




Stellaris® LM3S1960 Microcontroller

DATA SHEET

Copyright

Copyright © 2007-2009 Texas Instruments Incorporated All rights reserved. Stellaris and StellarisWare are registered trademarks of Texas Instruments Incorporated. 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.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

 Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

Texas Instruments Incorporated
108 Wild Basin, Suite 350
Austin, TX 78746
Main: +1-512-279-8800
Fax: +1-512-279-8879
<http://www.luminarymicro.com>



Table of Contents

Revision History	18
About This Document	21
Audience	21
About This Manual	21
Related Documents	21
Documentation Conventions	21
1 Architectural Overview	24
1.1 Product Features	24
1.2 Target Applications	31
1.3 High-Level Block Diagram	32
1.4 Functional Overview	34
1.4.1 ARM Cortex™-M3	34
1.4.2 Motor Control Peripherals	35
1.4.3 Analog Peripherals	35
1.4.4 Serial Communications Peripherals	36
1.4.5 System Peripherals	37
1.4.6 Memory Peripherals	38
1.4.7 Additional Features	38
1.4.8 Hardware Details	39
2 ARM Cortex-M3 Processor Core	40
2.1 Block Diagram	41
2.2 Functional Description	41
2.2.1 Serial Wire and JTAG Debug	41
2.2.2 Embedded Trace Macrocell (ETM)	42
2.2.3 Trace Port Interface Unit (TPIU)	42
2.2.4 ROM Table	42
2.2.5 Memory Protection Unit (MPU)	42
2.2.6 Nested Vectored Interrupt Controller (NVIC)	42
3 Memory Map	46
4 Interrupts	49
5 JTAG Interface	52
5.1 Block Diagram	53
5.2 Functional Description	53
5.2.1 JTAG Interface Pins	53
5.2.2 JTAG TAP Controller	55
5.2.3 Shift Registers	56
5.2.4 Operational Considerations	56
5.3 Initialization and Configuration	59
5.4 Register Descriptions	59
5.4.1 Instruction Register (IR)	59
5.4.2 Data Registers	61
6 System Control	64
6.1 Functional Description	64
6.1.1 Device Identification	64

6.1.2	Reset Control	64
6.1.3	Power Control	67
6.1.4	Clock Control	68
6.1.5	System Control	72
6.2	Initialization and Configuration	73
6.3	Register Map	73
6.4	Register Descriptions	74
7	Hibernation Module	128
7.1	Block Diagram	129
7.2	Functional Description	129
7.2.1	Register Access Timing	129
7.2.2	Clock Source	130
7.2.3	Battery Management	131
7.2.4	Real-Time Clock	131
7.2.5	Non-Volatile Memory	132
7.2.6	Power Control	132
7.2.7	Initiating Hibernate	132
7.2.8	Interrupts and Status	133
7.3	Initialization and Configuration	133
7.3.1	Initialization	133
7.3.2	RTC Match Functionality (No Hibernation)	133
7.3.3	RTC Match/Wake-Up from Hibernation	134
7.3.4	External Wake-Up from Hibernation	134
7.3.5	RTC/External Wake-Up from Hibernation	134
7.4	Register Map	134
7.5	Register Descriptions	135
8	Internal Memory	148
8.1	Block Diagram	148
8.2	Functional Description	148
8.2.1	SRAM Memory	148
8.2.2	Flash Memory	149
8.3	Flash Memory Initialization and Configuration	150
8.3.1	Flash Programming	150
8.3.2	Nonvolatile Register Programming	151
8.4	Register Map	152
8.5	Flash Register Descriptions (Flash Control Offset)	152
8.6	Flash Register Descriptions (System Control Offset)	159
9	General-Purpose Input/Outputs (GPIOs)	172
9.1	Functional Description	172
9.1.1	Data Control	173
9.1.2	Interrupt Control	174
9.1.3	Mode Control	175
9.1.4	Commit Control	175
9.1.5	Pad Control	175
9.1.6	Identification	175
9.2	Initialization and Configuration	175
9.3	Register Map	177
9.4	Register Descriptions	178

10	General-Purpose Timers	213
10.1	Block Diagram	214
10.2	Functional Description	214
10.2.1	GPTM Reset Conditions	215
10.2.2	32-Bit Timer Operating Modes	215
10.2.3	16-Bit Timer Operating Modes	216
10.3	Initialization and Configuration	220
10.3.1	32-Bit One-Shot/Periodic Timer Mode	220
10.3.2	32-Bit Real-Time Clock (RTC) Mode	221
10.3.3	16-Bit One-Shot/Periodic Timer Mode	221
10.3.4	16-Bit Input Edge Count Mode	222
10.3.5	16-Bit Input Edge Timing Mode	222
10.3.6	16-Bit PWM Mode	223
10.4	Register Map	223
10.5	Register Descriptions	224
11	Watchdog Timer	249
11.1	Block Diagram	250
11.2	Functional Description	250
11.3	Initialization and Configuration	251
11.4	Register Map	251
11.5	Register Descriptions	252
12	Universal Asynchronous Receivers/Transmitters (UARTs)	273
12.1	Block Diagram	274
12.2	Functional Description	274
12.2.1	Transmit/Receive Logic	274
12.2.2	Baud-Rate Generation	275
12.2.3	Data Transmission	276
12.2.4	Serial IR (SIR)	276
12.2.5	FIFO Operation	277
12.2.6	Interrupts	277
12.2.7	Loopback Operation	278
12.2.8	IrDA SIR block	278
12.3	Initialization and Configuration	278
12.4	Register Map	279
12.5	Register Descriptions	280
13	Synchronous Serial Interface (SSI)	314
13.1	Block Diagram	314
13.2	Functional Description	315
13.2.1	Bit Rate Generation	315
13.2.2	FIFO Operation	315
13.2.3	Interrupts	315
13.2.4	Frame Formats	316
13.3	Initialization and Configuration	323
13.4	Register Map	324
13.5	Register Descriptions	325
14	Inter-Integrated Circuit (I²C) Interface	351
14.1	Block Diagram	352

14.2	Functional Description	352
14.2.1	I ² C Bus Functional Overview	352
14.2.2	Available Speed Modes	354
14.2.3	Interrupts	355
14.2.4	Loopback Operation	356
14.2.5	Command Sequence Flow Charts	356
14.3	Initialization and Configuration	363
14.4	Register Map	364
14.5	Register Descriptions (I ² C Master)	365
14.6	Register Descriptions (I ² C Slave)	378
15	Analog Comparators	387
15.1	Block Diagram	388
15.2	Functional Description	388
15.2.1	Internal Reference Programming	389
15.3	Initialization and Configuration	390
15.4	Register Map	390
15.5	Register Descriptions	391
16	Pulse Width Modulator (PWM)	399
16.1	Block Diagram	400
16.2	Functional Description	401
16.2.1	PWM Timer	401
16.2.2	PWM Comparators	401
16.2.3	PWM Signal Generator	402
16.2.4	Dead-Band Generator	403
16.2.5	Interrupt Selector	403
16.2.6	Synchronization Methods	404
16.2.7	Fault Conditions	404
16.2.8	Output Control Block	404
16.3	Initialization and Configuration	404
16.4	Register Map	405
16.5	Register Descriptions	407
17	Quadrature Encoder Interface (QEI)	436
17.1	Block Diagram	436
17.2	Functional Description	437
17.3	Initialization and Configuration	439
17.4	Register Map	440
17.5	Register Descriptions	440
18	Pin Diagram	453
19	Signal Tables	455
19.1	100-Pin LQFP Package Pin Tables	455
19.2	108-Pin BGA Package Pin Tables	468
20	Operating Characteristics	483
21	Electrical Characteristics	484
21.1	DC Characteristics	484
21.1.1	Maximum Ratings	484
21.1.2	Recommended DC Operating Conditions	484

21.1.3	On-Chip Low Drop-Out (LDO) Regulator Characteristics	485
21.1.4	GPIO Module Characteristics	485
21.1.5	Power Specifications	485
21.1.6	Flash Memory Characteristics	487
21.1.7	Hibernation	487
21.2	AC Characteristics	487
21.2.1	Load Conditions	487
21.2.2	Clocks	488
21.2.3	JTAG and Boundary Scan	489
21.2.4	Reset	491
21.2.5	Hibernation Module	492
21.2.6	General-Purpose I/O (GPIO)	493
21.2.7	Synchronous Serial Interface (SSI)	494
21.2.8	Inter-Integrated Circuit (I ² C) Interface	495
21.2.9	Analog Comparator	496
A	Serial Flash Loader	497
A.1	Serial Flash Loader	497
A.2	Interfaces	497
A.2.1	UART	497
A.2.2	SSI	497
A.3	Packet Handling	498
A.3.1	Packet Format	498
A.3.2	Sending Packets	498
A.3.3	Receiving Packets	498
A.4	Commands	499
A.4.1	COMMAND_PING (0X20)	499
A.4.2	COMMAND_GET_STATUS (0x23)	499
A.4.3	COMMAND_DOWNLOAD (0x21)	499
A.4.4	COMMAND_SEND_DATA (0x24)	500
A.4.5	COMMAND_RUN (0x22)	500
A.4.6	COMMAND_RESET (0x25)	500
B	Register Quick Reference	502
C	Ordering and Contact Information	519
C.1	Ordering Information	519
C.2	Part Markings	519
C.3	Kits	520
C.4	Support Information	520
D	Package Information	521

List of Figures

Figure 1-1.	Stellaris® LM3S1960 Microcontroller High-Level Block Diagram	33
Figure 2-1.	CPU Block Diagram	41
Figure 2-2.	TPIU Block Diagram	42
Figure 5-1.	JTAG Module Block Diagram	53
Figure 5-2.	Test Access Port State Machine	56
Figure 5-3.	IDCODE Register Format	62
Figure 5-4.	BYPASS Register Format	62
Figure 5-5.	Boundary Scan Register Format	63
Figure 6-1.	Basic \overline{RST} Configuration	65
Figure 6-2.	External Circuitry to Extend Power-On Reset	66
Figure 6-3.	Reset Circuit Controlled by Switch	66
Figure 6-4.	Power Architecture	68
Figure 6-5.	Main Clock Tree	70
Figure 7-1.	Hibernation Module Block Diagram	129
Figure 7-2.	Clock Source Using Crystal	130
Figure 7-3.	Clock Source Using Dedicated Oscillator	131
Figure 8-1.	Flash Block Diagram	148
Figure 9-1.	GPIO Port Block Diagram	173
Figure 9-2.	GPIO DATA Write Example	174
Figure 9-3.	GPIO DATA Read Example	174
Figure 10-1.	GPTM Module Block Diagram	214
Figure 10-2.	16-Bit Input Edge Count Mode Example	218
Figure 10-3.	16-Bit Input Edge Time Mode Example	219
Figure 10-4.	16-Bit PWM Mode Example	220
Figure 11-1.	WDT Module Block Diagram	250
Figure 12-1.	UART Module Block Diagram	274
Figure 12-2.	UART Character Frame	275
Figure 12-3.	IrDA Data Modulation	277
Figure 13-1.	SSI Module Block Diagram	314
Figure 13-2.	TI Synchronous Serial Frame Format (Single Transfer)	317
Figure 13-3.	TI Synchronous Serial Frame Format (Continuous Transfer)	317
Figure 13-4.	Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0	318
Figure 13-5.	Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0	318
Figure 13-6.	Freescale SPI Frame Format with SPO=0 and SPH=1	319
Figure 13-7.	Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0	320
Figure 13-8.	Freescale SPI Frame Format (Continuous Transfer) with SPO=1 and SPH=0	320
Figure 13-9.	Freescale SPI Frame Format with SPO=1 and SPH=1	321
Figure 13-10.	MICROWIRE Frame Format (Single Frame)	322
Figure 13-11.	MICROWIRE Frame Format (Continuous Transfer)	323
Figure 13-12.	MICROWIRE Frame Format, SSIFss Input Setup and Hold Requirements	323
Figure 14-1.	I ² C Block Diagram	352
Figure 14-2.	I ² C Bus Configuration	352
Figure 14-3.	START and STOP Conditions	353
Figure 14-4.	Complete Data Transfer with a 7-Bit Address	353
Figure 14-5.	R/S Bit in First Byte	353

Figure 14-6.	Data Validity During Bit Transfer on the I ² C Bus	354
Figure 14-7.	Master Single SEND	357
Figure 14-8.	Master Single RECEIVE	358
Figure 14-9.	Master Burst SEND	359
Figure 14-10.	Master Burst RECEIVE	360
Figure 14-11.	Master Burst RECEIVE after Burst SEND	361
Figure 14-12.	Master Burst SEND after Burst RECEIVE	362
Figure 14-13.	Slave Command Sequence	363
Figure 15-1.	Analog Comparator Module Block Diagram	388
Figure 15-2.	Structure of Comparator Unit	389
Figure 15-3.	Comparator Internal Reference Structure	389
Figure 16-1.	PWM Unit Diagram	400
Figure 16-2.	PWM Module Block Diagram	401
Figure 16-3.	PWM Count-Down Mode	402
Figure 16-4.	PWM Count-Up/Down Mode	402
Figure 16-5.	PWM Generation Example In Count-Up/Down Mode	403
Figure 16-6.	PWM Dead-Band Generator	403
Figure 17-1.	QEI Block Diagram	437
Figure 17-2.	Quadrature Encoder and Velocity Predivider Operation	438
Figure 18-1.	100-Pin LQFP Package Pin Diagram	453
Figure 18-2.	108-Ball BGA Package Pin Diagram (Top View)	454
Figure 21-1.	Load Conditions	488
Figure 21-2.	JTAG Test Clock Input Timing	490
Figure 21-3.	JTAG Test Access Port (TAP) Timing	490
Figure 21-4.	JTAG TRST Timing	491
Figure 21-5.	External Reset Timing (\overline{RST})	491
Figure 21-6.	Power-On Reset Timing	492
Figure 21-7.	Brown-Out Reset Timing	492
Figure 21-8.	Software Reset Timing	492
Figure 21-9.	Watchdog Reset Timing	492
Figure 21-10.	Hibernation Module Timing	493
Figure 21-11.	SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement	494
Figure 21-12.	SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer	494
Figure 21-13.	SSI Timing for SPI Frame Format (FRF=00), with SPH=1	495
Figure 21-14.	I ² C Timing	496
Figure D-1.	100-Pin LQFP Package	521
Figure D-2.	108-Ball BGA Package	523

List of Tables

Table 1.	Revision History	18
Table 2.	Documentation Conventions	21
Table 3-1.	Memory Map	46
Table 4-1.	Exception Types	49
Table 4-2.	Interrupts	50
Table 5-1.	JTAG Port Pins Reset State	54
Table 5-2.	JTAG Instruction Register Commands	59
Table 6-1.	System Control Register Map	73
Table 7-1.	Hibernation Module Register Map	134
Table 8-1.	Flash Protection Policy Combinations	149
Table 8-2.	User-Programmable Flash Memory Resident Registers	151
Table 8-3.	Flash Register Map	152
Table 9-1.	GPIO Pad Configuration Examples	176
Table 9-2.	GPIO Interrupt Configuration Example	176
Table 9-3.	GPIO Register Map	177
Table 10-1.	Available CCP Pins	214
Table 10-2.	16-Bit Timer With Prescaler Configurations	217
Table 10-3.	Timers Register Map	223
Table 11-1.	Watchdog Timer Register Map	251
Table 12-1.	UART Register Map	279
Table 13-1.	SSI Register Map	324
Table 14-1.	Examples of I ² C Master Timer Period versus Speed Mode	355
Table 14-2.	Inter-Integrated Circuit (I ² C) Interface Register Map	364
Table 14-3.	Write Field Decoding for I2CMCS[3:0] Field (Sheet 1 of 3)	369
Table 15-1.	Internal Reference Voltage and ACREFLCTL Field Values	389
Table 15-2.	Analog Comparators Register Map	391
Table 16-1.	PWM Register Map	405
Table 17-1.	QEI Register Map	440
Table 19-1.	Signals by Pin Number	455
Table 19-2.	Signals by Signal Name	459
Table 19-3.	Signals by Function, Except for GPIO	464
Table 19-4.	GPIO Pins and Alternate Functions	466
Table 19-5.	Signals by Pin Number	468
Table 19-6.	Signals by Signal Name	473
Table 19-7.	Signals by Function, Except for GPIO	477
Table 19-8.	GPIO Pins and Alternate Functions	480
Table 20-1.	Temperature Characteristics	483
Table 20-2.	Thermal Characteristics	483
Table 21-1.	Maximum Ratings	484
Table 21-2.	Recommended DC Operating Conditions	484
Table 21-3.	LDO Regulator Characteristics	485
Table 21-4.	GPIO Module DC Characteristics	485
Table 21-5.	Detailed Power Specifications	486
Table 21-6.	Flash Memory Characteristics	487
Table 21-7.	Hibernation Module DC Characteristics	487
Table 21-8.	Phase Locked Loop (PLL) Characteristics	488

Table 21-9.	Actual PLL Frequency	488
Table 21-10.	Clock Characteristics	488
Table 21-11.	Crystal Characteristics	489
Table 21-12.	JTAG Characteristics	489
Table 21-13.	Reset Characteristics	491
Table 21-14.	Hibernation Module AC Characteristics	493
Table 21-15.	GPIO Characteristics	493
Table 21-16.	SSI Characteristics	494
Table 21-17.	I ² C Characteristics	495
Table 21-18.	Analog Comparator Characteristics	496
Table 21-19.	Analog Comparator Voltage Reference Characteristics	496
Table C-1.	Part Ordering Information	519

List of Registers

System Control	64
Register 1: Device Identification 0 (DID0), offset 0x000	75
Register 2: Brown-Out Reset Control (PBORCTL), offset 0x030	77
Register 3: LDO Power Control (LDOPCTL), offset 0x034	78
Register 4: Raw Interrupt Status (RIS), offset 0x050	79
Register 5: Interrupt Mask Control (IMC), offset 0x054	80
Register 6: Masked Interrupt Status and Clear (MISC), offset 0x058	81
Register 7: Reset Cause (RESC), offset 0x05C	82
Register 8: Run-Mode Clock Configuration (RCC), offset 0x060	83
Register 9: XTAL to PLL Translation (PLLCFG), offset 0x064	88
Register 10: Run-Mode Clock Configuration 2 (RCC2), offset 0x070	89
Register 11: Deep Sleep Clock Configuration (DSLPCCLKCFG), offset 0x144	91
Register 12: Device Identification 1 (DID1), offset 0x004	92
Register 13: Device Capabilities 0 (DC0), offset 0x008	94
Register 14: Device Capabilities 1 (DC1), offset 0x010	95
Register 15: Device Capabilities 2 (DC2), offset 0x014	97
Register 16: Device Capabilities 3 (DC3), offset 0x018	99
Register 17: Device Capabilities 4 (DC4), offset 0x01C	101
Register 18: Run Mode Clock Gating Control Register 0 (RCGC0), offset 0x100	103
Register 19: Sleep Mode Clock Gating Control Register 0 (SCGC0), offset 0x110	105
Register 20: Deep Sleep Mode Clock Gating Control Register 0 (DCGC0), offset 0x120	107
Register 21: Run Mode Clock Gating Control Register 1 (RCGC1), offset 0x104	109
Register 22: Sleep Mode Clock Gating Control Register 1 (SCGC1), offset 0x114	112
Register 23: Deep Sleep Mode Clock Gating Control Register 1 (DCGC1), offset 0x124	115
Register 24: Run Mode Clock Gating Control Register 2 (RCGC2), offset 0x108	118
Register 25: Sleep Mode Clock Gating Control Register 2 (SCGC2), offset 0x118	120
Register 26: Deep Sleep Mode Clock Gating Control Register 2 (DCGC2), offset 0x128	122
Register 27: Software Reset Control 0 (SRCR0), offset 0x040	124
Register 28: Software Reset Control 1 (SRCR1), offset 0x044	125
Register 29: Software Reset Control 2 (SRCR2), offset 0x048	127
Hibernation Module	128
Register 1: Hibernation RTC Counter (HIBRTCC), offset 0x000	136
Register 2: Hibernation RTC Match 0 (HIBRTCM0), offset 0x004	137
Register 3: Hibernation RTC Match 1 (HIBRTCM1), offset 0x008	138
Register 4: Hibernation RTC Load (HIBRTCLD), offset 0x00C	139
Register 5: Hibernation Control (HIBCTL), offset 0x010	140
Register 6: Hibernation Interrupt Mask (HIBIM), offset 0x014	142
Register 7: Hibernation Raw Interrupt Status (HIBRIS), offset 0x018	143
Register 8: Hibernation Masked Interrupt Status (HIBMIS), offset 0x01C	144
Register 9: Hibernation Interrupt Clear (HIBIC), offset 0x020	145
Register 10: Hibernation RTC Trim (HIBRTCT), offset 0x024	146
Register 11: Hibernation Data (HIBDATA), offset 0x030-0x12C	147
Internal Memory	148
Register 1: Flash Memory Address (FMA), offset 0x000	153
Register 2: Flash Memory Data (FMD), offset 0x004	154

Register 3:	Flash Memory Control (FMC), offset 0x008	155
Register 4:	Flash Controller Raw Interrupt Status (FCRIS), offset 0x00C	157
Register 5:	Flash Controller Interrupt Mask (FCIM), offset 0x010	158
Register 6:	Flash Controller Masked Interrupt Status and Clear (FCMISC), offset 0x014	159
Register 7:	USec Reload (USECRL), offset 0x140	160
Register 8:	Flash Memory Protection Read Enable 0 (FMPRE0), offset 0x130 and 0x200	161
Register 9:	Flash Memory Protection Program Enable 0 (FMPPE0), offset 0x134 and 0x400	162
Register 10:	User Debug (USER_DBG), offset 0x1D0	163
Register 11:	User Register 0 (USER_REG0), offset 0x1E0	164
Register 12:	User Register 1 (USER_REG1), offset 0x1E4	165
Register 13:	Flash Memory Protection Read Enable 1 (FMPRE1), offset 0x204	166
Register 14:	Flash Memory Protection Read Enable 2 (FMPRE2), offset 0x208	167
Register 15:	Flash Memory Protection Read Enable 3 (FMPRE3), offset 0x20C	168
Register 16:	Flash Memory Protection Program Enable 1 (FMPPE1), offset 0x404	169
Register 17:	Flash Memory Protection Program Enable 2 (FMPPE2), offset 0x408	170
Register 18:	Flash Memory Protection Program Enable 3 (FMPPE3), offset 0x40C	171
General-Purpose Input/Outputs (GPIOs)		172
Register 1:	GPIO Data (GPIODATA), offset 0x000	179
Register 2:	GPIO Direction (GPIODIR), offset 0x400	180
Register 3:	GPIO Interrupt Sense (GPIOIS), offset 0x404	181
Register 4:	GPIO Interrupt Both Edges (GPIOIBE), offset 0x408	182
Register 5:	GPIO Interrupt Event (GPIOIEV), offset 0x40C	183
Register 6:	GPIO Interrupt Mask (GPIOIM), offset 0x410	184
Register 7:	GPIO Raw Interrupt Status (GPIORIS), offset 0x414	185
Register 8:	GPIO Masked Interrupt Status (GPIOMIS), offset 0x418	186
Register 9:	GPIO Interrupt Clear (GPIOICR), offset 0x41C	187
Register 10:	GPIO Alternate Function Select (GPIOAFSEL), offset 0x420	188
Register 11:	GPIO 2-mA Drive Select (GPIODR2R), offset 0x500	190
Register 12:	GPIO 4-mA Drive Select (GPIODR4R), offset 0x504	191
Register 13:	GPIO 8-mA Drive Select (GPIODR8R), offset 0x508	192
Register 14:	GPIO Open Drain Select (GPIOODR), offset 0x50C	193
Register 15:	GPIO Pull-Up Select (GPIOPUR), offset 0x510	194
Register 16:	GPIO Pull-Down Select (GPIOPDR), offset 0x514	195
Register 17:	GPIO Slew Rate Control Select (GPIOSLR), offset 0x518	196
Register 18:	GPIO Digital Enable (GPIODEN), offset 0x51C	197
Register 19:	GPIO Lock (GPIOLOCK), offset 0x520	198
Register 20:	GPIO Commit (GPIOCR), offset 0x524	199
Register 21:	GPIO Peripheral Identification 4 (GPIOPeriphID4), offset 0xFD0	201
Register 22:	GPIO Peripheral Identification 5 (GPIOPeriphID5), offset 0xFD4	202
Register 23:	GPIO Peripheral Identification 6 (GPIOPeriphID6), offset 0xFD8	203
Register 24:	GPIO Peripheral Identification 7 (GPIOPeriphID7), offset 0xFDC	204
Register 25:	GPIO Peripheral Identification 0 (GPIOPeriphID0), offset 0xFE0	205
Register 26:	GPIO Peripheral Identification 1 (GPIOPeriphID1), offset 0xFE4	206
Register 27:	GPIO Peripheral Identification 2 (GPIOPeriphID2), offset 0xFE8	207
Register 28:	GPIO Peripheral Identification 3 (GPIOPeriphID3), offset 0xFEC	208
Register 29:	GPIO PrimeCell Identification 0 (GPIOPCellID0), offset 0xFF0	209
Register 30:	GPIO PrimeCell Identification 1 (GPIOPCellID1), offset 0xFF4	210
Register 31:	GPIO PrimeCell Identification 2 (GPIOPCellID2), offset 0xFF8	211

