the requested period minus one.
  - Write the PWM0LOAD register with a value of 0x0000.018F.
- 7. Set the pulse width of the PWM0 pin for a 25% duty cycle.

- Write the **PWM0CMPA** register with a value of 0x0000.012B.
- 8. Set the pulse width of the PWM1 pin for a 75% duty cycle.
  - Write the **PWM0CMPB** register with a value of 0x0000.0063.
- 9. Start the timers in PWM generator 0.
  - Write the **PWM0CTL** register with a value of 0x0000.0001.
- 10. Enable PWM outputs.
  - Write the **PWMENABLE** register with a value of 0x0000.0003.

# ww.DataSheet416.4 Register Map

Table 16-1 on page 406 lists the PWM registers. The offset listed is a hexadecimal increment to the register's address, relative to the PWM base address of 0x4002.8000.

Table 16-1. PWM Register Map

Offset	Name	Туре	Reset	Description	See page
0x000	PWMCTL	R/W	0x0000.0000	PWM Master Control	408
0x004	PWMSYNC	R/W	0x0000.0000	PWM Time Base Sync	409
0x008	PWMENABLE	R/W	0x0000.0000	PWM Output Enable	410
0x00C	PWMINVERT	R/W	0x0000.0000	PWM Output Inversion	411
0x010	PWMFAULT	R/W	0x0000.0000	PWM Output Fault	412
0x014	PWMINTEN	R/W	0x0000.0000	PWM Interrupt Enable	413
0x018	PWMRIS	RO	0x0000.0000	PWM Raw Interrupt Status	414
0x01C	PWMISC	R/W1C	0x0000.0000	PWM Interrupt Status and Clear	415
0x020	PWMSTATUS	RO	0x0000.0000	PWM Status	416
0x040	PWM0CTL	R/W	0x0000.0000	PWM0 Control	417
0x044	PWM0INTEN	R/W	0x0000.0000	PWM0 Interrupt Enable	419
0x048	PWM0RIS	RO	0x0000.0000	PWM0 Raw Interrupt Status	421
0x04C	PWM0ISC	R/W1C	0x0000.0000	PWM0 Interrupt Status and Clear	422
0x050	PWM0LOAD	R/W	0x0000.0000	PWM0 Load	423
0x054	PWM0COUNT	RO	0x0000.0000	PWM0 Counter	424
0x058	PWM0CMPA	R/W	0x0000.0000	PWM0 Compare A	425
0x05C	PWM0CMPB	R/W	0x0000.0000	PWM0 Compare B	426
0x060	PWM0GENA	R/W	0x0000.0000	PWM0 Generator A Control	427
0x064	PWM0GENB	R/W	0x0000.0000	PWM0 Generator B Control	430
0x068	PWM0DBCTL	R/W	0x0000.0000	PWM0 Dead-Band Control	433

www.DataSheet4U.com

Offset	Name	Туре	Reset	Description	See page
0x06C	PWM0DBRISE	R/W	0x0000.0000	PWM0 Dead-Band Rising-Edge Delay	434
0x070	PWM0DBFALL	R/W	0x0000.0000	PWM0 Dead-Band Falling-Edge-Delay	435

# 16.5 Register Descriptions

The remainder of this section lists and describes the PWM registers, in numerical order by address offset.

ww.DataSheet4U.com

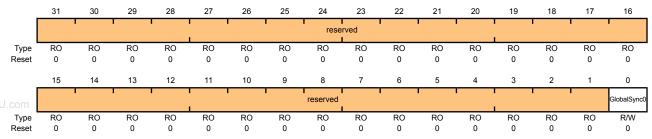
# Register 1: PWM Master Control (PWMCTL), offset 0x000

This register provides master control over the PWM generation block.

PWM Master Control (PWMCTL)

Base 0x4002.8000 Offset 0x000

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	GlobalSync0	R/W	0	Update PWM Generator 0

Setting this bit causes any queued update to a load or comparator register in PWM generator 0 to be applied the next time the corresponding counter becomes zero. This bit automatically clears when the updates have completed; it cannot be cleared by software.

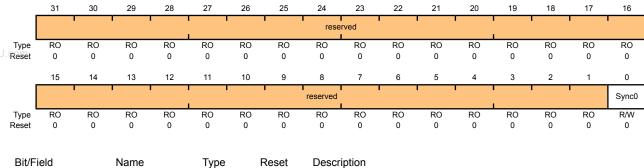
# Register 2: PWM Time Base Sync (PWMSYNC), offset 0x004

This register provides a method to perform synchronization of the counters in the PWM generation blocks. Writing a bit in this register to 1 causes the specified counter to reset back to 0; writing multiple bits resets multiple counters simultaneously. The bits auto-clear after the reset has occurred; reading them back as zero indicates that the synchronization has completed.

PWM Time Base Sync (PWMSYNC)

Base 0x4002.8000

Offset 0x004 Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	Sync0	R/W	0	Reset Generator 0 Counter

Performs a reset of the PWM generator 0 counter.

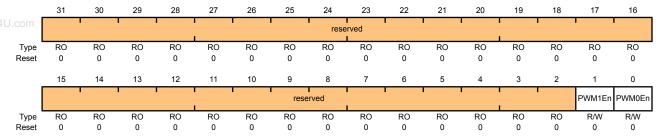
## Register 3: PWM Output Enable (PWMENABLE), offset 0x008

This register provides a master control of which generated PWM signals are output to device pins. By disabling a PWM output, the generation process can continue (for example, when the time bases are synchronized) without driving PWM signals to the pins. When bits in this register are set, the corresponding PWM signal is passed through to the output stage, which is controlled by the **PWMINVERT** register. When bits are not set, the PWM signal is replaced by a zero value which is also passed to the output stage.

#### PWM Output Enable (PWMENABLE)

Base 0x4002.8000 Offset 0x008

Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PWM1En	R/W	0	PWM1 Output Enable
				When set, allows the generated PWM1 signal to be passed to the device pin.
0	PWM0En	R/W	0	PWM0 Output Enable

When set, allows the generated PWM0 signal to be passed to the device pin.

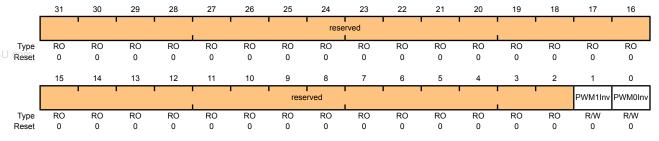
# Register 4: PWM Output Inversion (PWMINVERT), offset 0x00C

This register provides a master control of the polarity of the PWM signals on the device pins. The PWM signals generated by the PWM generator are active High; they can optionally be made active Low via this register. Disabled PWM channels are also passed through the output inverter (if so configured) so that inactive channels maintain the correct polarity.

#### PWM Output Inversion (PWMINVERT)

Base 0x4002.8000

Offset 0x00C Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	PWM1Inv	R/W	0	Invert PWM1 Signal
				When set, the generated PWM1 signal is inverted.
0	PWM0Inv	R/W	0	Invert PWM0 Signal

When set, the generated PWM0 signal is inverted.

# Register 5: PWM Output Fault (PWMFAULT), offset 0x010

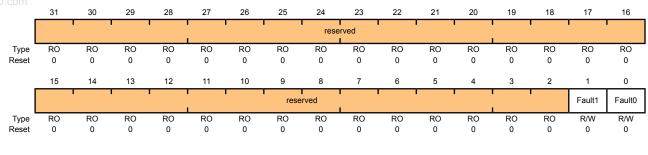
This register controls the behavior of the PWM outputs in the presence of fault conditions. Both the fault input and debug events are considered fault conditions. On a fault condition, each PWM signal can either be passed through unmodified or driven Low. For outputs that are configured for pass-through, the debug event handling on the corresponding PWM generator also determines if the PWM signal continues to be generated.

Fault condition control happens before the output inverter, so PWM signals driven Low on fault are inverted if the channel is configured for inversion (therefore, the pin is driven High on a fault condition).

#### PWM Output Fault (PWMFAULT)

Base 0x4002.8000

Offset 0x010 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	Fault1	R/W	0	PWM1 Driven Low on Fault
				When set, the PWM1 output signal is driven Low on a fault condition.
0	Fault0	R/W	0	PWM0 Driven Low on Fault

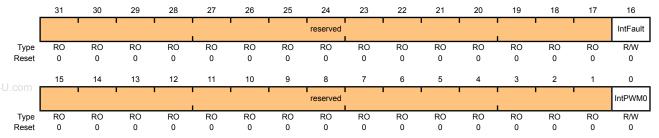
When set, the PWM0 output signal is driven Low on a fault condition.

## Register 6: PWM Interrupt Enable (PWMINTEN), offset 0x014

This register controls the global interrupt generation capabilities of the PWM module. The events that can cause an interrupt are the fault input and the individual interrupts from the PWM generator.

### PWM Interrupt Enable (PWMINTEN)

Base 0x4002.8000 Offset 0x014 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:17	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	IntFault	R/W	0	Fault Interrupt Enable  When 1, an interrupt occurs when the fault input is asserted.
15:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IntPWM0	R/W	0	PWM0 Interrupt Enable

When 1, an interrupt occurs when the PWM generator 0 block asserts an interrupt.

# Register 7: PWM Raw Interrupt Status (PWMRIS), offset 0x018

This register provides the current set of interrupt sources that are asserted, regardless of whether they cause an interrupt to be asserted to the controller. The fault interrupt is latched on detection; it must be cleared through the PWM Interrupt Status and Clear (PWMISC) register (see page 415). The PWM generator interrupts simply reflect the status of the PWM generator; they are cleared via the interrupt status register in the PWM generator block. Bits set to 1 indicate the events that are active; a zero bit indicates that the event in question is not active.

#### PWM Raw Interrupt Status (PWMRIS)

Base 0x4002.8000 Offset 0x018

Type RO,	reset 0	x0000.000	00													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
4U.com		'					'	reserved		_					'	IntFault
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	'				'	reserved		'					1	IntPWM0
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0
Bit/F	ield		Name		Туре		Reset	Descri	ption							
31:	17	1	reserved		RO		0x00	compa	tibility v	uld not re vith futur oss a rea	e produ	cts, the v	value of	a reserv		vide nould be
16	3		IntFault		RO		0		·	t Asserte		as been	asserted	I.		
15:	:1	1	reserved		RO		0x00	compa	atibility v	uld not re vith futur oss a rea	e produc	cts, the v	value of	a reserv		vide nould be
0		I	ntPWM0	)	RO		0	PWMC	) Interru	pt Asser	ted					

Indicates that the PWM generator 0 block is asserting its interrupt.

www.DataSheet4U.com

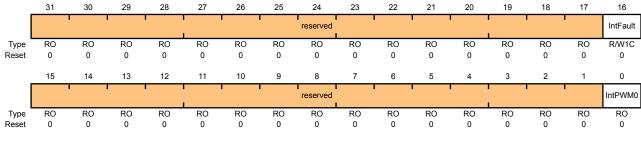
# Register 8: PWM Interrupt Status and Clear (PWMISC), offset 0x01C

This register provides a summary of the interrupt status of the PWM generator block. A bit set to 1 indicates that the generator block is asserting an interrupt. The individual interrupt status registers must be consulted to determine the reason for the interrupt, and used to clear the interrupt. For the fault interrupt, a write of 1 to that bit position clears the latched interrupt status.

PWM Interrupt Status and Clear (PWMISC)

Base 0x4002.8000 Offset 0x01C

Offset 0x01C Type R/W1C, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:17	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
16	IntFault	R/W1C	0	Fault Interrupt Asserted Indicates if the fault input is asserting an interrupt.
15:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	IntPWM0	RO	0	PWM0 Interrupt Status

Indicates if the PWM generator 0 block is asserting an interrupt.

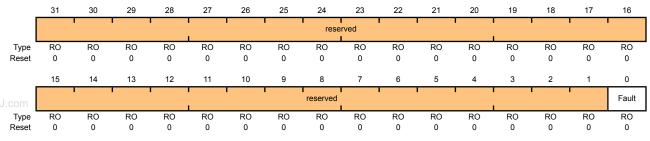
# Register 9: PWM Status (PWMSTATUS), offset 0x020

This register provides the status of the Fault input signal.

#### PWM Status (PWMSTATUS)

Base 0x4002.8000 Offset 0x020

Type RO, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	Fault	RO	0	Fault Interrupt Status

When set to 1, indicates the fault input is asserted.

www.DataSheet4U.com

### Register 10: PWM0 Control (PWM0CTL), offset 0x040

This register configures the PWM signal generation block. The Register Update mode, Debug mode, Counting mode, and Block Enable mode are all controlled via this register. The block produces the PWM signals, which can be either two independent PWM signals (from the same counter), or a paired set of PWM signals with dead-band delays added.

These registers configure the PWM signal generation blocks (PWM0CTL controls the PWM generator 0 block, and so on). The Register Update mode, Debug mode, Counting mode, and Block Enable mode are all controlled via these registers. The blocks produce the PWM signals, which can be either two independent PWM signals (from the same counter), or a paired set of PWM signals with dead-band delays added.

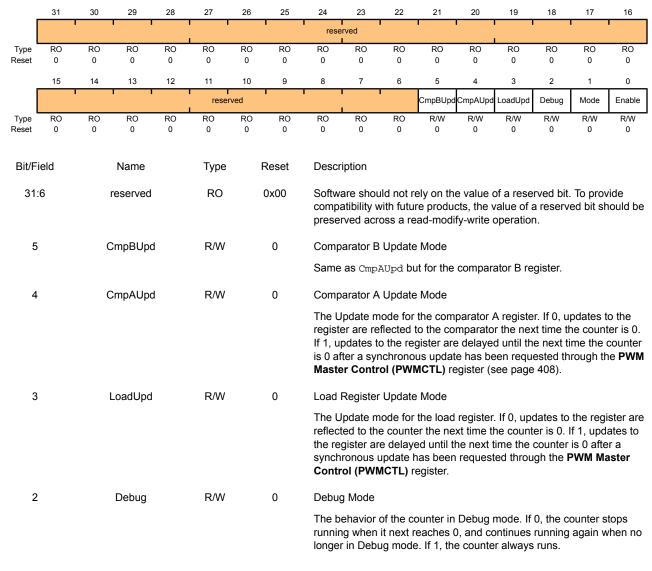
The PWM0 block produces the PWM0 and PWM1 outputs.

ww.DataSheet4U.com

#### PWM0 Control (PWM0CTL)

Base 0x4002.8000 Offset 0x040

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
1	Mode	R/W	0	Counter Mode
				The mode for the counter. If 0, the counter counts down from the load value to 0 and then wraps back to the load value (Count-Down mode). If 1, the counter counts up from 0 to the load value, back down to 0, and then repeats (Count-Up/Down mode).
0	Enable	R/W	0	PWM Block Enable
				Master enable for the PWM generation block. If 0, the entire block is disabled and not clocked. If 1, the block is enabled and produces PWM signals.

ww.DataSheet4U.con

## Register 11: PWM0 Interrupt Enable (PWM0INTEN), offset 0x044

This register controls the interrupt generation capabilities of the PWM generator. The events that can cause an interrupt are:

22

21

20

19

18

17

16

- The counter being equal to the load register
- The counter being equal to zero

28

27

- The counter being equal to the comparator A register while counting up
- The counter being equal to the comparator A register while counting down
- The counter being equal to the comparator B register while counting up

25

The counter being equal to the comparator B register while counting down

Any combination of these events can generate either an interrupt.

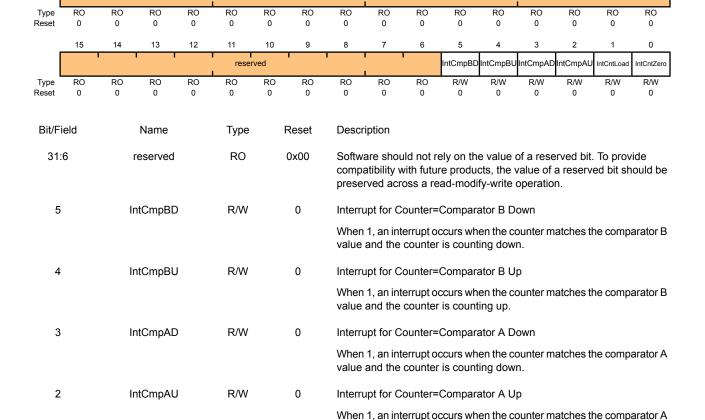
26

#### PWM0 Interrupt Enable (PWM0INTEN)

30

Base 0x4002.8000 Offset 0x044

Type R/W, reset 0x0000.0000



reserved

value and the counter is counting up.

Bit/Field	Name	Туре	Reset	Description
1	IntCntLoad	R/W	0	Interrupt for Counter=Load
				When 1, an interrupt occurs when the counter matches the <b>PWMnLOAD</b> register.
0	IntCntZero	R/W	0	Interrupt for Counter=0
				When 1, an interrupt occurs when the counter is 0.

ww.DataSheet4U.com

# Register 12: PWM0 Raw Interrupt Status (PWM0RIS), offset 0x048

This register provides the current set of interrupt sources that are asserted, regardless of whether they cause an interrupt to be asserted to the controller. Bits set to 1 indicate the latched events that have occurred; a 0 bit indicates that the event in question has not occurred.

#### PWM0 Raw Interrupt Status (PWM0RIS)

Base 0x4002.8000 Offset 0x048

Type RO, reset 0x0000.0000

_	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								rese	rved I		<u> </u>	<u> </u>				
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J.com _	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		'	1		rese		1	1			·	·	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
Туре	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit/Fi	eld		Name		Туре		Reset	Descr	iption							
31:	6		reserved	l	RO		0x00	compa	atibility w	ith futu	ely on the re produce ad-modi	cts, the v	alue of	a reserve		
5		ı	ntCmpBI	)	RO		0	Comp	arator B	Down	Interrupt	Status				
									ites that i		nter has	matched	d the cor	mparator	B value	while
4		1	ntCmpBl	J	RO		0	Comp	arator B	Up Inte	errupt Sta	atus				
									ites that i	the cou	nter has	matched	d the cor	nparator	B value	while
3		1	ntCmpA[	)	RO		0	Comp	arator A	Down	Interrupt	Status				
									ites that i		nter has	matched	d the cor	nparator	A value	while
2		1	ntCmpAl	J	RO		0	Comp	arator A	Up Inte	errupt Sta	atus				
									ites that i	the cou	nter has	matched	d the cor	mparator	A value	while
1		I	ntCntLoa	d	RO		0	Count	ter=Load	I Interru	pt Status	5				
								Indica	ites that	the cou	nter has	matched	d the <b>PW</b>	/MnLOA	<b>D</b> regist	ter.
0		I	ntCntZer	0	RO		0	Count	ter=0 Inte	errupt S	Status					
								Indica	ites that	the cou	nter has	matched	d 0.			

# Register 13: PWM0 Interrupt Status and Clear (PWM0ISC), offset 0x04C

This register provides the current set of interrupt sources that are asserted to the controller. Bits set to 1 indicate the latched events that have occurred; a 0 bit indicates that the event in question has not occurred. These are R/W1C registers; writing a 1 to a bit position clears the corresponding interrupt reason.