Register 32: GPIO PrimeCell Identification 3 (GPIOCellID3), offset 0xFFC	212
General-Purpose Timers	213
Register 1: GPTM Configuration (GPTMCFG), offset 0x000	225
Register 2: GPTM TimerA Mode (GPTMTAMR), offset 0x004	226
Register 3: GPTM TimerB Mode (GPTMTBMR), offset 0x008	228
Register 4: GPTM Control (GPTMCTL), offset 0x00C	230
Register 5: GPTM Interrupt Mask (GPTMIMR), offset 0x018	233
Register 6: GPTM Raw Interrupt Status (GPTMRIS), offset 0x01C	235
Register 7: GPTM Masked Interrupt Status (GPTMMIS), offset 0x020	236
Register 8: GPTM Interrupt Clear (GPTMICR), offset 0x024	237
Register 9: GPTM TimerA Interval Load (GPTMTAILR), offset 0x028	239
Register 10: GPTM TimerB Interval Load (GPTMTBILR), offset 0x02C	240
Register 11: GPTM TimerA Match (GPTMTAMATCHR), offset 0x030	241
Register 12: GPTM TimerB Match (GPTMTBMATCHR), offset 0x034	242
Register 13: GPTM TimerA Prescale (GPTMTAPR), offset 0x038	243
Register 14: GPTM TimerB Prescale (GPTMTBPR), offset 0x03C	244
Register 15: GPTM TimerA Prescale Match (GPTMTAPMR), offset 0x040	245
Register 16: GPTM TimerB Prescale Match (GPTMTBPMR), offset 0x044	246
Register 17: GPTM TimerA (GPTMTAR), offset 0x048	247
Register 18: GPTM TimerB (GPTMTBR), offset 0x04C	248
Watchdog Timer	249
Register 1: Watchdog Load (WDTLOAD), offset 0x000	253
Register 2: Watchdog Value (WDTVALUE), offset 0x004	254
Register 3: Watchdog Control (WDTCTL), offset 0x008	255
Register 4: Watchdog Interrupt Clear (WDTICR), offset 0x00C	256
Register 5: Watchdog Raw Interrupt Status (WDTRIS), offset 0x010	257
Register 6: Watchdog Masked Interrupt Status (WDTMIS), offset 0x014	258
Register 7: Watchdog Test (WDTTEST), offset 0x418	259
Register 8: Watchdog Lock (WDTLOCK), offset 0xC00	260
Register 9: Watchdog Peripheral Identification 4 (WDTPeriphID4), offset 0xFD0	261
Register 10: Watchdog Peripheral Identification 5 (WDTPeriphID5), offset 0xFD4	262
Register 11: Watchdog Peripheral Identification 6 (WDTPeriphID6), offset 0xFD8	263
Register 12: Watchdog Peripheral Identification 7 (WDTPeriphID7), offset 0xFDC	264
Register 13: Watchdog Peripheral Identification 0 (WDTPeriphID0), offset 0xFE0	265
Register 14: Watchdog Peripheral Identification 1 (WDTPeriphID1), offset 0xFE4	266
Register 15: Watchdog Peripheral Identification 2 (WDTPeriphID2), offset 0xFE8	267
Register 16: Watchdog Peripheral Identification 3 (WDTPeriphID3), offset 0xFEC	268
Register 17: Watchdog PrimeCell Identification 0 (WDTPCellID0), offset 0xFF0	269
Register 18: Watchdog PrimeCell Identification 1 (WDTPCellID1), offset 0xFF4	270
Register 19: Watchdog PrimeCell Identification 2 (WDTPCellID2), offset 0xFF8	271
Register 20: Watchdog PrimeCell Identification 3 (WDTPCellID3), offset 0xFFC	272
Universal Asynchronous Receivers/Transmitters (UARTs)	273
Register 1: UART Data (UARTDR), offset 0x000	281
Register 2: UART Receive Status/Error Clear (UARTRSR/UARTECR), offset 0x004	283
Register 3: UART Flag (UARTFR), offset 0x018	285
Register 4: UART IrDA Low-Power Register (UARTILPR), offset 0x020	287
Register 5: UART Integer Baud-Rate Divisor (UARTIBRD), offset 0x024	288
Register 6: UART Fractional Baud-Rate Divisor (UARTFBRD), offset 0x028	289

Register 7:	UART Line Control (UARTLCRH), offset 0x02C	290
Register 8:	UART Control (UARTCTL), offset 0x030	292
Register 9:	UART Interrupt FIFO Level Select (UARTIFLS), offset 0x034	294
Register 10:	UART Interrupt Mask (UARTIM), offset 0x038	296
Register 11:	UART Raw Interrupt Status (UARTRIS), offset 0x03C	298
Register 12:	UART Masked Interrupt Status (UARTMIS), offset 0x040	299
Register 13:	UART Interrupt Clear (UARTICR), offset 0x044	300
Register 14:	UART Peripheral Identification 4 (UARTPeriphID4), offset 0xFD0	302
Register 15:	UART Peripheral Identification 5 (UARTPeriphID5), offset 0xFD4	303
Register 16:	UART Peripheral Identification 6 (UARTPeriphID6), offset 0xFD8	304
Register 17:	UART Peripheral Identification 7 (UARTPeriphID7), offset 0xFDC	305
Register 18:	UART Peripheral Identification 0 (UARTPeriphID0), offset 0xFE0	306
Register 19:	UART Peripheral Identification 1 (UARTPeriphID1), offset 0xFE4	307
Register 20:	UART Peripheral Identification 2 (UARTPeriphID2), offset 0xFE8	308
Register 21:	UART Peripheral Identification 3 (UARTPeriphID3), offset 0xFEC	309
Register 22:	UART PrimeCell Identification 0 (UARTPCellID0), offset 0xFF0	310
Register 23:	UART PrimeCell Identification 1 (UARTPCellID1), offset 0xFF4	311
Register 24:	UART PrimeCell Identification 2 (UARTPCellID2), offset 0xFF8	312
Register 25:	UART PrimeCell Identification 3 (UARTPCellID3), offset 0xFFC	313
Synchronous Serial Interface (SSI)		314
Register 1:	SSI Control 0 (SSICR0), offset 0x000	326
Register 2:	SSI Control 1 (SSICR1), offset 0x004	328
Register 3:	SSI Data (SSIDR), offset 0x008	330
Register 4:	SSI Status (SSISR), offset 0x00C	331
Register 5:	SSI Clock Prescale (SSICPSR), offset 0x010	333
Register 6:	SSI Interrupt Mask (SSIIM), offset 0x014	334
Register 7:	SSI Raw Interrupt Status (SSIRIS), offset 0x018	336
Register 8:	SSI Masked Interrupt Status (SSIMIS), offset 0x01C	337
Register 9:	SSI Interrupt Clear (SSIICR), offset 0x020	338
Register 10:	SSI Peripheral Identification 4 (SSIPeriphID4), offset 0xFD0	339
Register 11:	SSI Peripheral Identification 5 (SSIPeriphID5), offset 0xFD4	340
Register 12:	SSI Peripheral Identification 6 (SSIPeriphID6), offset 0xFD8	341
Register 13:	SSI Peripheral Identification 7 (SSIPeriphID7), offset 0xFDC	342
Register 14:	SSI Peripheral Identification 0 (SSIPeriphID0), offset 0xFE0	343
Register 15:	SSI Peripheral Identification 1 (SSIPeriphID1), offset 0xFE4	344
Register 16:	SSI Peripheral Identification 2 (SSIPeriphID2), offset 0xFE8	345
Register 17:	SSI Peripheral Identification 3 (SSIPeriphID3), offset 0xFEC	346
Register 18:	SSI PrimeCell Identification 0 (SSIPCellID0), offset 0xFF0	347
Register 19:	SSI PrimeCell Identification 1 (SSIPCellID1), offset 0xFF4	348
Register 20:	SSI PrimeCell Identification 2 (SSIPCellID2), offset 0xFF8	349
Register 21:	SSI PrimeCell Identification 3 (SSIPCellID3), offset 0xFFC	350
Inter-Integrated Circuit (I²C) Interface		351
Register 1:	I ² C Master Slave Address (I2CMSA), offset 0x000	366
Register 2:	I ² C Master Control/Status (I2CMCS), offset 0x004	367
Register 3:	I ² C Master Data (I2CMDR), offset 0x008	371
Register 4:	I ² C Master Timer Period (I2CMTPR), offset 0x00C	372
Register 5:	I ² C Master Interrupt Mask (I2CMIMR), offset 0x010	373
Register 6:	I ² C Master Raw Interrupt Status (I2CMRIS), offset 0x014	374

Register 7:	I ² C Master Masked Interrupt Status (I2CMMIS), offset 0x018	375
Register 8:	I ² C Master Interrupt Clear (I2CMICR), offset 0x01C	376
Register 9:	I ² C Master Configuration (I2CMCR), offset 0x020	377
Register 10:	I ² C Slave Own Address (I2CSOAR), offset 0x000	379
Register 11:	I ² C Slave Control/Status (I2CSCSR), offset 0x004	380
Register 12:	I ² C Slave Data (I2CSDR), offset 0x008	382
Register 13:	I ² C Slave Interrupt Mask (I2CSIMR), offset 0x00C	383
Register 14:	I ² C Slave Raw Interrupt Status (I2CSRIS), offset 0x010	384
Register 15:	I ² C Slave Masked Interrupt Status (I2CSMIS), offset 0x014	385
Register 16:	I ² C Slave Interrupt Clear (I2CSICR), offset 0x018	386
Analog Comparators	387	
Register 1:	Analog Comparator Masked Interrupt Status (ACMIS), offset 0x000	392
Register 2:	Analog Comparator Raw Interrupt Status (ACRIS), offset 0x004	393
Register 3:	Analog Comparator Interrupt Enable (ACINTEN), offset 0x008	394
Register 4:	Analog Comparator Reference Voltage Control (ACREFCTL), offset 0x010	395
Register 5:	Analog Comparator Status 0 (ACSTAT0), offset 0x020	396
Register 6:	Analog Comparator Status 1 (ACSTAT1), offset 0x040	396
Register 7:	Analog Comparator Status 2 (ACSTAT2), offset 0x060	396
Register 8:	Analog Comparator Control 0 (ACCTL0), offset 0x024	397
Register 9:	Analog Comparator Control 1 (ACCTL1), offset 0x044	397
Register 10:	Analog Comparator Control 2 (ACCTL2), offset 0x064	397
Pulse Width Modulator (PWM)	399	
Register 1:	PWM Master Control (PWMCTL), offset 0x000	408
Register 2:	PWM Time Base Sync (PWMSYNC), offset 0x004	409
Register 3:	PWM Output Enable (PWMENABLE), offset 0x008	410
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:	PWM1 Control (PWM1CTL), offset 0x080	417
Register 12:	PWM2 Control (PWM2CTL), offset 0x0C0	417
Register 13:	PWM0 Interrupt Enable (PWM0INTEN), offset 0x044	419
Register 14:	PWM1 Interrupt Enable (PWM1INTEN), offset 0x084	419
Register 15:	PWM2 InterruptEnable (PWM2INTEN), offset 0x0C4	419
Register 16:	PWM0 Raw Interrupt Status (PWM0RIS), offset 0x048	421
Register 17:	PWM1 Raw Interrupt Status (PWM1RIS), offset 0x088	421
Register 18:	PWM2 Raw Interrupt Status (PWM2RIS), offset 0x0C8	421
Register 19:	PWM0 Interrupt Status and Clear (PWM0ISC), offset 0x04C	422
Register 20:	PWM1 Interrupt Status and Clear (PWM1ISC), offset 0x08C	422
Register 21:	PWM2 Interrupt Status and Clear (PWM2ISC), offset 0x0CC	422
Register 22:	PWM0 Load (PWM0LOAD), offset 0x050	423
Register 23:	PWM1 Load (PWM1LOAD), offset 0x090	423
Register 24:	PWM2 Load (PWM2LOAD), offset 0x0D0	423
Register 25:	PWM0 Counter (PWM0COUNT), offset 0x054	424

Register 26:	PWM1 Counter (PWM1COUNT), offset 0x094	424
Register 27:	PWM2 Counter (PWM2COUNT), offset 0x0D4	424
Register 28:	PWM0 Compare A (PWM0CMPA), offset 0x058	425
Register 29:	PWM1 Compare A (PWM1CMPA), offset 0x098	425
Register 30:	PWM2 Compare A (PWM2CMPA), offset 0x0D8	425
Register 31:	PWM0 Compare B (PWM0CMPB), offset 0x05C	426
Register 32:	PWM1 Compare B (PWM1CMPB), offset 0x09C	426
Register 33:	PWM2 Compare B (PWM2CMPB), offset 0x0DC	426
Register 34:	PWM0 Generator A Control (PWM0GENA), offset 0x060	427
Register 35:	PWM1 Generator A Control (PWM1GENA), offset 0x0A0	427
Register 36:	PWM2 Generator A Control (PWM2GENA), offset 0x0E0	427
Register 37:	PWM0 Generator B Control (PWM0GENB), offset 0x064	430
Register 38:	PWM1 Generator B Control (PWM1GENB), offset 0x0A4	430
Register 39:	PWM2 Generator B Control (PWM2GENB), offset 0x0E4	430
Register 40:	PWM0 Dead-Band Control (PWM0DBCTL), offset 0x068	433
Register 41:	PWM1 Dead-Band Control (PWM1DBCTL), offset 0x0A8	433
Register 42:	PWM2 Dead-Band Control (PWM2DBCTL), offset 0x0E8	433
Register 43:	PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE), offset 0x06C	434
Register 44:	PWM1 Dead-Band Rising-Edge Delay (PWM1DBRISE), offset 0x0AC	434
Register 45:	PWM2 Dead-Band Rising-Edge Delay (PWM2DBRISE), offset 0x0EC	434
Register 46:	PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL), offset 0x070	435
Register 47:	PWM1 Dead-Band Falling-Edge-Delay (PWM1DBFALL), offset 0x0B0	435
Register 48:	PWM2 Dead-Band Falling-Edge-Delay (PWM2DBFALL), offset 0x0F0	435
Quadrature Encoder Interface (QEI)		436
Register 1:	QEI Control (QEICTL), offset 0x000	441
Register 2:	QEI Status (QEISTAT), offset 0x004	443
Register 3:	QEI Position (QEIPPOS), offset 0x008	444
Register 4:	QEI Maximum Position (QEIMAXPOS), offset 0x00C	445
Register 5:	QEI Timer Load (QEILOAD), offset 0x010	446
Register 6:	QEI Timer (QEITIME), offset 0x014	447
Register 7:	QEI Velocity Counter (QEICOUNT), offset 0x018	448
Register 8:	QEI Velocity (QEISPEED), offset 0x01C	449
Register 9:	QEI Interrupt Enable (QEIINTEN), offset 0x020	450
Register 10:	QEI Raw Interrupt Status (QEIRIS), offset 0x024	451
Register 11:	QEI Interrupt Status and Clear (QEIISC), offset 0x028	452

Revision History

The revision history table notes changes made between the indicated revisions of the LM3S1960 data sheet.

Table 1. Revision History

Date	Revision	Description
March 2008	2550	Started tracking revision history.
April 2008	2881	<ul style="list-style-type: none"> ■ The Θ_{JA} value was changed from 55.3 to 34 in the "Thermal Characteristics" table in the Operating Characteristics chapter. ■ Bit 31 of the DC3 register was incorrectly described in prior versions of the data sheet. A reset of 1 indicates that an even CCP pin is present and can be used as a 32-KHz input clock. ■ Values for $I_{DD_HIBERNATE}$ were added to the "Detailed Power Specifications" table in the "Electrical Characteristics" chapter. ■ The "Hibernation Module DC Electricals" table was added to the "Electrical Characteristics" chapter. ■ The $T_{VDDRISE}$ parameter in the "Reset Characteristics" table in the "Electrical Characteristics" chapter was changed from a max of 100 to 250. ■ The maximum value on Core supply voltage (V_{DD25}) in the "Maximum Ratings" table in the "Electrical Characteristics" chapter was changed from 4 to 3. ■ The operational frequency of the internal 30-kHz oscillator clock source is 30 kHz \pm 50% (prior data sheets incorrectly noted it as 30 kHz \pm 30%). ■ A value of 0x3 in bits 5:4 of the MISC register ($OSCSRC$) indicates the 30-KHz internal oscillator is the input source for the oscillator. Prior data sheets incorrectly noted 0x3 as a reserved value. ■ The reset for bits 6:4 of the RCC2 register ($OSCSRC2$) is 0x1 (IOSC). Prior data sheets incorrectly noted the reset was 0x0 (MOSC). ■ Two figures on clock source were added to the "Hibernation Module": <ul style="list-style-type: none"> – Clock Source Using Crystal – Clock Source Using Dedicated Oscillator ■ The following notes on battery management were added to the "Hibernation Module" chapter: <ul style="list-style-type: none"> – Battery voltage is not measured while in Hibernate mode. – System level factors may affect the accuracy of the low battery detect circuit. The designer should consider battery type, discharge characteristics, and a test load during battery voltage measurements. ■ A note on high-current applications was added to the GPIO chapter: For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the VOL value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package or BGA pin group with the total number of high-current GPIO outputs not exceeding four for the entire package. ■ A note on Schmitt inputs was added to the GPIO chapter: Pins configured as digital inputs are Schmitt-triggered. ■ The Buffer type on the WAKE pin changed from OD to - in the Signal Tables.

Table 1. Revision History (continued)

Date	Revision	Description
		<ul style="list-style-type: none"> ■ The "Differential Sampling Range" figures in the ADC chapter were clarified. ■ The last revision of the data sheet (revision 2550) introduced two errors that have now been corrected: <ul style="list-style-type: none"> – The LQFP pin diagrams and pin tables were missing the comparator positive and negative input pins. – The base address was listed incorrectly in the FMPRE0 and FMPPE0 register bit diagrams. ■ Additional minor data sheet clarifications and corrections.
May 2008	2972	<ul style="list-style-type: none"> ■ As noted in the PCN, the option to provide V_{DD25} power from external sources was removed. Use the LDO output as the source of V_{DD25} input. ■ Additional minor data sheet clarifications and corrections.
July 2008	3108	<ul style="list-style-type: none"> ■ Additional minor data sheet clarifications and corrections.
August 2008	3447	<ul style="list-style-type: none"> ■ Added note on clearing interrupts to Interrupts chapter. ■ Added Power Architecture diagram to System Control chapter. ■ Additional minor data sheet clarifications and corrections.
October 2008	4149	<ul style="list-style-type: none"> ■ Corrected values for $DSOSCSRC$ bit field in Deep Sleep Clock Configuration (DSLCLKCFG) register. ■ The FMA value for the FMPRE3 register was incorrect in the Flash Resident Registers table in the Internal Memory chapter. The correct value is 0x0000.0006. ■ Incorrect Comparator Operating Modes tables were removed from the Analog Comparators chapter.
November 2008	4283	<ul style="list-style-type: none"> ■ Revised High-Level Block Diagram. ■ Additional minor data sheet clarifications and corrections were made.
January 2009	4660	<ul style="list-style-type: none"> ■ Corrected bit type for RELOAD bit field in SysTick Reload Value register; changed to R/W. ■ Clarification added as to what happens when the SSI in slave mode is required to transmit but there is no data in the TX FIFO. ■ Additional minor data sheet clarifications and corrections.
April 2009	5367	<ul style="list-style-type: none"> ■ Added JTAG/SWD clarification (see "Communication with JTAG/SWD" on page 58). ■ Added clarification that the PLL operates at 400 MHz, but is divided by two prior to the application of the output divisor. ■ Added "GPIO Module DC Characteristics" table (see Table 21-4 on page 485). ■ Additional minor data sheet clarifications and corrections.

Table 1. Revision History (continued)

Date	Revision	Description
July 2009	5902	<ul style="list-style-type: none"> ■ Clarified Power-on reset and $\overline{\text{RST}}$ pin operation; added new diagrams. ■ Corrected the reset value of the Hibernation Data (HIBDATA) and Hibernation Control (HIBCTL) registers. ■ Clarified explanation of nonvolatile register programming in Internal Memory chapter. ■ Added explanation of reset value to FMPRE0/1/2/3, FMPPE0/1/2/3, USER_DBG, and USER_REG0/1 registers. ■ Changed buffer type for $\overline{\text{WAKE}}$ pin to TTL and $\overline{\text{HIB}}$ pin to OD. ■ In ADC characteristics table, changed Max value for GAIN parameter from ± 1 to ± 3 and added E_{IR} (Internal voltage reference error) parameter. ■ Additional minor data sheet clarifications and corrections.
July 2009	5920	Corrected ordering numbers.
October 2009	6462	<ul style="list-style-type: none"> ■ Removed erroneous reference to the WRC bit in the Hibernation chapter. ■ Deleted reset value for 16-bit mode from GPTMTAILR, GPTMTAMATCHR, and GPTMTAR registers because the module resets in 32-bit mode. ■ Made these changes to the Electrical Characteristics chapter: <ul style="list-style-type: none"> – Removed V_{SIH} and V_{SIL} parameters from Operating Conditions table. – Added table showing actual PLL frequency depending on input crystal. – Changed the name of the $t_{\text{HIB_REG_WRITE}}$ parameter to $t_{\text{HIB_REG_ACCESS}}$. – Changed SSI set up and hold times to be expressed in system clocks, not ns.

About This Document

This data sheet provides reference information for the LM3S1960 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

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 Stellaris® 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*
- *Stellaris® Peripheral Driver Library User's Guide*
- *Stellaris® ROM User's Guide*

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 web site for additional documentation, including application notes and white papers.

Documentation Conventions

This document uses the conventions shown in Table 2 on page 21.

Table 2. Documentation Conventions

Notation	Meaning
General Register Notation	
REGISTER	APB registers are indicated in uppercase bold. For example, PBORCTL 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, SRCRn represents any (or all) of the three Software Reset Control registers: SRCR0 , SRCR1 , and SRCR2 .
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 46.

Table 2. Documentation Conventions (continued)

Notation	Meaning
Register <i>N</i>	Registers are numbered consecutively throughout the document to aid in referencing them. The register number has no meaning to software.
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.
R/W1S	Software can read or write a 1 to this field. A write of a 0 to a R/W1S bit does not affect the bit value in the register.
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 <i>SIGNAL</i> and <i>SIGNAL</i> below).
deassert a signal	Change the value of the signal from the logically True state to the logically False state.
<i>SIGNAL</i>	Signal names are in uppercase and in the Courier font. An overbar on a signal name indicates that it is active Low. To assert <i>SIGNAL</i> is to drive it Low; to deassert <i>SIGNAL</i> is to drive it High.
<i>SIGNAL</i>	Signal names are in uppercase and in the Courier font. An active High signal has no overbar. To assert <i>SIGNAL</i> is to drive it High; to deassert <i>SIGNAL</i> 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.

Table 2. Documentation Conventions (continued)

Notation	Meaning
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.

1 Architectural Overview

The Stellaris[®] family of microcontrollers—the first ARM[®] Cortex™-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[®] family offers efficient performance and extensive integration, favorably positioning the device into cost-conscious applications requiring significant control-processing and connectivity capabilities. The Stellaris[®] LM3S1000 series extends the Stellaris[®] family with larger on-chip memories, enhanced power management, and expanded I/O and control capabilities.

The LM3S1960 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.

For applications requiring extreme conservation of power, the LM3S1960 microcontroller features a battery-backed Hibernation module to efficiently power down the LM3S1960 to a low-power state during extended periods of inactivity. With a power-up/power-down sequencer, a continuous time counter (RTC), a pair of match registers, an APB interface to the system bus, and dedicated non-volatile memory, the Hibernation module positions the LM3S1960 microcontroller perfectly for battery applications.

In addition, the LM3S1960 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 LM3S1960 microcontroller is code-compatible to all members of the extensive Stellaris[®] family; providing flexibility to fit our customers' precise needs.

Texas Instruments 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. See “Ordering and Contact Information” on page 519 for ordering information for Stellaris[®] family devices.

1.1 Product Features

The LM3S1960 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
 - 50-MHz operation
 - Hardware-division and single-cycle-multiplication

- Integrated Nested Vectored Interrupt Controller (NVIC) providing deterministic interrupt handling
- 36 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
- ARM® Cortex™-M3 Processor Core
 - 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.
 - Deterministic, fast interrupt processing: always 12 cycles, or just 6 cycles with tail-chaining
 - Memory protection unit (MPU) to provide a privileged mode of operation for complex applications.
 - Migration from the ARM7™ processor family for better performance and power efficiency.
 - Full-featured debug solution
 - 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
 - Optimized for single-cycle flash usage
 - Three sleep modes with clock gating for low power
 - Single-cycle multiply instruction and hardware divide
 - Atomic operations
 - ARM Thumb2 mixed 16-/32-bit instruction set

- 1.25 DMIPS/MHz
- JTAG
 - IEEE 1149.1-1990 compatible Test Access Port (TAP) controller
 - Four-bit Instruction Register (IR) chain for storing JTAG instructions
 - IEEE standard instructions: BYPASS, IDCODE, SAMPLE/PRELOAD, EXTEST and INTEST
 - ARM additional instructions: APACC, DPACC and ABORT
 - Integrated ARM Serial Wire Debug (SWD)
- Hibernation
 - System power control using discrete external regulator
 - Dedicated pin for waking from an external signal
 - Low-battery detection, signaling, and interrupt generation
 - 32-bit real-time counter (RTC)
 - Two 32-bit RTC match registers for timed wake-up and interrupt generation
 - Clock source from a 32.768-kHz external oscillator or a 4.194304-MHz crystal
 - RTC predivider trim for making fine adjustments to the clock rate
 - 64 32-bit words of non-volatile memory
 - Programmable interrupts for RTC match, external wake, and low battery events
- Internal Memory
 - 256 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
 - 64 KB single-cycle SRAM
- GPIOs
 - 7-60 GPIOs, depending on configuration
 - 5-V-tolerant input/outputs
 - Programmable control for GPIO interrupts
 - Interrupt generation masking
 - Edge-triggered on rising, falling, or both

- Level-sensitive on High or Low values
- Bit masking in both read and write operations through address lines
- Pins configured as digital inputs are Schmitt-triggered.
- Programmable control for GPIO pad configuration
 - Weak pull-up or pull-down resistors
 - 2-mA, 4-mA, and 8-mA pad drive for digital communication; up to four pads can be configured with an 18-mA pad drive for high-current applications
 - Slew rate control for the 8-mA drive
 - Open drain enables
 - Digital input enables
- General-Purpose Timers
 - Four 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
 - 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
 - User-enabled stalling when the controller asserts CPU Halt flag during debug
 - 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
- UART
 - Three fully programmable 16C550-type UARTs with IrDA support
 - Separate 16x8 transmit (TX) and receive (RX) FIFOs to reduce CPU interrupt service loading
 - Programmable baud-rate generator allowing speeds up to 3.125 Mbps
 - 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
 - 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 μ s) bit durations
 - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration
- Synchronous Serial Interface (SSI)

- Two SSI modules, each with 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
- I²C
 - Two I²C modules, each with the following features:
 - Devices on the I²C bus can be designated as either a master or a slave
 - Supports both sending and receiving data as either a master or a slave
 - Supports simultaneous master and slave operation
 - Four I²C modes
 - Master transmit
 - Master receive
 - Slave transmit
 - Slave receive
 - Two transmission speeds: Standard (100 Kbps) and Fast (400 Kbps)
 - Master and slave interrupt generation
 - Master generates interrupts when a transmit or receive operation completes (or aborts due to an error)
 - Slave generates interrupts when data has been sent or requested by a master
 - Master with arbitration and clock synchronization, multimaster support, and 7-bit addressing mode
- 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
 - 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
- PWM
 - Three PWM generator blocks, each with one 16-bit counter, two PWM comparators, a PWM signal generator, a dead-band generator, and an interrupt selector
 - One fault input in hardware to promote low-latency shutdown
 - 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

- QEI
 - Two QEI modules, each with the following features:
 - Position integrator that tracks the encoder position
 - Velocity capture using built-in timer
 - The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 12.5 MHz for a 50-MHz system)
 - Interrupt generation on:
 - Index pulse
 - Velocity-timer expiration
 - Direction change
 - Quadrature error detection
- Power
 - On-chip Low Drop-Out (LDO) voltage regulator, with programmable output user-adjustable from 2.25 V to 2.75 V
 - Hibernation module handles the power-up/down 3.3 V sequencing and control for the core digital logic and analog circuits
 - Low-power options on controller: Sleep and Deep-sleep modes
 - Low-power options for peripherals: software controls shutdown of individual peripherals
 - 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
- Industrial and extended temperature 100-pin RoHS-compliant LQFP package
- Industrial-range 108-ball RoHS-compliant BGA package

1.2 Target Applications

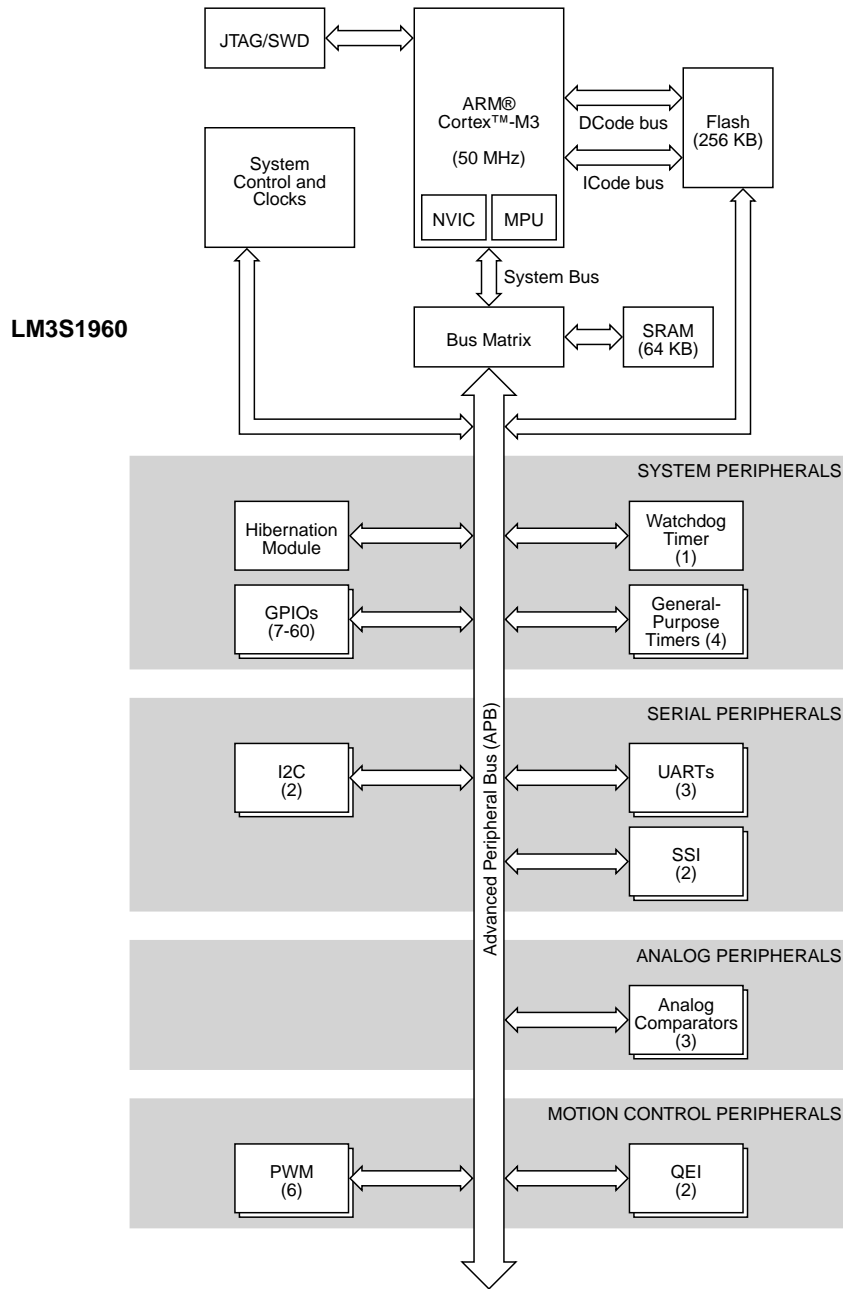
- Remote monitoring

- 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
- Transportation

1.3 High-Level Block Diagram

Figure 1-1 on page 33 depicts the features on the Stellaris[®] LM3S1960 microcontroller.

Figure 1-1. Stellaris® LM3S1960 Microcontroller High-Level Block Diagram



1.4 Functional Overview

The following sections provide an overview of the features of the LM3S1960 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 519.

1.4.1 ARM Cortex™-M3

1.4.1.1 Processor Core (see page 40)

All members of the Stellaris® product family, including the LM3S1960 microcontroller, are designed around an 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.

“ARM Cortex-M3 Processor Core” on page 40 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) (see page 43)

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) (see page 49)

The LM3S1960 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 36 interrupts.

“Interrupts” on page 49 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 LM3S1960 controller features Pulse Width Modulation (PWM) outputs and the Quadrature Encoder Interface (QEI).

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 LM3S1960, 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 399)

The LM3S1960 PWM module consists of three 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 219)

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.

Fault Pin (see page 404)

The LM3S1960 PWM module includes one fault-condition handling input to quickly provide low-latency shutdown and prevent damage to the motor being controlled.

1.4.2.2 QEI (see page 436)

A quadrature encoder, also known as a 2-channel incremental encoder, converts linear displacement into a pulse signal. By monitoring both the number of pulses and the relative phase of the two signals, you can track the position, direction of rotation, and speed. In addition, a third channel, or index signal, can be used to reset the position counter.

The Stellaris quadrature encoder with index (QEI) module interprets the code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel. The LM3S1960 microcontroller includes two QEI modules, which enables control of two motors at the same time.

1.4.3 Analog Peripherals

For support of analog signals, the LM3S1960 microcontroller offers three analog comparators.

1.4.3.1 Analog Comparators (see page 387)

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

The LM3S1960 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 LM3S1960 controller supports both asynchronous and synchronous serial communications with:

- Three fully programmable 16C550-type UARTs
- Two SSI modules
- Two I²C modules

1.4.4.1 UART (see page 273)

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 LM3S1960 controller includes three fully programmable 16C550-type UARTs that support data transfer speeds up to 3.125 Mbps. (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 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 314)

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

The LM3S1960 controller includes two SSI modules that provide 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.

Each 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.

Each 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.

Each 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 I²C (see page 351)

The Inter-Integrated Circuit (I²C) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL).

The I²C bus interfaces to external I²C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I²C bus may also be used for system testing and diagnostic purposes in product development and manufacture.

The LM3S1960 controller includes two I²C modules that provide the ability to communicate to other IC devices over an I²C bus. The I²C bus supports devices that can both transmit and receive (write and read) data.

Devices on the I²C bus can be designated as either a master or a slave. Each I²C 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 I²C modes are: Master Transmit, Master Receive, Slave Transmit, and Slave Receive.

A Stellaris® I²C module can operate at two speeds: Standard (100 Kbps) and Fast (400 Kbps).

Both the I²C master and slave can generate interrupts. The I²C master generates interrupts when a transmit or receive operation completes (or aborts due to an error). The I²C slave generates interrupts when data has been sent or requested by a master.

1.4.5 System Peripherals

1.4.5.1 Programmable GPIOs (see page 172)

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

The Stellaris® GPIO module is comprised 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 7-60 programmable input/output pins. The number of GPIOs available depends on the peripherals being used (see "Signal Tables" on page 455 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. Pins configured as digital inputs are Schmitt-triggered.

1.4.5.2 Four Programmable Timers (see page 213)

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

The Stellaris® General-Purpose Timer Module (GPTM) contains four 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 249)

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[®] Watchdog Timer module consists of a 32-bit down counter, a programmable load register, interrupt generation logic, and a locking register.

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 LM3S1960 controller offers both single-cycle SRAM and single-cycle Flash memory.

1.4.6.1 SRAM (see page 148)

The LM3S1960 static random access memory (SRAM) controller supports 64 KB SRAM. The internal SRAM of the Stellaris[®] 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 149)

The LM3S1960 Flash controller supports 256 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 46)

A memory map lists the location of instructions and data in memory. The memory map for the LM3S1960 controller can be found in “Memory Map” on page 46. 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 52)

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 composed of the standard five pins: $\overline{\text{TRST}}$, 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 Stellaris® 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 Stellaris® JTAG instructions select the Stellaris® TDO outputs. The multiplexer is controlled by the Stellaris® JTAG controller, which has comprehensive programming for the ARM, Stellaris®, and unimplemented JTAG instructions.

1.4.7.3 System Control and Clocks (see page 64)

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.

1.4.7.4 Hibernation Module (see page 128)

The Hibernation module provides logic to switch power off to the main processor and peripherals, and to wake on external or time-based events. The Hibernation module includes power-sequencing logic, a real-time clock with a pair of match registers, low-battery detection circuitry, and interrupt signalling to the processor. It also includes 64 32-bit words of non-volatile memory that can be used for saving state during hibernation.

1.4.8 Hardware Details

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

- “Pin Diagram” on page 453
- “Signal Tables” on page 455
- “Operating Characteristics” on page 483
- “Electrical Characteristics” on page 484
- “Package Information” on page 521

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:

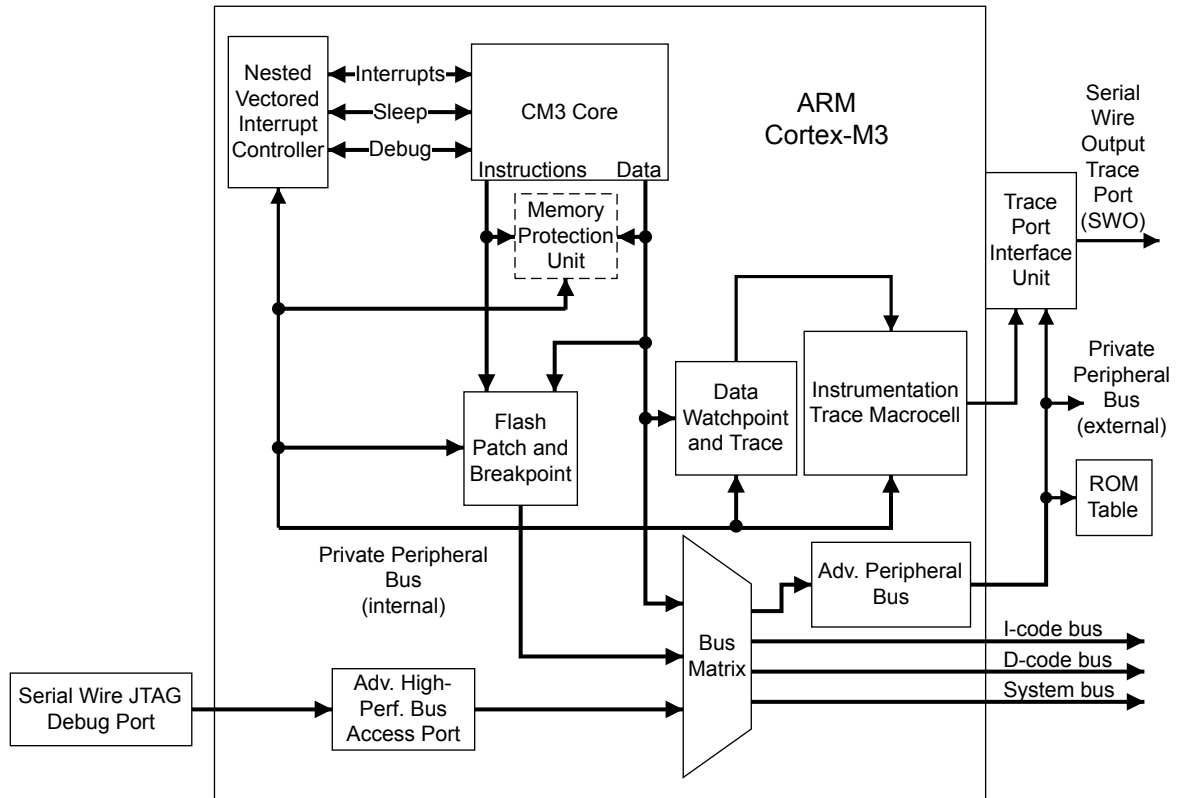
- Compact core.
- Thumb-2 instruction set, delivering the high-performance expected of an ARM core in the memory size usually associated with 8- and 16-bit devices; typically in the range of a few kilobytes of memory for microcontroller class applications.
- Rapid application execution through Harvard architecture characterized by separate buses for instruction and data.
- Exceptional interrupt handling, by implementing the register manipulations required for handling an interrupt in hardware.
- Deterministic, fast interrupt processing: always 12 cycles, or just 6 cycles with tail-chaining
- Memory protection unit (MPU) to provide a privileged mode of operation for complex applications.
- Migration from the ARM7™ processor family for better performance and power efficiency.
- Full-featured debug solution
 - 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
- Optimized for single-cycle flash usage
- Three sleep modes with clock gating for low power
- Single-cycle multiply instruction and hardware divide
- Atomic operations
- ARM Thumb2 mixed 16-/32-bit instruction set
- 1.25 DMIPS/MHz

The Stellaris® 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.

Texas Instruments has implemented the ARM Cortex-M3 core as shown in Figure 2-1 on page 41. 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

Texas Instruments 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® 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.

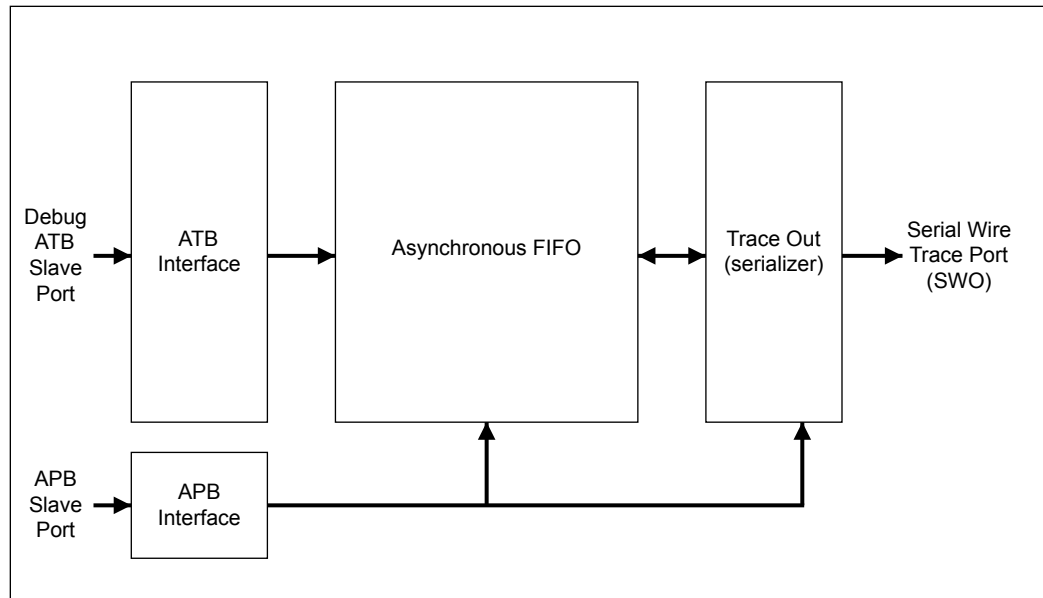
2.2.2 Embedded Trace Macrocell (ETM)

ETM was not implemented in the Stellaris[®] 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[®] devices have implemented TPIU as shown in Figure 2-2 on page 42. 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 LM3S1960 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

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.

2.2.6.1 Interrupts

The *ARM® Cortex™-M3 Technical Reference Manual* describes the maximum number of interrupts and interrupt priorities. The LM3S1960 microcontroller supports 36 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® 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	Type	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	Count Flag 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-only 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	Clock Source Value Description 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	Tick Interrupt Value Description 0 Counting down to 0 does not generate the interrupt request to the NVIC. Software can use the COUNTFLAG to determine if ever counted to 0. 1 Counting down to 0 pends the SysTick handler.
0	ENABLE	R/W	0	Enable Value Description 0 Counter disabled. 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.

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	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	RELOAD	R/W	-	Reload 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	Type	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 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.

SysTick Calibration Value Register

The SysTick Calibration Value register is not implemented.

3 Memory Map

The memory map for the LM3S1960 controller is provided in Table 3-1 on page 46.

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*.

Table 3-1. Memory Map^a

Start	End	Description	For details on registers, see page ...
Memory			
0x0000.0000	0x0003.FFFF	On-chip flash ^b	152
0x0004.0000	0x1FFF.FFFF	Reserved	-
0x2000.0000	0x2000.FFFF	Bit-banded on-chip SRAM ^c	152
0x2001.0000	0x21FF.FFFF	Reserved	-
0x2200.0000	0x221F.FFFF	Bit-band alias of 0x2000.0000 through 0x200F.FFFF	148
0x2220.0000	0x3FFF.FFFF	Reserved	-
FIRM Peripherals			
0x4000.0000	0x4000.0FFF	Watchdog timer	252
0x4000.1000	0x4000.3FFF	Reserved	-
0x4000.4000	0x4000.4FFF	GPIO Port A	178
0x4000.5000	0x4000.5FFF	GPIO Port B	178
0x4000.6000	0x4000.6FFF	GPIO Port C	178
0x4000.7000	0x4000.7FFF	GPIO Port D	178
0x4000.8000	0x4000.8FFF	SSI0	325
0x4000.9000	0x4000.9FFF	SSI1	325
0x4000.A000	0x4000.BFFF	Reserved	-
0x4000.C000	0x4000.CFFF	UART0	280
0x4000.D000	0x4000.DFFF	UART1	280
0x4000.E000	0x4000.EFFF	UART2	280
0x4000.F000	0x4001.FFFF	Reserved	-
Peripherals			
0x4002.0000	0x4002.07FF	I2C Master 0	365
0x4002.0800	0x4002.0FFF	I2C Slave 0	378
0x4002.1000	0x4002.17FF	I2C Master 1	365
0x4002.1800	0x4002.1FFF	I2C Slave 1	378
0x4002.2000	0x4002.3FFF	Reserved	-
0x4002.4000	0x4002.4FFF	GPIO Port E	178
0x4002.5000	0x4002.5FFF	GPIO Port F	178
0x4002.6000	0x4002.6FFF	GPIO Port G	178
0x4002.7000	0x4002.7FFF	GPIO Port H	178
0x4002.8000	0x4002.8FFF	PWM	407
0x4002.9000	0x4002.BFFF	Reserved	-

Table 3-1. Memory Map (continued)

Start	End	Description	For details on registers, see page ...
0x4002.C000	0x4002.CFFF	QEI0	440
0x4002.D000	0x4002.DFFF	QEI1	440
0x4002.E000	0x4002.FFFF	Reserved	-
0x4003.0000	0x4003.0FFF	Timer0	224
0x4003.1000	0x4003.1FFF	Timer1	224
0x4003.2000	0x4003.2FFF	Timer2	224
0x4003.3000	0x4003.3FFF	Timer3	224
0x4003.4000	0x4003.BFFF	Reserved	-
0x4003.C000	0x4003.CFFF	Analog Comparators	387
0x4003.D000	0x400F.BFFF	Reserved	-
0x400F.C000	0x400F.CFFF	Hibernation Module	135
0x400F.D000	0x400F.DFFF	Flash control	152
0x400F.E000	0x400F.EFFF	System control	74
0x400F.F000	0x41FF.FFFF	Reserved	-
0x4200.0000	0x43FF.FFFF	Bit-banded alias of 0x4000.0000 through 0x400F.FFFF	-
0x4400.0000	0xDFFF.FFFF	Reserved	-
Private Peripheral Bus			
0xE000.0000	0xE000.0FFF	Instrumentation Trace Macrocell (ITM)	ARM® Cortex™-M3 Technical Reference Manual
0xE000.1000	0xE000.1FFF	Data Watchpoint and Trace (DWT)	ARM® Cortex™-M3 Technical Reference Manual
0xE000.2000	0xE000.2FFF	Flash Patch and Breakpoint (FPB)	ARM® Cortex™-M3 Technical Reference Manual
0xE000.3000	0xE000.DFFF	Reserved	-
0xE000.E000	0xE000.EFFF	Nested Vectored Interrupt Controller (NVIC)	ARM® Cortex™-M3 Technical Reference Manual
0xE000.F000	0xE003.FFFF	Reserved	-
0xE004.0000	0xE004.0FFF	Trace Port Interface Unit (TPIU)	ARM® Cortex™-M3 Technical Reference Manual
0xE004.1000	0xFFFF.FFFF	Reserved	-

- 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.

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 49 lists all exception types. Software can set eight priority levels on seven of these exceptions (system handlers) as well as on 36 interrupts (listed in Table 4-2 on page 50).

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 also can group priorities by splitting priority levels into pre-emption priorities and subpriorities. All of 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 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.

Important: It may take several processor cycles after a write to clear an interrupt source in order for NVIC to see the interrupt source de-assert. This means if the interrupt clear is done as the last action in an interrupt handler, it is possible for the interrupt handler to complete while NVIC sees the interrupt as still asserted, causing the interrupt handler to be re-entered errantly. This can be avoided by either clearing the interrupt source at the beginning of the interrupt handler or by performing a read or write after the write to clear the interrupt source (and flush the write buffer).

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.

Table 4-1. Exception Types

Exception Type	Vector Number	Priority ^a	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 Interrupt Control State 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.

Table 4-1. Exception Types (continued)

Exception Type	Vector Number	Priority ^a	Description
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.
SVCcall	11	settable	System service call with SVC instruction. This is synchronous.
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 50 lists the interrupts on the LM3S1960 controller.

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

Table 4-2. Interrupts

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Description
0-15	-	Processor exceptions
16	0	GPIO Port A
17	1	GPIO Port B
18	2	GPIO Port C
19	3	GPIO Port D
20	4	GPIO Port E
21	5	UART0
22	6	UART1
23	7	SSI0
24	8	I2C0
25	9	PWM Fault
26	10	PWM Generator 0
27	11	PWM Generator 1
28	12	PWM Generator 2
29	13	QEI0
30-33	14-17	Reserved
34	18	Watchdog timer
35	19	Timer0 A
36	20	Timer0 B
37	21	Timer1 A
38	22	Timer1 B

Table 4-2. Interrupts (*continued*)

Vector Number	Interrupt Number (Bit in Interrupt Registers)	Description
39	23	Timer2 A
40	24	Timer2 B
41	25	Analog Comparator 0
42	26	Analog Comparator 1
43	27	Analog Comparator 2
44	28	System Control
45	29	Flash Control
46	30	GPIO Port F
47	31	GPIO Port G
48	32	GPIO Port H
49	33	UART2
50	34	SSI1
51	35	Timer3 A
52	36	Timer3 B
53	37	I2C1
54	38	QE11
55-58	39-42	Reserved
59	43	Hibernation Module
60-70	44-54	Reserved

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 five pins: $\overline{\text{TRST}}$, 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 Stellaris[®] 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 Stellaris[®] JTAG instructions select the Stellaris[®] TDO outputs. The multiplexer is controlled by the Stellaris[®] JTAG controller, which has comprehensive programming for the ARM, Stellaris[®], and unimplemented JTAG instructions.

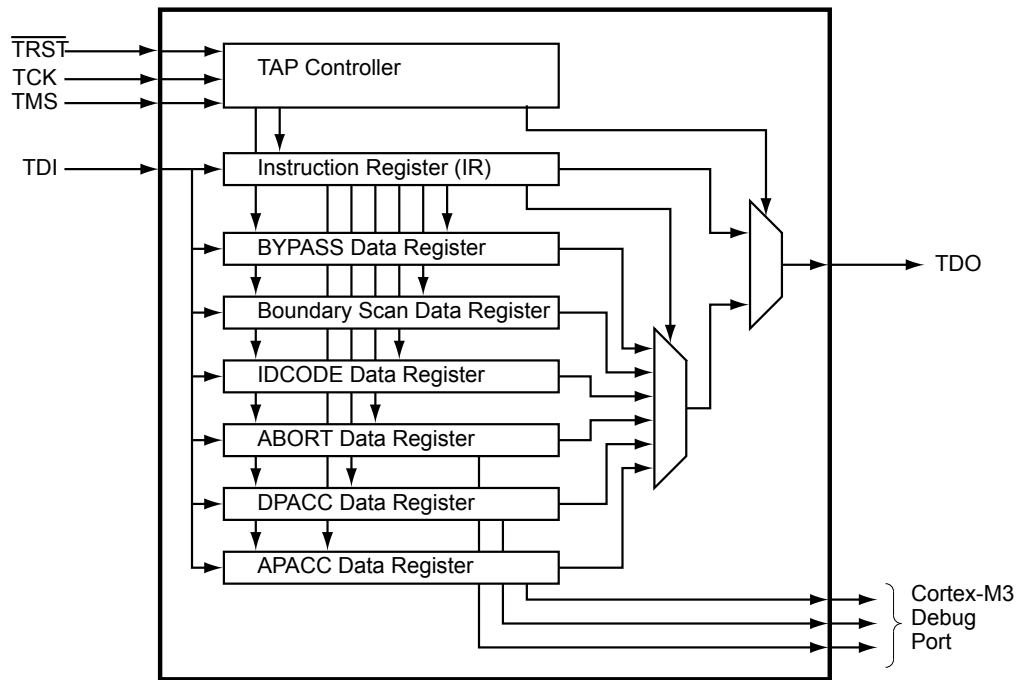
The Stellaris[®] JTAG module has the following features:

- IEEE 1149.1-1990 compatible Test Access Port (TAP) controller
- Four-bit Instruction Register (IR) chain for storing JTAG instructions
- IEEE standard instructions: BYPASS, IDCODE, SAMPLE/PRELOAD, EXTEST and INTEST
- ARM additional instructions: APACC, DPACC and ABORT
- 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 53. 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 $\overline{\text{TRST}}$, TCK and TMS inputs. The current state of the TAP controller depends on the current value of $\overline{\text{TRST}}$ 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 59 for a list of implemented instructions).

See “JTAG and Boundary Scan” on page 489 for JTAG timing diagrams.