PWM0 Interrupt Status and Clear (PWM0ISC)

IntCntZero

R/W1C

0

Base 0x4002.8000

Offset 0x04C Type R/W1C, reset 0x0000.0000

Type RO	_	31	3	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset   0   0   0   0   0   0   0   0   0			1		<u> </u>				1	rese	rved							
15																		
Type Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset	0	(	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Type Roset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	_	15	1	4	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit/Field   Name   Type   Reset   Description			1		<u> </u>		resen	/ed	1	'			IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
Bit/Field Name Type Reset Description  31:6 reserved RO 0x00 Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.  5 IntCmpBD R/W1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.																		
31:6 reserved RO 0x00 Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.  5 IntCmpBD RW1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU RW1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD RW1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU RW1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad RW1C 0 Counter=Load Interrupt	Reset	0	(	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31:6 reserved RO 0x00 Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.  5 IntCmpBD RW1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU RW1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD RW1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU RW1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad RW1C 0 Counter=Load Interrupt																		
compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.  5 IntCmpBD R/W1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	Bit/F	ield			Name		Type		Reset	Descr	iption							
compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.  5 IntCmpBD R/W1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	31:	:6		r	eserved		RO		0x00	Softwa	are shou	ld not re	elv on the	e value o	of a rese	rved bit.	To prov	ride
5 IntCmpBD R/W1C 0 Comparator B Down Interrupt Indicates that the counter has matched the comparator B value while counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt																		
Indicates that the counter has matched the comparator B value while counting down.  R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  Indicates that the counter has matched the comparator B value while counting up.  Indicates that the counter has matched the comparator A value while counting down.  Indicates that the counter has matched the comparator A value while counting down.  Indicates that the counter has matched the comparator A value while counting up.  Indicates that the counter has matched the comparator A value while counting up.										prese	rved acro	oss a re	ad-modi	fy-write	operatio	n.		
counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	5	5 IntCmpBD		)	R/W1C		0	Comp	arator B	Down I	nterrupt							
counting down.  4 IntCmpBU R/W1C 0 Comparator B Up Interrupt Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt							Indica	tes that t	the cou	nter has	matched	the cor	mnaratoi	r R value	while			
Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt													inter nas	materiet	i ti ic coi	прагато	D value	Willic
Indicates that the counter has matched the comparator B value while counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt				1	. 40 DI		D 4440		0	0	t D	11 14-						
counting up.  3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	4			ır	itCmpBC	J	R/W1C		0	Comp	arator B	Up Inte	errupt					
3 IntCmpAD R/W1C 0 Comparator A Down Interrupt Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt												the cou	nter has	matched	the cor	mparato	B value	while
Indicates that the counter has matched the comparator A value while counting down.  2 IntCmpAU RW1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad RW1C 0 Counter=Load Interrupt										counti	ng up.							
counting down.  2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	3			In	ntCmpAD	)	R/W1C		0	Comp	arator A	Down I	nterrupt					
2 IntCmpAU R/W1C 0 Comparator A Up Interrupt Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt										Indica	tes that t	the cou	nter has	matched	the cor	nparato	A value	while
Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt										counti	ng down	١.						
Indicates that the counter has matched the comparator A value while counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	2			In	ntCmpAL	J	R/W1C		0	Comp	arator A	Up Inte	errupt					
counting up.  1 IntCntLoad R/W1C 0 Counter=Load Interrupt	_											•	•					
·												ine cou	nter nas	matched	tne cor	nparatoi	A value	wniie
Indicates that the counter has matched the PWMnI OAD register	1			In	tCntLoad	d	R/W1C		0	Count	er=Load	Interru	pt					
										Indica	tes that t	the cou	nter has	matched	d the <b>PW</b>	/MnLO4	D regis	ter

Counter=0 Interrupt

Indicates that the counter has matched 0.

www.DataSheet4U.com

www.DataSheet4U.com

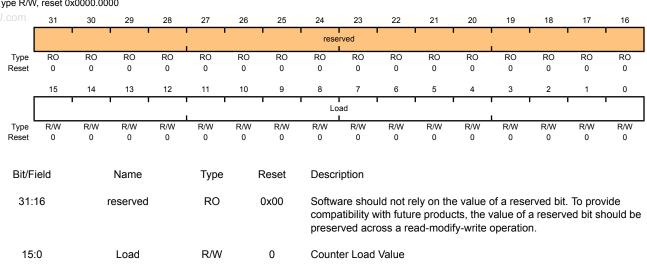
### Register 14: PWM0 Load (PWM0LOAD), offset 0x050

This register contains the load value for the PWM counter. Based on the counter mode, either this value is loaded into the counter after it reaches zero, or it is the limit of up-counting after which the counter decrements back to zero. If the Load Value Update mode is immediate, this value is used the next time the counter reaches zero; if the mode is synchronous, it is used the next time the counter reaches zero after a synchronous update has been requested through the PWM Master Control (PWMCTL) register (see page 408). If this register is re-written before the actual update occurs, the previous value is never used and is lost.

#### PWM0 Load (PWM0LOAD)

Base 0x4002.8000 Offset 0x050

Type R/W, reset 0x0000.0000



The counter load value.

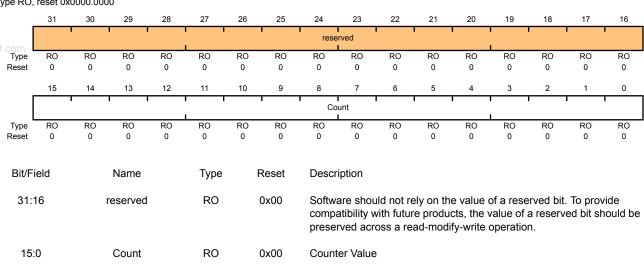
# Register 15: PWM0 Counter (PWM0COUNT), offset 0x054

This register contains the current value of the PWM counter. When this value matches the load register, a pulse is output; this can drive the generation of a PWM signal (via the PWMnGENA/PWMnGENB registers, see page 427 and page 430) or drive an interrupt (via the **PWMnINTEN** register, see page 419). A pulse with the same capabilities is generated when this value is zero.

#### PWM0 Counter (PWM0COUNT)

Base 0x4002.8000 Offset 0x054

Type RO, reset 0x0000.0000



The current value of the counter.

# Register 16: PWM0 Compare A (PWM0CMPA), offset 0x058

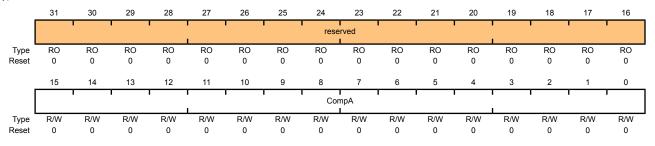
This register contains a value to be compared against the counter. When this value matches the counter, a pulse is output; this can drive the generation of a PWM signal (via the **PWMnGENA/PWMnGENB** registers) or drive an interrupt (via the **PWMnINTEN** register). If the value of this register is greater than the **PWMnLOAD** register (see page 423), then no pulse is ever output.

If the comparator A update mode is immediate (based on the CmpAUpd bit in the **PWMnCTL** register), then this 16-bit CompA value is used the next time the counter reaches zero. If the update mode is synchronous, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 408). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

#### PWM0 Compare A (PWM0CMPA)

Base 0x4002.8000 Offset 0x058

Type R/W, reset 0x0000.0000



Bit/Field	Name	Туре	Reset	Description
31:16	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	CompA	R/W	0x00	Comparator A Value

The value to be compared against the counter.

## Register 17: PWM0 Compare B (PWM0CMPB), offset 0x05C

This register contains a value to be compared against the counter. When this value matches the counter, a pulse is output; this can drive the generation of a PWM signal (via the **PWMnGENA/PWMnGENB** registers) or drive an interrupt (via the **PWMnINTEN** register). If the value of this register is greater than the **PWMnLOAD** register, then no pulse is ever output.

IF the comparator B update mode is immediate (based on the <code>CmpBUpd</code> bit in the **PWMnCTL** register), then this 16-bit <code>CompB</code> value is used the next time the counter reaches zero. If the update mode is synchronous, it is used the next time the counter reaches zero after a synchronous update has been requested through the **PWM Master Control (PWMCTL)** register (see page 408). If this register is rewritten before the actual update occurs, the previous value is never used and is lost.

#### PWM0 Compare B (PWM0CMPB)

Base 0x4002.8000 Offset 0x05C

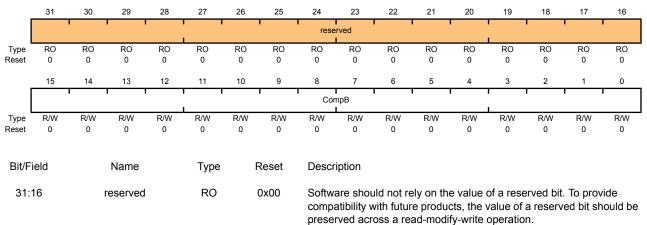
15:0

CompB

R/W

0x00

Type R/W, reset 0x0000.0000



Comparator B Value

The value to be compared against the counter.

### Register 18: PWM0 Generator A Control (PWM0GENA), offset 0x060

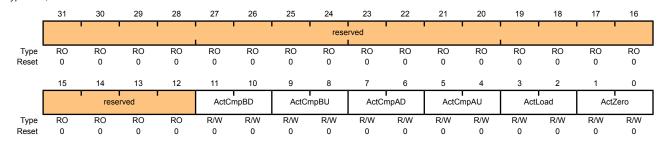
This register controls the generation of the PWMnA signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators. When the counter is running in Count-Down mode, only four of these events occur; when running in Count-Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the PWM signal that is produced.

The **PWM0GENA** register controls generation of the PWM0A signal.

If a zero or load event coincides with a compare A or compare B event, the zero or load action is taken and the compare A or compare B action is ignored. If a compare A event coincides with a compare B event, the compare A action is taken and the compare B action is ignored.

### www.DataSheet4PWM0 Generator A Control (PWM0GENA)

Base 0x4002.8000 Offset 0x060 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:10	ActCmpBD	R/W	0x0	Action for Comparator B Down

The action to be taken when the counter matches comparator  $\ensuremath{\mathsf{B}}$  while counting down.

The table below defines the effect of the event on the output signal.

www.DataSheet4U.com

Value Description

0x0 Do nothing.

0x1 Invert the output signal.

0x2 Set the output signal to 0.

0x3 Set the output signal to 1.

Bit/Field	Name	Туре	Reset	Description
9:8	ActCmpBU	R/W	0x0	Action for Comparator B Up
				The action to be taken when the counter matches comparator B while counting up. Occurs only when the Mode bit in the <b>PWMnCTL</b> register (see page 417) is set to 1.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
4U.com				0x3 Set the output signal to 1.
7:6	ActCmpAD	R/W	0x0	Action for Comparator A Down
				The action to be taken when the counter matches comparator A while counting down.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.
5:4	ActCmpAU	R/W	0x0	Action for Comparator A Up
				The action to be taken when the counter matches comparator A while counting up. Occurs only when the Mode bit in the <b>PWMnCTL</b> register is set to 1.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.
3:2	ActLoad	R/W	0x0	Action for Counter=Load
				The action to be taken when the counter matches the load value.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.

Bit/Field	Name	Туре	Reset	Description
1:0	ActZero	R/W	0x0	Action for Counter=0
				The action to be taken when the counter is zero.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.

ww.DataSheet4U.com

## Register 19: PWM0 Generator B Control (PWM0GENB), offset 0x064

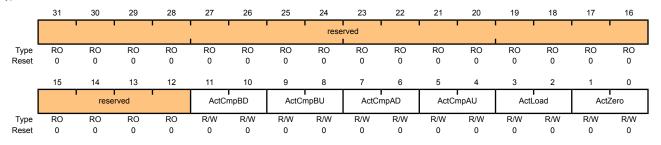
This register controls the generation of the PWMnB signal based on the load and zero output pulses from the counter, as well as the compare A and compare B pulses from the comparators. When the counter is running in Down mode, only four of these events occur; when running in Up/Down mode, all six occur. These events provide great flexibility in the positioning and duty cycle of the PWM signal that is produced.

The **PWM0GENB** register controls generation of the PWM0B signal.

If a zero or load event coincides with a compare A or compare B event, the zero or load action is taken and the compare A or compare B action is ignored. If a compare A event coincides with a compare B event, the compare B action is taken and the compare A action is ignored.

#### www.DataSheet4PWM0 Generator B Control (PWM0GENB)

Base 0x4002.8000 Offset 0x064 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:12	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
11:10	ActCmpBD	R/W	0x0	Action for Comparator B Down

The action to be taken when the counter matches comparator B while counting down.

The table below defines the effect of the event on the output signal.

www.DataSheet4U.com

Value Description

0x0 Do nothing.

0x1 Invert the output signal.

0x2 Set the output signal to 0.

0x3 Set the output signal to 1.

Bit/Field	Name	Туре	Reset	Description
9:8	ActCmpBU	R/W	0x0	Action for Comparator B Up
				The action to be taken when the counter matches comparator B while counting up. Occurs only when the Mode bit in the <b>PWMnCTL</b> register is set to 1.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
4U.com				0x3 Set the output signal to 1.
7:6	ActCmpAD	R/W	0x0	Action for Comparator A Down
				The action to be taken when the counter matches comparator A while counting down.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.
5:4	ActCmpAU	R/W	0x0	Action for Comparator A Up
				The action to be taken when the counter matches comparator A while counting up. Occurs only when the Mode bit in the <b>PWMnCTL</b> register is set to 1.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.
3:2	ActLoad	R/W	0x0	Action for Counter=Load
				The action to be taken when the counter matches the load value.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.

Bit/Field	Name	Type	Reset	Description
1:0	ActZero	R/W	0x0	Action for Counter=0
				The action to be taken when the counter is 0.
				The table below defines the effect of the event on the output signal.
				Value Description
				0x0 Do nothing.
				0x1 Invert the output signal.
				0x2 Set the output signal to 0.
				0x3 Set the output signal to 1.

ww.DataSheet4U.con

## Register 20: PWM0 Dead-Band Control (PWM0DBCTL), offset 0x068

The **PWM0DBCTL** register controls the dead-band generator, which produces the PWM0 and PWM1 signals based on the PWM0A and PWM0B signals. When disabled, the PWM0A signal passes through to the PWM0 signal and the PWM0B signal passes through to the PWM1 signal. When enabled and inverting the resulting waveform, the PWM0B signal is ignored; the PWM0 signal is generated by delaying the rising edge(s) of the PWM0A signal by the value in the **PWM0DBRISE** register (see page 434), and the PWM1 signal is generated by delaying the falling edge(s) of the PWM0A signal by the value in the **PWM0DBFALL** register (see page 435).

#### PWM0 Dead-Band Control (PWM0DBCTL)

Base 0x4002.8000 Offset 0x068

Type R/W, reset 0x0000.0000

U.com	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				•				rese	rved I					'		•
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								reserved					<u>'</u>	'		Enable
Type Reset	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	RO 0	R/W 0							

Bit/Field	Name	Type	Reset	Description
31:1	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
0	Enable	R/W	0	Dead-Band Generator Enable

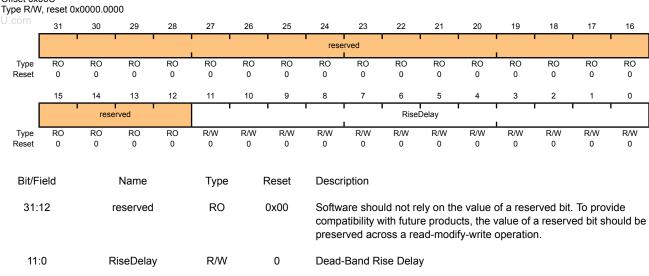
When set, the dead-band generator inserts dead bands into the output signals; when clear, it simply passes the PWM signals through.

#### Register 21: PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE), offset 0x06C

The **PWM0DBRISE** register contains the number of clock ticks to delay the rising edge of the PWM0A signal when generating the PWM0 signal. If the dead-band generator is disabled through the PWMnDBCTL register, the PWM0DBRISE register is ignored. If the value of this register is larger than the width of a High pulse on the input PWM signal, the rising-edge delay consumes the entire High time of the signal, resulting in no High time on the output. Care must be taken to ensure that the input High time always exceeds the rising-edge delay.

PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE)

Base 0x4002.8000 Offset 0x06C



The number of clock ticks to delay the rising edge.

# Register 22: PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL), offset 0x070

The **PWM0DBFALL** register contains the number of clock ticks to delay the falling edge of the PWM0A signal when generating the PWM1 signal. If the dead-band generator is disabled, this register is ignored. If the value of this register is larger than the width of a Low pulse on the input PWM signal, the falling-edge delay consumes the entire Low time of the signal, resulting in no Low time on the output. Care must be taken to ensure that the input Low time always exceeds the falling-edge delay.

PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL)

FallDelay

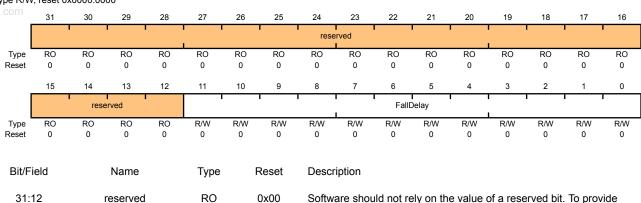
R/W

0x00

Base 0x4002.8000 Offset 0x070

11:0

Type R/W, reset 0x0000.0000



The number of clock ticks to delay the falling edge.

preserved across a read-modify-write operation.

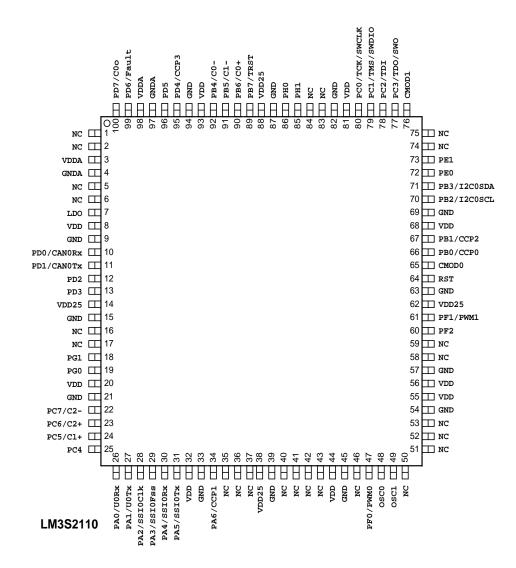
Dead-Band Fall Delay

compatibility with future products, the value of a reserved bit should be

## 17 Pin Diagram

Figure 17-1 on page 436 shows the pin diagram and pin-to-signal-name mapping.

Figure 17-1. Pin Connection Diagram



## 18 Signal Tables

The following tables list the signals available for each pin. Functionality is enabled by software with the **GPIOAFSEL** register.

Important: All multiplexed pins are GPIOs by default, with the exception of the five JTAG pins (PB7 and PC[3:0]) which default to the JTAG functionality.

Table 18-1 on page 437 shows the pin-to-signal-name mapping, including functional characteristics of the signals. Table 18-2 on page 441 lists the signals in alphabetical order by signal name.

Table 18-3 on page 444 groups the signals by functionality, except for GPIOs. Table 18-4 on page 447 lists the GPIO pins and their alternate functionality.

Table 18-1. Signals by Pin Number

Pin Number **Pin Name** Pin Type Description **Buffer Type** 1 NC No connect 2 NC No connect 3 Power The positive supply (3.3 V) for the analog MDDA circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions. 4 GNDA Power The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions. NC No connect 5 6 NC No connect 7 Power LDO Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 µF or greater. When the on-chip LDO is used to provide power to the logic, the LDO pin must also be connected to the VDD25 pins at the board level in addition to the decoupling capacitor(s). Positive supply for I/O and some logic. 8 VDD Power 9 Power Ground reference for logic and I/O pins. GND 10 I/O TTL GPIO port D bit 0 PD0 ı TTL CAN module 0 receive CANORx I/O TTL GPIO port D bit 1 11 PD1 0 TTL CAN module 0 transmit CAN0Tx 12 PD2 I/O TTI GPIO port D bit 2 I/O TTL GPIO port D bit 3 13 PD3 Power 14 Positive supply for most of the logic function, VDD25 including the processor core and most peripherals. 15 Power Ground reference for logic and I/O pins. GND 16 NC No connect NC 17 No connect

Pin Number	Pin Name	Pin Type	Buffer Type	Description
18	PG1	I/O	TTL	GPIO port G bit 1
19	PG0	I/O	TTL	GPIO port G bit 0
20	VDD	-	Power	Positive supply for I/O and some logic.
21	GND	-	Power	Ground reference for logic and I/O pins.
22	PC7	I/O	TTL	GPIO port C bit 7
	C2-	I	Analog	Analog comparator 2 negative input
23	PC6	I/O	TTL	GPIO port C bit 6
	C2+	I	Analog	Analog comparator positive input
24	PC5	I/O	TTL	GPIO port C bit 5
	C1+	I	Analog	Analog comparator positive input
25	PC4	I/O	TTL	GPIO port C bit 4
26	PA0	I/O	TTL	GPIO port A bit 0
	U0Rx	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
27	PA1	I/O	TTL	GPIO port A bit 1
	UOTx	0	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
28	PA2	I/O	TTL	GPIO port A bit 2
	SSIOClk	I/O	TTL	SSI module 0 clock
29	PA3	I/O	TTL	GPIO port A bit 3
	SSI0Fss	I/O	TTL	SSI module 0 frame
30	PA4	I/O	TTL	GPIO port A bit 4
	SSI0Rx	I	TTL	SSI module 0 receive
31	PA5	I/O	TTL	GPIO port A bit 5
	SSIOTx	0	TTL	SSI module 0 transmit
32	VDD	-	Power	Positive supply for I/O and some logic.
33	GND	-	Power	Ground reference for logic and I/O pins.
34	PA6	I/O	TTL	GPIO port A bit 6
	CCP1	I/O	TTL	Capture/Compare/PWM 1
35	NC	-	-	No connect
36	NC	-	-	No connect
37	NC	-	-	No connect
38	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
39	GND	-	Power	Ground reference for logic and I/O pins.
40	NC	-	-	No connect
41	NC	-	-	No connect
42	NC	-	-	No connect
43	NC	-	-	No connect
44	VDD	-	Power	Positive supply for I/O and some logic.
45	GND	-	Power	Ground reference for logic and I/O pins.
46	NC	-	-	No connect

nunu DataChaat411 aam

Pin Number	Pin Name	Pin Type	Buffer Type	Description
47	PF0	I/O	TTL	GPIO port F bit 0
	PWM0	0	TTL	PWM 0
48	osc0	1	Analog	Main oscillator crystal input or an external clock reference input.
49	OSC1	I	Analog	Main oscillator crystal output.
50	NC	-	-	No connect
51	NC	-	-	No connect
52	NC	-	-	No connect
53	NC	-	-	No connect
54	GND	-	Power	Ground reference for logic and I/O pins.
55	VDD	-	Power	Positive supply for I/O and some logic.
56	VDD	-	Power	Positive supply for I/O and some logic.
57	GND	-	Power	Ground reference for logic and I/O pins.
58	NC	-	-	No connect
59	NC	-	-	No connect
60	PF2	I/O	TTL	GPIO port F bit 2
61	PF1	I/O	TTL	GPIO port F bit 1
	PWM1	0	TTL	PWM 1
62	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
63	GND	-	Power	Ground reference for logic and I/O pins.
64	RST	I	TTL	System reset input.
65	CMOD0	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
66	PB0	I/O	TTL	GPIO port B bit 0
	CCP0	I/O	TTL	Capture/Compare/PWM 0
67	PB1	I/O	TTL	GPIO port B bit 1
	CCP2	I/O	TTL	Capture/Compare/PWM 2
68	VDD	-	Power	Positive supply for I/O and some logic.
69	GND	-	Power	Ground reference for logic and I/O pins.
70	PB2	I/O	TTL	GPIO port B bit 2
	I2C0SCL	I/O	OD	I2C module 0 clock
71	PB3	I/O	TTL	GPIO port B bit 3
	I2C0SDA	I/O	OD	I2C module 0 data
72	PE0	I/O	TTL	GPIO port E bit 0
73	PE1	I/O	TTL	GPIO port E bit 1
74	NC	-	-	No connect
75	NC	-	-	No connect
76	CMOD1	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
77	PC3	I/O	TTL	GPIO port C bit 3
	TDO	0	TTL	JTAG TDO and SWO
	SWO	0	TTL	JTAG TDO and SWO

Pin Number	Pin Name	Pin Type	Buffer Type	Description
78	PC2	I/O	TTL	GPIO port C bit 2
	TDI	I	TTL	JTAG TDI
79	PC1	I/O	TTL	GPIO port C bit 1
	TMS	I/O	TTL	JTAG TMS and SWDIO
	SWDIO	I/O	TTL	JTAG TMS and SWDIO
80	PC0	I/O	TTL	GPIO port C bit 0
	TCK	I	TTL	JTAG/SWD CLK
	SWCLK	I	TTL	JTAG/SWD CLK
81	VDD	-	Power	Positive supply for I/O and some logic.
82	GND	-	Power	Ground reference for logic and I/O pins.
83	NC	-	-	No connect
84	NC	-	-	No connect
85	PH1	I/O	TTL	GPIO port H bit 1
86	РН0	I/O	TTL	GPIO port H bit 0
87	GND	-	Power	Ground reference for logic and I/O pins.
88	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
89	PB7	I/O	TTL	GPIO port B bit 7
	TRST	I	TTL	JTAG TRSTn
90	PB6	I/O	TTL	GPIO port B bit 6
	C0+	I	Analog	Analog comparator 0 positive input
91	PB5	I/O	TTL	GPIO port B bit 5
	C1-	I	Analog	Analog comparator 1 negative input
92	PB4	I/O	TTL	GPIO port B bit 4
	C0-	I	Analog	Analog comparator 0 negative input
93	VDD	-	Power	Positive supply for I/O and some logic.
94	GND	-	Power	Ground reference for logic and I/O pins.
95	PD4	I/O	TTL	GPIO port D bit 4
	CCP3	I/O	TTL	Capture/Compare/PWM 3
96	PD5	I/O	TTL	GPIO port D bit 5
97	GNDA	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
98	VDDA	-	Power	The positive supply (3.3 V) for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions.
99	PD6	I/O	TTL	GPIO port D bit 6
	Fault	I	TTL	PWM Fault
100	PD7	I/O	TTL	GPIO port D bit 7
	C0o	0	TTL	Analog comparator 0 output

/ww.DataSheet4U.com

Table 18-2. Signals by Signal Name

Pin Name	Pin Number	Pin Type	Buffer Type	Description
C0+	90	I	Analog	Analog comparator 0 positive input
C0-	92	Į	Analog	Analog comparator 0 negative input
COo	100	0	TTL	Analog comparator 0 output
C1+	24	Į	Analog	Analog comparator positive input
C1-	91	I	Analog	Analog comparator 1 negative input
C2+	23	I	Analog	Analog comparator positive input
C2-	22	I	Analog	Analog comparator 2 negative input
CAN0Rx	10	I	TTL	CAN module 0 receive
CAN0Tx	11	0	TTL	CAN module 0 transmit
CCP0	66	I/O	TTL	Capture/Compare/PWM 0
CCP1	34	I/O	TTL	Capture/Compare/PWM 1
CCP2	67	I/O	TTL	Capture/Compare/PWM 2
CCP3	95	I/O	TTL	Capture/Compare/PWM 3
CMOD0	65	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	76	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
Fault	99	I	TTL	PWM Fault
GND	9	-	Power	Ground reference for logic and I/O pins.
GND	15	-	Power	Ground reference for logic and I/O pins.
GND	21	-	Power	Ground reference for logic and I/O pins.
GND	33	-	Power	Ground reference for logic and I/O pins.
GND	39	-	Power	Ground reference for logic and I/O pins.
GND	45	-	Power	Ground reference for logic and I/O pins.
GND	54	-	Power	Ground reference for logic and I/O pins.
GND	57	-	Power	Ground reference for logic and I/O pins.
GND	63	-	Power	Ground reference for logic and I/O pins.
GND	69	-	Power	Ground reference for logic and I/O pins.
GND	82	-	Power	Ground reference for logic and I/O pins.
GND	87	-	Power	Ground reference for logic and I/O pins.
GND	94	-	Power	Ground reference for logic and I/O pins.
GNDA	4	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
GNDA	97	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
I2C0SCL	70	I/O	OD	I2C module 0 clock
I2C0SDA	71	I/O	OD	I2C module 0 data

Pin Name	Pin Number	Pin Type	Buffer Type	Description
LDO	7	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 $\mu$ F or greater. When the on-chip LDO is used to provide power to the logic, the LDO pin must also be connected to the VDD25 pins at the board level in addition to the decoupling capacitor(s).
NC	1	-	-	No connect
NC	2	-	-	No connect
NC	5	-	-	No connect
NC	6	-	-	No connect
NC	16	-	-	No connect
NC	17	-	-	No connect
NC	35	-	-	No connect
NC	36	-	-	No connect
NC	37	-	-	No connect
NC	40	-	-	No connect
NC	41	-	-	No connect
NC	42	-	-	No connect
NC	43	-	-	No connect
NC	46	-	-	No connect
NC	50	-	-	No connect
NC	51	-	-	No connect
NC	52	-	-	No connect
NC	53	-	-	No connect
NC	58	-	-	No connect
NC	59	-	-	No connect
NC	74	-	-	No connect
NC	75	-	-	No connect
NC	83	-	-	No connect
NC	84	-	-	No connect
OSC0	48	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	49	I	Analog	Main oscillator crystal output.
PA0	26	I/O	TTL	GPIO port A bit 0
PA1	27	I/O	TTL	GPIO port A bit 1
PA2	28	I/O	TTL	GPIO port A bit 2
PA3	29	I/O	TTL	GPIO port A bit 3
PA4	30	I/O	TTL	GPIO port A bit 4
PA5	31	I/O	TTL	GPIO port A bit 5
PA6	34	I/O	TTL	GPIO port A bit 6
PB0	66	I/O	TTL	GPIO port B bit 0
PB1	67	I/O	TTL	GPIO port B bit 1
PB2	70	I/O	TTL	GPIO port B bit 2
PB3	71	I/O	TTL	GPIO port B bit 3

/ww.DataSheet4U.con

Pin Name	Pin Number	Pin Type	Buffer Type	Description
PB4	92	I/O	TTL	GPIO port B bit 4
PB5	91	I/O	TTL	GPIO port B bit 5
PB6	90	I/O	TTL	GPIO port B bit 6
PB7	89	I/O	TTL	GPIO port B bit 7
PC0	80	I/O	TTL	GPIO port C bit 0
PC1	79	I/O	TTL	GPIO port C bit 1
PC2	78	I/O	TTL	GPIO port C bit 2
PC3	77	I/O	TTL	GPIO port C bit 3
PC4	25	I/O	TTL	GPIO port C bit 4
PC5	24	I/O	TTL	GPIO port C bit 5
PC6	23	I/O	TTL	GPIO port C bit 6
PC7	22	I/O	TTL	GPIO port C bit 7
PD0	10	I/O	TTL	GPIO port D bit 0
PD1	11	I/O	TTL	GPIO port D bit 1
PD2	12	I/O	TTL	GPIO port D bit 2
PD3	13	I/O	TTL	GPIO port D bit 3
PD4	95	I/O	TTL	GPIO port D bit 4
PD5	96	I/O	TTL	GPIO port D bit 5
PD6	99	I/O	TTL	GPIO port D bit 6
PD7	100	I/O	TTL	GPIO port D bit 7
PE0	72	I/O	TTL	GPIO port E bit 0
PE1	73	I/O	TTL	GPIO port E bit 1
PF0	47	I/O	TTL	GPIO port F bit 0
PF1	61	I/O	TTL	GPIO port F bit 1
PF2	60	I/O	TTL	GPIO port F bit 2
PG0	19	I/O	TTL	GPIO port G bit 0
PG1	18	I/O	TTL	GPIO port G bit 1
PH0	86	I/O	TTL	GPIO port H bit 0
PH1	85	I/O	TTL	GPIO port H bit 1
PWM0	47	0	TTL	PWM 0
PWM1	61	0	TTL	PWM 1
RST	64	I	TTL	System reset input.
SSIOClk	28	I/O	TTL	SSI module 0 clock
SSIOFss	29	I/O	TTL	SSI module 0 frame
SSI0Rx	30	I	TTL	SSI module 0 receive
SSI0Tx	31	0	TTL	SSI module 0 transmit
SWCLK	80	I	TTL	JTAG/SWD CLK
SWDIO	79	I/O	TTL	JTAG TMS and SWDIO
SWO	77	0	TTL	JTAG TDO and SWO
TCK	80	I	TTL	JTAG/SWD CLK
TDI	78	I	TTL	JTAG TDI
TDO	77	0	TTL	JTAG TDO and SWO

Pin Name	Pin Number	Pin Type	Buffer Type	Description
TMS	79	I/O	TTL	JTAG TMS and SWDIO
TRST	89	I	TTL	JTAG TRSTn
U0Rx	26	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
UOTx	27	0	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
VDD	8	-	Power	Positive supply for I/O and some logic.
VDD	20	-	Power	Positive supply for I/O and some logic.
VDD	32	-	Power	Positive supply for I/O and some logic.
VDD	44	-	Power	Positive supply for I/O and some logic.
VDD	55	-	Power	Positive supply for I/O and some logic.
VDD	56	-	Power	Positive supply for I/O and some logic.
VDD	68	-	Power	Positive supply for I/O and some logic.
VDD	81	-	Power	Positive supply for I/O and some logic.
VDD	93	-	Power	Positive supply for I/O and some logic.
VDD25	14	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
VDD25	38	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
VDD25	62	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
VDD25	88	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
VDDA	3	-	Power	The positive supply (3.3 V) for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions.
VDDA	98	-	Power	The positive supply (3.3 V) for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions.

Table 18-3. Signals by Function, Except for GPIO

Function	Pin Name	Pin Number	Pin Type	Buffer Type	Description
Analog	C0+	90	I	Analog	Analog comparator 0 positive input
Comparators	C0-	92	I	Analog	Analog comparator 0 negative input
	C0o	100	0	TTL	Analog comparator 0 output
	C1+	24	I	Analog	Analog comparator positive input
	C1-	91	I	Analog	Analog comparator 1 negative input
	C2+	23	I	Analog	Analog comparator positive input
	C2-	22	I	Analog	Analog comparator 2 negative input

Function	Pin Name	Pin Number	Pin Type	Buffer Type	Description
Controller Area	CAN0Rx	10	I	TTL	CAN module 0 receive
Network	CAN0Tx	11	0	TTL	CAN module 0 transmit
General-Purpose	CCP0	66	I/O	TTL	Capture/Compare/PWM 0
Timers	CCP1	34	I/O	TTL	Capture/Compare/PWM 1
	CCP2	67	I/O	TTL	Capture/Compare/PWM 2
	CCP3	95	I/O	TTL	Capture/Compare/PWM 3
I2C	I2C0SCL	70	I/O	OD	I2C module 0 clock
	I2C0SDA	71	I/O	OD	I2C module 0 data
JTAG/SWD/SWO	SWCLK	80	I	TTL	JTAG/SWD CLK
	SWDIO	79	I/O	TTL	JTAG TMS and SWDIO
	SWO	77	0	TTL	JTAG TDO and SWO
	TCK	80	ļ	TTL	JTAG/SWD CLK
	TDI	78	Į	TTL	JTAG TDI
	TDO	77	0	TTL	JTAG TDO and SWO
	TMS	79	I/O	TTL	JTAG TMS and SWDIO
PWM	Fault	99	I	TTL	PWM Fault
	PWM0	47	0	TTL	PWM 0
	PWM1	61	0	TTL	PWM 1

Function	Pin Name	Pin Number	Pin Type	Buffer Type	Description
Power	GND	9	-	Power	Ground reference for logic and I/O pins.
	GND	15	-	Power	Ground reference for logic and I/O pins.
	GND	21	-	Power	Ground reference for logic and I/O pins.
	GND	33	-	Power	Ground reference for logic and I/O pins.
	GND	39	-	Power	Ground reference for logic and I/O pins.
	GND	45	-	Power	Ground reference for logic and I/O pins.
	GND	54	-	Power	Ground reference for logic and I/O pins.
	GND	57	-	Power	Ground reference for logic and I/O pins.
	GND	63	-	Power	Ground reference for logic and I/O pins.
	GND	69	-	Power	Ground reference for logic and I/O pins.
	GND	82	-	Power	Ground reference for logic and I/O pins.
	GND	87	-	Power	Ground reference for logic and I/O pins.
	GND	94	-	Power	Ground reference for logic and I/O pins.
	GNDA	4	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
	GNDA	97	-	Power	The ground reference for the analog circuits (ADC, Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
	LDO	7	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 $\mu$ F or greater. When the on-chip LDO is used to provide power to the logic, the LDO pin must also be connected to the VDD25 pins at the board level in addition to the decoupling capacitor(s).
	VDD	8	-	Power	Positive supply for I/O and some logic.
	VDD	20	-	Power	Positive supply for I/O and some logic.
	VDD	32	-	Power	Positive supply for I/O and some logic.
	VDD	44	-	Power	Positive supply for I/O and some logic.
	VDD	55	-	Power	Positive supply for I/O and some logic.
	VDD	56	-	Power	Positive supply for I/O and some logic.
	VDD	68	-	Power	Positive supply for I/O and some logic.
	VDD	81	-	Power	Positive supply for I/O and some logic.
	VDD	93	-	Power	Positive supply for I/O and some logic.
	VDD25	14	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDD25	38	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDD25	62	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDD25	88	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDDA	3	-	Power	

MAN DataSheet411 con

www.DataSheet4U.com

Function	Pin Name	Pin Number	Pin Type	Buffer Type	Description
					The positive supply (3.3 V) for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions.
	VDDA	98	-	Power	The positive supply (3.3 V) for the analog circuits (ADC, Analog Comparators, etc.). These are separated from VDD to minimize the electrical noise contained on VDD from affecting the analog functions.
SSI	SSI0Clk	28	I/O	TTL	SSI module 0 clock
	SSI0Fss	29	I/O	TTL	SSI module 0 frame
	SSIORx	30	I	TTL	SSI module 0 receive
	SSIOTx	31	0	TTL	SSI module 0 transmit
System Control & Clocks	CMOD0	65	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
	CMOD1	76	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
	osc0	48	I	Analog	Main oscillator crystal input or an external clock reference input.
	osc1	49	I	Analog	Main oscillator crystal output.
	RST	64	I	TTL	System reset input.
	TRST	89	I	TTL	JTAG TRSTn
UART	U0Rx	26	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	UOTx	27	0	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.

**Table 18-4. GPIO Pins and Alternate Functions** 

GPIO Pin	Pin Number	Multiplexed Function	Multiplexed Function
PA0	26	U0Rx	
PA1	27	UOTx	
PA2	28	SSI0Clk	
PA3	29	SSI0Fss	
PA4	30	SSI0Rx	
PA5	31	SSIOTx	
PA6	34	CCP1	
PB0	66	CCP0	
PB1	67	CCP2	
PB2	70	I2C0SCL	
PB3	71	I2C0SDA	
PB4	92	C0-	
PB5	91	C1-	
PB6	90	C0+	
PB7	89	TRST	
PC0	80	TCK	SWCLK

November 29, 2007

Bratiminan

GPIO Pin	Pin Number	Multiplexed Function	Multiplexed Function
PC1	79	TMS	SWDIO
PC2	78	TDI	
PC3	77	TDO	SWO
PC4	25		
PC5	24	C1+	
PC6	23	C2+	
PC7	22	C2-	
PD0	10	CAN0Rx	
PD1	11	CAN0Tx	
PD2	12		
PD3	13		
PD4	95	CCP3	
PD5	96		
PD6	99	Fault	
PD7	100	COo	
PE0	72		
PE1	73		
PF0	47	PWM0	
PF1	61	PWM1	
PF2	60		
PG0	19		
PG1	18		
PHO	86		
PH1	85		

ww.DataSheet4U.com

## 19 Operating Characteristics

**Table 19-1. Temperature Characteristics** 

Characteristic	Symbol	Value	Unit
Operating temperature range <sup>a</sup>	T <sub>A</sub>	-40 to +85	°C

a. Maximum storage temperature is 150°C.