5.2.1 JTAG Interface Pins

The JTAG interface consists of five standard pins: $\overline{\text{TRST}}$, TCK, TMS, TDI, and TDO. These pins and their associated reset state are given in Table 5-1 on page 54. 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
$\overline{\text{TRST}}$	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

5.2.1.1 Test Reset Input ($\overline{\text{TRST}}$)

The $\overline{\text{TRST}}$ pin is an asynchronous active Low input signal for initializing and resetting the JTAG TAP controller and associated JTAG circuitry. When $\overline{\text{TRST}}$ is asserted, the TAP controller resets to the Test-Logic-Reset state and remains there while $\overline{\text{TRST}}$ 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 $\overline{\text{TRST}}$ 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/ $\overline{\text{TRST}}$; otherwise JTAG communication could be lost.

5.2.1.2 Test Clock Input (TCK)

The 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, TCK is driven by a free-running clock with a nominal 50% duty cycle. When necessary, TCK can be stopped at 0 or 1 for extended periods of time. While 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 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 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 $\overline{\text{TRST}}$. The JTAG Test Access Port state machine can be seen in its entirety in Figure 5-2 on page 56.

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)

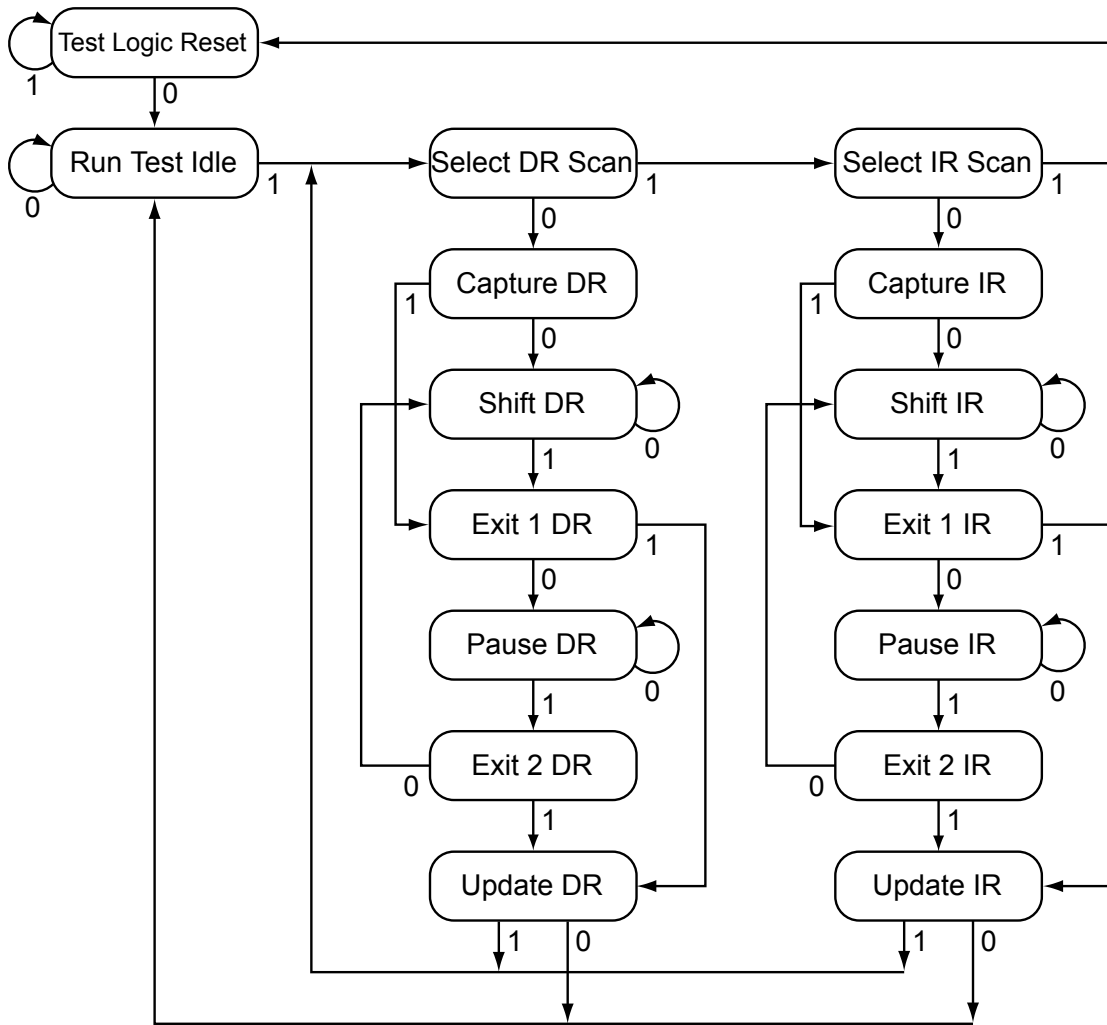
The TDO pin provides an output stream of serial information from the IR chain or the DR chains. The value of TDO depends on the current TAP state, the current instruction, and the data in the chain being accessed. In order to save power when the JTAG port is not being used, the TDO pin is placed in an 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 56. 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 $\overline{\text{TRST}}$. 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*.

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 59.

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 $\overline{\text{RST}}$, 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 **GIOPUR** to 1), and enabling the alternate hardware function (setting **GPIOAFSEL** to 1) for the PB7 and $\text{PC}[3:0]$ JTAG/SWD pins.

It is possible for software to configure these pins as GPIOs after reset by writing 0s to PB7 and $\text{PC}[3:0]$ 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 – 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 GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is currently provided for the five JTAG/SWD pins (PB7 and $\text{PC}[3:0]$). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 188) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 198) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 199) have been set to 1.

Recovering a "Locked" Device

Note: Performing the sequence below causes the nonvolatile registers discussed in “Nonvolatile Register Programming” on page 151 to be restored to their factory default values. The mass erase of the flash memory caused by the below sequence occurs prior to the nonvolatile registers being restored.

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 $\overline{\text{RST}}$ signal.
2. Perform the JTAG-to-SWD switch sequence.
3. Perform the SWD-to-JTAG switch sequence.
4. Perform the JTAG-to-SWD switch sequence.
5. Perform the SWD-to-JTAG switch sequence.
6. Perform the JTAG-to-SWD switch sequence.
7. Perform the SWD-to-JTAG switch sequence.
8. Perform the JTAG-to-SWD switch sequence.
9. Perform the SWD-to-JTAG switch sequence.
10. Perform the JTAG-to-SWD switch sequence.

11. Perform the SWD-to-JTAG switch sequence.
12. Release the $\overline{\text{RST}}$ signal.
13. Wait 400 ms.
14. Power-cycle the device.

The JTAG-to-SWD and SWD-to-JTAG switch sequences are described in “ARM Serial Wire Debug (SWD)” on page 58. When performing switch sequences for the purpose of recovering the debug capabilities of the device, only steps 1 and 2 of the switch sequence in the section called “JTAG-to-SWD Switching” on page 58 must be performed.

5.2.4.2 Communication with JTAG/SWD

Because the debug clock and the system clock can be running at different frequencies, care must be taken to maintain reliable communication with the JTAG/SWD interface. In the Capture-DR state, the result of the previous transaction, if any, is returned, together with a 3-bit ACK response. Software should check the ACK response to see if the previous operation has completed before initiating a new transaction. Alternatively, if the system clock is at least 8 times faster than the debug clock (TCK or SWCLK), the previous operation has enough time to complete and the ACK bits do not have to be checked.

5.2.4.3 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 switching 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 the switching preamble 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 b1110011100111100, 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 ($\overline{\text{RST}}$), 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. In addition to enabling the alternate functions, any other changes to the GPIO pad configurations on the five JTAG pins (PB7 and PC[3:0]) should be reverted to their default settings.

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 connected between the JTAG TDI and TDO pins with a parallel load register. 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 59. A detailed explanation of each instruction, along with its associated Data Register, follows.

Table 5-2. JTAG Instruction Register Commands

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.

Table 5-2. JTAG Instruction Register Commands (continued)

IR[3:0]	Instruction	Description
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 is not associated with its own 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. While the EXTEST instruction is present in the Instruction Register, the Boundary Scan Data Register can be accessed to sample and shift out the current data and load new data into the Boundary Scan Data Register.

5.4.1.2 INTEST Instruction

The INTEST instruction is not associated with its own 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{RST} input pin is on the Boundary Scan Data Register chain, it is only observable. While the INTEXT instruction is present in the Instruction Register, the Boundary Scan Data Register can be accessed to sample and shift out the current data and load new data into the Boundary Scan Data Register.

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 62 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 63 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 63 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 63 for more information.

5.4.1.7 IDCODE Instruction

The IDCODE instruction connects the associated IDCODE Data Register chain between TDI and TDO. 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, TRST is asserted, or the Test-Logic-Reset state is entered. Please see “IDCODE Data Register” on page 62 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 62 for more information.

5.4.2 Data Registers

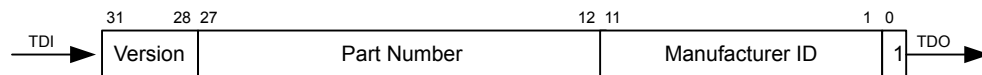
The JTAG module contains six Data Registers. These include: IDCODE, BYPASS, Boundary Scan, APACC, DPACC, and ABORT serial Data Register chains. Each of these Data Registers is discussed in the following sections.

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 62. 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 0x3BA0.0477. This allows the debuggers to automatically configure themselves to work correctly with the Cortex-M3 during debug.

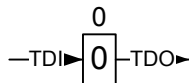
Figure 5-3. IDCODE Register Format



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 62. 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

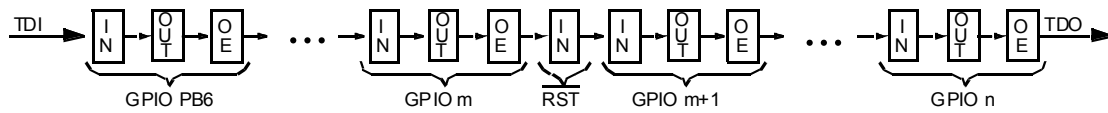


5.4.2.3 Boundary Scan Data Register

The format of the Boundary Scan Data Register is shown in Figure 5-5 on page 63. Each GPIO pin, starting with a GPIO pin next to 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.

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 T_{CK} 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



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*.

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 64
- Local control, such as reset (see “Reset Control” on page 64), power (see “Power Control” on page 67) and clock control (see “Clock Control” on page 68)
- System control (Run, Sleep, and Deep-Sleep modes), see “System Control” on page 72

6.1.1 Device Identification

Several 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 internal use for testing the microcontroller 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:

1. External reset input pin ($\overline{\text{RST}}$) assertion, see “External $\overline{\text{RST}}$ Pin” on page 65.
2. Power-on reset (POR), see “Power-On Reset (POR)” on page 64.
3. Internal brown-out (BOR) detector, see “Brown-Out Reset (BOR)” on page 66.
4. Software-initiated reset (with the software reset registers), see “Software Reset” on page 67.
5. A watchdog timer reset condition violation, see “Watchdog Timer Reset” on page 67.

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 Power-On Reset (POR)

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

The internal Power-On Reset (POR) circuit monitors the power supply voltage (V_{DD}) and generates a reset signal to all of the internal logic including JTAG when the power supply ramp reaches a threshold value (V_{TH}). The microcontroller must be operating within the specified operating parameters when the on-chip power-on reset pulse is complete. The 3.3-V power supply to the microcontroller must reach 3.0 V within 10 msec of V_{DD} crossing 2.0 V to guarantee proper operation. For applications that require the use of an external reset signal to hold the microcontroller in reset longer than the internal POR, the \overline{RST} input may be used as discussed in “External \overline{RST} Pin” on page 65.

The Power-On Reset sequence is as follows:

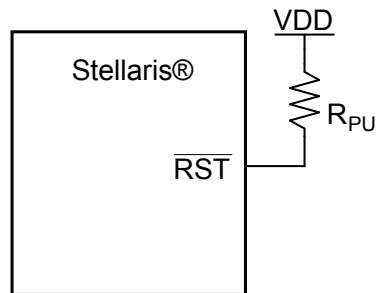
1. The microcontroller waits for internal POR to go inactive.
2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

The internal POR is only active on the initial power-up of the microcontroller. The Power-On Reset timing is shown in Figure 21-6 on page 492.

6.1.2.4 External \overline{RST} Pin

If the application only uses the internal POR circuit, the \overline{RST} input must be connected to the power supply (V_{DD}) through an optional pull-up resistor (0 to 10K Ω) as shown in Figure 6-1 on page 65.

Figure 6-1. Basic \overline{RST} Configuration

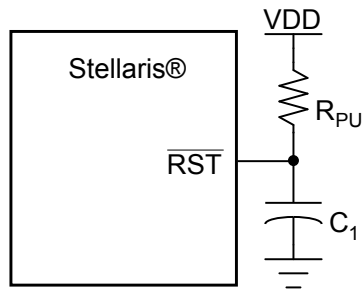


$R_{PU} = 0$ to 100 k Ω

The external reset pin (\overline{RST}) resets the microcontroller including the core and all the on-chip peripherals except the JTAG TAP controller (see “JTAG Interface” on page 52). The external reset sequence is as follows:

1. The external reset pin (\overline{RST}) is asserted for the duration specified by T_{MIN} and then de-asserted (see “Reset” on page 491).
2. The internal reset is released and the core loads from memory the initial stack pointer, the initial program counter, and the first instruction designated by the program counter, and then begins execution.

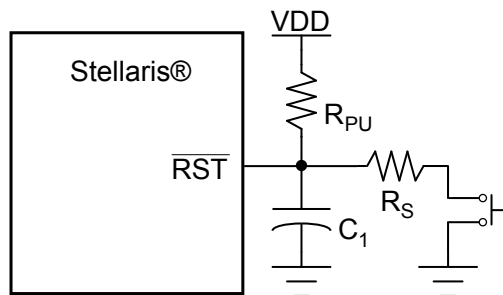
To improve noise immunity and/or to delay reset at power up, the \overline{RST} input may be connected to an RC network as shown in Figure 6-2 on page 66.

Figure 6-2. External Circuitry to Extend Power-On Reset

$R_{PU} = 1 \text{ k}\Omega \text{ to } 100 \text{ k}\Omega$

$C_1 = 1 \text{ nF to } 10 \text{ }\mu\text{F}$

If the application requires the use of an external reset switch, Figure 6-3 on page 66 shows the proper circuitry to use.

Figure 6-3. Reset Circuit Controlled by Switch

Typical $R_{PU} = 10 \text{ k}\Omega$

Typical $R_S = 470 \text{ }\Omega$

$C_1 = 10 \text{ nF}$

The R_{PU} and C_1 components define the power-on delay.

The external reset timing is shown in Figure 21-5 on page 491.

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 equivalent to an assertion of the external \overline{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 21-7 on page 492.

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 72). 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:

1. 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.
3. 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 21-8 on page 492.

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.
3. 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.

The watchdog reset timing is shown in Figure 21-9 on page 492.

6.1.3 Power Control

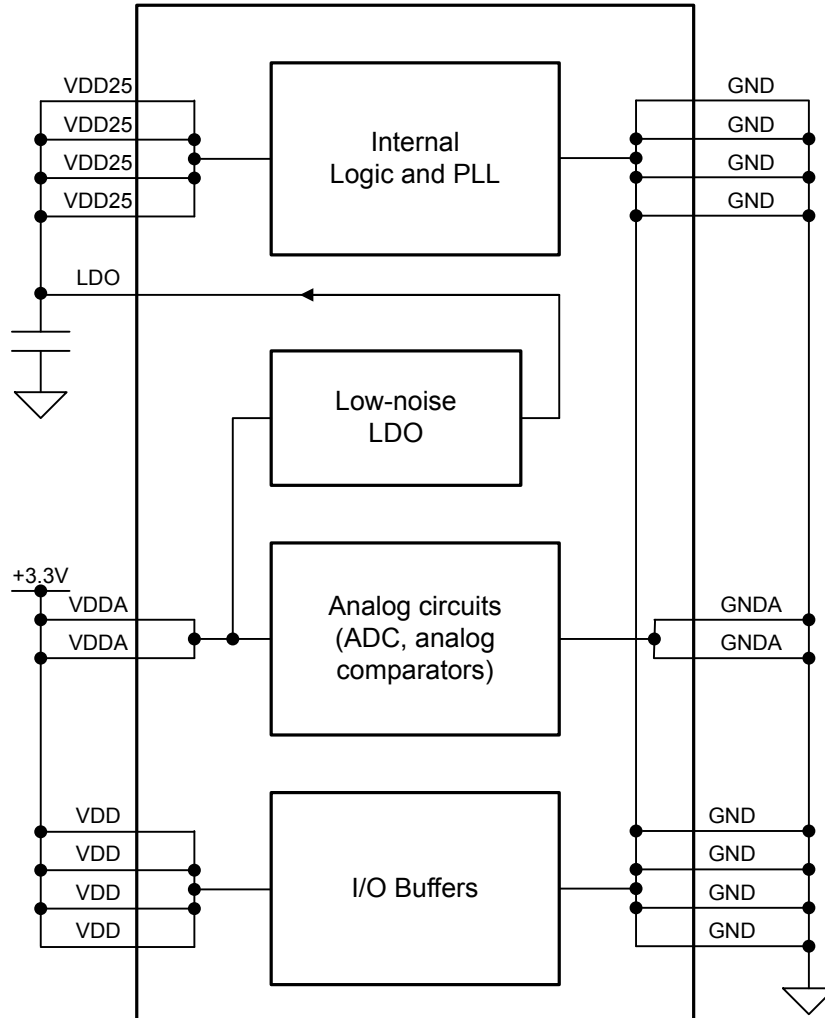
The Stellaris® microcontroller provides an integrated LDO regulator that may be used to provide power to the majority of the controller's internal logic. For power reduction, the LDO regulator provides

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 $2.5\text{ V} \pm 10\%$. The adjustment is made by changing the value of the V_{ADJ} field in the **LDO Power Control (LDOPCTL)** register.

Figure 6-4 on page 68 shows the power architecture.

Note: On the printed circuit board, use the LDO output as the source of V_{DD25} input. In addition, the LDO requires decoupling capacitors. See “On-Chip Low Drop-Out (LDO) Regulator Characteristics” on page 485.

Figure 6-4. Power Architecture



6.1.4 Clock Control

System control determines the control of clocks in this part.

6.1.4.1 Fundamental Clock Sources

There are multiple clock sources for use in the device:

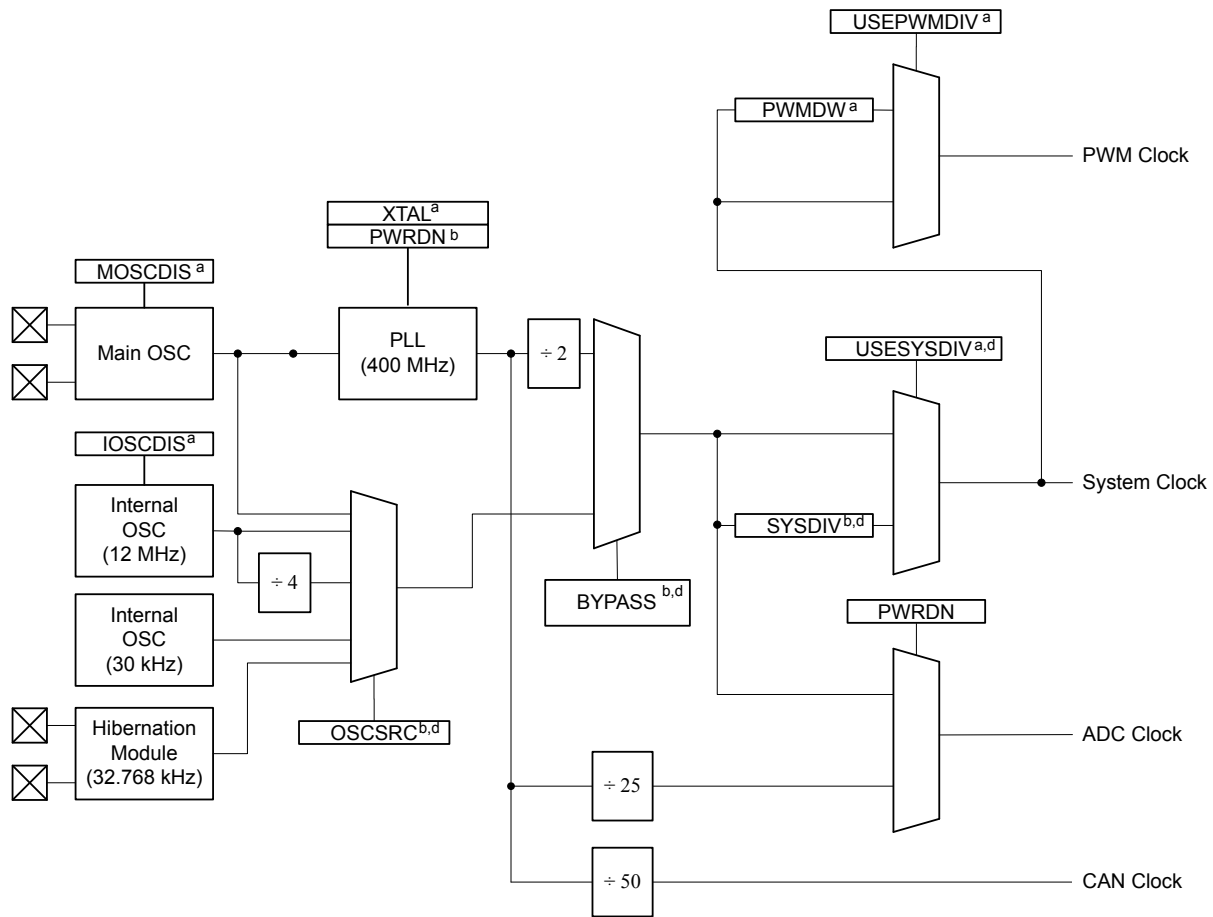
- **Internal Oscillator (IOSC).** The internal oscillator is an on-chip clock source. It does not require the use of any external components. The frequency of the internal oscillator is 12 MHz \pm 30%. Applications that do not depend on accurate clock sources may use this clock source to reduce system 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 (MOSC).** 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. If the PLL is being used, the crystal value 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 field in the **RCC** register (see page 83).
- **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 \pm 50%. 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.
- **External Real-Time Oscillator.** The external real-time oscillator provides a low-frequency, accurate clock reference. It is intended to provide the system with a real-time clock source. The real-time oscillator is part of the Hibernation Module (see “Hibernation Module” on page 128) and may also provide an accurate source of Deep-Sleep or Hibernate mode power savings.

The internal system clock (SysClk), is derived from any of the above sources plus two others: the output of the main 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 used by the logic over the corresponding field in the **RCC** register. In particular, **RCC2** provides for a larger assortment of clock configuration options.

Figure 6-5 on page 70 shows the logic for the main clock tree. The peripheral blocks are driven by the system clock signal and can be individually enabled/disabled. The PWM clock signal is a synchronous divide of the system clock to provide the PWM circuit with more range (set with **PWMDIV** in **RCC**).

Figure 6-5. Main Clock Tree



- a. Control provided by RCC register bit/field.
 b. Control provided by RCC register bit/field or RCC2 register bit/field, if overridden with RCC2 register bit USERCC2.
 c. Control provided by RCC2 register bit/field.
 d. Also may be controlled by DSLPCLKCFG when in deep sleep mode.

Note: The figure above shows all features available on all Stellaris® Fury-class devices.

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 83) 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 Main PLL Frequency Configuration

The main PLL is disabled by default during power-on reset and is enabled later by software if required. Software specifies the output divisor to set the system clock frequency, and enables the main PLL to drive the output. The PLL operates at 400 MHz, but is divided by two prior to the application of the output divisor.

If the main oscillator provides the clock reference to the main 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 88). The internal translation provides a translation within $\pm 1\%$ of the targeted PLL VCO frequency. Table 21-9 on page 488 shows the actual PLL frequency and error for a given crystal choice.

The Crystal Value field (*XTAL*) on page 83 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.

To configure the external 32-kHz real-time oscillator as the PLL input reference, program the *OSCR2* field in the **Run-Mode Clock Configuration 2 (RCC2)** register to be 0x7.

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 83 and page 89).

6.1.4.5 PLL Operation

If a 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_{READY} (see Table 21-8 on page 488). During the relock time, the affected 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, $\sim 600 \mu\text{s}$ 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.

If the main PLL is enabled and the system clock is switched to use the PLL in one step, the system control hardware continues to clock the controller from the oscillator selected by the **RCC/RCC2** register until the main PLL is stable (T_{READY} time met), after which it changes to the PLL. Software can use many methods to ensure that the system is clocked from the main PLL, including periodically polling the *PLLRIS* bit in the **Raw Interrupt Status (RIS)** register, and enabling the PLL Lock interrupt.

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.

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

- **Run Mode.** In Run mode, the controller actively executes code. 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.** In Sleep mode, the clock frequency of the active peripherals is unchanged, but the processor and the memory subsystem are not clocked and therefore no longer execute code. 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.

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.** 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. 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 **DSLPCCLKCFG** register if one is enabled. When the **DSLPCCLKCFG** 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 determined by the **DSDIVORIDE** setting in the **DSLPCCLKCFG** register, up to /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.

- **Hibernate Mode.** In this mode, the power supplies are turned off to the main part of the device and only the Hibernation module's circuitry is active. An external wake event or RTC event is required to bring the device back to Run mode. The Cortex-M3 processor and peripherals outside of the Hibernation module see a normal "power on" sequence and the processor starts running code. It can determine that it has been restarted from Hibernate mode by inspecting the Hibernation module registers.

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:

1. 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 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.
4. 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 73 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. Software should not modify any reserved memory address.