#### **Table 19-2. Thermal Characteristics**

Characteristic	Symbol	Value	Unit
Thermal resistance (junction to ambient) <sup>a</sup>	$\Theta_{JA}$	55.3	°C/W
Average junction temperature <sup>b</sup>	T <sub>J</sub>	$T_A + (P_{AVG} \cdot \Theta_{JA})$	°C

a. Junction to ambient thermal resistance  $\boldsymbol{\theta}_{JA}$  numbers are determined by a package simulator.

b. Power dissipation is a function of temperature.

## 20 Electrical Characteristics

#### 20.1 DC Characteristics

## 20.1.1 Maximum Ratings

The maximum ratings are the limits to which the device can be subjected without permanently damaging the device.

**Note:** The device is not guaranteed to operate properly at the maximum ratings.

Table 20-1. Maximum Ratings

Characteristic	Symbol	Value		Unit
ü		Min	Max	
I/O supply voltage (V <sub>DD</sub> )	V <sub>DD</sub>	0	4	٧
Core supply voltage (V <sub>DD25</sub> )	V <sub>DD25</sub>	0	4	٧
Analog supply voltage (V <sub>DDA</sub> )	$V_{DDA}$	0	4	٧
Input voltage	V <sub>IN</sub>	-0.3	5.5	V
Maximum current per output pins	I	-	25	mA

a. Voltages are measured with respect to GND.

Important: This device contains circuitry to protect the inputs against damage due to high-static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either GND or VDD).

## 20.1.2 Recommended DC Operating Conditions

**Table 20-2. Recommended DC Operating Conditions** 

Parameter	Parameter Name	Min	Nom	Max	Unit
V <sub>DD</sub>	I/O supply voltage	3.0	3.3	3.6	V
V <sub>DD25</sub>	Core supply voltage	2.25	2.5	2.75	V
$V_{DDA}$	Analog supply voltage	3.0	3.3	3.6	V
V <sub>IH</sub>	High-level input voltage	2.0	-	5.0	V
V <sub>IL</sub>	Low-level input voltage	-0.3	-	1.3	V
V <sub>SIH</sub>	High-level input voltage for Schmitt trigger inputs	0.8 * V <sub>DD</sub>	-	$V_{DD}$	V
V <sub>SIL</sub>	Low-level input voltage for Schmitt trigger inputs	0	-	0.2 * V <sub>DD</sub>	V
V <sub>OH</sub>	High-level output voltage	2.4	-	-	V
V <sub>OL</sub>	Low-level output voltage	-	-	0.4	V
I <sub>OH</sub>	High-level source current, V <sub>OH</sub> =2.4 V				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA

Parameter	Parameter Name		Min	Nom	Max	Unit
I <sub>OL</sub>	Low-level sink current, V <sub>OL</sub> =0.4 V					
	2-	mA Drive	2.0	-	-	mA
	4-	mA Drive	4.0	-	-	mA
	8-	mA Drive	8.0	-	-	mA

## 20.1.3 On-Chip Low Drop-Out (LDO) Regulator Characteristics

#### **Table 20-3. LDO Regulator Characteristics**

Parameter	Parameter Name	Min	Nom	Max	Unit
V <sub>LDOOUT</sub>	Programmable internal (logic) power supply output value	2.25	2.5	2.75	V
	Output voltage accuracy	-	2%	-	%
t <sub>PON</sub>	Power-on time	-	-	100	μs
t <sub>ON</sub>	Time on	-	-	200	μs
t <sub>OFF</sub>	Time off	-	-	100	μs
V <sub>STEP</sub>	Step programming incremental voltage	-	50	-	mV
C <sub>LDO</sub>	External filter capacitor size for internal power supply	1.0	-	3.0	μF

#### 20.1.4 Power Specifications

The power measurements specified in the tables that follow are run on the core processor using SRAM with the following specifications (except as noted):

- $V_{DD} = 3.3 \text{ V}$
- V<sub>DD25</sub> = 2.50 V
- $V_{DDA} = 3.3 \text{ V}$
- Temperature = 25°C
- Clock Source (MOSC) =3.579545 MHz Crystal Oscillator
- Main oscillator (MOSC) = enabled
- Internal oscillator (IOSC) = disabled

**Table 20-4. Detailed Power Specifications** 

Parameter	Parameter Name	Conditions	$0.3 \text{ V V}_{\text{DD}}, \text{ V}_{\text{DDA}}, $ $\text{V}_{\text{DDPHY}}$				V V <sub>DD25</sub>	Unit														
			Nom	Max	Nom	Max																
I <sub>DD_RUN</sub>	Run mode 1 (Flash	V <sub>DD25</sub> = 2.50 V	3	pending <sup>a</sup>	64	pendinga	mA															
	loop)	Code= while(1){} executed in Flash																				
		Peripherals = All ON																				
		System Clock = 25 MHz (with PLL)																				
	Run mode 2 (Flash	V <sub>DD25</sub> = 2.50 V	0	pending <sup>a</sup>	33	pendinga	mA															
	loop)	Code= while(1){} executed in Flash																				
		Peripherals = All OFF																				
		System Clock = 25 MHz (with PLL)																				
	Run mode 1 (SRAM	V <sub>DD25</sub> = 2.50 V	3	pending <sup>a</sup> 57	pending <sup>a</sup> 5	pending <sup>a</sup>	57	pendinga	mA													
	loop)	Code= while(1){} executed in SRAM																				
		Peripherals = All ON																				
		System Clock = 25 MHz (with PLL)																				
	Run mode 2 (SRAM	V <sub>DD25</sub> = 2.50 V	0	pending <sup>a</sup>	27	pendinga	mA															
	loop)	Code= while(1){} executed in SRAM																				
		Peripherals = All OFF																				
		System Clock = 25 MHz (with PLL)																				
I <sub>DD_SLEEP</sub>	Sleep mode	V <sub>DD25</sub> = 2.50 V	0	pending <sup>a</sup>	12	pendinga	mA															
		Peripherals = All OFF																				
		System Clock = 25 MHz (with PLL)																				
I <sub>DD_DEEPSLEEP</sub>	Deep-Sleep mode	LDO = 2.25 V	0.14	pending <sup>a</sup>	0.18	pendinga	mA															
		Peripherals = All OFF																				
		System Clock = IOSC30KHZ/64																				

a. Pending characterization completion.

## 20.1.5 Flash Memory Characteristics

**Table 20-5. Flash Memory Characteristics** 

Parameter	Parameter Name	Min	Nom	Max	Unit
PE <sub>CYC</sub>	Number of guaranteed program/erase cycles before failure <sup>a</sup>	10,000	100,000	-	cycles
T <sub>RET</sub>	Data retention at average operating temperature of 85°C	10	-	-	years
T <sub>PROG</sub>	Word program time	20	-	-	μs
T <sub>ERASE</sub>	Page erase time	20	-	-	ms
T <sub>ME</sub>	Mass erase time	200	-	-	ms

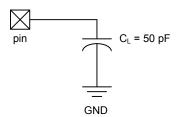
a. A program/erase cycle is defined as switching the bits from 1-> 0 -> 1.

## 20.2 AC Characteristics

#### 20.2.1 Load Conditions

Unless otherwise specified, the following conditions are true for all timing measurements. Timing measurements are for 4-mA drive strength.

Figure 20-1. Load Conditions



w.DataSneet4U.com

#### 20.2.2 Clocks

Table 20-6. Phase Locked Loop (PLL) Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
f <sub>ref_crystal</sub>	Crystal reference <sup>a</sup>	3.579545	-	8.192	MHz
f <sub>ref_ext</sub>	External clock reference <sup>a</sup>	3.579545	-	8.192	MHz
f <sub>pll</sub>	PLL frequency <sup>b</sup>	-	400	-	MHz
T <sub>READY</sub>	PLL lock time	-	-	0.5	ms

a. The exact value is determined by the crystal value programmed into the XTAL field of the **Run-Mode Clock Configuration** (RCC) register.

**Table 20-7. Clock Characteristics** 

Parameter	Parameter Name	Min	Nom	Max	Unit
f <sub>IOSC</sub>	Internal 12 MHz oscillator frequency	8.4	12	15.6	MHz
f <sub>IOSC30KHZ</sub>	Internal 30 KHz oscillator frequency	21	30	39	KHz
f <sub>MOSC</sub>	Main oscillator frequency	1	-	8	MHz
t <sub>MOSC_per</sub>	Main oscillator period	125	-	1000	ns
f <sub>ref_crystal_bypass</sub>	Crystal reference using the main oscillator (PLL in BYPASS mode)	1	-	8	MHz
f <sub>ref_ext_bypass</sub>	External clock reference (PLL in BYPASS mode)	0	-	25	MHz
f <sub>system_clock</sub>	System clock	0	-	25	MHz

**Table 20-8. Crystal Characteristics** 

Parameter Name		Value				
Frequency	8	6	4	3.5	MHz	
Frequency tolerance	±50	±50	±50	±50	ppm	
Aging	±5	±5	±5	±5	ppm/yr	
Oscillation mode	Parallel	Parallel	Parallel	Parallel		
Temperature stability (0 - 85 °C)	±25	±25	±25	±25	ppm	
Motional capacitance (typ)	27.8	37.0	55.6	63.5	pF	

b. PLL frequency is automatically calculated by the hardware based on the  ${\tt XTAL}$  field of the  ${\tt RCC}$  register.

Parameter Name		Units			
Motional inductance (typ)	14.3	19.1	28.6	32.7	mH
Equivalent series resistance (max)	120	160	200	220	Ω
Shunt capacitance (max)	10	10	10	10	pF
Load capacitance (typ)	16	16	16	16	pF
Drive level (typ)	100	100	100	100	μW

## 20.2.3 Analog Comparator

#### **Table 20-9. Analog Comparator Characteristics**

Parameter	Parameter Name	Min	Nom	Max	Unit
V <sub>OS</sub>	Input offset voltage	-	±10	±25	mV
V <sub>CM</sub>	Input common mode voltage range	0	-	V <sub>DD</sub> -1.5	٧
C <sub>MRR</sub>	Common mode rejection ratio	50	-	-	dB
T <sub>RT</sub>	Response time	-	-	1	μs
T <sub>MC</sub>	Comparator mode change to Output Valid	-	-	10	μs

#### **Table 20-10. Analog Comparator Voltage Reference Characteristics**

Parameter	Parameter Name	Min	Nom	Max	Unit
R <sub>HR</sub>	Resolution high range	-	V <sub>DD</sub> /32	-	LSB
R <sub>LR</sub>	Resolution low range	-	V <sub>DD</sub> /24	-	LSB
A <sub>HR</sub>	Absolute accuracy high range	-	-	±1/2	LSB
A <sub>LR</sub>	Absolute accuracy low range	-	-	±1/4	LSB

## 20.2.4 I<sup>2</sup>C

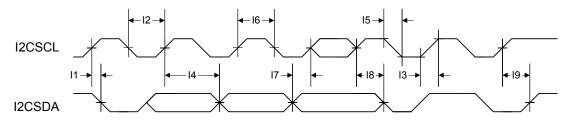
#### Table 20-11, I<sup>2</sup>C Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
I1 <sup>a</sup>	t <sub>SCH</sub>	Start condition hold time	36	-	-	system clocks
I2 <sup>a</sup>	t <sub>LP</sub>	Clock Low period	36	-	-	system clocks
I3 <sup>b</sup>	t <sub>SRT</sub>	<code>I2CSCL/I2CSDA</code> rise time (V $_{IL}$ =0.5 V to V $_{IH}$ =2.4 V)	-	-	(see note b)	ns
I4 <sup>a</sup>	t <sub>DH</sub>	Data hold time	2	-	-	system clocks
I5 <sup>c</sup>	t <sub>SFT</sub>	<code>I2CSCL/I2CSDA</code> fall time (V $_{IH}$ =2.4 V to V $_{IL}$ =0.5 V)	-	9	10	ns
I6 <sup>a</sup>	t <sub>HT</sub>	Clock High time	24	-	-	system clocks
I7 <sup>a</sup>	t <sub>DS</sub>	Data setup time	18	-	-	system clocks
I8 <sup>a</sup>	t <sub>SCSR</sub>	Start condition setup time (for repeated start condition only)	36	-	-	system clocks
19 <sup>a</sup>	t <sub>SCS</sub>	Stop condition setup time	24	-	-	system clocks

- a. Values depend on the value programmed into the TPR bit in the I<sup>2</sup>C Master Timer Period (I2CMTPR) register; a TPR programmed for the maximum I2CSCL frequency (TPR=0x2) results in a minimum output timing as shown in the table above. The I<sup>2</sup>C interface is designed to scale the actual data transition time to move it to the middle of the I2CSCL Low period. The actual position is affected by the value programmed into the TPR; however, the numbers given in the above values are minimum values.
- b. Because I2CSCL and I2CSDA are open-drain-type outputs, which the controller can only actively drive Low, the time I2CSCL or I2CSDA takes to reach a high level depends on external signal capacitance and pull-up resistor values.
- c. Specified at a nominal 50 pF load.

454 November 29, 2007

Figure 20-2. I<sup>2</sup>C Timing

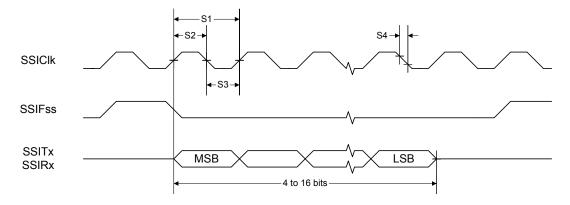


## 20.2.5 Synchronous Serial Interface (SSI)

**Table 20-12. SSI Characteristics** 

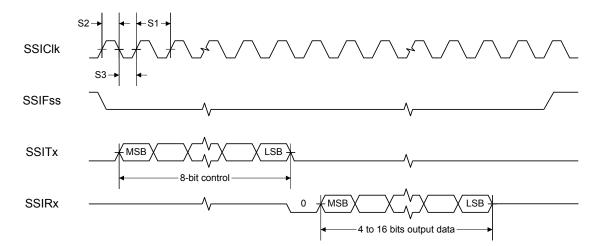
Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
S1	t <sub>clk_per</sub>	SSIC1k cycle time	2	-	65024	system clocks
S2	t <sub>clk_high</sub>	SSIC1k high time	-	1/2	-	t clk_per
S3	t <sub>clk_low</sub>	SSIC1k low time	-	1/2	-	t clk_per
S4	t <sub>clkrf</sub>	SSIC1k rise/fall time	-	7.4	26	ns
S5	t <sub>DMd</sub>	Data from master valid delay time	0	-	20	ns
S6	t <sub>DMs</sub>	Data from master setup time	20	-	-	ns
S7	t <sub>DMh</sub>	Data from master hold time	40	-	-	ns
S8	t <sub>DSs</sub>	Data from slave setup time	20	-	-	ns
S9	t <sub>DSh</sub>	Data from slave hold time	40	-	-	ns

Figure 20-3. SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement



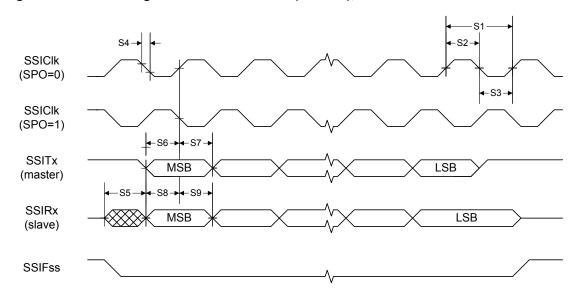
November 29, 2007 455

Figure 20-4. SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer



ww.DataSheet4U.com

Figure 20-5. SSI Timing for SPI Frame Format (FRF=00), with SPH=1



### 20.2.6 JTAG and Boundary Scan

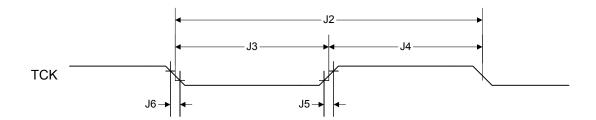
**Table 20-13. JTAG Characteristics** 

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J1	f <sub>TCK</sub>	TCK operational clock frequency	0	-	10	MHz
J2	t <sub>TCK</sub>	TCK operational clock period	100	-	-	ns
J3	t <sub>TCK_LOW</sub>	TCK clock Low time	-	t <sub>TCK</sub>	-	ns

www.DataSheet4U.com

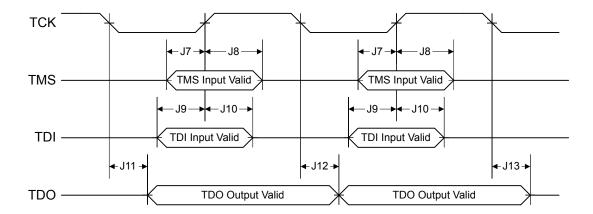
Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J4	t <sub>TCK_HIGH</sub>	TCK clock High time	-	t <sub>TCK</sub>	-	ns
J5	t <sub>TCK_R</sub>	TCK rise time	0	-	10	ns
J6	t <sub>TCK_F</sub>	тск fall time	0	-	10	ns
J7	t <sub>TMS_SU</sub>	тмs setup time to тск rise	20	-	-	ns
J8	t <sub>TMS_HLD</sub>	TMS hold time from TCK rise	20	-	-	ns
J9	t <sub>TDI_</sub> SU	TDI setup time to TCK rise	25	-	-	ns
J10	t <sub>TDI_HLD</sub>	TDI hold time from TCK rise	25	-	-	ns
J11	TCK fall to Data Valid from High-Z	2-mA drive	-	23	35	ns
t <sub>TDO_ZDV</sub>		4-mA drive		15	26	ns
_		8-mA drive		14	25	ns
		8-mA drive with slew rate control		18	29	ns
J12	TCK fall to Data Valid from Data Valid	2-mA drive	-	21	35	ns
t <sub>TDO_DV</sub>		4-mA drive		14	25	ns
_		8-mA drive		13	24	ns
		8-mA drive with slew rate control		18	28	ns
J13	TCK fall to High-Z from Data Valid	2-mA drive	-	9	11	ns
t <sub>TDO_DVZ</sub>		4-mA drive		7	9	ns
_		8-mA drive		6	8	ns
		8-mA drive with slew rate control		7	9	ns
J14	t <sub>TRST</sub>	TRST assertion time	100	-	-	ns
J15	t <sub>TRST_SU</sub>	TRST setup time to TCK rise	10	-	-	ns

Figure 20-6. JTAG Test Clock Input Timing



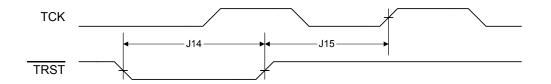
November 29, 2007 457

Figure 20-7. JTAG Test Access Port (TAP) Timing



w.DataSneet40.com

Figure 20-8. JTAG TRST Timing



## 20.2.7 General-Purpose I/O

Note: All GPIOs are 5 V-tolerant.