Table 6-1. System Control Register Map

Offset	Name	Type	Reset	Description	See page
0x000	DID0	RO	-	Device Identification 0	75
0x004	DID1	RO	-	Device Identification 1	92
0x008	DC0	RO	0x00FF.007F	Device Capabilities 0	94
0x010	DC1	RO	0x0010.30DF	Device Capabilities 1	95
0x014	DC2	RO	0x070F.5337	Device Capabilities 2	97
0x018	DC3	RO	0xBF00.FFFF	Device Capabilities 3	99
0x01C	DC4	RO	0x0000.C0FF	Device Capabilities 4	101
0x030	PBORCTL	R/W	0x0000.7FFD	Brown-Out Reset Control	77
0x034	LDOPCTL	R/W	0x0000.0000	LDO Power Control	78
0x040	SRCR0	R/W	0x00000000	Software Reset Control 0	124
0x044	SRCR1	R/W	0x00000000	Software Reset Control 1	125
0x048	SRCR2	R/W	0x00000000	Software Reset Control 2	127
0x050	RIS	RO	0x0000.0000	Raw Interrupt Status	79

Table 6-1. System Control Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x054	IMC	R/W	0x0000.0000	Interrupt Mask Control	80
0x058	MISC	R/W1C	0x0000.0000	Masked Interrupt Status and Clear	81
0x05C	RESC	R/W	-	Reset Cause	82
0x060	RCC	R/W	0x078E.3AD1	Run-Mode Clock Configuration	83
0x064	PLLCFG	RO	-	XTAL to PLL Translation	88
0x070	RCC2	R/W	0x0780.2810	Run-Mode Clock Configuration 2	89
0x100	RCGC0	R/W	0x00000040	Run Mode Clock Gating Control Register 0	103
0x104	RCGC1	R/W	0x00000000	Run Mode Clock Gating Control Register 1	109
0x108	RCGC2	R/W	0x00000000	Run Mode Clock Gating Control Register 2	118
0x110	SCGC0	R/W	0x00000040	Sleep Mode Clock Gating Control Register 0	105
0x114	SCGC1	R/W	0x00000000	Sleep Mode Clock Gating Control Register 1	112
0x118	SCGC2	R/W	0x00000000	Sleep Mode Clock Gating Control Register 2	120
0x120	DCGC0	R/W	0x00000040	Deep Sleep Mode Clock Gating Control Register 0	107
0x124	DCGC1	R/W	0x00000000	Deep Sleep Mode Clock Gating Control Register 1	115
0x128	DCGC2	R/W	0x00000000	Deep Sleep Mode Clock Gating Control Register 2	122
0x144	DSLPCCLKCFG	R/W	0x0780.0000	Deep Sleep Clock Configuration	91

6.4 Register Descriptions

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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved	VER			reserved				CLASS							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MAJOR								MINOR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit/Field	Name	Type	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	<p>DID0 Version</p> <p>This field defines the DID0 register format version. The version number is numeric. The value of the <code>VER</code> field is encoded as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x1</td> <td>Second version of the DID0 register format.</td> </tr> </tbody> </table>	Value	Description	0x1	Second version of the DID0 register format.
Value	Description							
0x1	Second version of the DID0 register format.							
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	<p>Device Class</p> <p>The <code>CLASS</code> field value identifies the internal design from which all mask sets are generated for all devices in a particular product line. The <code>CLASS</code> field value is changed for new product lines, for changes in fab process (for example, a remap or shrink), or any case where the <code>MAJOR</code> or <code>MINOR</code> fields require differentiation from prior devices. The value of the <code>CLASS</code> field is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x1</td> <td>Stellaris® Fury-class devices.</td> </tr> </tbody> </table>	Value	Description	0x1	Stellaris® Fury-class devices.
Value	Description							
0x1	Stellaris® Fury-class devices.							

Bit/Field	Name	Type	Reset	Description								
15:8	MAJOR	RO	-	<p>Major Revision</p> <p>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:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0</td><td>Revision A (initial device)</td></tr><tr><td>0x1</td><td>Revision B (first base layer revision)</td></tr><tr><td>0x2</td><td>Revision C (second base layer revision)</td></tr></tbody></table> <p>and so on.</p>	Value	Description	0x0	Revision A (initial device)	0x1	Revision B (first base layer revision)	0x2	Revision C (second base layer revision)
Value	Description											
0x0	Revision A (initial device)											
0x1	Revision B (first base layer revision)											
0x2	Revision C (second base layer revision)											
7:0	MINOR	RO	-	<p>Minor Revision</p> <p>This field specifies the minor revision number of the device. The minor revision reflects changes to the metal layers of the design. The <code>MINOR</code> field value is reset when the <code>MAJOR</code> field is changed. This field is numeric and is encoded as follows:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0</td><td>Initial device, or a major revision update.</td></tr><tr><td>0x1</td><td>First metal layer change.</td></tr><tr><td>0x2</td><td>Second metal layer change.</td></tr></tbody></table> <p>and so on.</p>	Value	Description	0x0	Initial device, or a major revision update.	0x1	First metal layer change.	0x2	Second metal layer change.
Value	Description											
0x0	Initial device, or a major revision update.											
0x1	First metal layer change.											
0x2	Second metal layer change.											

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															BORIOR	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

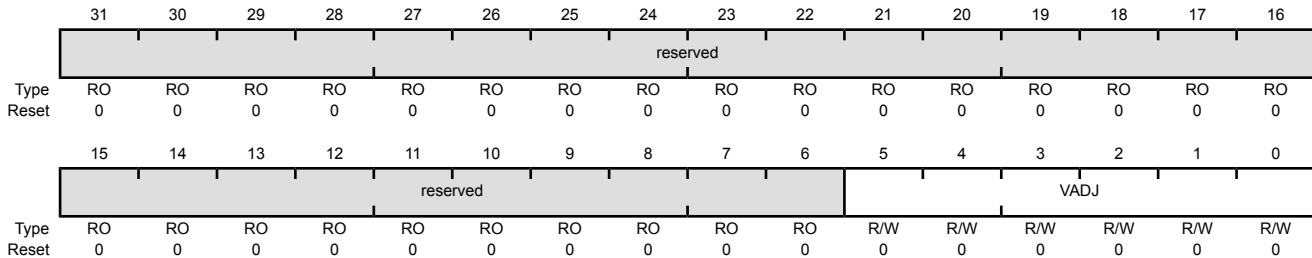
Bit/Field	Name	Type	Reset	Description
31:2	reserved	RO	0x0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
1	BORIOR	R/W	0	BOR Interrupt or Reset This bit controls 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 V_{ADJ} 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	Type	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 V_{ADJ} 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved										PLLLRIS	reserved				BORRIS	reserved
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	

Bit/Field	Name	Type	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 IMC register is set and the BORIOR bit in the PBORCTL 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved										PLLLIM	reserved				BORIM	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	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
31:7	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 PLL Lock interrupt is promoted to a controller interrupt. If set, an interrupt is generated if <code>PLLLRIS</code> in RIS 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 <code>BORRIS</code> 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

On a read, this register gives the current masked status value of the corresponding interrupt. All of the bits are R/W1C and this action also clears the corresponding raw interrupt bit in the **RIS** register (see page 79).

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	
	reserved																
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	
	reserved										PLLLMIS	reserved				BORMIS	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	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
31:7	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	PLLLMIS	R/W1C	0	PLL Lock Masked Interrupt Status This bit is set when the PLL T _{READY} timer asserts. The interrupt is cleared by writing a 1 to this bit.
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	BORMIS	R/W1C	0	BOR Masked Interrupt Status The BORMIS is simply the BORRIS ANDed with the mask value, BORIM.
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 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 power-on reset is the cause, in which case, all bits other than POR in the **RESC** register are cleared.

Reset Cause (RESC)

Base 0x400F.E000

Offset 0x05C

Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved												SW	WDT	BOR	POR	EXT
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0	0	0	0	-	-	-	-	-	

Bit/Field	Name	Type	Reset	Description
31:5	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	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 (\overline{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 0x078E.3AD1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved				ACG	SYSDIV				USESYSDIV	reserved	USEPWMDIV	PWMDIV			reserved
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved		PWRDN	reserved	BYPASS	reserved	XTAL				OSCSRC		reserved		IOSCDIS	MOSCDIS
Type	RO	RO	R/W	RO	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W	R/W
Reset	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1

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	<p>Auto Clock Gating</p> <p>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.</p> <p>The RCGCn registers are always used to control the clocks in Run mode.</p> <p>This allows peripherals to consume less power when the controller is in a sleep mode and the peripheral is unused.</p>

Bit/Field	Name	Type	Reset	Description																																																			
26:23	SYSDIV	R/W	0xF	<p>System Clock Divisor</p> <p>Specifies which divisor is used to generate the system clock from the PLL output.</p> <p>Although the PLL VCO frequency is 400 MHz, it is predivided by 2 before the divisor is applied.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Divisor (BYPASS=1)</th> <th>Frequency (BYPASS=0)</th> </tr> </thead> <tbody> <tr><td>0x0</td><td>reserved</td><td>reserved</td></tr> <tr><td>0x1</td><td>/2</td><td>reserved</td></tr> <tr><td>0x2</td><td>/3</td><td>reserved</td></tr> <tr><td>0x3</td><td>/4</td><td>50 MHz</td></tr> <tr><td>0x4</td><td>/5</td><td>40 MHz</td></tr> <tr><td>0x5</td><td>/6</td><td>33.33 MHz</td></tr> <tr><td>0x6</td><td>/7</td><td>28.57 MHz</td></tr> <tr><td>0x7</td><td>/8</td><td>25 MHz</td></tr> <tr><td>0x8</td><td>/9</td><td>22.22 MHz</td></tr> <tr><td>0x9</td><td>/10</td><td>20 MHz</td></tr> <tr><td>0xA</td><td>/11</td><td>18.18 MHz</td></tr> <tr><td>0xB</td><td>/12</td><td>16.67 MHz</td></tr> <tr><td>0xC</td><td>/13</td><td>15.38 MHz</td></tr> <tr><td>0xD</td><td>/14</td><td>14.29 MHz</td></tr> <tr><td>0xE</td><td>/15</td><td>13.33 MHz</td></tr> <tr><td>0xF</td><td>/16</td><td>12.5 MHz (default)</td></tr> </tbody> </table> <p>When reading the Run-Mode Clock Configuration (RCC) register (see page 83), the SYSDIV value is MINSYSDIV if a lower divider was requested and the PLL is being used. This lower value is allowed to divide a non-PLL source.</p>	Value	Divisor (BYPASS=1)	Frequency (BYPASS=0)	0x0	reserved	reserved	0x1	/2	reserved	0x2	/3	reserved	0x3	/4	50 MHz	0x4	/5	40 MHz	0x5	/6	33.33 MHz	0x6	/7	28.57 MHz	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)
Value	Divisor (BYPASS=1)	Frequency (BYPASS=0)																																																					
0x0	reserved	reserved																																																					
0x1	/2	reserved																																																					
0x2	/3	reserved																																																					
0x3	/4	50 MHz																																																					
0x4	/5	40 MHz																																																					
0x5	/6	33.33 MHz																																																					
0x6	/7	28.57 MHz																																																					
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)																																																					
22	USESYSCLK	R/W	0	<p>Enable System Clock Divider</p> <p>Use the system clock divider as the source for the system clock. The system clock divider is forced to be used when the PLL is selected as the source.</p>																																																			
21	reserved	RO	0	<p>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.</p>																																																			
20	USEPWMDIV	R/W	0	<p>Enable PWM Clock Divisor</p> <p>Use the PWM clock divider as the source for the PWM clock.</p>																																																			

Bit/Field	Name	Type	Reset	Description
19:17	PWMDIV	R/W	0x7	<p>PWM Unit Clock Divisor</p> <p>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.</p> <p>Value Divisor</p> <p>0x0 /2</p> <p>0x1 /4</p> <p>0x2 /8</p> <p>0x3 /16</p> <p>0x4 /32</p> <p>0x5 /64</p> <p>0x6 /64</p> <p>0x7 /64 (default)</p>
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	<p>PLL Power Down</p> <p>This bit connects to the PLL PWRDN input. The reset value of 1 powers down the PLL.</p>
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	<p>PLL Bypass</p> <p>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.</p>
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	Type	Reset	Description																																																			
9:6	XTAL	R/W	0xB	<p>Crystal Value</p> <p>This field specifies the crystal value attached to the main oscillator. The encoding for this field is provided below.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Crystal Frequency (MHz) Not Using the PLL</th> <th>Crystal Frequency (MHz) Using the PLL</th> </tr> </thead> <tbody> <tr><td>0x0</td><td>1.000</td><td>reserved</td></tr> <tr><td>0x1</td><td>1.8432</td><td>reserved</td></tr> <tr><td>0x2</td><td>2.000</td><td>reserved</td></tr> <tr><td>0x3</td><td>2.4576</td><td>reserved</td></tr> <tr><td>0x4</td><td></td><td>3.579545 MHz</td></tr> <tr><td>0x5</td><td></td><td>3.6864 MHz</td></tr> <tr><td>0x6</td><td></td><td>4 MHz</td></tr> <tr><td>0x7</td><td></td><td>4.096 MHz</td></tr> <tr><td>0x8</td><td></td><td>4.9152 MHz</td></tr> <tr><td>0x9</td><td></td><td>5 MHz</td></tr> <tr><td>0xA</td><td></td><td>5.12 MHz</td></tr> <tr><td>0xB</td><td></td><td>6 MHz (reset value)</td></tr> <tr><td>0xC</td><td></td><td>6.144 MHz</td></tr> <tr><td>0xD</td><td></td><td>7.3728 MHz</td></tr> <tr><td>0xE</td><td></td><td>8 MHz</td></tr> <tr><td>0xF</td><td></td><td>8.192 MHz</td></tr> </tbody> </table>	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	0x4		3.579545 MHz	0x5		3.6864 MHz	0x6		4 MHz	0x7		4.096 MHz	0x8		4.9152 MHz	0x9		5 MHz	0xA		5.12 MHz	0xB		6 MHz (reset value)	0xC		6.144 MHz	0xD		7.3728 MHz	0xE		8 MHz	0xF		8.192 MHz
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																																																					
0x4		3.579545 MHz																																																					
0x5		3.6864 MHz																																																					
0x6		4 MHz																																																					
0x7		4.096 MHz																																																					
0x8		4.9152 MHz																																																					
0x9		5 MHz																																																					
0xA		5.12 MHz																																																					
0xB		6 MHz (reset value)																																																					
0xC		6.144 MHz																																																					
0xD		7.3728 MHz																																																					
0xE		8 MHz																																																					
0xF		8.192 MHz																																																					
5:4	OSCSRC	R/W	0x1	<p>Oscillator Source</p> <p>Selects the input source for the OSC. The values are:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Input Source</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>MOSC Main oscillator</td> </tr> <tr> <td>0x1</td> <td>IOSC Internal oscillator (default)</td> </tr> <tr> <td>0x2</td> <td>IOSC/4 Internal oscillator / 4 (this is necessary if used as input to PLL)</td> </tr> <tr> <td>0x3</td> <td>30 kHz 30-KHz internal oscillator</td> </tr> </tbody> </table> <p>For additional oscillator sources, see the RCC2 register.</p>	Value	Input Source	0x0	MOSC Main oscillator	0x1	IOSC Internal oscillator (default)	0x2	IOSC/4 Internal oscillator / 4 (this is necessary if used as input to PLL)	0x3	30 kHz 30-KHz internal oscillator																																									
Value	Input Source																																																						
0x0	MOSC Main oscillator																																																						
0x1	IOSC Internal oscillator (default)																																																						
0x2	IOSC/4 Internal oscillator / 4 (this is necessary if used as input to PLL)																																																						
0x3	30 kHz 30-KHz internal oscillator																																																						
3:2	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	IOSCDIS	R/W	0	<p>Internal Oscillator Disable</p> <p>0: Internal oscillator (IOSC) is enabled.</p> <p>1: Internal oscillator is disabled.</p>																																																			

Bit/Field	Name	Type	Reset	Description
0	MOSCDIS	R/W	1	Main Oscillator Disable 0: Main oscillator is enabled . 1: Main oscillator 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 83).

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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved		F					R								
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-

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.

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

This register overrides the **RCC** equivalent register fields when the **USERCC2** bit is set, allowing the extended capabilities of the **RCC2** register to be used 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 2 bits wider than the **SYSDIV** field in the **RCC** register so that additional larger divisors are possible, allowing a lower system clock frequency for improved Deep Sleep power consumption. The PLL VCO frequency is 400 MHz.

Run-Mode Clock Configuration 2 (RCC2)

Base 0x400F.E000

Offset 0x070

Type R/W, reset 0x0780.2810

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	USERCC2	reserved			SYSDIV2						reserved					
Type	R/W	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
	reserved		PWRDN2	reserved	BYPASS2	reserved				OSCSRC2			reserved			
Type	RO	RO	R/W	RO	R/W	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO
Reset	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31	USERCC2	R/W	0	Use RCC2 When set, overrides the RCC register fields.
30:29	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.
28:23	SYSDIV2	R/W	0x0F	System Clock Divisor Specifies which divisor is used to generate the system clock from the PLL output. Although the PLL VCO frequency is 400 MHz, it is predivided by 2 before the divisor is applied. This field is wider than the RCC register SYSDIV field in order to provide additional divisor values. This permits the system clock to be run at much lower frequencies during Deep Sleep mode. For example, where the RCC register SYSDIV encoding of 1111 provides /16, the RCC2 register SYSDIV2 encoding of 111111 provides /64.
22:14	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	PWRDN2	R/W	1	Power-Down PLL When set, powers down the PLL.
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.

Bit/Field	Name	Type	Reset	Description																		
11	BYPASS2	R/W	1	Bypass PLL When set, bypasses the PLL for the clock source.																		
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	0x1	Oscillator Source Selects the input source for the OSC. The values are: <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>MOSC Main oscillator</td> </tr> <tr> <td>0x1</td> <td>IOSC Internal oscillator</td> </tr> <tr> <td>0x2</td> <td>IOSC/4 Internal oscillator / 4</td> </tr> <tr> <td>0x3</td> <td>30 kHz 30-kHz internal oscillator</td> </tr> <tr> <td>0x4</td> <td>Reserved</td> </tr> <tr> <td>0x5</td> <td>Reserved</td> </tr> <tr> <td>0x6</td> <td>Reserved</td> </tr> <tr> <td>0x7</td> <td>32 kHz 32.768-kHz external oscillator</td> </tr> </tbody> </table>	Value	Description	0x0	MOSC Main oscillator	0x1	IOSC Internal oscillator	0x2	IOSC/4 Internal oscillator / 4	0x3	30 kHz 30-kHz internal oscillator	0x4	Reserved	0x5	Reserved	0x6	Reserved	0x7	32 kHz 32.768-kHz external oscillator
Value	Description																					
0x0	MOSC Main oscillator																					
0x1	IOSC Internal oscillator																					
0x2	IOSC/4 Internal oscillator / 4																					
0x3	30 kHz 30-kHz internal oscillator																					
0x4	Reserved																					
0x5	Reserved																					
0x6	Reserved																					
0x7	32 kHz 32.768-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 (DSLPCCLKCFG), offset 0x144

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

Deep Sleep Clock Configuration (DSLPCCLKCFG)

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			DSDIVORIDE						reserved						
Type	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
	reserved						DSOSCSRC				reserved					
Type	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/Field	Name	Type	Reset	Description																		
31:29	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.																		
28:23	DSDIVORIDE	R/W	0x0F	Divider Field Override 6-bit system divider field to override when Deep-Sleep occurs with PLL running.																		
22:7	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	DSOSCSRC	R/W	0x0	Clock Source Specifies the clock source during Deep-Sleep mode. <table border="0"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>MOSC Use main oscillator as source.</td> </tr> <tr> <td>0x1</td> <td>IOSC Use internal 12-MHz oscillator as source.</td> </tr> <tr> <td>0x2</td> <td>Reserved</td> </tr> <tr> <td>0x3</td> <td>30 kHz Use 30-kHz internal oscillator as source.</td> </tr> <tr> <td>0x4</td> <td>Reserved</td> </tr> <tr> <td>0x5</td> <td>Reserved</td> </tr> <tr> <td>0x6</td> <td>Reserved</td> </tr> <tr> <td>0x7</td> <td>32 kHz Use 32.768-kHz external oscillator as source.</td> </tr> </tbody> </table>	Value	Description	0x0	MOSC Use main oscillator as source.	0x1	IOSC Use internal 12-MHz oscillator as source.	0x2	Reserved	0x3	30 kHz Use 30-kHz internal oscillator as source.	0x4	Reserved	0x5	Reserved	0x6	Reserved	0x7	32 kHz Use 32.768-kHz external oscillator as source.
Value	Description																					
0x0	MOSC Use main oscillator as source.																					
0x1	IOSC Use internal 12-MHz oscillator as source.																					
0x2	Reserved																					
0x3	30 kHz Use 30-kHz internal oscillator as source.																					
0x4	Reserved																					
0x5	Reserved																					
0x6	Reserved																					
0x7	32 kHz Use 32.768-kHz external oscillator as source.																					
3:0	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.																		

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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VER				FAM				PARTNO							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	1	0	1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PINCOUNT			reserved				TEMP			PKG		ROHS	QUAL		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	0	0	0	0	0	-	-	-	-	-	1	-	-

Bit/Field	Name	Type	Reset	Description				
31:28	VER	RO	0x1	<p>DID1 Version</p> <p>This field defines the DID1 register format version. The version number is numeric. The value of the <code>VER</code> field is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x1</td> <td>Second version of the DID1 register format.</td> </tr> </tbody> </table>	Value	Description	0x1	Second version of the DID1 register format.
Value	Description							
0x1	Second version of the DID1 register format.							
27:24	FAM	RO	0x0	<p>Family</p> <p>This field provides the family identification of the device within the Luminary Micro product portfolio. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Stellaris family of microcontrollers, that is, all devices with external part numbers starting with LM3S.</td> </tr> </tbody> </table>	Value	Description	0x0	Stellaris family of microcontrollers, that is, all devices with external part numbers starting with LM3S.
Value	Description							
0x0	Stellaris family of microcontrollers, that is, all devices with external part numbers starting with LM3S.							
23:16	PARTNO	RO	0xB5	<p>Part Number</p> <p>This field provides the part number of the device within the family. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0xB5</td> <td>LM3S1960</td> </tr> </tbody> </table>	Value	Description	0xB5	LM3S1960
Value	Description							
0xB5	LM3S1960							
15:13	PINCOUNT	RO	0x2	<p>Package Pin Count</p> <p>This field specifies the number of pins on the device package. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x2</td> <td>100-pin or 108-ball package</td> </tr> </tbody> </table>	Value	Description	0x2	100-pin or 108-ball package
Value	Description							
0x2	100-pin or 108-ball package							

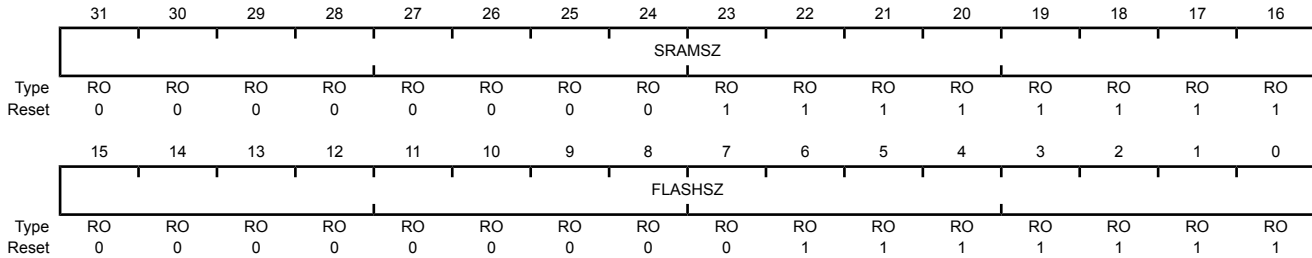
Bit/Field	Name	Type	Reset	Description								
12:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.								
7:5	TEMP	RO	-	<p>Temperature Range</p> <p>This field specifies the temperature rating of the device. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Commercial temperature range (0°C to 70°C)</td> </tr> <tr> <td>0x1</td> <td>Industrial temperature range (-40°C to 85°C)</td> </tr> <tr> <td>0x2</td> <td>Extended temperature range (-40°C to 105°C)</td> </tr> </tbody> </table>	Value	Description	0x0	Commercial temperature range (0°C to 70°C)	0x1	Industrial temperature range (-40°C to 85°C)	0x2	Extended temperature range (-40°C to 105°C)
Value	Description											
0x0	Commercial temperature range (0°C to 70°C)											
0x1	Industrial temperature range (-40°C to 85°C)											
0x2	Extended temperature range (-40°C to 105°C)											
4:3	PKG	RO	-	<p>Package Type</p> <p>This field specifies the package type. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>SOIC package</td> </tr> <tr> <td>0x1</td> <td>LQFP package</td> </tr> <tr> <td>0x2</td> <td>BGA package</td> </tr> </tbody> </table>	Value	Description	0x0	SOIC package	0x1	LQFP package	0x2	BGA package
Value	Description											
0x0	SOIC package											
0x1	LQFP package											
0x2	BGA package											
2	ROHS	RO	1	<p>RoHS-Compliance</p> <p>This bit specifies whether the device is RoHS-compliant. A 1 indicates the part is RoHS-compliant.</p>								
1:0	QUAL	RO	-	<p>Qualification Status</p> <p>This field specifies the qualification status of the device. The value is encoded as follows (all other encodings are reserved):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Engineering Sample (unqualified)</td> </tr> <tr> <td>0x1</td> <td>Pilot Production (unqualified)</td> </tr> <tr> <td>0x2</td> <td>Fully Qualified</td> </tr> </tbody> </table>	Value	Description	0x0	Engineering Sample (unqualified)	0x1	Pilot Production (unqualified)	0x2	Fully Qualified
Value	Description											
0x0	Engineering Sample (unqualified)											
0x1	Pilot Production (unqualified)											
0x2	Fully Qualified											

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 0x00FF.007F



Bit/Field	Name	Type	Reset	Description
31:16	SRAMSZ	RO	0x00FF	SRAM Size Indicates the size of the on-chip SRAM memory. Value Description 0x00FF 64 KB of SRAM
15:0	FLASHSZ	RO	0x007F	Flash Size Indicates the size of the on-chip flash memory. Value Description 0x007F 256 KB of Flash

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 0x0010.30DF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved											PWM	reserved			
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	1	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MINSYSDIV			reserved					MPU	HIB	reserved	PLL	WDT	SWO	SWD	JTAG
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	1	1	0	0	0	0	0	1	1	0	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:21	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	RO	1	PWM Module Present When set, indicates that the PWM module is present.
19:16	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.
15:12	MINSYSDIV	RO	0x3	System Clock Divider Minimum 4-bit divider value for system clock. The reset value is hardware-dependent. See the RCC register for how to change the system clock divisor using the SYSDIV bit. Value Description 0x3 Specifies a 50-MHz CPU clock with a PLL divider of 4.
11:8	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	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	HIB	RO	1	Hibernation Module Present When set, indicates that the Hibernation module is present.

Bit/Field	Name	Type	Reset	Description
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 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)

Base 0x400F.E000

Offset 0x014

Type RO, reset 0x070F.5337

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved					COMP2	COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QE1	QE10	reserved		SSI1	SSI0	reserved	UART2	UART1	UART0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	1	0	1	0	0	1	1	0	0	1	1	0	1	1	1

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	RO	1	Analog Comparator 2 Present When set, indicates that analog comparator 2 is present.
25	COMP1	RO	1	Analog Comparator 1 Present When set, indicates that analog comparator 1 is present.
24	COMP0	RO	1	Analog Comparator 0 Present When set, indicates that analog comparator 0 is present.
23:20	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.
19	TIMER3	RO	1	Timer 3 Present When set, indicates that General-Purpose Timer module 3 is present.
18	TIMER2	RO	1	Timer 2 Present When set, indicates that General-Purpose Timer module 2 is present.
17	TIMER1	RO	1	Timer 1 Present When set, indicates that General-Purpose Timer module 1 is present.
16	TIMER0	RO	1	Timer 0 Present When set, indicates that General-Purpose Timer module 0 is present.
15	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	Type	Reset	Description
14	I2C1	RO	1	I2C Module 1 Present When set, indicates that I2C module 1 is present.
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	RO	1	I2C Module 0 Present When set, indicates that I2C module 0 is present.
11:10	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.
9	QE11	RO	1	QE11 Present When set, indicates that QE1 module 1 is present.
8	QE10	RO	1	QE10 Present When set, indicates that QE1 module 0 is present.
7:6	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	SSI1	RO	1	SSI1 Present When set, indicates that SSI module 1 is present.
4	SSI0	RO	1	SSI0 Present When set, indicates that SSI module 0 is present.
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	UART2	RO	1	UART2 Present When set, indicates that UART module 2 is present.
1	UART1	RO	1	UART1 Present When set, indicates that UART module 1 is present.
0	UART0	RO	1	UART0 Present When set, indicates that UART module 0 is present.

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 0xBF00.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	32KHZ	reserved	CCP5	CCP4	CCP3	CCP2	CCP1	CCP0	reserved							
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWMFAULT	C2O	C2PLUS	C2MINUS	C1O	C1PLUS	C1MINUS	C0O	C0PLUS	C0MINUS	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31	32KHZ	RO	1	32KHz Input Clock Available When set, indicates an even CCP pin is present and can be used as a 32-KHz input clock.
30	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.
29	CCP5	RO	1	CCP5 Pin Present When set, indicates that Capture/Compare/PWM pin 5 is present.
28	CCP4	RO	1	CCP4 Pin Present When set, indicates that Capture/Compare/PWM pin 4 is present.
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.

Bit/Field	Name	Type	Reset	Description
14	C2O	RO	1	C2o Pin Present When set, indicates that the analog comparator 2 output pin is present.
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.
11	C1O	RO	1	C1o Pin Present When set, indicates that the analog comparator 1 output pin is present.
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	C0O	RO	1	C0o Pin Present 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	PWM5	RO	1	PWM5 Pin Present When set, indicates that the PWM pin 5 is present.
4	PWM4	RO	1	PWM4 Pin Present When set, indicates that the PWM pin 4 is present.
3	PWM3	RO	1	PWM3 Pin Present When set, indicates that the PWM pin 3 is present.
2	PWM2	RO	1	PWM2 Pin Present When set, indicates that the PWM pin 2 is present.
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
Type RO, reset 0x0000.C0FF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	CCP7	CCP6	reserved					GPIOH	GPIOG	GPIOF	GPIOE	GIOD	GPIOC	GPIOB	GPIOA	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:16	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	CCP7	RO	1	CCP7 Pin Present When set, indicates that Capture/Compare/PWM pin 7 is present.
14	CCP6	RO	1	CCP6 Pin Present When set, indicates that Capture/Compare/PWM pin 6 is present.
13:8	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	RO	1	GPIO Port H Present When set, indicates that GPIO Port H is present.
6	GPIOG	RO	1	GPIO Port G Present When set, indicates that GPIO Port G is present.
5	GPIOF	RO	1	GPIO Port F Present When set, indicates that GPIO Port F is present.
4	GPIOE	RO	1	GPIO Port E Present When set, indicates that GPIO Port E is present.
3	GIOD	RO	1	GPIO Port D Present When set, indicates that GPIO Port D is present.
2	GPIOC	RO	1	GPIO Port C Present When set, indicates that GPIO Port C is present.

Bit/Field	Name	Type	Reset	Description
1	GPIOB	RO	1	GPIO Port B Present When set, indicates that GPIO Port B is present.
0	GPIOA	RO	1	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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

Run Mode Clock Gating Control Register 0 (RCGC0)

Base 0x400F.E000

Offset 0x100

Type R/W, reset 0x00000040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved											PWM	reserved				
Type	RO	RO	RO	RO	RO	RO	RO	RO	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	
	reserved											HIB	reserved		WDT	reserved	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	RO	RO	RO	
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	

Bit/Field	Name	Type	Reset	Description
31:21	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	<p>PWM Clock Gating Control</p> <p>This bit controls the clock gating for the PWM module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.</p>
19:7	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	HIB	R/W	1	<p>HIB Clock Gating Control</p> <p>This bit controls the clock gating for the Hibernation module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled.</p>
5:4	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	<p>WDT Clock Gating Control</p> <p>This bit controls the clock gating for the WDT module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.</p>

Bit/Field	Name	Type	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.

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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

Sleep Mode Clock Gating Control Register 0 (SCGC0)

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved											PWM	reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	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
	reserved										HIB	reserved		WDT	reserved	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:21	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 Clock Gating Control This bit controls the clock gating for the PWM module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.
19:7	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	HIB	R/W	1	HIB Clock Gating Control This bit controls the clock gating for the Hibernation module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled.
5:4	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.

Bit/Field	Name	Type	Reset	Description
3	WDT	R/W	0	WDT Clock Gating Control This bit controls the clock gating for the WDT module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.
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 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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

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
	reserved											PWM	reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	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
	reserved										HIB	reserved		WDT	reserved	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	RO	RO	R/W	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:21	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 Clock Gating Control This bit controls the clock gating for the PWM module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.
19:7	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	HIB	R/W	1	HIB Clock Gating Control This bit controls the clock gating for the Hibernation module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled.
5:4	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.

Bit/Field	Name	Type	Reset	Description
3	WDT	R/W	0	WDT Clock Gating Control This bit controls the clock gating for the WDT module. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, a read or write to the unit generates a bus fault.
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 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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

Run Mode Clock Gating Control Register 1 (RCGC1)

Base 0x400F.E000

Offset 0x104

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				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QE1	QE10	reserved		SSI1	SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W	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/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	<p>Analog Comparator 2 Clock Gating</p> <p>This bit controls the clock gating for analog comparator 2. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
25	COMP1	R/W	0	<p>Analog Comparator 1 Clock Gating</p> <p>This bit controls the clock gating for analog comparator 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
24	COMP0	R/W	0	<p>Analog Comparator 0 Clock Gating</p> <p>This bit controls the clock gating for analog comparator 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
23:20	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	Type	Reset	Description
19	TIMER3	R/W	0	<p>Timer 3 Clock Gating Control</p> <p>This bit controls the clock gating for General-Purpose Timer module 3. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
18	TIMER2	R/W	0	<p>Timer 2 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
17	TIMER1	R/W	0	<p>Timer 1 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
16	TIMER0	R/W	0	<p>Timer 0 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
15	reserved	RO	0	<p>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.</p>
14	I2C1	R/W	0	<p>I2C1 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
13	reserved	RO	0	<p>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.</p>
12	I2C0	R/W	0	<p>I2C0 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
11:10	reserved	RO	0	<p>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.</p>
9	QE11	R/W	0	<p>QE11 Clock Gating Control</p> <p>This bit controls the clock gating for QE1 module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>

Bit/Field	Name	Type	Reset	Description
8	QEI0	R/W	0	<p>QEI0 Clock Gating Control</p> <p>This bit controls the clock gating for QEI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
7:6	reserved	RO	0	<p>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.</p>
5	SSI1	R/W	0	<p>SSI1 Clock Gating Control</p> <p>This bit controls the clock gating for SSI module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
4	SSI0	R/W	0	<p>SSI0 Clock Gating Control</p> <p>This bit controls the clock gating for SSI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
3	reserved	RO	0	<p>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.</p>
2	UART2	R/W	0	<p>UART2 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
1	UART1	R/W	0	<p>UART1 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
0	UART0	R/W	0	<p>UART0 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>

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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

Sleep Mode Clock Gating Control Register 1 (SCGC1)

Base 0x400F.E000
Offset 0x114
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				TIMER3	TIMER2	TIMER1	TIMER0	
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QE1	QE0	reserved		SSI1	SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W	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/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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
23:20	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	Type	Reset	Description
19	TIMER3	R/W	0	<p>Timer 3 Clock Gating Control</p> <p>This bit controls the clock gating for General-Purpose Timer module 3. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
18	TIMER2	R/W	0	<p>Timer 2 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
17	TIMER1	R/W	0	<p>Timer 1 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
16	TIMER0	R/W	0	<p>Timer 0 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
15	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.
14	I2C1	R/W	0	<p>I2C1 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
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	<p>I2C0 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
11:10	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.
9	QE11	R/W	0	<p>QE11 Clock Gating Control</p> <p>This bit controls the clock gating for QE1 module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>

Bit/Field	Name	Type	Reset	Description
8	QEI0	R/W	0	QEI0 Clock Gating Control This bit controls the clock gating for QEI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
7:6	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	SSI1	R/W	0	SSI1 Clock Gating Control This bit controls the clock gating for SSI module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
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	UART2	R/W	0	UART2 Clock Gating Control This bit controls the clock gating for UART module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
1	UART1	R/W	0	UART1 Clock Gating Control This bit controls the clock gating for UART module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.

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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

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
	reserved					COMP2	COMP1	COMP0	reserved				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QE1	QE0	reserved		SSI1	SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W	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/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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
23:20	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	Type	Reset	Description
19	TIMER3	R/W	0	<p>Timer 3 Clock Gating Control</p> <p>This bit controls the clock gating for General-Purpose Timer module 3. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
18	TIMER2	R/W	0	<p>Timer 2 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
17	TIMER1	R/W	0	<p>Timer 1 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
16	TIMER0	R/W	0	<p>Timer 0 Clock Gating Control</p> <p>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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
15	reserved	RO	0	<p>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.</p>
14	I2C1	R/W	0	<p>I2C1 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
13	reserved	RO	0	<p>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.</p>
12	I2C0	R/W	0	<p>I2C0 Clock Gating Control</p> <p>This bit controls the clock gating for I2C module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
11:10	reserved	RO	0	<p>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.</p>
9	QE11	R/W	0	<p>QE11 Clock Gating Control</p> <p>This bit controls the clock gating for QE1 module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>

Bit/Field	Name	Type	Reset	Description
8	QEI0	R/W	0	<p>QEI0 Clock Gating Control</p> <p>This bit controls the clock gating for QEI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
7:6	reserved	RO	0	<p>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.</p>
5	SSI1	R/W	0	<p>SSI1 Clock Gating Control</p> <p>This bit controls the clock gating for SSI module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
4	SSI0	R/W	0	<p>SSI0 Clock Gating Control</p> <p>This bit controls the clock gating for SSI module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
3	reserved	RO	0	<p>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.</p>
2	UART2	R/W	0	<p>UART2 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 2. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
1	UART1	R/W	0	<p>UART1 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 1. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>
0	UART0	R/W	0	<p>UART0 Clock Gating Control</p> <p>This bit controls the clock gating for UART module 0. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.</p>

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.

Run Mode Clock Gating Control Register 2 (RCGC2)

Base 0x400F.E000

Offset 0x108

Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	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	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	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	Type	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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	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	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	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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.

Bit/Field	Name	Type	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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled (saving power). If the unit is unlocked, reads or writes to the unit will generate a bus fault. The reset state of these bits is 0 (unlocked) 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.

Deep Sleep Mode Clock Gating Control Register 2 (DCGC2)

Base 0x400F.E000

Offset 0x128

Type R/W, reset 0x00000000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	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	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	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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.

Bit/Field	Name	Type	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 unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, reads or writes to the unit will generate a bus fault.
1	GPIOB	R/W	0	Port B Clock Gating Control This bit controls the clock gating for Port B. If set, the unit receives a clock and functions. Otherwise, the unit is unlocked and disabled. If the unit is unlocked, 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 unlocked and disabled. If the unit is unlocked, 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												PWM	reserved			
Type	RO	RO	RO	RO	RO	RO	RO	RO	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	
	reserved										HIB	reserved		WDT	reserved		
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	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/Field	Name	Type	Reset	Description
31:21	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.
19:7	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	HIB	R/W	0	HIB Reset Control Reset control for the Hibernation module.
5:4	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 0x044Writes 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				TIMER3	TIMER2	TIMER1	TIMER0
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved	I2C1	reserved	I2C0	reserved		QE1	QE10	reserved		SSI1	SSI0	reserved	UART2	UART1	UART0
Type	RO	R/W	RO	R/W	RO	RO	R/W	R/W	RO	RO	R/W	R/W	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/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 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	COMP0	R/W	0	Analog Comp 0 Reset Control Reset control for analog comparator 0.
23:20	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.
19	TIMER3	R/W	0	Timer 3 Reset Control Reset control for General-Purpose Timer module 3.
18	TIMER2	R/W	0	Timer 2 Reset Control Reset control for General-Purpose Timer module 2.
17	TIMER1	R/W	0	Timer 1 Reset Control Reset control for General-Purpose Timer module 1.
16	TIMER0	R/W	0	Timer 0 Reset Control Reset control for General-Purpose Timer module 0.
15	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.
14	I2C1	R/W	0	I2C1 Reset Control Reset control for I2C unit 1.

Bit/Field	Name	Type	Reset	Description
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 Reset Control Reset control for I2C unit 0.
11:10	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.
9	QEI1	R/W	0	QEI1 Reset Control Reset control for QEI unit 1.
8	QEI0	R/W	0	QEI0 Reset Control Reset control for QEI unit 0.
7:6	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	SSI1	R/W	0	SSI1 Reset Control Reset control for SSI unit 1.
4	SSI0	R/W	0	SSI0 Reset Control Reset control for SSI unit 0.
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	UART2	R/W	0	UART2 Reset Control Reset control for UART unit 2.
1	UART1	R/W	0	UART1 Reset Control Reset control for UART unit 1.
0	UART0	R/W	0	UART0 Reset Control Reset control for UART unit 0.

Register 29: Software Reset Control 2 (SRCR2), offset 0x048Writes 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	17	16
	reserved															
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
	reserved								GPIOH	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA
Type	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	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	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 Hibernation Module

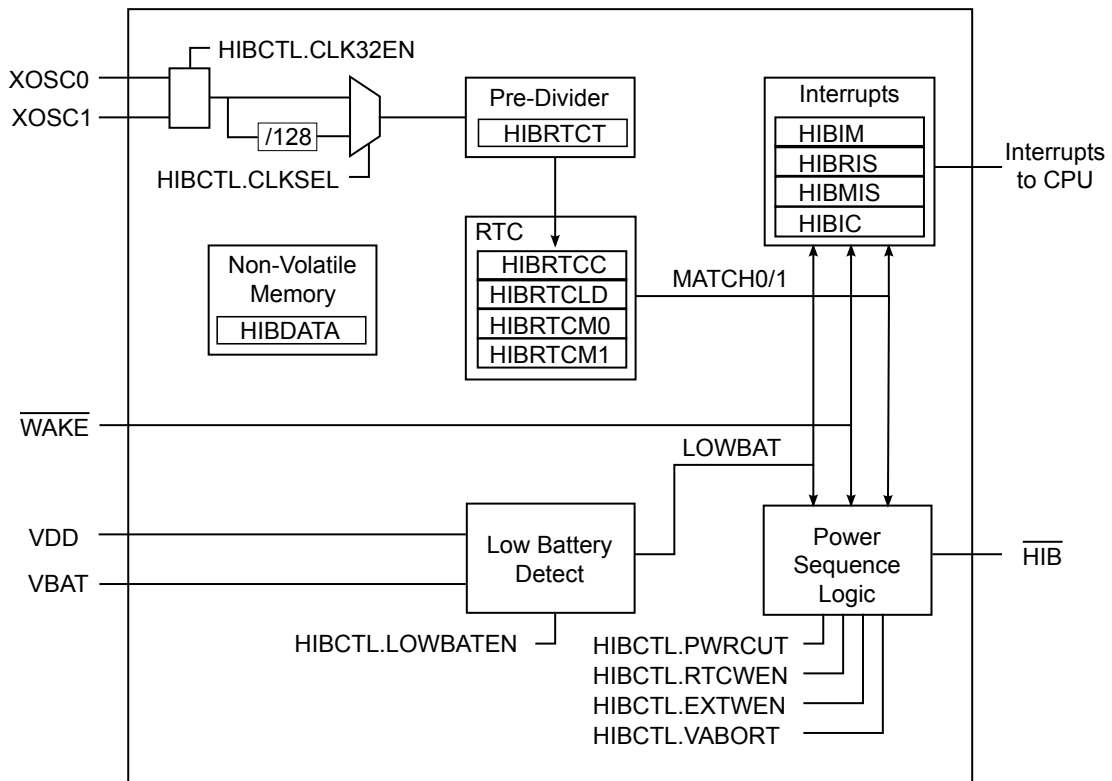
The Hibernation Module manages removal and restoration of power to provide a means for reducing power consumption. When the processor and peripherals are idle, power can be completely removed with only the Hibernation module remaining powered. Power can be restored based on an external signal, or at a certain time using the built-in Real-Time Clock (RTC). The Hibernation module can be independently supplied from a battery or an auxiliary power supply.

The Hibernation module has the following features:

- System power control using discrete external regulator
- Dedicated pin for waking from an external signal
- Low-battery detection, signaling, and interrupt generation
- 32-bit real-time counter (RTC)
- Two 32-bit RTC match registers for timed wake-up and interrupt generation
- Clock source from a 32.768-kHz external oscillator or a 4.194304-MHz crystal
- RTC predivider trim for making fine adjustments to the clock rate
- 64 32-bit words of non-volatile memory
- Programmable interrupts for RTC match, external wake, and low battery events

7.1 Block Diagram

Figure 7-1. Hibernation Module Block Diagram



7.2 Functional Description

The Hibernation module controls the power to the processor with an enable signal ($\overline{\text{HIB}}$) that signals an external voltage regulator to turn off.

The Hibernation module power source is determined dynamically. The supply voltage of the Hibernation module is the larger of the main voltage source (VDD) or the battery/auxiliary voltage source (VBAT). A voting circuit indicates the larger and an internal power switch selects the appropriate voltage source. The Hibernation module also has a separate clock source to maintain a real-time clock (RTC). Once in hibernation, the module signals an external voltage regulator to turn back on the power when an external pin ($\overline{\text{WAKE}}$) is asserted, or when the internal RTC reaches a certain value. The Hibernation module can also detect when the battery voltage is low, and optionally prevent hibernation when this occurs.

Power-up from a power cut to code execution is defined as the regulator turn-on time (specified at $t_{\text{HIB_TO_VDD}}$ maximum) plus the normal chip POR (see "Hibernation Module" on page 492).

7.2.1 Register Access Timing

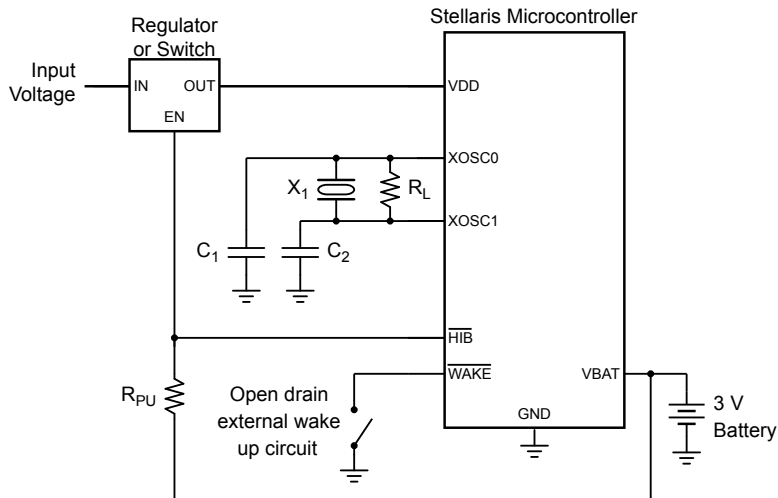
Because the Hibernation module has an independent clocking domain, certain registers must be written only with a timing gap between accesses. The delay time is $t_{\text{HIB_REG_WRITE}}$, therefore software must guarantee that a delay of $t_{\text{HIB_REG_WRITE}}$ is inserted between back-to-back writes to certain Hibernation registers, or between a write followed by a read to those same registers. There is no restriction on timing for back-to-back reads from the Hibernation module.

7.2.2 Clock Source

The Hibernation module must be clocked by an external source, even if the RTC feature is not used. An external oscillator or crystal can be used for this purpose. To use a crystal, a 4.194304-MHz crystal is connected to the XOSC0 and XOSC1 pins. This clock signal is divided by 128 internally to produce the 32.768-kHz clock reference. For an alternate clock source, a 32.768-kHz oscillator can be connected to the XOSC0 pin. See Figure 7-2 on page 130 and Figure 7-3 on page 131. Note that these diagrams only show the connection to the Hibernation pins and not to the full system. See “Hibernation Module” on page 492 for specific values.

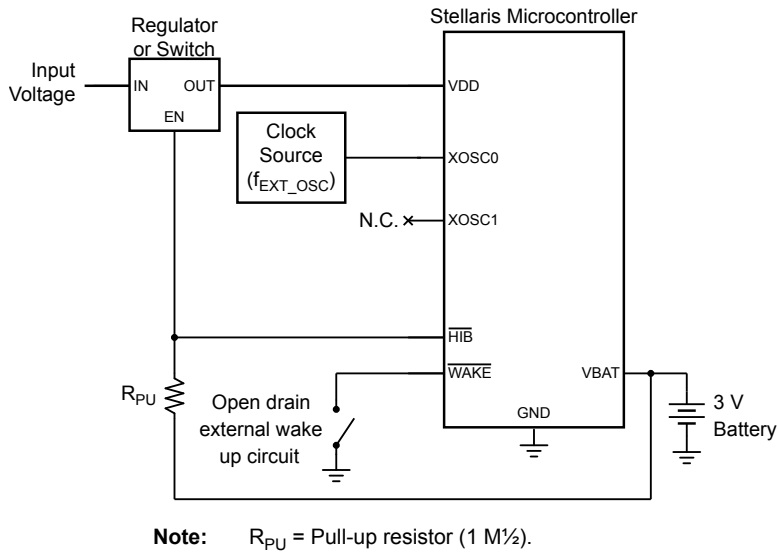
The clock source is enabled by setting the CLK32EN bit of the HIBCTL register. The type of clock source is selected by setting the CLKSEL bit to 0 for a 4.194304-MHz clock source, and to 1 for a 32.768-kHz clock source. If the bit is set to 0, the 4.194304-MHz input clock is divided by 128, resulting in a 32.768-kHz clock source. If a crystal is used for the clock source, the software must leave a delay of t_{XOSC_SETTLE} after setting the CLK32EN bit and before any other accesses to the Hibernation module registers. The delay allows the crystal to power up and stabilize. If an oscillator is used for the clock source, no delay is needed.

Figure 7-2. Clock Source Using Crystal



- Note:**
- X_1 = Crystal frequency is f_{XOSC_XTAL} .
 - $C_{1,2}$ = Capacitor value derived from crystal vendor load capacitance specifications.
 - R_L = Load resistor is R_{XOSC_LOAD} .
 - R_{PU} = Pull-up resistor (1 M Ω).
- See “Hibernation Module” on page 492 for specific parameter values.

Figure 7-3. Clock Source Using Dedicated Oscillator



7.2.3 Battery Management

The Hibernation module can be independently powered by a battery or an auxiliary power source. The module can monitor the voltage level of the battery and detect when the voltage drops below V_{LOWBAT} . When this happens, an interrupt can be generated. The module can also be configured so that it will not go into Hibernate mode if the battery voltage drops below this threshold. Battery voltage is not measured while in Hibernate mode.

Important: System level factors may affect the accuracy of the low battery detect circuit. The designer should consider battery type, discharge characteristics, and a test load during battery voltage measurements.

Note that the Hibernation module draws power from whichever source (V_{BAT} or V_{DD}) has the higher voltage. Therefore, it is important to design the circuit to ensure that V_{DD} is higher than V_{BAT} under nominal conditions or else the Hibernation module draws power from the battery even when V_{DD} is available.

The Hibernation module can be configured to detect a low battery condition by setting the $LOWBATEN$ bit of the **HIBCTL** register. In this configuration, the $LOWBAT$ bit of the **HIBRIS** register will be set when the battery level is low. If the $VABORT$ bit is also set, then the module is prevented from entering Hibernation mode when a low battery is detected. The module can also be configured to generate an interrupt for the low-battery condition (see “Interrupts and Status” on page 133).

7.2.4 Real-Time Clock

The Hibernation module includes a 32-bit counter that increments once per second with a proper clock source and configuration (see “Clock Source” on page 130). The 32.768-kHz clock signal is fed into a predivider register which counts down the 32.768-kHz clock ticks to achieve a once per second clock rate for the RTC. The rate can be adjusted to compensate for inaccuracies in the clock source by using the predivider trim register, **HIBRTCT**. This register has a nominal value of 0x7FFF, and is used for one second out of every 64 seconds to divide the input clock. This allows the software to make fine corrections to the clock rate by adjusting the predivider trim register up or down from 0x7FFF. The predivider trim should be adjusted up from 0x7FFF in order to slow down the RTC rate, and down from 0x7FFF in order to speed up the RTC rate.

The Hibernation module includes two 32-bit match registers that are compared to the value of the RTC counter. The match registers can be used to wake the processor from hibernation mode, or to generate an interrupt to the processor if it is not in hibernation.

The RTC must be enabled with the `RTCEN` bit of the `HIBCTL` register. The value of the RTC can be set at any time by writing to the `HIBRTCLD` register. The predivider trim can be adjusted by reading and writing the `HIBRTCT` register. The predivider uses this register once every 64 seconds to adjust the clock rate. The two match registers can be set by writing to the `HIBRTCM0` and `HIBRTCM1` registers. The RTC can be configured to generate interrupts by using the interrupt registers (see “Interrupts and Status” on page 133).

7.2.5 Non-Volatile Memory

The Hibernation module contains 64 32-bit words of memory which are retained during hibernation. This memory is powered from the battery or auxiliary power supply during hibernation. The processor software can save state information in this memory prior to hibernation, and can then recover the state upon waking. The non-volatile memory can be accessed through the `HIBDATA` registers.

7.2.6 Power Control

Important: The Hibernation Module requires special system implementation considerations when using `HIB` to control power, as it is intended to power-down all other sections of its host device. All system signals and power supplies that connect to the chip must be driven to 0 V_{DC} or powered down with the same regulator controlled by `HIB`. See “Hibernation Module” on page 492 for more details.