**Table 20-14. GPIO Characteristics** 

Parameter	Parameter Name	Condition	Min	Nom	Max	Unit
t <sub>GPIOR</sub>	GPIO Rise Time (from 20% to 80% of V <sub>DD</sub> )	2-mA drive	-	17	26	ns
		4-mA drive		9	13	ns
		8-mA drive		6	9	ns
		8-mA drive with slew rate control		10	12	ns
t <sub>GPIOF</sub>	GPIO Fall Time (from 80% to 20% of V <sub>DD</sub> )	2-mA drive	-	17	25	ns
		4-mA drive		8	12	ns
		8-mA drive		6	10	ns
		8-mA drive with slew rate control		11	13	ns

#### 20.2.8 Reset

**Table 20-15. Reset Characteristics** 

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
R1	$V_{TH}$	Reset threshold	-	2.0	-	V

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
R2	V <sub>BTH</sub>	Brown-Out threshold	2.85	2.9	2.95	٧
R3	T <sub>POR</sub>	Power-On Reset timeout	-	10	-	ms
R4	T <sub>BOR</sub>	Brown-Out timeout	-	500	-	μs
R5	T <sub>IRPOR</sub>	Internal reset timeout after POR	6	-	11	ms
R6	T <sub>IRBOR</sub>	Internal reset timeout after BOR <sup>a</sup>	0	-	1	μs
R7	T <sub>IRHWR</sub>	Internal reset timeout after hardware reset (RST pin)	0	-	1	ms
R8	T <sub>IRSWR</sub>	Internal reset timeout after software-initiated system reset a	2.5	-	20	μs
R9	T <sub>IRWDR</sub>	Internal reset timeout after watchdog reset <sup>a</sup>	2.5	-	20	μs
R10	T <sub>VDDRISE</sub>	Supply voltage (V <sub>DD</sub> ) rise time (0V-3.3V)	-	-	100	ms
R11	T <sub>MIN</sub>	Minimum RST pulse width	2	-	-	μs

Figure 20-9. External Reset Timing (RST)

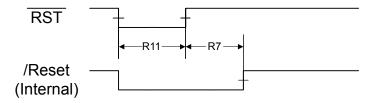


Figure 20-10. Power-On Reset Timing

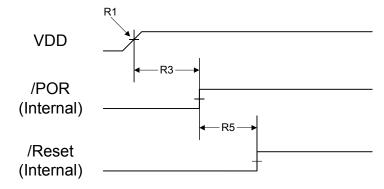
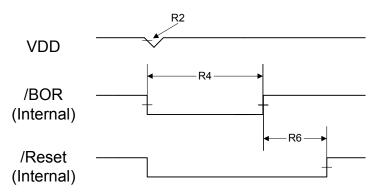


Figure 20-11. Brown-Out Reset Timing



a. 20 \* t <sub>MOSC\_per</sub>

Figure 20-12. Software Reset Timing

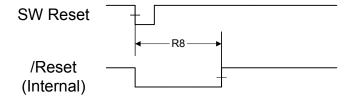
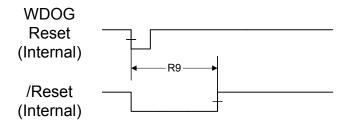


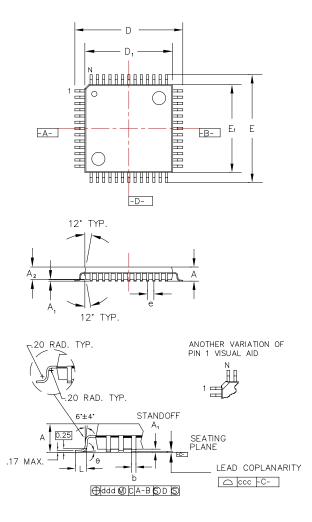
Figure 20-13. Watchdog Reset Timing



ww.DataSheet4U.com

## 21 Package Information

Figure 21-1. 100-Pin LQFP Package



**Note:** The following notes apply to the package drawing.

- 1. All dimensions shown in mm.
- 2. Dimensions shown are nominal with tolerances indicated.
- 3. Foot length 'L' is measured at gage plane 0.25 mm above seating plane.

Body +2.00 mm	Footprint, 1.4 mm	package thickness				
Symbols	Leads	100L				
A	Max.	1.60				
A <sub>1</sub>		0.05 Min./0.15 Max.				
A <sub>2</sub>	±0.05	1.40				
D	±0.20	16.00				
D <sub>1</sub>	±0.05	14.00				
E	±0.20	16.00				
E <sub>1</sub>	±0.05	14.00				
L	±0.15/-0.10	0.60				
е	BASIC	0.50				
b	±0.05	0.22				
θ	===	0°~7°				
ddd	Max.	0.08				
ccc	Max.	0.08				
JEDEC Refer	MS-026					
Variation [	BED					

/ww.DataSheet4U.com

## A Serial Flash Loader

#### A.1 Serial Flash Loader

The Stellaris<sup>®</sup> serial flash loader is a preprogrammed flash-resident utility used to download code to the flash memory of a device without the use of a debug interface. The serial flash loader uses a simple packet interface to provide synchronous communication with the device. The flash loader runs off the crystal and does not enable the PLL, so its speed is determined by the crystal used. The two serial interfaces that can be used are the UART0 and SSI0 interfaces. For simplicity, both the data format and communication protocol are identical for both serial interfaces.

#### A.2 Interfaces

Once communication with the flash loader is established via one of the serial interfaces, that interface is used until the flash loader is reset or new code takes over. For example, once you start communicating using the SSI port, communications with the flash loader via the UART are disabled until the device is reset.

#### A.2.1 UART

The Universal Asynchronous Receivers/Transmitters (UART) communication uses a fixed serial format of 8 bits of data, no parity, and 1 stop bit. The baud rate used for communication is automatically detected by the flash loader and can be any valid baud rate supported by the host and the device. The auto detection sequence requires that the baud rate should be no more than 1/32 the crystal frequency of the board that is running the serial flash loader. This is actually the same as the hardware limitation for the maximum baud rate for any UART on a Stellaris<sup>®</sup> device which is calculated as follows:

Max Baud Rate = System Clock Frequency / 16

In order to determine the baud rate, the serial flash loader needs to determine the relationship between its own crystal frequency and the baud rate. This is enough information for the flash loader to configure its UART to the same baud rate as the host. This automatic baud-rate detection allows the host to use any valid baud rate that it wants to communicate with the device.

The method used to perform this automatic synchronization relies on the host sending the flash loader two bytes that are both 0x55. This generates a series of pulses to the flash loader that it can use to calculate the ratios needed to program the UART to match the host's baud rate. After the host sends the pattern, it attempts to read back one byte of data from the UART. The flash loader returns the value of 0xCC to indicate successful detection of the baud rate. If this byte is not received after at least twice the time required to transfer the two bytes, the host can resend another pattern of 0x55, 0x55, and wait for the 0xCC byte again until the flash loader acknowledges that it has received a synchronization pattern correctly. For example, the time to wait for data back from the flash loader should be calculated as at least 2\*(20(bits/sync)/baud rate (bits/sec)). For a baud rate of 115200, this time is 2\*(20/115200) or 0.35 ms.

#### A.2.2 SSI

The Synchronous Serial Interface (SSI) port also uses a fixed serial format for communications, with the framing defined as Motorola format with SPH set to 1 and SPO set to 1. See "Frame Formats" on page 277 in the SSI chapter for more information on formats for this transfer protocol. Like the UART, this interface has hardware requirements that limit the maximum speed that the SSI clock can run. This allows the SSI clock to be at most 1/12 the crystal frequency of the board running

the flash loader. Since the host device is the master, the SSI on the flash loader device does not need to determine the clock as it is provided directly by the host.

## A.3 Packet Handling

All communications, with the exception of the UART auto-baud, are done via defined packets that are acknowledged (ACK) or not acknowledged (NAK) by the devices. The packets use the same format for receiving and sending packets, including the method used to acknowledge successful or unsuccessful reception of a packet.

#### A.3.1 Packet Format

All packets sent and received from the device use the following byte-packed format.

ww.DataSheet4U.com

```
struct
{
  unsigned char ucSize;
  unsigned char ucCheckSum;
  unsigned char Data[];
};
```

ucSize The first byte received holds the total size of the transfer including

the size and checksum bytes.

ucChecksum This holds a simple checksum of the bytes in the data buffer only.

The algorithm is Data[0]+Data[1]+...+ Data[ucSize-3].

Data This is the raw data intended for the device, which is formatted in

some form of command interface. There should be ucSize-2 bytes of data provided in this buffer to or from the device.

#### A.3.2 Sending Packets

The actual bytes of the packet can be sent individually or all at once; the only limitation is that commands that cause flash memory access should limit the download sizes to prevent losing bytes during flash programming. This limitation is discussed further in the section that describes the serial flash loader command, COMMAND\_SEND\_DATA (see "COMMAND\_SEND\_DATA (0x24)" on page 466).

Once the packet has been formatted correctly by the host, it should be sent out over the UART or SSI interface. Then the host should poll the UART or SSI interface for the first non-zero data returned from the device. The first non-zero byte will either be an ACK (0xCC) or a NAK (0x33) byte from the device indicating the packet was received successfully (ACK) or unsuccessfully (NAK). This does not indicate that the actual contents of the command issued in the data portion of the packet were valid, just that the packet was received correctly.

#### A.3.3 Receiving Packets

The flash loader sends a packet of data in the same format that it receives a packet. The flash loader may transfer leading zero data before the first actual byte of data is sent out. The first non-zero byte is the size of the packet followed by a checksum byte, and finally followed by the data itself. There is no break in the data after the first non-zero byte is sent from the flash loader. Once the device communicating with the flash loader receives all the bytes, it must either ACK or NAK the packet to indicate that the transmission was successful. The appropriate response after sending a NAK to the flash loader is to resend the command that failed and request the data again. If needed, the host may send leading zeros before sending down the ACK/NAK signal to the flash loader, as the

www.DataSheet4U.com

flash loader only accepts the first non-zero data as a valid response. This zero padding is needed by the SSI interface in order to receive data to or from the flash loader.

#### A.4 Commands

The next section defines the list of commands that can be sent to the flash loader. The first byte of the data should always be one of the defined commands, followed by data or parameters as determined by the command that is sent.

#### A.4.1 COMMAND\_PING (0X20)

This command simply accepts the command and sets the global status to success. The format of the packet is as follows:

Byte[0] = 0x03;
Byte[1] = checksum(Byte[2]);
Byte[2] = COMMAND\_PING;

The ping command has 3 bytes and the value for COMMAND\_PING is 0x20 and the checksum of one byte is that same byte, making Byte[1] also 0x20. Since the ping command has no real return status, the receipt of an ACK can be interpreted as a successful ping to the flash loader.

### A.4.2 COMMAND\_GET\_STATUS (0x23)

This command returns the status of the last command that was issued. Typically, this command should be sent after every command to ensure that the previous command was successful or to properly respond to a failure. The command requires one byte in the data of the packet and should be followed by reading a packet with one byte of data that contains a status code. The last step is to ACK or NAK the received data so the flash loader knows that the data has been read.

Byte[0] = 0x03
Byte[1] = checksum(Byte[2])
Byte[2] = COMMAND\_GET\_STATUS

### A.4.3 COMMAND\_DOWNLOAD (0x21)

This command is sent to the flash loader to indicate where to store data and how many bytes will be sent by the COMMAND\_SEND\_DATA commands that follow. The command consists of two 32-bit values that are both transferred MSB first. The first 32-bit value is the address to start programming data into, while the second is the 32-bit size of the data that will be sent. This command also triggers an erase of the full area to be programmed so this command takes longer than other commands. This results in a longer time to receive the ACK/NAK back from the board. This command should be followed by a COMMAND\_GET\_STATUS to ensure that the Program Address and Program size are valid for the device running the flash loader.

The format of the packet to send this command is a follows:

Byte[0] = 11
Byte[1] = checksum(Bytes[2:10])
Byte[2] = COMMAND\_DOWNLOAD
Byte[3] = Program Address [31:24]
Byte[4] = Program Address [23:16]
Byte[5] = Program Address [15:8]
Byte[6] = Program Address [7:0]
Byte[7] = Program Size [31:24]

```
Byte[8] = Program Size [23:16]
Byte[9] = Program Size [15:8]
Byte[10] = Program Size [7:0]
```

#### A.4.4 COMMAND\_SEND\_DATA (0x24)

This command should only follow a COMMAND\_DOWNLOAD command or another COMMAND SEND DATA command if more data is needed. Consecutive send data commands automatically increment address and continue programming from the previous location. The caller should limit transfers of data to a maximum 8 bytes of packet data to allow the flash to program successfully and not overflow input buffers of the serial interfaces. The command terminates programming once the number of bytes indicated by the COMMAND\_DOWNLOAD command has been received. Each time this function is called it should be followed by a COMMAND GET STATUS to ensure that the data was successfully programmed into the flash. If the flash loader sends a NAK to this command, the flash loader does not increment the current address to allow retransmission of the previous data.

```
Byte[0] = 11
Byte[1] = checksum(Bytes[2:10])
Byte[2] = COMMAND_SEND_DATA
Byte[3] = Data[0]
Byte[4] = Data[1]
Byte[5] = Data[2]
Byte[6] = Data[3]
Byte[7] = Data[4]
Byte[8] = Data[5]
Byte[9] = Data[6]
Byte[10] = Data[7]
```

#### A.4.5 COMMAND RUN (0x22)

This command is used to tell the flash loader to execute from the address passed as the parameter in this command. This command consists of a single 32-bit value that is interpreted as the address to execute. The 32-bit value is transmitted MSB first and the flash loader responds with an ACK signal back to the host device before actually executing the code at the given address. This allows the host to know that the command was received successfully and the code is now running.

```
Byte[0] = 7
Byte[1] = checksum(Bytes[2:6])
Byte[2] = COMMAND RUN
Byte[3] = Execute Address[31:24]
Byte[4] = Execute Address[23:16]
Byte[5] = Execute Address[15:8]
Byte[6] = Execute Address[7:0]
```

#### A.4.6 COMMAND RESET (0x25)

This command is used to tell the flash loader device to reset. This is useful when downloading a new image that overwrote the flash loader and wants to start from a full reset. Unlike the COMMAND\_RUN command, this allows the initial stack pointer to be read by the hardware and set up for the new code. It can also be used to reset the flash loader if a critical error occurs and the host device wants to restart communication with the flash loader.

466 November 29, 2007 **Preliminary** 

```
Byte[0] = 3
Byte[1] = checksum(Byte[2])
Byte[2] = COMMAND_RESET
```

The flash loader responds with an ACK signal back to the host device before actually executing the software reset to the device running the flash loader. This allows the host to know that the command was received successfully and the part will be reset.

# **B** Register Quick Reference

31															
	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
System Base 0x40															
DID0, type	RO, offset 0	0x000, res	et -												
		VER									CLA	ASS			
MAJOR							MINOR								
PBORCTL,	type R/W, o	offset 0x03	30, reset 0	x0000.7FFC	)										
														BORIOR	
LDOPCTL,	type R/W, o	offset 0x03	34, reset 0	k0000.0000											
D.00111															
												VA	DJ		
RIS, type R	RO, offset 0x	c050, rese	t 0x0000.0	000											
									PLLLRIS					BORRIS	
IMC, type F	R/W, offset 0	0x054, res	et 0x0000.	0000											
									PLLLIM					BORIM	
MISC, type	R/W1C, offs	set 0x058	, reset 0x0	000.000											
									PLLLMIS					BORMIS	
RESC, type	e R/W, offset	t 0x05C, r	eset -												
										LDO	SW	WDT	BOR	POR	EXT
RCC, type	R/W, offset	0x060, res	set 0x07AE												
				ACG		SYS	SDIV		USESYSDIV		USEPWMDIV		PWMDIV		
		PWRDN		BYPASS			XT	AL		osc	SRC			IOSCDIS	MOSCE
PLLCFG, ty	ype RO, offs	set 0x064,	reset -												
						_									
DOOC :	- DAM	4.00=0		0.0000		F							R		
		τ υχυ70, re	2561 112078	v.2800											
	e R/W, offset		JOCT OXOTO		2)/2	DIV /0									
USERCC2			3301 0207 0		SYS	DIV2				OSCSBCS					
USERCC2	F	PWRDN2		BYPASS2		DIV2				OSCSRC2					
USERCC2		PWRDN2		BYPASS2	0000					OSCSRC2					
USERCC2	F	PWRDN2		BYPASS2	0000	DIV2									
USERCC2  DSLPCLKC	FCFG, type R/	PWRDN2	0x144, res	BYPASS2	0000					OSCSRC2	;				
USERCC2  DSLPCLKO	CFG, type R/	PWRDN2 /W, offset 0x004, res	0x144, res	BYPASS2	DSDIV	ORIDE						TNO			
DSLPCLKO	RO, offset 0	PWRDN2 /W, offset 0x004, res	0x144, res	BYPASS2	DSDIV				С		PAR'		ROHS	OI	JAL
DSLPCLKO DID1, type	RO, offset 0 VER	PWRDN2 /W, offset  0x004, res	0x144, res	BYPASS2 set 0x0780.0	DSDIV	ORIDE							ROHS	QL	JAL
DSLPCLKO DID1, type	RO, offset 0	PWRDN2 /W, offset  0x004, res	0x144, res	BYPASS2 set 0x0780.0	DSDIV	ORIDE	SRA	MSZ	С		PAR'		ROHS	QL	JAL
DSLPCLKO DID1, type	RO, offset 0 VER	PWRDN2 /W, offset  0x004, res	0x144, res	BYPASS2 set 0x0780.0	DSDIV	ORIDE		MSZ SHSZ	С		PAR'		ROHS	QL	JAL
DSLPCLKO DID1, type PDC0, type F	RO, offset 0	PWRDN2 /W, offset 0x004, res	0x144, res et - et 0x003F.0	BYPASS2 set 0x0780.0	DSDIV	ORIDE			С		PAR'		ROHS	QL	JAL
DSLPCLKO DID1, type PDC0, type F	RO, offset 0 VER	PWRDN2 /W, offset 0x004, res	0x144, res et - et 0x003F.0	BYPASS2 set 0x0780.0	DSDIV	ORIDE			С		PAR'		ROHS	QL	JAL
DSLPCLKO DID1, type PDC0, type F	RO, offset 0	PWRDN2 /W, offset 0x004, rese 0x008, rese	0x144, res et - et 0x003F.0	BYPASS2 set 0x0780.0	DSDIV	ORIDE	FLAS		С		PAR'		ROHS	QL	
DSLPCLKO DID1, type P DC0, type F	RO, offset 0:  RO, offset 0:  RO, offset 0:  MINSYS	PWRDN2 /W, offset 0x004, rese 0x008, rese 0x010, rese	0x144, res et - et 0x003F.C	BYPASS2 set 0x0780.0	DSDIV	ORIDE	FLAS	SHSZ	С		PAR' Pk	KG			
DSLPCLKO DID1, type PDC0, type F	RO, offset 0:	PWRDN2 /W, offset 0x004, rese 0x008, rese 0x010, rese	0x144, res et - et 0x003F.C	BYPASS2 set 0x0780.0	DSDIV	ORIDE	CAN0	SHSZ	С		PAR' Pk	KG	SWO		JTAG
DSLPCLKO DID1, type  P DC0, type F	RO, offset 0:  RO, offset 0:  RO, offset 0:  MINSYS	PWRDN2 /W, offset 0x004, rese 0x008, rese 0x010, rese	0x144, res et - et 0x003F.C	BYPASS2 set 0x0780.0	DSDIV	ORIDE	FLAS	SHSZ	С		PAR' Pk	KG		SWD	JAL JTAG TIMER UARTI
DID1, type  F DC0, type F  DC1, type F	RO, offset 0:  RO, offset 0:  MINSYS RO, offset 0:	PWRDN2 /W, offset  0x004, rese  0x008, rese  0x010, rese  SDIV	0x144, res et - et 0x003F.0 et 0x0110.7	BYPASS2 set 0x0780.0	DSDIV	ORIDE	CAN0	SHSZ	С		PAR Pk PWM PLL	KG	SWO	SWD	JTAG
DID1, type  F DC0, type F  DC1, type F	RO, offset 0:  RO, offset 0:  RO, offset 0:  MINSYS	PWRDN2 /W, offset  0x004, rese  0x008, rese  0x010, rese  SDIV	0x144, res et - et 0x003F.0 et 0x0110.7	BYPASS2 set 0x0780.0	DSDIV	ORIDE	CAN0	SHSZ	С		PAR Pk PWM PLL	KG	SWO	SWD	JTAG