The Hibernation module controls power to the microcontroller through the use of the `HIB` pin. This pin is intended to be connected to the enable signal of the external regulator(s) providing 3.3 V and/or 2.5 V to the microcontroller. When the `HIB` signal is asserted by the Hibernation module, the external regulator is turned off and no longer powers the system. The Hibernation module remains powered from the `VBAT` supply (which could be a battery or an auxiliary power source) until a Wake event. Power to the device is restored by deasserting the `HIB` signal, which causes the external regulator to turn power back on to the chip.

7.2.7 Initiating Hibernate

Hibernation mode is initiated by the microcontroller setting the `HIBREQ` bit of the `HIBCTL` register. Prior to doing this, a wake-up condition must be configured, either from the external `WAKE` pin, or by using an RTC match.

The Hibernation module is configured to wake from the external `WAKE` pin by setting the `PINWEN` bit of the `HIBCTL` register. It is configured to wake from RTC match by setting the `RTCWEN` bit. Either one or both of these bits can be set prior to going into hibernation. The `WAKE` pin includes a weak internal pull-up. Note that both the `HIB` and `WAKE` pins use the Hibernation module's internal power supply as the logic 1 reference.

When the Hibernation module wakes, the microcontroller will see a normal power-on reset. Software can detect that the power-on was due to a wake from hibernation by examining the raw interrupt status register (see “Interrupts and Status” on page 133) and by looking for state data in the non-volatile memory (see “Non-Volatile Memory” on page 132).

When the `HIB` signal deasserts, enabling the external regulator, the external regulator must reach the operating voltage within $t_{HIB_TO_VDD}$.

7.2.8 Interrupts and Status

The Hibernation module can generate interrupts when the following conditions occur:

- Assertion of $\overline{\text{WAKE}}$ pin
- RTC match
- Low battery detected

All of the interrupts are ORed together before being sent to the interrupt controller, so the Hibernation module can only generate a single interrupt request to the controller at any given time. The software interrupt handler can service multiple interrupt events by reading the **HIBMIS** register. Software can also read the status of the Hibernation module at any time by reading the **HIBRIS** register which shows all of the pending events. This register can be used at power-on to see if a wake condition is pending, which indicates to the software that a hibernation wake occurred.

The events that can trigger an interrupt are configured by setting the appropriate bits in the **HIBIM** register. Pending interrupts can be cleared by writing the corresponding bit in the **HIBIC** register.

7.3 Initialization and Configuration

The Hibernation module can be set in several different configurations. The following sections show the recommended programming sequence for various scenarios. The examples below assume that a 32.768-kHz oscillator is used, and thus always show bit 2 (**CLKSEL**) of the **HIBCTL** register set to 1. If a 4.194304-MHz crystal is used instead, then the **CLKSEL** bit remains cleared. Because the Hibernation module runs at 32.768 kHz and is asynchronous to the rest of the system, software must allow a delay of $t_{\text{HIB_REG_WRITE}}$ after writes to certain registers (see “Register Access Timing” on page 129). The registers that require a delay are listed in a note in “Register Map” on page 134 as well as in each register description.

7.3.1 Initialization

The Hibernation module clock source must be enabled first, even if the RTC feature is not used. If a 4.194304-MHz crystal is used, perform the following steps:

1. Write 0x40 to the **HIBCTL** register at offset 0x10 to enable the crystal and select the divide-by-128 input path.
2. Wait for a time of $t_{\text{XOSC_SETTLE}}$ for the crystal to power up and stabilize before performing any other operations with the Hibernation module.

If a 32.678-kHz oscillator is used, then perform the following steps:

1. Write 0x44 to the **HIBCTL** register at offset 0x10 to enable the oscillator input.
2. No delay is necessary.

The above is only necessary when the entire system is initialized for the first time. If the processor is powered due to a wake from hibernation, then the Hibernation module has already been powered up and the above steps are not necessary. The software can detect that the Hibernation module and clock are already powered by examining the **CLK32EN** bit of the **HIBCTL** register.

7.3.2 RTC Match Functionality (No Hibernation)

Use the following steps to implement the RTC match functionality of the Hibernation module:

1. Write the required RTC match value to one of the **HIBRTCMn** registers at offset 0x004 or 0x008.
2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.
3. Set the required RTC match interrupt mask in the **RTCALT0** and **RTCALT1** bits (bits 1:0) in the **HIBIM** register at offset 0x014.
4. Write 0x0000.0041 to the **HIBCTL** register at offset 0x010 to enable the RTC to begin counting.

7.3.3 RTC Match/Wake-Up from Hibernation

Use the following steps to implement the RTC match and wake-up functionality of the Hibernation module:

1. Write the required RTC match value to the **HIBRTCMn** registers at offset 0x004 or 0x008.
2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.
3. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.
4. Set the RTC Match Wake-Up and start the hibernation sequence by writing 0x0000.004F to the **HIBCTL** register at offset 0x010.

7.3.4 External Wake-Up from Hibernation

Use the following steps to implement the Hibernation module with the external $\overline{\text{WAKE}}$ pin as the wake-up source for the microcontroller:

1. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.
2. Enable the external wake and start the hibernation sequence by writing 0x0000.0056 to the **HIBCTL** register at offset 0x010.

7.3.5 RTC/External Wake-Up from Hibernation

1. Write the required RTC match value to the **HIBRTCMn** registers at offset 0x004 or 0x008.
2. Write the required RTC load value to the **HIBRTCLD** register at offset 0x00C.
3. Write any data to be retained during power cut to the **HIBDATA** register at offsets 0x030-0x12C.
4. Set the RTC Match/External Wake-Up and start the hibernation sequence by writing 0x0000.005F to the **HIBCTL** register at offset 0x010.

7.4 Register Map

Table 7-1 on page 134 lists the Hibernation registers. All addresses given are relative to the Hibernation Module base address at 0x400F.C000.

Table 7-1. Hibernation Module Register Map

Offset	Name	Type	Reset	Description	See page
0x000	HIBRTCC	RO	0x0000.0000	Hibernation RTC Counter	136
0x004	HIBRTCM0	R/W	0xFFFF.FFFF	Hibernation RTC Match 0	137

Table 7-1. Hibernation Module Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x008	HIBRTCM1	R/W	0xFFFF.FFFF	Hibernation RTC Match 1	138
0x00C	HIBRTCLD	R/W	0xFFFF.FFFF	Hibernation RTC Load	139
0x010	HIBCTL	R/W	0x8000.0000	Hibernation Control	140
0x014	HIBIM	R/W	0x0000.0000	Hibernation Interrupt Mask	142
0x018	HIBRIS	RO	0x0000.0000	Hibernation Raw Interrupt Status	143
0x01C	HIBMIS	RO	0x0000.0000	Hibernation Masked Interrupt Status	144
0x020	HIBIC	R/W1C	0x0000.0000	Hibernation Interrupt Clear	145
0x024	HIBRTCT	R/W	0x0000.7FFF	Hibernation RTC Trim	146
0x030- 0x12C	HIBDATA	R/W	-	Hibernation Data	147

7.5 Register Descriptions

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

Register 1: Hibernation RTC Counter (HIBRTCC), offset 0x000

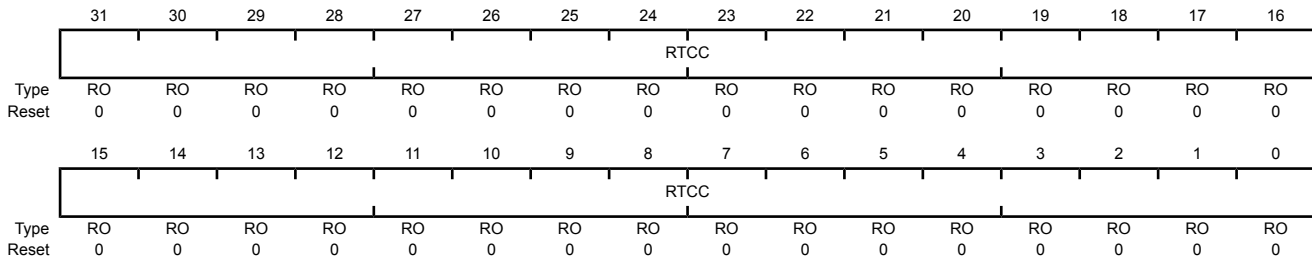
This register is the current 32-bit value of the RTC counter.

Hibernation RTC Counter (HIBRTCC)

Base 0x400F.C000

Offset 0x000

Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:0	RTCC	RO	0x0000.0000	RTC Counter

A read returns the 32-bit counter value. This register is read-only. To change the value, use the **HIBRTCLD** register.

Register 2: Hibernation RTC Match 0 (HIBRTCM0), offset 0x004

This register is the 32-bit match 0 register for the RTC counter.

Hibernation RTC Match 0 (HIBRTCM0)

Base 0x400F.C000

Offset 0x004

Type R/W, reset 0xFFFF.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	RTCM0															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCM0															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit/Field	Name	Type	Reset	Description
31:0	RTCM0	R/W	0xFFFF.FFFF	RTC Match 0

A write loads the value into the RTC match register.

A read returns the current match value.

Register 3: Hibernation RTC Match 1 (HIBRTCM1), offset 0x008

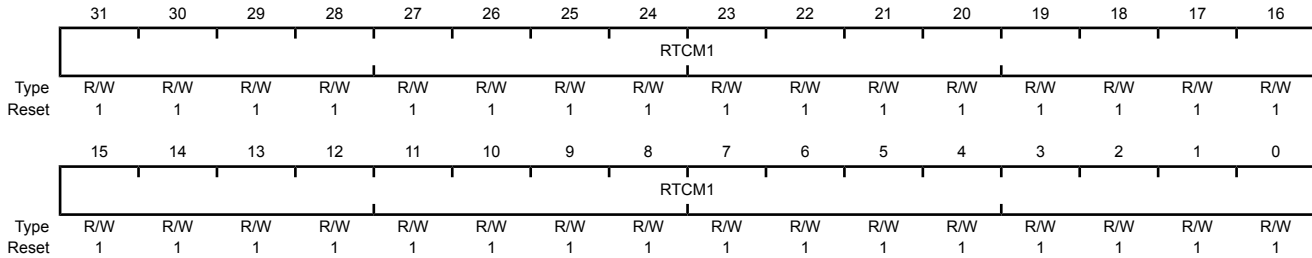
This register is the 32-bit match 1 register for the RTC counter.

Hibernation RTC Match 1 (HIBRTCM1)

Base 0x400F.C000

Offset 0x008

Type R/W, reset 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:0	RTCM1	R/W	0xFFFF.FFFF	RTC Match 1

A write loads the value into the RTC match register.

A read returns the current match value.

Register 4: Hibernation RTC Load (HIBRTCLD), offset 0x00C

This register is the 32-bit value loaded into the RTC counter.

Hibernation RTC Load (HIBRTCLD)

Base 0x400F.C000

Offset 0x00C

Type R/W, reset 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	RTCLD	R/W	0xFFFF.FFFF	RTC Load

A write loads the current value into the RTC counter (RTCC).

A read returns the 32-bit load value.

Register 5: Hibernation Control (HIBCTL), offset 0x010

This register is the control register for the Hibernation module.

Hibernation Control (HIBCTL)

Base 0x400F.C000
 Offset 0x010
 Type R/W, reset 0x8000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								VABORT	CLK32EN	LOWBATEN	PINWEN	RTCWEN	CLKSEL	HIBREQ	RTCEN
Type	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	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	VABORT	R/W	0	Power Cut Abort Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Power cut occurs during a low-battery alert.</td> </tr> <tr> <td>1</td> <td>Power cut is aborted.</td> </tr> </table>	Value	Description	0	Power cut occurs during a low-battery alert.	1	Power cut is aborted.
Value	Description									
0	Power cut occurs during a low-battery alert.									
1	Power cut is aborted.									
6	CLK32EN	R/W	0	Clocking Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </table> <p>This bit must be enabled to use the Hibernation module. If a crystal is used, then software should wait 20 ms after setting this bit to allow the crystal to power up and stabilize.</p>	Value	Description	0	Disabled	1	Enabled
Value	Description									
0	Disabled									
1	Enabled									
5	LOWBATEN	R/W	0	Low Battery Monitoring Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </table> <p>When set, low battery voltage detection is enabled ($V_{BAT} < V_{LOWBAT}$).</p>	Value	Description	0	Disabled	1	Enabled
Value	Description									
0	Disabled									
1	Enabled									
4	PINWEN	R/W	0	External \overline{WAKE} Pin Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </table> <p>When set, an external event on the \overline{WAKE} pin will re-power the device.</p>	Value	Description	0	Disabled	1	Enabled
Value	Description									
0	Disabled									
1	Enabled									

Bit/Field	Name	Type	Reset	Description						
3	RTCWEN	R/W	0	RTC Wake-up Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </table> <p>When set, an RTC match event (RTCM0 or RTCM1) will re-power the device based on the RTC counter value matching the corresponding match register 0 or 1.</p>	Value	Description	0	Disabled	1	Enabled
Value	Description									
0	Disabled									
1	Enabled									
2	CLKSEL	R/W	0	Hibernation Module Clock Select <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Use Divide by 128 output. Use this value for a 4.194304-MHz crystal.</td> </tr> <tr> <td>1</td> <td>Use raw output. Use this value for a 32.768-kHz oscillator.</td> </tr> </table>	Value	Description	0	Use Divide by 128 output. Use this value for a 4.194304-MHz crystal.	1	Use raw output. Use this value for a 32.768-kHz oscillator.
Value	Description									
0	Use Divide by 128 output. Use this value for a 4.194304-MHz crystal.									
1	Use raw output. Use this value for a 32.768-kHz oscillator.									
1	HIBREQ	R/W	0	Hibernation Request <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Hibernation initiated</td> </tr> </table> <p>After a wake-up event, this bit is cleared by hardware.</p>	Value	Description	0	Disabled	1	Hibernation initiated
Value	Description									
0	Disabled									
1	Hibernation initiated									
0	RTCEN	R/W	0	RTC Timer Enable <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </table>	Value	Description	0	Disabled	1	Enabled
Value	Description									
0	Disabled									
1	Enabled									

Register 6: Hibernation Interrupt Mask (HIBIM), offset 0x014

This register is the interrupt mask register for the Hibernation module interrupt sources.

Hibernation Interrupt Mask (HIBIM)

Base 0x400F.C000
 Offset 0x014
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0	
Type	RO	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	0

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x000.0000	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 preserved across a read-modify-write operation.
3	EXTW	R/W	0	External Wake-Up Interrupt Mask
				Value Description
				0 Masked
				1 Unmasked
2	LOWBAT	R/W	0	Low Battery Voltage Interrupt Mask
				Value Description
				0 Masked
				1 Unmasked
1	RTCALT1	R/W	0	RTC Alert1 Interrupt Mask
				Value Description
				0 Masked
				1 Unmasked
0	RTCALT0	R/W	0	RTC Alert0 Interrupt Mask
				Value Description
				0 Masked
				1 Unmasked

Register 7: Hibernation Raw Interrupt Status (HIBRIS), offset 0x018

This register is the raw interrupt status for the Hibernation module interrupt sources.

Hibernation Raw Interrupt Status (HIBRIS)

Base 0x400F.C000

Offset 0x018

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
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

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x000.0000	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 preserved across a read-modify-write operation.
3	EXTW	RO	0	External Wake-Up Raw Interrupt Status
2	LOWBAT	RO	0	Low Battery Voltage Raw Interrupt Status
1	RTCALT1	RO	0	RTC Alert1 Raw Interrupt Status
0	RTCALT0	RO	0	RTC Alert0 Raw Interrupt Status

Register 8: Hibernation Masked Interrupt Status (HIBMIS), offset 0x01C

This register is the masked interrupt status for the Hibernation module interrupt sources.

Hibernation Masked Interrupt Status (HIBMIS)

Base 0x400F.C000

Offset 0x01C

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
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

Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0x000.0000	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 preserved across a read-modify-write operation.
3	EXTW	RO	0	External Wake-Up Masked Interrupt Status
2	LOWBAT	RO	0	Low Battery Voltage Masked Interrupt Status
1	RTCALT1	RO	0	RTC Alert1 Masked Interrupt Status
0	RTCALT0	RO	0	RTC Alert0 Masked Interrupt Status

Register 9: Hibernation Interrupt Clear (HIBIC), offset 0x020

This register is the interrupt write-one-to-clear register for the Hibernation module interrupt sources.

Hibernation Interrupt Clear (HIBIC)

Base 0x400F.C000

Offset 0x020

Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved												EXTW	LOWBAT	RTCALT1	RTCALT0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
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	0x000.0000	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 preserved across a read-modify-write operation.
3	EXTW	R/W1C	0	External Wake-Up Masked Interrupt Clear Reads return an indeterminate value.
2	LOWBAT	R/W1C	0	Low Battery Voltage Masked Interrupt Clear Reads return an indeterminate value.
1	RTCALT1	R/W1C	0	RTC Alert1 Masked Interrupt Clear Reads return an indeterminate value.
0	RTCALT0	R/W1C	0	RTC Alert0 Masked Interrupt Clear Reads return an indeterminate value.

Register 10: Hibernation RTC Trim (HIBRTCT), offset 0x024

This register contains the value that is used to trim the RTC clock predivider. It represents the computed underflow value that is used during the trim cycle. It is represented as $0x7FFF \pm N$ clock cycles.

Hibernation RTC Trim (HIBRTCT)

Base 0x400F.C000
 Offset 0x024
 Type R/W, reset 0x0000.7FFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	TRIM															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	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.
15:0	TRIM	R/W	0x7FFF	RTC Trim Value This value is loaded into the RTC predivider every 64 seconds. It is used to adjust the RTC rate to account for drift and inaccuracy in the clock source. The compensation is made by software by adjusting the default value of 0x7FFF up or down.

Register 11: Hibernation Data (HIBDATA), offset 0x030-0x12C

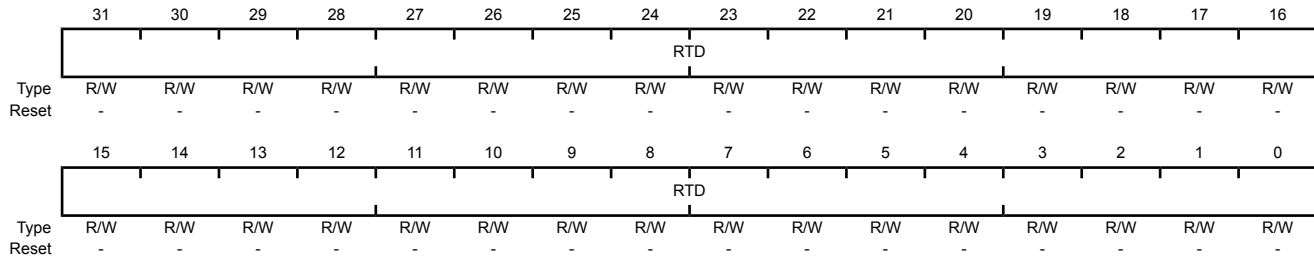
This address space is implemented as a 64x32-bit memory (256 bytes). It can be loaded by the system processor in order to store any non-volatile state data and will not lose power during a power cut operation.

Hibernation Data (HIBDATA)

Base 0x400F.C000

Offset 0x030-0x12C

Type R/W, reset -



Bit/Field	Name	Type	Reset	Description
31:0	RTD	R/W	-	Hibernation Module NV Registers[63:0]

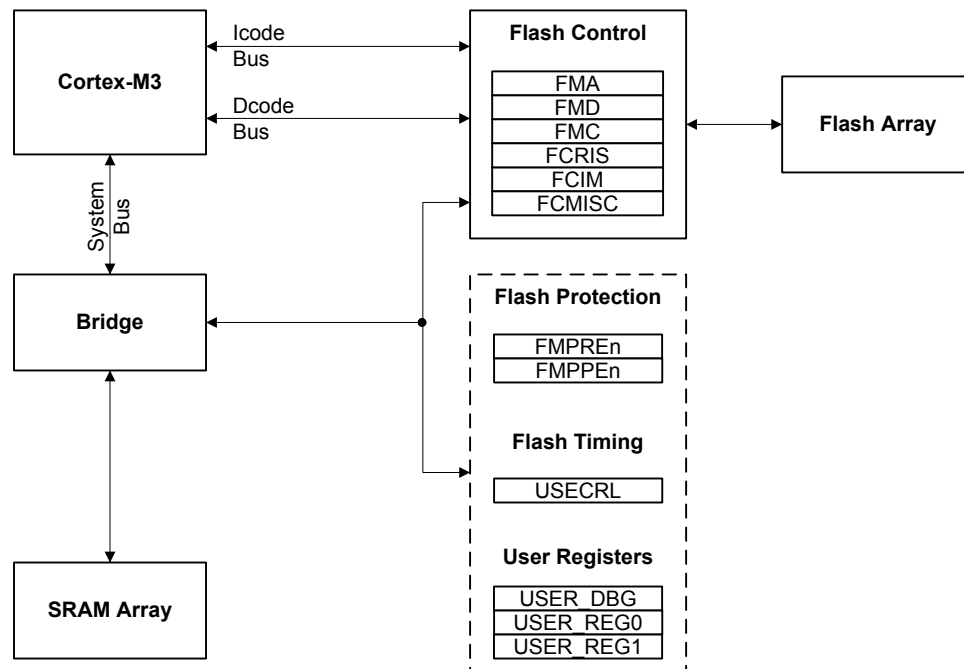
8 Internal Memory

The LM3S1960 microcontroller comes with 64 KB of bit-banded SRAM and 256 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.

8.1 Block Diagram

Figure 8-1 on page 148 illustrates the Flash functions. The dashed boxes in the figure indicate registers residing in the System Control module rather than the Flash Control module.

Figure 8-1. Flash Block Diagram



8.2 Functional Description

This section describes the functionality of the SRAM and Flash memories.

8.2.1 SRAM Memory

The internal SRAM of the Stellaris[®] 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:

$$\text{bit-band alias} = \text{bit-band base} + (\text{byte offset} * 32) + (\text{bit number} * 4)$$

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*.

8.2.2 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 497 for a preprogrammed flash-resident utility used to download code to the flash memory of a device without the use of a debug interface.

8.2.2.1 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.

8.2.2.2 Flash Memory Protection

The user is provided two forms of flash protection per 2-KB flash blocks in four pairs 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 and contents of the memory block are prohibited from being accessed as data.

The policies may be combined as shown in Table 8-1 on page 149.

Table 8-1. Flash Protection Policy Combinations

FMPPEn	FMPREn	Protection
0	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.

Table 8-1. Flash Protection Policy Combinations (continued)

FMPPEn	FMPREn	Protection
0	1	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 151.

8.3 Flash Memory Initialization and Configuration

8.3.1 Flash Programming

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

8.3.1.1 To program a 32-bit word

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

8.3.1.2 To perform an erase of a 1-KB page

1. Write the page address to the **FMA** register.
2. 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.

8.3.1.3 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.

8.3.2 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 memory array and are not affected by an ERASE or MASS ERASE operation. The bits in these registers can be changed from 1 to 0 with a write operation. The register contents are unaffected by any reset condition except power-on reset, which returns the register contents to 0xFFFF.FFFF. By committing the register values using the **COMT** bit in the **FMC** register, the register contents become nonvolatile and are therefore retained following power cycling. Once the register contents are committed, the **FMPREx** and **FMPPEx** registers can be restored to their factory default values by performing the sequence described in the section called "Recovering a "Locked" Device" on page 57. However, the **USER_REGx** and **USER_DBG** registers can never be restored to the factory default values.

With the exception of the **USER_DBG** register, the settings in these registers can be tested before committing them to Flash memory. For the **USER_DBG** register, the data to be written is loaded into the **FMD** register before it is committed. The **FMD** register is read only and does not allow the **USER_DBG** operation to be tried before committing it to nonvolatile memory.

Important: These registers can only have bits changed from 1 to 0 by user programming. Once committed, the **FMPREx** and **FMPPEx** registers can only be restored to their factory default values only by performing the sequence described in the section called "Recovering a "Locked" Device" on page 57. The mass erase of the main Flash memory array caused by the sequence is performed prior to restoring these registers.

In addition, the **USER_REG0**, **USER_REG1**, **USER_REG2**, **USER_REG3**, and **USER_DBG** registers each use bit 31 (**NW**) to indicate that they have not been committed and bits in the register may be changed from 1 to 0. These five registers can only be committed once whereas the Flash memory protection registers may be committed multiple times. Table 8-2 on page 151 provides the **FMA** address required for commitment of each of the registers and the source of the data to be written when 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 8-2. User-Programmable Flash Memory Resident Registers

Register to be Committed	FMA Value	Data Source
FMPRE0	0x0000.0000	FMPRE0
FMPRE1	0x0000.0002	FMPRE1
FMPRE2	0x0000.0004	FMPRE2
FMPRE3	0x0000.0006	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_REG2	0x8000.0002	USER_REG2
USER_REG3	0x8000.0003	USER_REG3
USER_DBG	0x7510.0000	FMD

8.4 Register Map

Table 8-3 on page 152 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** register offsets are relative to the Flash memory control base address of 0x400F.D000. The Flash memory protection register offsets are relative to the System Control base address of 0x400F.E000.

Table 8-3. Flash Register Map

Offset	Name	Type	Reset	Description	See page
Flash Memory Control Registers (Flash Control Offset)					
0x000	FMA	R/W	0x0000.0000	Flash Memory Address	153
0x004	FMD	R/W	0x0000.0000	Flash Memory Data	154
0x008	FMC	R/W	0x0000.0000	Flash Memory Control	155
0x00C	FCRIS	RO	0x0000.0000	Flash Controller Raw Interrupt Status	157
0x010	FCIM	R/W	0x0000.0000	Flash Controller Interrupt Mask	158
0x014	FCMISC	R/W1C	0x0000.0000	Flash Controller Masked Interrupt Status and Clear	159
Flash Memory Protection Registers (System Control Offset)					
0x130	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	161
0x200	FMPRE0	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 0	161
0x134	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	162
0x400	FMPPE0	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 0	162
0x140	USECRL	R/W	0x31	USec Reload	160
0x1D0	USER_DBG	R/W	0xFFFF.FFFE	User Debug	163
0x1E0	USER_REG0	R/W	0xFFFF.FFFF	User Register 0	164
0x1E4	USER_REG1	R/W	0xFFFF.FFFF	User Register 1	165
0x204	FMPRE1	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 1	166
0x208	FMPRE2	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 2	167
0x20C	FMPRE3	R/W	0xFFFF.FFFF	Flash Memory Protection Read Enable 3	168
0x404	FMPPE1	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 1	169
0x408	FMPPE2	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 2	170
0x40C	FMPPE3	R/W	0xFFFF.FFFF	Flash Memory Protection Program Enable 3	171

8.5 Flash Register Descriptions (Flash Control Offset)

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

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved														OFFSET	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	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
	OFFSET															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	Reset	Description
31:18	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.
17:0	OFFSET	R/W	0x0	Address Offset Address offset in flash where operation is performed, except for nonvolatile registers (see "Nonvolatile Register Programming" on page 151 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DATA															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	Reset	Description
31:0	DATA	R/W	0x0	Data Value 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 153). If the access is a write access, the data contained in the **Flash Memory Data (FMD)** register (see page 154) 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	WRKEY																
Type	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	
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												COMT	MERASE	ERASE	WRITE	
Type	RO	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	0

Bit/Field	Name	Type	Reset	Description
31:16	WRKEY	WO	0x0	Flash Write Key This field contains a write key, which is used to minimize the incidence of accidental flash writes. The value 0xA442 must be written into this field for a write to occur. Writes to the FMC register without this <code>WRKEY</code> value are ignored. A read of this field returns the value 0.
15:4	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	COMT	R/W	0	Commit Register Value Commit (write) of register value to nonvolatile storage. A write of 0 has no effect on the state of this bit. If read, the state of the previous commit access is provided. If the previous commit access is complete, a 0 is returned; otherwise, if the commit access is not complete, a 1 is returned. This can take up to 50 μ s.
2	MERASE	R/W	0	Mass Erase Flash Memory If this bit is set, the flash main memory of the device is all erased. A write of 0 has no effect on the state of this bit. If read, the state of the previous mass erase access is provided. If the previous mass erase access is complete, a 0 is returned; otherwise, if the previous mass erase access is not complete, a 1 is returned. This can take up to 250 ms.