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
15 C4 type	14	13	12 set 0x0000.	11	10	9	8	7	6	5	4	3	2	1
DO4, type	NO, onset	0,010,16	361 020000.											
								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB
RCGC0, ty	pe R/W, of	ffset 0x100	, reset 0x0	0000040			1			1	1		1	
							CAN0				PWM			
												WDT		
SCGC0, ty	pe R/W, of	ffset 0x110	, reset 0x00	0000040										
							CAN0				PWM			
												WDT		
DCGC0, ty	pe R/W, of	ffset 0x120	, reset 0x0	0000040			1					1		
							CAN0				PWM	WDT		
DCCC4 4	ma DAM at	FF4 0×404	L ====+ 0×0	0000000								WDT		
ROGOT, ty	pe R/VV, OI	nset UX104	, reset 0x0		COMP2	COMP1	COMP0						TIMER2	TIMER1
			I2C0		CONFZ	CONFI	CONFU				SSI0		TIMERZ	INVERT
SCGC1. tv	pe R/W. of	fset 0x114	, reset 0x00	0000000							23.0			
, • • •			,		COMP2	COMP1	COMP0						TIMER2	TIMER1
			I2C0								SSI0			
DCGC1, ty	pe R/W, of	ffset 0x124	, reset 0x0	0000000										
					COMP2	COMP1	COMP0						TIMER2	TIMER1
			I2C0								SSI0			
RCGC2, ty	/pe R/W, of	ffset 0x108	, reset 0x0	0000000										
								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB
SCGC2, ty	pe R/W, of	ffset 0x118	, reset 0x00	000000										
								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB
DCGC2 to	no P/M of	ffeat Av128	, reset 0x0	000000				GFIOH	GFIOG	GFIOF	GFIOE	GFIOD	GFIOC	GFIOB
D0002, ty	pe latt, of	11361 02 120	, 16361 020											
								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB
SRCR0, ty	pe R/W, of	fset 0x040	, reset 0x00	000000										
							CAN0				PWM			
												WDT		
SRCR1, ty	pe R/W, of	fset 0x044	, reset 0x00	0000000										
					COMP2	COMP1	COMP0						TIMER2	TIMER1
			I2C0								SSI0			
SRCR2, ty	pe R/W, of	fset 0x048	, reset 0x00	0000000										
								00:2::	00:00	00:00	00:07	00:00	00:00	05:5-
								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB
	Memor													
	ontrol C													
	100F.D000			0000										
riviA, type	K/W, offse	et uxuu0, re	eset 0x0000	.0000										
							OFF	SET						
FMD type	R/W offer	ot 0x004 ==	eset 0x0000	0000			OFF	UL I						
, type	, Olise	UAUU4, IE	JOSE GAUGUE				DA	TA						
								TA						
FMC, type	R/W, offse	et 0x008, re	eset 0x0000	0.0000										
							WR	KEY						

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
FCRIS, ty	pe RO, offs	et 0x00C,	reset 0x000	30.0000										
														PRIS
FCIM, typ	e R/W, offse	et 0x010, r	eset 0x000	0.0000										
														PMA
FCMISC, 1	type R/W1C	C, offset 0x	.014, reset	0x0000.00	00									
														PMI
Interna	l Memory	y												
	Control													
Base 0x4	100F.E000	)												
USECRL,	type R/W, o	offset 0x14	0, reset 0x	16										
											119	SEC		
FMPRE0.	type R/W. c	offset 0x13	0 and 0x20	)0, reset 0:	kFFFF.FFFF						0.	020		
	, ,			-,			READ_	ENABLE						
							READ_	ENABLE						
FMPPE0,	type R/W, o	offset 0x13	4 and 0x40	10, reset 0x	(FFFF.FFFF									
								ENABLE ENABLE						
USER DE	G, type R/V	W, offset 0	x1D0. rese	t 0xFFFF.F	FFE		FRUG_	LINUDEE						
NW	.,.,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	,						DATA						
						D	ATA							DB
	G0, type R	!/W, offset	0x1E0, rese	et 0xFFFF.	FFFF									
NW								DATA						
USER RE	G1, type R	Z/W. offset	0x1E4. res	et 0xFFFF	FFFF		D.	ATA						
NW	, ., pe 10	-,						DATA						
							D	ATA						
FMPRE1,	type R/W, c	offset 0x20	4, reset 0x	0000.0000										
								ENABLE ENABLE						
FMPRE2	type R/W, c	offset 0x20	)8. reset 0×	0000.0000			KEAD_	LINABLE						
,	.,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		-,				READ_	ENABLE						
								ENABLE						
FMPRE3,	type R/W, c	offset 0x20	C, reset 0x	.0000.0000										
								ENABLE						
FMPPF1	type R/W, o	offset OxAO	4. reset fly	0000 0000			KEAD_	ENABLE						
. m. FLI,	.5 pe 10 vv, 0	J.1361 UA4U	-, 1030t UX				PROG	ENABLE						
								ENABLE						
FMPPE2,	type R/W, o	offset 0x40	8, reset 0x	0000.0000										
								ENABLE						
I	tuno P/M -	offeet 0×40	C rocat A	,0000 0000			PROG_	ENABLE						
EMDDE2	LVIJE IS/VV. C	J11561 0X40	o, reset ux	.0000.0000	'									
FMPPE3,	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,						PROG	ENABLE						

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rt A base:														
	rt B base: rt C base:														
GPIO Po	rt D base:	0x4000.7	7000												
	rt E base:														
	rt F base: rt G base:														
	rt H base:														
			(000, reset 0	×0000 0000	n										
OI IODAIA	i, typo rain	, 011001 02	1000, 10001 0		•										
											DF	ATA			
GPIODIR,	type R/W,	offset 0x40	00, reset 0x(	0000.0000											
											D	IR			
GPIOIS, ty	pe R/W, of	fset 0x404	, reset 0x00	000.000											
											ı	S			
GPIOIBE	type R/W	offset 0x40	08, reset 0x(	0000.0000				1							
,	., p , ,		.,												
											IE.	BE .			
GPIOIEV,	type R/W, o	offset 0x40	C, reset 0x0	0000.0000											
											IE	ΕV			
GPIOIM, ty	ype R/W, o	ffset 0x410	), reset 0x00	000.000											
											IN	л ИЕ			
GPIORIS	type RO. o	ffset 0x41	4, reset 0x0	000.0000											
or iorao,	type ite, e	IIIOCC UX 4 I													
											K	RIS			
GPIOMIS,	type RO, o	offset 0x41	8, reset 0x0	000.0000								1			
											M	IIS			
GPIOICR,	type W1C,	offset 0x4	1C, reset 0	k0000.0000											
										1	1	C			
GPIOAFSI	FL type R/	W. offset (	0x420, reset	_											
0. 107 0.	, . <b>, po</b>	,	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,												
											A.E.	CEI			
ODIOS	D. 4:	v -85 : 5	-500		-						AF:	SEL			
GPIODR2	K, type R/V	v, offset 0x	(500, reset (	JX0000.00F	F										
											DF	RV2			
GPIODR4	R, type R/V	V, offset 0x	(504, reset (	0x0000.000	0										
											DF	RV4			
GPIODR	R. type R/M	V. offset 0x	(508, reset (	0x0000.000	0										
	,		,												
											רי	2)/8			
onie				*****							DF	RV8			
GPIOODR	, type R/W,	offset 0x	50C, reset 0	x0000.0000											
												DE			
GPIOPUR	, type R/W,	offset 0x5	510, reset -												
											DI	UE			
											г	<u>-</u>			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIOPDR	, type R/W,	offset 0x5	14, reset 0>	k0000.0000											
											PI	DE			
GPIOSLR,	type R/W,	offset 0x5	18, reset 0x	0000.0000											
											SI	RL			
GPIODEN	, type R/W,	offset 0x5	1C, reset -												
0010100	K 6 D.04	1 - ff 4 O	500	00000 0000							Di	EN			
GPIOLOC	K, type K/W	i, oπset ux	520, reset	0x0000.000	1		1.0	OCK							
								OCK							
GPIOCR 1	type -, offse	ot 0x524 re	eset -												
Or IOOK, I	type -, onse	, t 0, 52 , T t	-												
											C	R			
GPIOPerig	ohID4, type	RO, offset	t 0xFD0, res	set 0x0000.0	0000			1							
·															
											PI	D4			
GPIOPerip	ohID5, type	RO, offset	t 0xFD4, res	set 0x0000.0	0000										
											PI	D5			
GPIOPerip	ohID6, type	RO, offset	t 0xFD8, res	set 0x0000.0	0000										
											PI	D6			
GPIOPerip	ohID7, type	RO, offset	t 0xFDC, re	set 0x0000.	0000										
00100 :	LIDO	<b>50</b> "									PI	D7			
GPIOPerip	oniDu, type	RO, offset	UXFEU, res	set 0x0000.0	J061										
											PI	D0			
GPIOPerir	nhID1 tyne	RO offset	OxFF4 res	set 0x0000.0	2000										
O. 101 C.I.	л <b>Б</b> 1, <b>сур</b> е	110, 011001	- OXI E-1, 100		,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,										
											PI	L D1			
GPIOPerip	ohID2, type	RO, offset	t 0xFE8, res	set 0x0000.0	0018										
											PI	D2			
GPIOPerip	ohID3, type	RO, offset	t 0xFEC, re	set 0x0000.	0001										
											PI	D3			
GPIOPCel	IID0, type F	RO, offset (	0xFF0, rese	et 0x0000.00	00D										
											CI	D0			
GPIOPCel	IID1, type F	RO, offset (	0xFF4, rese	et 0x0000.00	)F0										
											0.1	D1			
CDIODO	IID2 4:	00 6#	0vEF0	- 0v0000 00	NO.E						CI	D1			
GPIOPCE	ווטצ, type F	kU, offset (	UXFF8, rese	et 0x0000.00	105										
											CI	D2			
GDIODCAL	IID3 type 5	O offers	OVEEC room	et 0x0000.00	nR1			L			CI	D4			
OF IOP CEI	יים, type ה	o, onset	JAI 1-0, res	. 0.0000.00	, U										
											CI	D3			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	oase: 0x40 oase: 0x40														
	oase: 0x40														
GPTMCF	G, type R/W	, offset 0x	000, reset 0	x0000.000	)										
														GPTMCFG	
GPTMTAI	MR, type R/	W, offset 0	x004, reset	0x0000.00	00										
												TAAMS	TACMR	TA	MR
GPTMTB	MR, type R/	W, offset 0	x008, reset	0x0000.00	00										
												TBAMS	TBCMR	ТВ	MR
GPTMCT	L, type R/W	offset 0x0	00C, reset 0	x0000.000	)							ı			
	TBPWML				/ENT	TBSTALL	TBEN		TAPWML	TAOTE	RTCEN	IAE	/ENT	TASTALL	TAEN
GPTMIMF	R, type R/W,	offset 0x0	118, reset 02	KUOOO.0000											
					CDE"	CDA41A4	TDTOWA					DTOWA	CAFINA	CALAIN	TATO
CDTMDIC	tune BC	effoot 0::04	C rocat C	0000 0000	CBEIM	СВМІМ	TBTOIM					RTCIM	CAEIM	CAMIM	TATOIN
GPIWKIS	S, type RO, o	omset uxu1	C, reset ux	0000.0000											
					CBERIS	CBMDIS	TBTORIS					RTCRIS	CAERIS	CAMRIS	TATODI
CDTMMIS	S, type RO,	offeet Ov03	On reset Ov	0000 0000	OBLINO	ODIVINO	IBIONIO					KTOKIO	OALINO	OAWING	IAION
OI TIVIIVII	J, type ItO,	JIISEL UAUZ	LO, TOSEL OX												
					CBEMIS	CBMMIS	TRTOMIS					RTCMIS	CAEMIS	CAMMIS	TATOMI
GPTMICE	R, type W1C	offset 0x0	024. reset 0	×0000.0000		02						1	07 12.11.10	0,	., ., ., ., .,
C	t, type ii ie	, onoce ox	100010												
					CBECINT	CBMCINT	TBTOCINT					RTCCINT	CAECINT	CAMCINT	TATOCIN
GPTMTAI	ILR, type R/	W. offset 0	x028. reset	0x0000.FF				FF (32-bit	mode)						
	, ,,,,	,	,				TAII								
							TAI								
GPTMTB	ILR, type R/	W, offset 0	x02C, rese	t 0x0000.FI	FF										
		-													
				I			TBI	LRL				I			
GPTMTAI	MATCHR, ty	pe R/W, of	fset 0x030,	reset 0x00	00.FFFF (1	6-bit mode	) and 0xFF	FF.FFFF (3	32-bit mode)						
							TAN	/IRH							
							TAN	//RL							
GPTMTB	MATCHR, ty	pe R/W, of	ffset 0x034,	reset 0x00	00.FFFF										
							TBN	//RL							
GPTMTAI	PR, type R/\	V, offset 0	x038, reset	0x0000.000	00										
											TAF	PSR			
GPTMTB	PR, type R/\	N, offset 0	x03C, reset	0x0000.00	00										
											TBI	PSR			
GPTMTAI	PMR, type R	/W, offset	0x040, rese	et 0x0000.0	000										
											TAP	SMR			
GPTMTB	PMR, type F	R/W, offset	0x044, rese	et 0x0000.0	000										
											TBP	SMR			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	R, type RO,	offset 0x0	48, reset 0x	0000.FFFF	(16-bit mo	ode) and 0x	FFFF.FFF	(32-bit mo	de)	1		1	1	-	1
							TA	ARH							
							T	ARL							
GPTMTBI	R, type RO	offset 0x0	4C, reset 0	k0000.FFFI	F										
								 BRL							
Motobo	don Time						11	DKL .							
	dog Time 4000.0000														
			000, reset 0	xFFFF.FF	F										
							WD.	TLoad							
							WD.	TLoad							
WDTVALI	UE, type RO	O, offset 0x	(004, reset 0	xFFFF.FFI	FF										
								ΓValue ΓValue							
WDTCTI	tyne R/W	offset OxO(	08, reset 0x(	0000 0000			VVD	value							
,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	typo rati,		, 10301 020												
														RESEN	INTEN
WDTICR,	type WO, o	offset 0x00	C, reset -												
								ΓIntClr							
							WD.	ΓIntClr							
WDTRIS,	type RO, o	ffset 0x010	), reset 0x00	000.0000											
															WDTRI
WDTMIS,	type RO, o	ffset 0x014	4, reset 0x00	000.0000											
<u> </u>															
															WDTM
WDTTEST	T, type R/W	, offset 0x4	418, reset 0	k0000.0000	)										
							OTALL								
WDTI OC	K tune B/V	V offeet Ov	C00, reset (	~^^^	10		STALL								
WDILOC	r, type rv	v, onset ox	Coo, reser (	JA0000.00C			WD	TLock							
								TLock							
WDTPerip	phID4, type	RO, offset	t 0xFD0, res	et 0x0000.	0000										
											Р	ID4			
WDTPerip	phID5, type	RO, offset	t 0xFD4, res	et 0x0000.	0000			1							
											P	ID5			
WDTPerig	phID6, type	RO, offset	t 0xFD8, res	et 0x0000.	0000						•				
		,													
											Р	ID6			
WDTPerip	ohID7, type	RO, offset	t 0xFDC, res	set 0x0000	.0000										
WDTDari	ahIDO 4	PO offer	OVEEN ===	ot 0v0000	0005						Р	ID7			
wu i Perip	priiDu, type	KU, offset	t 0xFE0, res	et ux0000.	0005										
											Р	ID0			
WDTPerip	phID1, type	RO, offset	t 0xFE4, res	et 0x0000.	0018										
											Р	ID1			
WDTPerip	phID2, type	RO, offset	t 0xFE8, res	et 0x0000.	0018										
												ID0			
											P	ID2			

31	30	29	20	27	26	25	24	22	22	24	20	19	10	17	16
15	14	13	28 12	27 11	10	25 9	24 8	23 7	22 6	21 5	20	3	18	17	0
	phID3, type I							<u> </u>			<u> </u>				
	, ,,,,,,														
											PI	D3			
WDTPCel	IID0, type R	O, offset (	0xFF0, rese	t 0x0000.0	00D										
											CI	D0			
WDTPCel	IID1, type R	O, offset (	0xFF4, rese	t 0x0000.0	0F0										
											CI	D1			
WDTPCel	IID2, type R	O, offset (	0xFF8, rese	t 0x0000.0	005			1				1			
											CI	D2			
WDTPCel	IID3, type R	O offset (	DyFFC rese	 	0B1						- Ci	D2			
11011 061	mbs, type it	0, 011361 (	JX11 0, 1636		VD1										
											CI	I D3			
Univers	sal Asyno	chrono	us Receiv	vers/Tra	nsmitter	s (UAR)	rs)								
	pase: 0x400			. J. 5/ 11a		5 (5/111)	-,								
UARTDR,	type R/W, o	offset 0x00	00, reset 0x(	0000.0000											
				OE	BE	PE	FE				DA	TA			
UARTRSF	R/UARTECR	, type RO	, offset 0x00	04, reset 0	x0000.0000										
												OE	BE	PE	FE
UARTRSF	R/UARTECR	type WC	), offset 0x0	04, reset 0	x0000.0000							I			
											D.4	Τ.			
HADTED	type RO, of	foot 0v019	P rooot OvO	000 0000							DP	ATA			
UAKIFK,	type KO, of	iset uxu i	b, reset uxu												
								TXFE	RXFF	TXFF	RXFE	BUSY			
UARTILPI	R, type R/W,	offset 0x	020, reset 0	x0000.000	0							I.			
		•													
											ILPD	VSR			
UARTIBR	D, type R/W	, offset 0x	(024, reset (	0x0000.000	00										
							DI\	/INT							
UARTFBF	RD, type R/W	V, offset 0	x028, reset	0x0000.00	00										
												DIVE	RAC		
UARILGE	RH, type R/W	v, offset u	x02C, reset	0x0000.00	100										
								SPS	WI	_EN	FEN	STP2	EPS	PEN	BRK
HARTCTI	_, type R/W,	offset Oxi	130 reset 0:	×0000 0300	)			] 010	***		1 214	0112	LIO	I LIV	DIVIC
O/IIII O	-, t <b>ypo</b> 1011,	OHOUL UX	, , , , , , , , , , , , , , , , , , , ,												
						RXE	TXE	LBE					SIRLP	SIREN	UARTEN
UARTIFL	S, type R/W,	offset 0x	034, reset 0	x0000.001	2										-
											RXIFLSEL			TXIFLSEL	
UARTIM,	type R/W, of	ffset 0x03	8, reset 0x0	000.0000						-					
					OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM				
UARTRIS	, type RO, o	ffset 0x03	C, reset 0x0	0000.000F											
					OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS				

November 29, 2007 475 Preliminary

31 50 29 28 27 28 27 28 28 27 28 28 27 28 28 27 28 28 27 28 28 27 28 27 28 28 27 28 27 27 28 28 27 28 28 27 28 27 28 28 27 28 28 27 28 28 27 28 28 27 28 28 27 28 28 28 27 28 28 28 28 28 28 28 28 28 28 28 28 28																
ARTIFICIAL TO A CONTROL Oxford	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OEMIS BEMIS PEMIS PEMIS RITINS TOWNS PROMIS  ARTTCR, type WIC. offset 0x044, reset 0x0000.0000  ARTTPeriphID4, type RO, offset 0x040, reset 0x0000.0000  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID5  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID6  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID7  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID7  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID7  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  PID3  ARTTPeriphID5, type RO, offset 0x040, reset 0x0000.0000  SCID2  ARTTPC-dilID2, type RO, offset 0x040, reset 0x0000.0000  SCID2  ARTTPC-dilID3, type RO, offset 0x0400, reset 0x00000.0000  SCR0, type RW, offset 0x000, reset 0x00000.0000	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  PIDS  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCellIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCellIDS, type RO, offset 0xFD6, reset 0x0000.00000  PIDS  ARTPCELIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCELIDS, type RW, offset 0x0000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000	UARTMIS,	type RO, o	offset 0x04	0, reset 0x0	0000.0000											
ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD4, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  PIDS  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  ARTPeriphIDS, type RO, offset 0xFD5, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPERIPHIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCellIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCellIDS, type RO, offset 0xFD6, reset 0x0000.00000  PIDS  ARTPCELIDS, type RO, offset 0xFD6, reset 0x0000.0000  PIDS  ARTPCELIDS, type RW, offset 0x0000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000																
OBIC   BEIC   PEIC   FEIC   RTIC   TXIC   RXIC						OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS				
ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID7  ARTPeriphID6, type RO, offset 0xFE6, reset 0x0000.0000  PID7  ARTPeriphID6, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPeriphID8, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPERIPHID8, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPERIPHID8, type RO, offset 0xFE6, reset 0x0000.0000  CID8  ARTPCOIIID9, type RO, offset 0xFE6, reset 0x0000.0000  CID8  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000	UARTICR,	type W1C,	offset 0x0	)44, reset 0:	×0000.0000	)							•			
ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphID6, type RO, offset 0xFD6, reset 0x0000.0000  PID7  ARTPeriphID6, type RO, offset 0xFE6, reset 0x0000.0000  PID7  ARTPeriphID6, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPeriphID8, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPERIPHID8, type RO, offset 0xFE6, reset 0x0000.0000  PID8  ARTPERIPHID8, type RO, offset 0xFE6, reset 0x0000.0000  CID8  ARTPCOIIID9, type RO, offset 0xFE6, reset 0x0000.0000  CID8  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000																
ARTPeriphIDS, type RO, offset 0xFDs, reset 0x0000.0000  PIDS  ARTPeriphIDS, type RO, offset 0xFDs, reset 0x0000.0000  PIDS  ARTPeriphIDS, type RO, offset 0xFDs, reset 0x0000.0000  PIDS  ARTPeriphIDS, type RO, offset 0xFEs, reset 0x0000.0000  CIDS  ARTPCOIIIDS, type RO, offset 0xFEs, reset 0x0000.0000  CIDS  ARTPCOIIIDS, type RO, offset 0xFEs, reset 0x0000.0005  CIDS  ARTPCOIIIDS, type RO, offset 0xFEs, reset 0x0000.0005  CIDS  SCR SPH SPO FRF DSS  SICRI, type RIW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SIDR, type RIW, offset 0x000, reset 0x0000.0000						OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC				
ARTPeriphID5, type RO, offset 0xFD4, reset 0x0000,0000  ARTPeriphID5, type RO, offset 0xFD6, reset 0x0000,0000  ARTPeriphID7, type RO, offset 0xFD6, reset 0x0000,0000  ARTPeriphID7, type RO, offset 0xFD6, reset 0x0000,0000  ARTPeriphID7, type RO, offset 0xFE6, reset 0x0000,0000  ARTPeriphID2, type RO, offset 0xFE6, reset 0x0000,0000  ARTPeriphID3, type RO, offset 0xFE6, reset 0x0000,0000  ARTPeriphID3, type RO, offset 0xFE6, reset 0x0000,0001  PID3  ARTPeriphID3, type RO, offset 0xFE6, reset 0x0000,0000  ARTPCeIIID3, type RO, offset 0xFE6, reset 0x0000,0000  CID0  ARTPCeIIID3, type RO, offset 0xFE6, reset 0x0000,0005  CID1  ARTPCeIIID3, type RO, offset 0xFE6, reset 0x0000,0005  CID2  ARTPCeIIID3, type RO, offset 0xFE6, reset 0x0000,0005  SCR0, type RW, offset 0x000, reset 0x0000,0000  SCR0, type RW, offset 0x004, reset 0x0000,0000	UARTPeri	ohID4. type	RO. offse	t 0xFD0. re	set 0x0000	0.0000										
ARTPeriphiD5, type R0, offset 0xFD4, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFB6, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD1, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD2, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID1, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE6, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SOR SSE LBM  SIDR, type RW, offset 0x008, reset 0x0000.0000		<b>-</b> ., ., p.	, , , , , , , , ,													
ARTPeriphiD5, type R0, offset 0xFD4, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFB6, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD1, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD2, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID1, type R0, offset 0xFE6, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE6, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR0, type RW, offset 0x000, reset 0x0000.0000  SOR SSE LBM  SIDR, type RW, offset 0x008, reset 0x0000.0000												DI	ID4			
ARTPeriphiD5, type R0, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD7, type R0, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD1, type R0, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD2, type R0, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE0, reset 0x0000.0018  ARTPeriphiD3, type R0, offset 0xFE0, reset 0x0000.0019  ARTPeriphiD3, type R0, offset 0xFE0, reset 0x0000.0019  ARTPCellID3, type R0, offset 0xFE0, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE0, reset 0x0000.0000  ARTPCellID3, type R0, offset 0xFE0, reset 0x0000.0005  ARTPCellID3, type R0, offset 0xFE0, reset 0x0000.0005  ARTPCellID3, type R0, offset 0xFE0, reset 0x0000.0005  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x000, reset 0x0000.0000  SOD MS SSE LBM  SICR1, type RW, offset 0x000, reset 0x0000.0000	HADTDaris	-bIDE from	DO effec	4.0vFD4. ==		0000										
ARTPeriphiD6, type R0, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphiD7, type R0, offset 0xFD6, reset 0x0000.0000  PID7  ARTPeriphiD9, type R0, offset 0xFD6, reset 0x0000.0001  PID0  ARTPeriphiD2, type R0, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  PID2  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type R0, offset 0xFE6, reset 0x0000.0001  ARTPCellID1, type R0, offset 0xFE6, reset 0x0000.0000  CID2  ARTPCellID2, type R0, offset 0xFF6, reset 0x0000.0005  CID2  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0005  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0005  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0006  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0006  CID3  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000	UAKTPEN	опірэ, турє	e KO, onse	t uxrb4, re	set uxuuuu	1.0000			I							
ARTPeriphiD6, type R0, offset 0xFD6, reset 0x0000.0000  PID6  ARTPeriphiD7, type R0, offset 0xFD6, reset 0x0000.0000  PID7  ARTPeriphiD9, type R0, offset 0xFD6, reset 0x0000.0001  PID0  ARTPeriphiD2, type R0, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  PID2  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD3, type R0, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type R0, offset 0xFE6, reset 0x0000.0001  ARTPCellID1, type R0, offset 0xFE6, reset 0x0000.0000  CID2  ARTPCellID2, type R0, offset 0xFF6, reset 0x0000.0005  CID2  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0005  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0005  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0006  CID3  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0006  CID3  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000													 			
ARTPeriphiD7, type RO, offset 0xFDC, reset 0x0000.0000  ARTPeriphiD7, type RO, offset 0xFE0, reset 0x0000.0011  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RO, offset 0xFE5, reset 0x0000.0001  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID4, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID5, type RO, offset 0xFE6, reset 0x0000.0000  CID0  ARTPCellID5, type RO, offset 0xFE6, reset 0x0000.0000  CID0  ARTPCellID5, type RO, offset 0xFE6, reset 0x0000.0000  CID0  CID1  ARTPCellID5, type RO, offset 0xFE6, reset 0x0000.0000  CID2  ARTPCellID5, type RO, offset 0xFE6, reset 0x0000.0001  CID3  SRCR, type RW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SOD MIS SSE LEM  SIDR, type RW, offset 0x006, reset 0x0000.0000	Lcom											PI	D5			
ARTPeriphiD7, type RO, offset 0xFDC, reset 0x0000.0000  ARTPeriphiD7, type RO, offset 0xFE6, reset 0x0000.0011  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE6, reset 0x0000.0001  CID0  ARTPCellID2, type RO, offset 0xFF6, reset 0x0000.0005  CID1  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.0005  CID3  SIGR1, type RW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SIGR1, type RW, offset 0x004, reset 0x0000.0000  SOD MIS SSE LBM  SIDR, type RW, offset 0x006, reset 0x0000.0000	UARTPeri	ohID6, type	RO, offse	t 0xFD8, re	set 0x0000	0.0000										
ARTPeriphiD7, type RO, offset 0xFDC, reset 0x0000.0000  ARTPeriphiD7, type RO, offset 0xFE6, reset 0x0000.0011  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE6, reset 0x0000.0001  CID0  ARTPCellID2, type RO, offset 0xFF6, reset 0x0000.0005  CID1  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.0005  CID3  SIGR1, type RW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SIGR1, type RW, offset 0x004, reset 0x0000.0000  SOD MIS SSE LBM  SIDR, type RW, offset 0x006, reset 0x0000.0000																
ARTPeriphiD0, type RO, offset 0xFE0, reset 0x0000.0000  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD1, type RO, offset 0xFE5, reset 0x0000.0010  ARTPeriphiD2, type RO, offset 0xFE6, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  ARTPCelliD0, type RO, offset 0xFF6, reset 0x0000.0001  ARTPCelliD1, type RO, offset 0xFF6, reset 0x0000.0000  CID0  ARTPCelliD2, type RO, offset 0xFF6, reset 0x0000.0000  CID1  ARTPCelliD3, type RO, offset 0xFF6, reset 0x0000.0001  CID2  ARTPCelliD3, type RO, offset 0xFF6, reset 0x0000.0001  CID2  ARTPCelliD3, type RO, offset 0xFF6, reset 0x0000.0001  CID2  ARTPCelliD3, type RO, offset 0xFF6, reset 0x0000.0001  CID3  SCR0, type RW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type RW, offset 0x008, reset 0x0000.0000												PI	D6			
ARTPeriphiD0, type RO, offset 0xFE0, reset 0x0000.0010  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RO, offset 0xFE5, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE5, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE0, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE0, reset 0x0000.000D  ARTPCellID1, type RO, offset 0xFE4, reset 0x0000.000D  CID0  ARTPCellID2, type RO, offset 0xFE4, reset 0x0000.00F0  CID1  ARTPCellID3, type RO, offset 0xFE6, reset 0x0000.00E0  CID2  ARTPCellID3, type RO, offset 0xFE6, reset 0x0000.00B1  CID3  CID3  CID3  CID3  CID3  CID3  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000	UARTPeri	ohID7, type	RO, offse	t 0xFDC, re	set 0x0000	0.0000										
ARTPeriphiD0, type RO, offset 0xFE0, reset 0x0000.0010  ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RO, offset 0xFE5, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE5, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE0, reset 0x0000.0001  ARTPCellID0, type RO, offset 0xFE0, reset 0x0000.000D  ARTPCellID1, type RO, offset 0xFE4, reset 0x0000.000D  CID0  ARTPCellID2, type RO, offset 0xFE4, reset 0x0000.00F0  CID1  ARTPCellID3, type RO, offset 0xFE6, reset 0x0000.00E0  CID2  ARTPCellID3, type RO, offset 0xFE6, reset 0x0000.00B1  CID3  CID3  CID3  CID3  CID3  CID3  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000																
ARTPeriphiD1, type RO, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RO, offset 0xFE8, reset 0x0000.0018  ARTPeriphiD3, type RO, offset 0xFE6, reset 0x0000.0001  PID2  ARTPCellID0, type RO, offset 0xFEC, reset 0x0000.0000  CID0  ARTPCellID1, type RO, offset 0xFF6, reset 0x0000.000D  CID0  ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.0081  CID3  Ynchronous Serial Interface (SSI) SIO base: 0x4000.8000  SICR0, type RW, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type RW, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIOR, type RW, offset 0x008, reset 0x0000.0000												PI	D7			
ARTPeriphiD1, type RQ, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RQ, offset 0xFE8, reset 0x0000.0018  ARTPeriphiD3, type RQ, offset 0xFEC, reset 0x0000.0001  ARTPCellID0, type RQ, offset 0xFEC, reset 0x0000.0000  ARTPCellID1, type RQ, offset 0xFE4, reset 0x0000.0000  CID0  ARTPCellID2, type RQ, offset 0xFE4, reset 0x0000.0000  CID1  ARTPCellID3, type RQ, offset 0xFE4, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID3  CID4  CID5	UARTPeri	phID0, type	RO, offse	t 0xFE0, re	set 0x0000	.0011										
ARTPeriphiD1, type RQ, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RQ, offset 0xFE8, reset 0x0000.0018  ARTPeriphiD3, type RQ, offset 0xFEC, reset 0x0000.0001  ARTPCellID0, type RQ, offset 0xFEC, reset 0x0000.0000  ARTPCellID1, type RQ, offset 0xFE4, reset 0x0000.0000  CID0  ARTPCellID2, type RQ, offset 0xFE4, reset 0x0000.0000  CID1  ARTPCellID3, type RQ, offset 0xFE4, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID3  CID4  CID5			-													
ARTPeriphiD1, type RQ, offset 0xFE4, reset 0x0000.0000  ARTPeriphiD2, type RQ, offset 0xFE8, reset 0x0000.0018  ARTPeriphiD3, type RQ, offset 0xFEC, reset 0x0000.0001  ARTPCellID0, type RQ, offset 0xFEC, reset 0x0000.0000  ARTPCellID1, type RQ, offset 0xFE4, reset 0x0000.0000  CID0  ARTPCellID2, type RQ, offset 0xFE4, reset 0x0000.0000  CID1  ARTPCellID3, type RQ, offset 0xFE4, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID2  ARTPCellID3, type RQ, offset 0xFE6, reset 0x0000.0005  CID3  CID4  CID5												PI	ID0			
ARTPeriphiD2, type RO, offset 0xFE8, reset 0x0000.0018  PID2  ARTPeriphiD3, type RO, offset 0xFEC, reset 0x0000.0001  PID3  ARTPCellID0, type RO, offset 0xFF0, reset 0x0000.000D  ARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID0  ARTPCellID2, type RO, offset 0xFF4, reset 0x0000.00F0  CID2  ARTPCellID3, type RO, offset 0xFF6, reset 0x0000.00B1  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID4  CID5  CID5  CID5  CID5  CID5  CID6  CID7  CID	IIARTPoris	ohID1 type	PO offee	t OvEE4 ro	eat OvOOOO	0000										
ARTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018  ARTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001  PID3  ARTPCeIIID0, type RO, offset 0xFF0, reset 0x0000.000D  CID0  ARTPCeIIID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCeIIID2, type RO, offset 0xFF8, reset 0x0000.00F0  CID2  ARTPCeIIID3, type RO, offset 0xFF6, reset 0x0000.00B1  CID2  ARTPCeIIID3, type RO, offset 0xFF6, reset 0x0000.00B1  CID3  CID4  CID5  CID5  CID5  CID7  CID	UAKTEH	ливт, турс	r KO, Olise	T UXI L4, 16	Set uxuuuu	.0000										
ARTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018  ARTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001  PID3  ARTPCeIIID0, type RO, offset 0xFF0, reset 0x0000.000D  CID0  ARTPCeIIID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCeIIID2, type RO, offset 0xFF8, reset 0x0000.00F0  CID2  ARTPCeIIID3, type RO, offset 0xFF6, reset 0x0000.00B1  CID2  ARTPCeIIID3, type RO, offset 0xFF6, reset 0x0000.00B1  CID3  CID4  CID5  CID5  CID5  CID7  CID												DI	L			
ARTPCeIIID1, type RQ, offset 0xFEQ, reset 0x0000.0000  ARTPCeIIID1, type RQ, offset 0xFF4, reset 0x0000.00F0  CID0  ARTPCeIIID2, type RQ, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCeIIID3, type RQ, offset 0xFF4, reset 0x0000.00F0  CID2  ARTPCeIIID3, type RQ, offset 0xFF6, reset 0x0000.00B1  CID2  ARTPCeIIID3, type RQ, offset 0xFFC, reset 0x0000.00B1  CID3  CID3  CID3  CID3  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000												PI	וטו			
ARTPCellID0, type R0, offset 0xFEC, reset 0x0000.000D  ARTPCellID1, type R0, offset 0xFF0, reset 0x0000.000D  ARTPCellID1, type R0, offset 0xFF4, reset 0x0000.00F0  CID0  ARTPCellID2, type R0, offset 0xFF8, reset 0x0000.00F0  CID1  ARTPCellID3, type R0, offset 0xFF8, reset 0x0000.00B1  CID2  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.00B1  CID3  CID4  CID5	UARTPeri	ohID2, type	RO, offse	t 0xFE8, re	set 0x0000	.0018										
ARTPCellID0, type R0, offset 0xFEC, reset 0x0000.000D  ARTPCellID1, type R0, offset 0xFF0, reset 0x0000.000D  ARTPCellID1, type R0, offset 0xFF4, reset 0x0000.00F0  CID0  ARTPCellID2, type R0, offset 0xFF8, reset 0x0000.00F0  CID1  ARTPCellID3, type R0, offset 0xFF8, reset 0x0000.00B1  CID2  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.00B1  CID3  CID4  CID5																
ARTPCellID1, type R0, offset 0xFF0, reset 0x0000.000D  ARTPCellID1, type R0, offset 0xFF4, reset 0x0000.00F0  ARTPCellID2, type R0, offset 0xFF8, reset 0x0000.0005  ARTPCellID3, type R0, offset 0xFF6, reset 0x0000.0005  CID2  ARTPCellID3, type R0, offset 0xFFC, reset 0x0000.00B1  CID3  Type R0, offset 0xFFC, reset 0x0000.00B1  SICR0, type RW, offset 0x000, reset 0x0000.0000  SICR0, type RW, offset 0x004, reset 0x0000.0000  SICR1, type RW, offset 0x004, reset 0x0000.0000  SICR1, type RW, offset 0x004, reset 0x0000.0000  SIDR, type RW, offset 0x008, reset 0x0000.0000												PI	ID2			
ARTPCellID1, type R0, offset 0xFF0, reset 0x0000.000D  CID0  ARTPCellID1, type R0, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCellID2, type R0, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type R0, offset 0xFFC, reset 0x0000.00B1  CID2  ARTPCellID3, type R0, offset 0xFFC, reset 0x0000.00B1  CID3  CID4  CID5  C	UARTPeri	ohID3, type	RO, offse	t 0xFEC, re	set 0x0000	0.0001										
ARTPCellID1, type R0, offset 0xFF0, reset 0x0000.000D  CID0  ARTPCellID1, type R0, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCellID2, type R0, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type R0, offset 0xFFC, reset 0x0000.00B1  CID2  ARTPCellID3, type R0, offset 0xFFC, reset 0x0000.00B1  CID3  CID4  CID5  C																
CID0  ARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID4  CID5  CID5  CID5  CID5  CID5  CID5  CID5  CID5  CID5  CID6  CID7  CID												PI	D3			
ARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID	UARTPCel	IID0, type	RO, offset	0xFF0, rese	et 0x0000.0	000D										
ARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0  CID1  ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID																
ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID3  CYNChronous Serial Interface (SSI)  SIO base: 0x4000.8000  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000												CI	ID0			
ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID3  CYNChronous Serial Interface (SSI)  SIO base: 0x4000.8000  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000	UARTPCel	IID1, type	RO, offset	0xFF4, rese	et 0x0000.0	00F0										
ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID3				,												
ARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005  CID2  ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  CID3												CI	I ID1			
ARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1  CID3  Synchronous Serial Interface (SSI) SIO base: 0x4000.8000  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000	ΠΑΡΤΡΟΝ	IIID2 tupe	RO offect	OxFE8 room	et Oyooo o	0005			<u> </u>							
ARTPCelIID3, type RO, offset 0xFFC, reset 0x0000.00B1    CID3	UAKTECEI	IIIDZ, type	NO, Oliset	UXI I O, I est		,003 										
ARTPCelIID3, type RO, offset 0xFFC, reset 0x0000.00B1    CID3												01	ID0			
CID3												CI	ID2			
SID base: 0x4000.8000  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SIDR, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000	UARTPCel	IIID3, type	RO, offset	0xFFC, res	et 0x0000.	00B1										
SID base: 0x4000.8000  SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SIDR, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000																
SICR0, type R/W, offset 0x000, reset 0x0000.0000  SICR1, type R/W, offset 0x004, reset 0x0000.0000												CI	ID3			
SICR0, type R/W, offset 0x000, reset 0x0000.0000  SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000				erface (S	SSI)											
SCR SPH SPO FRF DSS  SICR1, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000	SSICR0, tv	pe R/W, of	fset 0x000	, reset 0x00	000.0000											
SICR1, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000																
SICR1, type R/W, offset 0x004, reset 0x0000.0000  SOD MS SSE LBM  SIDR, type R/W, offset 0x008, reset 0x0000.0000				SC	CR				SPH	SPO	FF	RF		D	SS	
SIDR, type R/W, offset 0x008, reset 0x0000.0000	SSICD4 #	ne P/M -4	feat Nynn4						1 5. 11	5. 0			1			
SIDR, type R/W, offset 0x008, reset 0x0000.0000	ooiori, ty	pe ivivi, oi	1361 03004	, reset uxut												
SIDR, type R/W, offset 0x008, reset 0x0000.0000													207	110	005	10
													SOD	MS	SSE	LBM
DATA	SSIDR, typ	e R/W, off	set 0x008,	reset 0x000	00.0000											
DATA																
								DA	ATA							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSISR, ty	pe RO, offs	et 0x00C,	reset 0x000	0.0003				1				1			
											BSY	RFF	RNE	TNF	TFE
SSICPSR	, type R/W, o	offset 0x0	10, reset 0x	0000.0000				1				1			
											CPS	DVSR			
SSIIM, typ	pe R/W, offs	et 0x014, ı	reset 0x000	0.0000				1				1			
												TXIM	RXIM	RTIM	RORIM
SSIRIS, ty	ype RO, offs	et 0x018,	reset 0x000	00.0008				1				1			
_Lcom_												TXRIS	RXRIS	RTRIS	RORRIS
SSIMIS, t	ype RO, offs	set 0x01C,	reset 0x00	00.0000				I				I			
												T)/1410	DVANO	DTM	DODLUG
201127												TXMIS	RXMIS	RTMIS	RORMIS
SSIICK, ty	ype W1C, of	tset uxu2u	), reset uxu	000.0000											
														DTIC	DODIC
	1544 5													RTIC	RORIC
SSIPeripi	nID4, type R	O, offset 0	0xFD0, rese	t 0x0000.00	000										
											DI	  D4			
	15- / 5										PI	ID4			
SSIPeripi	nID5, type R	O, offset C	DXFD4, rese	t 0x0000.00	J00			I							
											PI	ID5			
SSIPeripi	nID6, type R	O, offset 0	0xFD8, rese	t 0x0000.00	000			I				I			
											DI	 			
0010	.ID7 4 D	0 -554.6		4.00000	000						PI	D6			
SSIPeripr	nID7, type R	O, onset c	JXFDC, rese	T UXUUUU.UI	000										
											DI	  D7			
0010	-ID0 4 D	0 -554.6	\	4.00000.00	100						PI	D7			
SSIPeripi	nID0, type R	O, offset C	JXFEU, rese	t 0x0000.00	122										
											DI	 			
CCIDarial	-ID4 turns D	O offeet (	\v_FF4 ====	4.0~0000.00	100						PI	D0			
SSIPERIPI	nID1, type R	O, onset t	JXFE4, rese	UXUUUU.UU	)UU							1			
											DI	D1			
SSIDorink	ID2 type P	O offect (	VEES roso	+ 0~0000 00	118							101			
SSIFETIPI	nID2, type R	O, onset t	JAFEO, TESE		710										
											DI	  D2			
CCIDorink	nID3, type R	O offeet (	VEEC roos	+ 0×0000 0	001										
Someripi	ווטס, type R	o, onset (	AFEU, Fese	, UXUUUU.00 	UU I										
											Di	D3			
eelbe-iii	D0, type RO	offers C	EE0 ====	0×0000 000	n						PI	נטו			
SOIPCEIII	υυ, type RO	, onset 0x	rru, reset	UXUUUU.UUU 	U										
											0	ID0			
eeino-iii	D4 6:=== D2	offer t	EE4	020000 000	·n						CI	וטטו			
SSIPCEIII	D1, type RO	, orrset 0x	rr4, reset	UXUUUU.00F	·U										
											-	ID4			
					_						CI	ID1			
SSIPCelli	D2, type RO	), offset 0x	rF8, reset	0x0000.000	5										
											CI	ID2			

0.4				1 07		0.5		1 00	00			1 40	40	47	40
31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23	22 6	21 5	20	19 3	18	17	16 0
				0x0000.00E		3	0		0		7				0
	, ,,,,	,			-										
											CI	D3			
Inter-Int	egrated	Circuit	(I <sup>2</sup> C) Inte	erface											
I <sup>2</sup> C Mast			` '												
I2C Maste	er 0 base:	0x4002.0	0000												
I2CMSA, ty	pe R/W, of	fset 0x000	, reset 0x0	000.0000											
											SA				R/S
I2CMCS, ty	pe RO, off	set 0x004,	reset 0x00	000.0000								I			
									BUSBSY	IDLE	ARBLST	DATACK	ADRACK	ERROR	BUSY
J.com I2CMCS, ty	me WO of	Feat NyNNA	reset OvO	000 0000					BUSBS 1	IDLE	ARBLST	DATACK	ADRACK	ERRUR	возт
12011100, ty	, pe 110, oi	1361 02004	, 16361 020												
												ACK	STOP	START	RUN
I2CMDR, ty	/pe R/W, of	fset 0x008	3, reset 0x0	000.0000											
											DA	TA			
I2CMTPR,	type R/W,	offset 0x00	C, reset 0	k0000.0001											
											TF	PR			
I2CMIMR, t	ype R/W, c	offset 0x01	0, reset 0x	0000.0000											
															IM
I2CMRIS, t	vpe RO. of	fset 0x014	. reset 0x0	000.0000											IIVI
,	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		,												
															RIS
I2CMMIS, t	ype RO, of	fset 0x018	, reset 0x0	000.0000											
															MIS
I2CMICR, t	ype WO, o	ffset 0x010	C, reset 0x0	0000.0000											
IOCMOD &	D/M	0×000	) ====± 0×0	000 0000											IC
I2CMCR, ty	/pe R/vv, o	iset uxuzu	, reset uxu												
										SFE	MFE				LPBK
Inter-Int	egrated	Circuit	(I <sup>2</sup> C) Inte	erface								<u> </u>			
I <sup>2</sup> C Slav		on our	(1 0)	311430											
I2C Slave		x4002.08	300												
I2CSOAR,	type R/W,	offset 0x00	00, reset 0x	0000.0000											
												OAR			
I2CSCSR, 1	type RO, o	ffset 0x004	4, reset 0x0	0000.0000											
1200000	h.m. 1440	ff4 0:-00	4 =====================================	0000 0000									FBR	TREQ	RREQ
I2CSCSR, 1	type WO, d	niset uxuu	4, reset ux	UUUU.UUUU 											
															DA
IOOODD 4	pe R/W. of	fset 0x008	, reset 0x0	000.0000											DA
IZCSDR. TV															
IZCSDR, ty															

				07		0.5		1 00		- 04		10	10		10
31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20	19 3	18	17	16 0
			C, reset 0x0		10	9	0	,	0	3	4	] 3	2		0
izooniiit,	type rave,	JIISEL UXUU	o, reset ox												
															IM
I2CSRIS,	type RO, o	ffset 0x010	, reset 0x00	000.0000											
															RIS
I2CSMIS,	type RO, o	ffset 0x014	, reset 0x00	000.0000											
															MIS
I2CSICR,	type WO, o	ffset 0x018	, reset 0x0	000.000											
Lcom															IC
Contro	ller Area	Networ	k (CAN)	Module											
CAN0 ba	ase: 0x400	04.0000													
CANCTL,	type R/W,	offset 0x00	0, reset 0x0	0000.0001											
								Test	CCE	DAR		EIE	SIE	IE	INIT
CANSTS,	type R/W,	offset 0x00	4, reset 0x0	0000.0000											
								BOff	EWarn	EPass	RxOK	TxOK		LEC	
CANEDD	tuno BO d	effect 0v00	3, reset 0x0	000 0000				BOII	Evvaiii	EPass	RXUK	IXOK		LEC	
CANERR,	, type KO, t	JIISEL UXUUC	o, reset uxu												
RP				REC							TI	I EC			
	type R/W. o	ffset 0x00C	, reset 0x0												
	1		,												
		TSeg2			TS	eg1		SJ	W			В	RP		
CANINT, t	type RO, of	fset 0x010,	reset 0x00	00.0000											
							lr	ntld				•			
CANTST,	type R/W,	offset 0x014	4, reset 0x0	0000.0000											
								Rx	٦	x	LBack	Silent	Basic		
CANBRPI	E, type R/W	, offset 0x0	)18, reset 0:	x0000.0000											
													BRI	PE	
CANIF1C	RQ, type R	O, offset 0x	(020, reset (	0x0000.0001	<u> </u>										
Duest													IUM		
Busy	PO tuno P	O officet fly	1000 rooot (	0x0000.0001								IVII	IOW		
CANIFZU	κω, type K	o, onset 0x	oou, reset	0.0000.000	•										
Busy												MN	IUM		
	MSK. type	RO, offset (	0x024. rese	t 0x0000.00	00							1411			
	, -, po	_,													
								WRNRD	Mask	Arb	Control	CirintPnd	TxRqstNewDat	DataA	DataB
CANIF2C	MSK, type	RO, offset (	0x084, rese	t 0x0000.00	00				<u> </u>						1
								WRNRD	Mask	Arb	Control	CirintPnd	TxRqstNewDat	DataA	DataB
CANIF1M	ISK1, type I	RO, offset 0	x028, reset	t 0x0000.FFI	FF										
							N	/sk		<u> </u>	<u> </u>	·			
CANIF2M	SK1, type I	RO, offset 0	)x088, reset	t 0x0000.FFI	FF		N	/Isk							
CANIF2M	ISK1, type I	RO, offset 0	0x088, reset	t 0x0000.FFI	FF		N	/Isk							

November 29, 2007 479 Preliminary

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANIF1M	SK2, type R	O, offset 0	0x02C, rese	t 0x0000.FF	FF										
MXtd	MDir								Msk						
CANIF2M	SK2, type R	O, offset 0	x08C, rese	t 0x0000.FI	FF										
MXtd	MDir								Msk						
CANIF1A	RB1, type R	O, offset 0	)x030, reset	0x0000.00	00										
							I	D							
CANIF2A	RB1, type R	O, offset 0	)x090, reset	0x0000.00	00			ı							
Lcom_							I	D							
CANIF1A	RB2, type R	O, offset 0	0x034, reset	0x0000.00	00			I							
MsgVal	Xtd	Dir							ID						
CANIF2A	RB2, type R	U, offset 0	xu94, reset	UXUUU0.00	UU										
Mag\/al	Vtd	Dia							ID						
MsgVal	Xtd	Dir			•				ID						
CANIFIN	CTL, type R	o, onset t	JXU36, rese	UXUUUU.UU	00										
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB					DI	_C	
	CTL, type R					TAIRLI	TXINGSU	LOD							
CAIVII ZIVI	CTE, type N	o, onset c	7,030, 1636	0.00000.00	00										
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB					DI	_C	
	A1, type R/V					Tuncen	TATEGOT	Lob				<u> </u>			
OAIIII ID	A1, type IV	v, 011361 02	, 1030, 10301	0.0000.00	<b>,</b>										
							Da	l ata							
CANIF1D	A2, type R/V	V. offset 0:	x040. reset	0x0000.000	10										
	, 1,	.,	,		· ·										
							Da	ı ata							
CANIF1D	B1, type R/V	V, offset 0:	x044, reset	0x0000.000	10										
		<u>,                                      </u>	,												
							Da	ata							
CANIF1D	B2, type R/V	V, offset 0:	x048, reset	0x0000.000	10										
			1			1	Da	ata				1	1		
CANIF2D	A1, type R/V	V, offset 0	x09C, reset	0x0000.00	00										
							Da	ata							
CANIF2D	A2, type R/V	V, offset 0	x0A0, reset	0x0000.00	00										
							Da	ata							
CANIF2D	B1, type R/V	V, offset 0	x0A4, reset	0x0000.00	00										
							Da	ata							
CANIF2D	B2, type R/V	V, offset 0	x0A8, reset	0x0000.00	00										
							Da	ata							
CANTXR	Q1, type RO	, offset 0x	100, reset 0	x0000.0000	)										
							TxF	Rqst							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CANTXRO	2, type RO	, offset 0x	104, reset 0	x0000.0000	)			1							
							Tx	Rqst							
CANNWD	A1, type R0	D, offset 0x	(120, reset	0x0000.000	0										
							Ne	wDat				•			
CANNWD	A2, type R0	O, offset 0x	(124, reset	0x0000.000	0										
							Ne	wDat				•			
CANMSG1	IINT, type F	RO, offset (	0x140, rese	t 0x0000.00	00										
Lcom							Int	Pnd							
CANMSG2	2INT, type F	RO, offset (	0x144, rese	t 0x0000.00	00										
							Int	Pnd							
CANMSG1	IVAL, type	RO, offset	0x160, res	et 0x0000.00	000										
							Ms	gVal							
CANMSG2	2VAL, type	RO, offset	0x164, rese	et 0x0000.00	000										
							Ms	gVal							
	Compar														
Base 0x4	003.C000	)													
ACMIS, ty	pe R/W1C,	offset 0x0	0, reset 0x0	000.0000											
													IN2	IN1	IN0
ACRIS, typ	pe RO, offs	et 0x04, re	set 0x0000	.0000											
													IN2	IN1	IN0
ACINTEN,	type R/W,	offset 0x08	8, reset 0x0	000.0000											
													INIO	INIA	INIO
			10 10										IN2	IN1	IN0
ACREFUI	L, type R/V	v, offset ux	(10, reset 0)	x0000.0000								1			
						EN	RNG						\/[	REF	
ACSTATO	tuno BO o	effoot 0v20	, reset 0x00	00 0000		EIN	KNG						VF	NEF	
ACSTATU,	туре ко, с	JIISEL UXZU	, reset uxuu	00.000											
														OVAL	
ACSTAT1	type RO o	offset 0x40	, reset 0x00	000.0000											
	-,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		,	2.2.20											
														OVAL	
ACSTAT2.	type RO. c	offset 0x60	, reset 0x00	000.0000				I							
	J		,												
														OVAL	
ACCTL0. t	type R/W, o	ffset 0x24.	, reset 0x00	00.0000				I							
					ASF	RCP					ISLVAL	15	SEN	CINV	
ACCTL1, t	type R/W, o	ffset 0x44.	, reset 0x00	00.000				1				1			
					ASF	RCP					ISLVAL	15	SEN	CINV	
ACCTL2. t	type R/W, o	ffset 0x64.	, reset 0x00	00.0000											
,	, ,	,													
					ASF	RCP					ISLVAL	18	SEN	CINV	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	17	0
Pulse V	Vidth Mo	dulator	(PWM)												
Base 0x4	002.8000														
PWMCTL,	type R/W,	offset 0x00	00, reset 0x0	0000.0000											
															GlobalSync0
PWMSYN	C, type R/M	, offset 0x	004, reset 0	x0000.000	0										Globaloyno
	, ,,	<u>,                                      </u>													
															Sync0
PWMENA	BLE, type F	R/W, offset	0x008, rese	et 0x0000.0	000							1			
														DWM1En	PWM0En
PWMINVE	RT. type R/	W. offset (	0x00C, reset	t 0x0000.00	000									FVVIVITLII	FVVIVIOLII
J.COM1 -	, ., ,	,													
														PWM1Inv	PWM0Inv
PWMFAUL	_T, type R/V	V, offset 0	x010, reset (	0x0000.000	0										
														F144	F140
DWMINITE	N type P/V	V offect ()	x014, reset (	220000 000	<u> </u>									Fault1	Fault0
	it, type itt	, onset 07	1014, 16361	7,0000.000											IntFault
															IntPWM0
PWMRIS,	type RO, o	fset 0x018	3, reset 0x00	000.000											
															IntFault
DIAMAGO		- # 4 O	.040	00000 000	<u> </u>										IntPWM0
PWWISC,	type R/W10	, onset u	x01C, reset	UXUUUU.UU	JU										IntFault
															IntPWM0
PWMSTAT	US, type R	O, offset 0	0x020, reset	0x0000.00	00										
															Fault
PWM0CTL	_, type R/W	offset 0x0	040, reset 0	k0000.0000											
										CmpBUpd	CmpAUpd	LoadUpd	Debug	Mode	Enable
PWM0INT	EN, type R/	W, offset (	0x044, reset	0x0000.00	100			1							
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0RIS	, type RO, o	offset 0x04	48, reset 0x0	0000.0000				1							
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0ISC	, type R/W1	C, offset (	0x04C, reset	t 0x0000.00	000										
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0LOA	AD, type R/	W, offset 0	0x050, reset	0x0000.00	00			1							
							L	oad							
PWM0COI	UNT, type F	O, offset (	0x054, reset	0x0000.00	000										
							Co	ount							
PWM0CMI	PA, type R/	W, offset 0	0x058, reset	0x0000.00	00										
							Co	mpA							
PWM0CMI	PB, type R/	W, offset (	0x05C, reset	0x0000.00	000			шргч							
	, .yp = 10	,	, , , , ,		-										
							Co	mpB							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PWM0GEN	NA, type R/	W, offset 0	x060, reset	0x0000.00	00											
				ActCmpBD		ActCı	ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM0GEN	NB, type R/	W, offset 0	x064, reset	0x0000.00	00											
				ActCmpBD		ActCmpBU		ActC	ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM0DB0	CTL, type F	R/W, offset	0x068, rese	t 0x0000.0	000											
															Enable	
PWMODRE	RISE type	R/W offset	0x06C, res	et 0x0000	0000										2.10010	
1 WINODDI	tio⊑, type	ivvi, onset	0,000,163		0000											
LLcom									Rise	Delay						
PWM0DBF	ALL, type	R/W, offset	t 0x070, res	et 0x0000.	0000											
				FallDelay												

# C Ordering and Contact Information

### C.1 Ordering Information

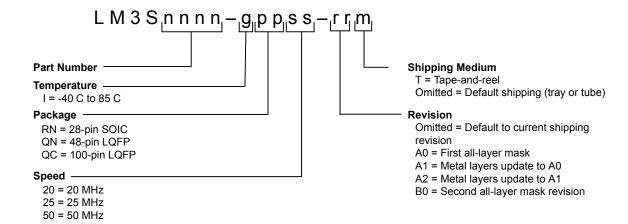


Table C-1. Part Ordering Information

Orderable Part Number	Description
LM3S2110-IQC25	Stellaris® LM3S2110 Microcontroller
LM3S2110-IQC25(T)	Stellaris <sup>®</sup> LM3S2110 Microcontroller

#### C.2 Kits

The Luminary Micro Stellaris<sup>®</sup> Family provides the hardware and software tools that engineers need to begin development quickly.

- Reference Design Kits accelerate product development by providing ready-to-run hardware, and comprehensive documentation including hardware design files:
  - http://www.luminarymicro.com/products/reference\_design\_kits/
- Evaluation Kits provide a low-cost and effective means of evaluating Stellaris® microcontrollers before purchase:
  - http://www.luminarymicro.com/products/evaluation kits/
- Development Kits provide you with all the tools you need to develop and prototype embedded applications right out of the box:
  - http://www.luminarymicro.com/products/boards.html

See the Luminary Micro website for the latest tools available or ask your Luminary Micro distributor.

## C.3 Company Information

Luminary Micro, Inc. designs, markets, and sells ARM Cortex-M3-based microcontrollers (MCUs). Austin, Texas-based Luminary Micro is the lead partner for the Cortex-M3 processor, delivering the world's first silicon implementation of the Cortex-M3 processor. Luminary Micro's introduction of the

Stellaris® family of products provides 32-bit performance for the same price as current 8- and 16-bit microcontroller designs. With entry-level pricing at \$1.00 for an ARM technology-based MCU, Luminary Micro's Stellaris product line allows for standardization that eliminates future architectural upgrades or software tool changes.

Luminary Micro, Inc. 108 Wild Basin, Suite 350 Austin, TX 78746 Main: +1-512-279-8800 Fax: +1-512-279-8879 http://www.luminarymicro.com sales@luminarymicro.com

### www.DataSheet4 C:4 Support Information

For support on Luminary Micro products, contact: support@luminarymicro.com +1-512-279-8800, ext. 3