Bit/Field	Name	Type	Reset	Description
1	ERASE	R/W	0	<p>Erase a Page of Flash Memory</p> <p>If this bit is set, the page of flash main memory as specified by the contents of FMA is erased. A write of 0 has no effect on the state of this bit.</p> <p>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.</p> <p>This can take up to 25 ms.</p>
0	WRITE	R/W	0	<p>Write a Word into Flash Memory</p> <p>If this bit is set, the data stored in FMD is written into the location as specified by the contents of FMA. A write of 0 has no effect on the state of this bit.</p> <p>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.</p> <p>This can take up to 50 μs.</p>

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															PRIS	ARIS
Type	RO	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	0

Bit/Field	Name	Type	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	PRIS	RO	0	<p>Programming Raw Interrupt Status</p> <p>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 Flash Memory Control (FMC) register bits (see page 155).</p>
0	ARIS	RO	0	<p>Access Raw Interrupt Status</p> <p>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.</p>

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved														PMASK	AMASK	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W
Reset	0	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	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	PMASK	R/W	0	<p>Programming Interrupt Mask</p> <p>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.</p>
0	AMASK	R/W	0	<p>Access Interrupt Mask</p> <p>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.</p>

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	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															PMISC	AMISC
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	
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	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	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 FCRIS register (see page 157) 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.

8.6 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- μ s 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 0x31

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								USEC							
Type	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	1	1	0	0	0	1

Bit/Field	Name	Type	Reset	Description
31:8	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.
7:0	USEC	R/W	0x31	Microsecond Reload Value MHz -1 of the controller clock when the flash is being erased or programmed. If the maximum system frequency is being used, USEC should be set to 0x31 (50 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 compatibility.

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 (**FMPPE_n** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPREN** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPREN** and **FMPPE_n** 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. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the "Recover Locked Device" sequence detailed in the JTAG chapter. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Read Enable 0 (FMPRE0)

Base 0x400F.E000
Offset 0x130 and 0x200
Type R/W, reset 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable. Enables 2-KB Flash memory blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".

Value	Description
0xFFFFFFFF	Bits [31:0] each enable protection on a 2-KB block of Flash memory up to the total of 64 KB.

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

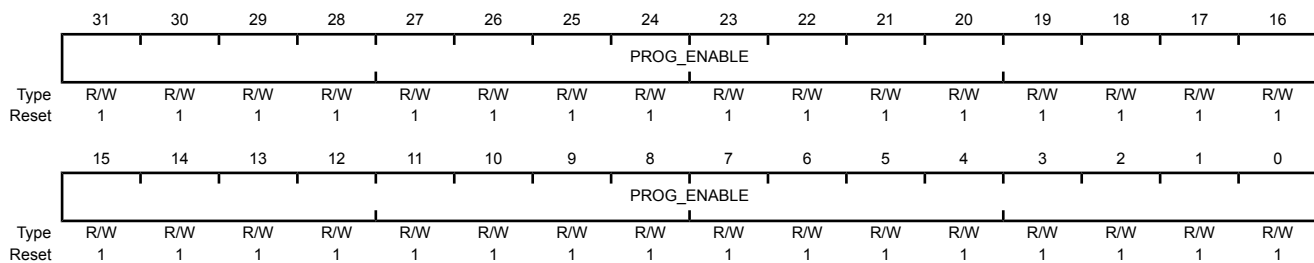
Note: This register is aliased for backwards compatability.

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). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPEN** registers (if any) provide protection for other 64K blocks. 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. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the "Recover Locked Device" sequence detailed in the JTAG chapter. For additional information, see the "Flash Memory Protection" section.

Flash Memory Protection Program Enable 0 (FMPPE0)

Base 0x400F.E000
 Offset 0x134 and 0x400
 Type R/W, reset 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable

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
0xFFFFFFFF	Bits [31:0] each enable protection on a 2-KB block of Flash memory up to the total of 64 KB.

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 NW bit (bit 31) indicates that the register has not yet been committed and is controlled through hardware to ensure that the register is only committed once. Prior to being committed, bits can only be changed from 1 to 0. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, this register cannot be restored to the factory default value.

User Debug (USER_DBG)

Base 0x400F.E000

Offset 0x1D0

Type R/W, reset 0xFFFF.FFFE

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	NW	DATA															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	DATA														DBG1	DBG0	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	User Debug Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:2	DATA	R/W	0x1FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed once.
1	DBG1	R/W	1	Debug Control 1 The DBG1 bit must be 1 and DBG0 must be 0 for debug to be available.
0	DBG0	R/W	0	Debug Control 0 The DBG1 bit must be 1 and DBG0 must be 0 for debug to be available.

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 committed once. Bit 31 indicates that the register is available to be committed and is controlled through hardware to ensure that the register is only committed once. Prior to being committed, bits can only be changed from 1 to 0. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. 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. Once committed, this register cannot be restored to the factory default value.

User Register 0 (USER_REG0)

Base 0x400F.E000
 Offset 0x1E0
 Type R/W, reset 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed 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 committed once. Bit 31 indicates that the register is available to be committed and is controlled through hardware to ensure that the register is only committed once. Prior to being committed, bits can only be changed from 1 to 0. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. 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. Once committed, this register cannot be restored to the factory default value.

User Register 1 (USER_REG1)

Base 0x400F.E000

Offset 0x1E4

Type R/W, reset 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31	NW	R/W	1	Not Written When set, this bit indicates that this 32-bit register has not been committed. When clear, this bit specifies that this register has been committed and may not be committed again.
30:0	DATA	R/W	0x7FFFFFFF	User Data Contains the user data value. This field is initialized to all 1s and can only be committed 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 (**FMPPE_n** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPRE_n** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPRE_n** and **FMPPE_n** 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. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the "Recover Locked Device" sequence detailed in the JTAG chapter. If the Flash memory size on the device is less than 64 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. 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 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable. Enables 2-KB Flash memory blocks to be executed or read. The policies may be combined as shown in the table "Flash Protection Policy Combinations".
				Value Description
			0xFFFFFFFF	Bits [31:0] each enable protection on a 2-KB block of Flash memory in memory range from 65 to 128 KB.

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 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable

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
0xFFFFFFFF	Enables 256 KB of flash.

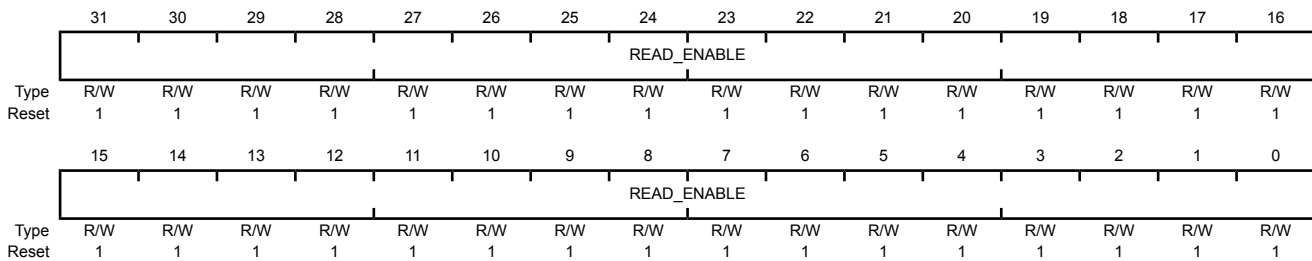
Register 15: Flash Memory Protection Read Enable 3 (FMPRE3), offset 0x20C

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 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
-----------	------	------	-------	-------------

31:0	READ_ENABLE	R/W	0xFFFFFFFF	Flash Read Enable
------	-------------	-----	------------	-------------------

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
0xFFFFFFFF	Enables 256 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 (**FMPPE_n** stores the execute-only bits). Flash memory up to a total of 64 KB is controlled by this register. Other **FMPPE_n** registers (if any) provide protection for other 64K blocks. This register is loaded during the power-on reset sequence. The factory settings for the **FMPPE_n** and **FMPPE_n** 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. The reset value shown only applies to power-on reset; any other type of reset does not affect this register. Once committed, the only way to restore the factory default value of this register is to perform the "Recover Locked Device" sequence detailed in the JTAG chapter. If the Flash memory size on the device is less than 64 KB, this register usually reads as zeroes, but software should not rely on these bits to be zero. 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 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable

Value	Description
0xFFFFFFFF	Bits [31:0] each enable protection on a 2-KB block of Flash memory in memory range from 65 to 128 KB.

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

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

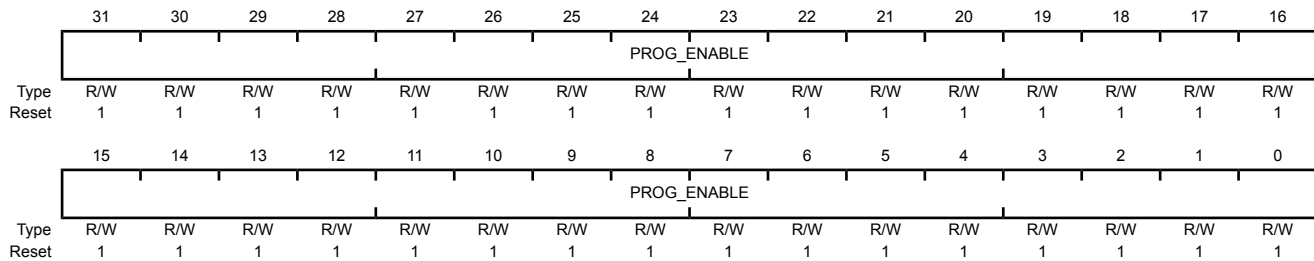
This register stores the execute-only protection bits for each 2-KB flash block (**FMPPE_n** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPPE_n** and **FMPPE_n** registers are a value of 1 for all implemented banks. This achieves 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 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable

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
0xFFFFFFFF	Enables 256 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 (**FMPPE_n** stores the execute-only bits). This register is loaded during the power-on reset sequence. The factory settings for the **FMPPE_n** and **FMPREN** 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 0xFFFF.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	PROG_ENABLE	R/W	0xFFFFFFFF	Flash Programming Enable

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
0xFFFFFFFF	Enables 256 KB of flash.

9 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, Port H). The GPIO module supports 7-60 programmable input/output pins, depending on the peripherals being used.

The GPIO module has the following features:

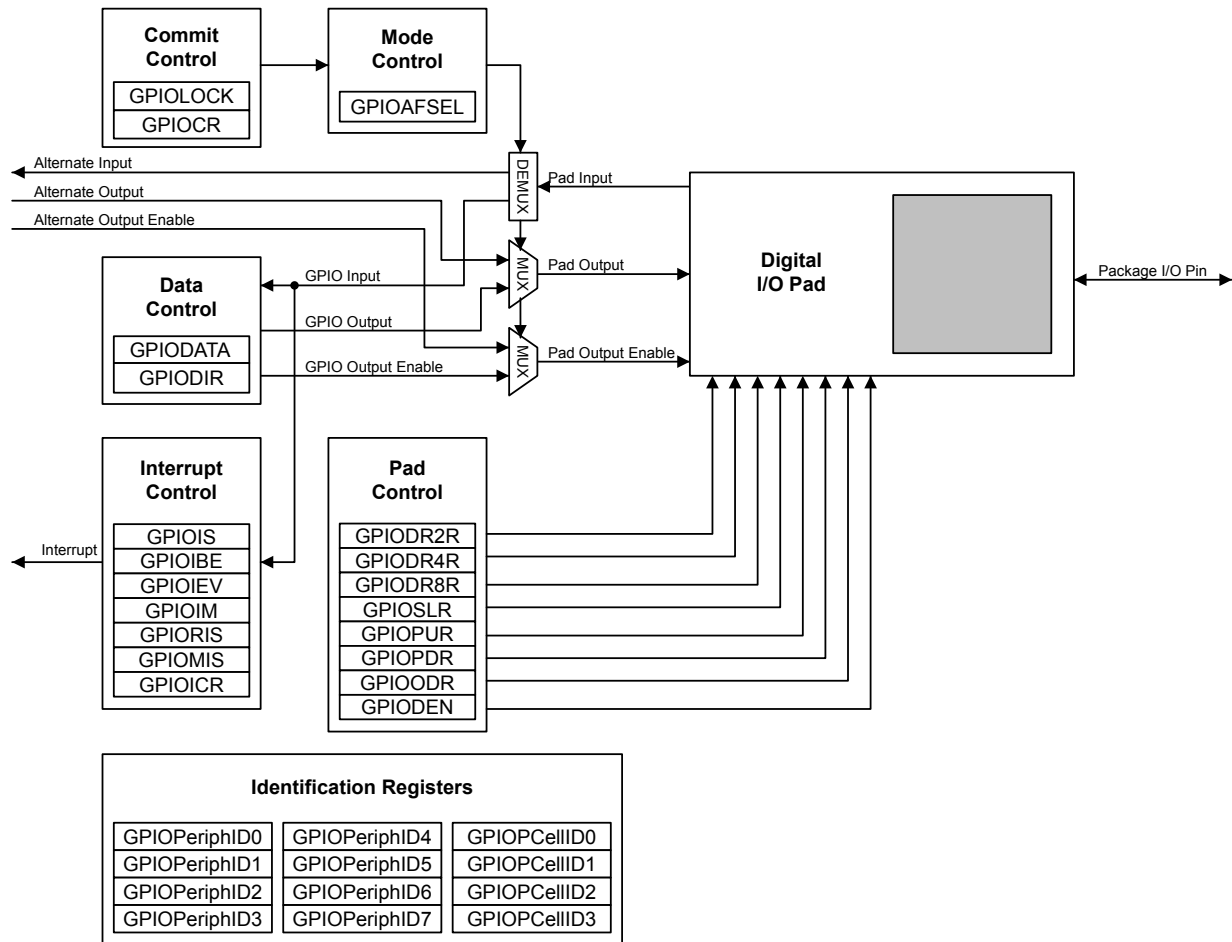
- 7-60 GPIOs, depending on configuration
- 5-V-tolerant input/outputs
- Programmable control for GPIO interrupts
 - Interrupt generation masking
 - Edge-triggered on rising, falling, or both
 - Level-sensitive on High or Low values
- Bit masking in both read and write operations through address lines
- Pins configured as digital inputs are Schmitt-triggered.
- Programmable control for GPIO pad configuration
 - Weak pull-up or pull-down resistors
 - 2-mA, 4-mA, and 8-mA pad drive for digital communication; up to four pads can be configured with an 18-mA pad drive for high-current applications
 - Slew rate control for the 8-mA drive
 - Open drain enables
 - Digital input enables

9.1 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 9-1 on page 173). The LM3S1960 microcontroller contains eight ports and thus eight of these physical GPIO blocks.

Figure 9-1. GPIO Port Block Diagram



9.1.1 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.

9.1.1.1 Data Direction Operation

The **GPIO Direction (GPIODIR)** register (see page 180) 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.

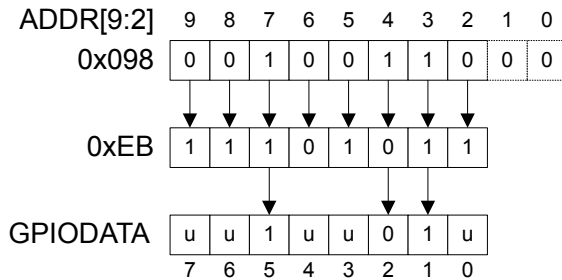
9.1.1.2 Data Register Operation

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 179) 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 **GPIO DATA** register is altered. If it is cleared to 0, it is left unchanged.

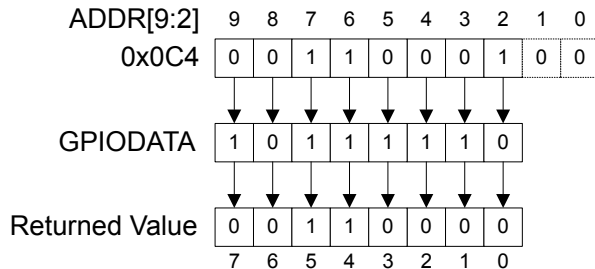
For example, writing a value of 0xEB to the address **GPIO DATA** + 0x098 would yield as shown in Figure 9-2 on page 174, where *u* is data unchanged by the write.

Figure 9-2. GPIO DATA 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 **GPIO DATA** + 0x0C4 yields as shown in Figure 9-3 on page 174.

Figure 9-3. GPIO DATA Read Example



9.1.2 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 181)
- **GPIO Interrupt Both Edges (GPIOIBE)** register (see page 182)
- **GPIO Interrupt Event (GPIOIEV)** register (see page 183)

Interrupts are enabled/disabled via the **GPIO Interrupt Mask (GPIOIM)** register (see page 184).

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 185 and page 186). 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 appropriate bit of the **GPIO Interrupt Clear (GPIOICR)** register (see page 187).

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.

9.1.3 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 188), the pin state is controlled by its alternate function (that is, the peripheral). Software control corresponds to GPIO mode, where the **GPIO DATA** register is used to read/write the corresponding pins.

9.1.4 Commit Control

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is currently provided for the five JTAG/SWD pins ($PB7$ and $PC[3:0]$). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 188) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 198) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 199) have been set to 1.

9.1.5 Pad Control

The pad control registers allow for GPIO pad configuration by software based on the application requirements. The pad control registers include the **GPIO DR2R**, **GPIO DR4R**, **GPIO DR8R**, **GPIO DR**, **GPIO PUR**, **GPIO PDR**, **GPIO SLR**, and **GPIO DEN** registers. These registers control drive strength, open-drain configuration, pull-up and pull-down resistors, slew-rate control and digital input enable.

For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the V_{OL} value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package or BGA pin group with the total number of high-current GPIO outputs not exceeding four for the entire package.

9.1.6 Identification

The identification registers configured at reset allow software to detect and identify the module as a GPIO block. The identification registers include the **GPIO PeriphID0-GPIO PeriphID7** registers as well as the **GPIO CellID0-GPIO CellID3** registers.

9.2 Initialization and Configuration

To use the GPIO, the peripheral clock must be enabled by setting the appropriate GPIO Port bit field ($GPIO_n$) 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**, **GPIO DEN=0**, **GPIO PDR=0**, and **GPIO PUR=0**. Table 9-1 on page 176 shows all possible configurations of the GPIO pads and the control register settings required to achieve them. Table 9-2 on page 176 shows how a rising edge interrupt would be configured for pin 2 of a GPIO port.

Table 9-1. GPIO Pad Configuration Examples

Configuration	GPIO Register Bit Value ^a									
	AFSEL	DIR	ODR	DEN	PUR	PDR	DR2R	DR4R	DR8R	SLR
Digital Input (GPIO)	0	0	0	1	?	?	X	X	X	X
Digital Output (GPIO)	0	1	0	1	?	?	?	?	?	?
Open Drain Output (GPIO)	0	1	1	1	X	X	?	?	?	?
Open Drain Input/Output (I ² C)	1	X	1	1	X	X	?	?	?	?
Digital Input (Timer CCP)	1	X	0	1	?	?	X	X	X	X
Digital Input (QEI)	1	X	0	1	?	?	X	X	X	X
Digital Output (PWM)	1	X	0	1	?	?	?	?	?	?
Digital Output (Timer PWM)	1	X	0	1	?	?	?	?	?	?
Digital Input/Output (SSI)	1	X	0	1	?	?	?	?	?	?
Digital Input/Output (UART)	1	X	0	1	?	?	?	?	?	?
Analog Input (Comparator)	0	0	0	0	0	0	X	X	X	X
Digital Output (Comparator)	1	X	0	1	?	?	?	?	?	?

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

?=Can be either 0 or 1, depending on the configuration

Table 9-2. GPIO Interrupt Configuration Example

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

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

9.3 Register Map

Table 9-3 on page 177 lists the GPIO registers. The offset listed is a hexadecimal increment to the register's address, relative to that GPIO port's base address:

- GPIO Port A: 0x4000.4000
- GPIO Port B: 0x4000.5000
- GPIO Port C: 0x4000.6000
- 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-committable. 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 9-3. GPIO Register Map

Offset	Name	Type	Reset	Description	See page
0x000	GPIODATA	R/W	0x0000.0000	GPIO Data	179
0x400	GPIODIR	R/W	0x0000.0000	GPIO Direction	180
0x404	GPIOIS	R/W	0x0000.0000	GPIO Interrupt Sense	181
0x408	GPIOIBE	R/W	0x0000.0000	GPIO Interrupt Both Edges	182
0x40C	GPIOIEV	R/W	0x0000.0000	GPIO Interrupt Event	183
0x410	GPIOIM	R/W	0x0000.0000	GPIO Interrupt Mask	184
0x414	GPIORIS	RO	0x0000.0000	GPIO Raw Interrupt Status	185
0x418	GPIOMIS	RO	0x0000.0000	GPIO Masked Interrupt Status	186
0x41C	GPIOICR	W1C	0x0000.0000	GPIO Interrupt Clear	187

Table 9-3. GPIO Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x420	GPIOAFSEL	R/W	-	GPIO Alternate Function Select	188
0x500	GPIODR2R	R/W	0x0000.00FF	GPIO 2-mA Drive Select	190
0x504	GPIODR4R	R/W	0x0000.0000	GPIO 4-mA Drive Select	191
0x508	GPIODR8R	R/W	0x0000.0000	GPIO 8-mA Drive Select	192
0x50C	GPIOODR	R/W	0x0000.0000	GPIO Open Drain Select	193
0x510	GPIOPUR	R/W	-	GPIO Pull-Up Select	194
0x514	GPIOPDR	R/W	0x0000.0000	GPIO Pull-Down Select	195
0x518	GPIOSLR	R/W	0x0000.0000	GPIO Slew Rate Control Select	196
0x51C	GIODEN	R/W	-	GPIO Digital Enable	197
0x520	GPIOLOCK	R/W	0x0000.0001	GPIO Lock	198
0x524	GPIOCR	-	-	GPIO Commit	199
0xFD0	GPIOPeriphID4	RO	0x0000.0000	GPIO Peripheral Identification 4	201
0xFD4	GPIOPeriphID5	RO	0x0000.0000	GPIO Peripheral Identification 5	202
0xFD8	GPIOPeriphID6	RO	0x0000.0000	GPIO Peripheral Identification 6	203
0xFDC	GPIOPeriphID7	RO	0x0000.0000	GPIO Peripheral Identification 7	204
0xFE0	GPIOPeriphID0	RO	0x0000.0061	GPIO Peripheral Identification 0	205
0xFE4	GPIOPeriphID1	RO	0x0000.0000	GPIO Peripheral Identification 1	206
0xFE8	GPIOPeriphID2	RO	0x0000.0018	GPIO Peripheral Identification 2	207
0xFEC	GPIOPeriphID3	RO	0x0000.0001	GPIO Peripheral Identification 3	208
0xFF0	GPIOPCelID0	RO	0x0000.000D	GPIO PrimeCell Identification 0	209
0xFF4	GPIOPCelID1	RO	0x0000.00F0	GPIO PrimeCell Identification 1	210
0xFF8	GPIOPCelID2	RO	0x0000.0005	GPIO PrimeCell Identification 2	211
0xFFC	GPIOPCelID3	RO	0x0000.00B1	GPIO PrimeCell Identification 3	212

9.4 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 180).

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: 0x4002.6000
 GPIO Port H base: 0x4002.7000
 Offset 0x000
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DATA							
Type	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	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

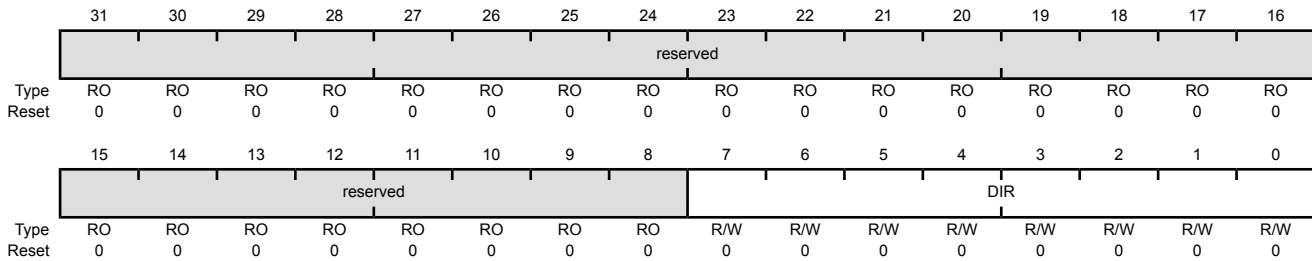
This register is virtually mapped to 256 locations in the address space. To facilitate the reading and writing of data to these registers by independent drivers, the data read from and the data written to the registers are masked by the eight address lines `ipaddr[9:2]`. Reads from this register return its current state. Writes to this register only affect bits that are not masked by `ipaddr[9:2]` and are configured as outputs. See "Data Register Operation" on page 173 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
 Offset 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:

Value	Description
0	Pins are inputs.
1	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								IS							
Type	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	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 181) 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 183). 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								IBE							
Type	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	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
0	Interrupt generation is controlled by the GPIO Interrupt Event (GPIOIEV) register (see page 183).
1	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 181). 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								IEV							
Type	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	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:

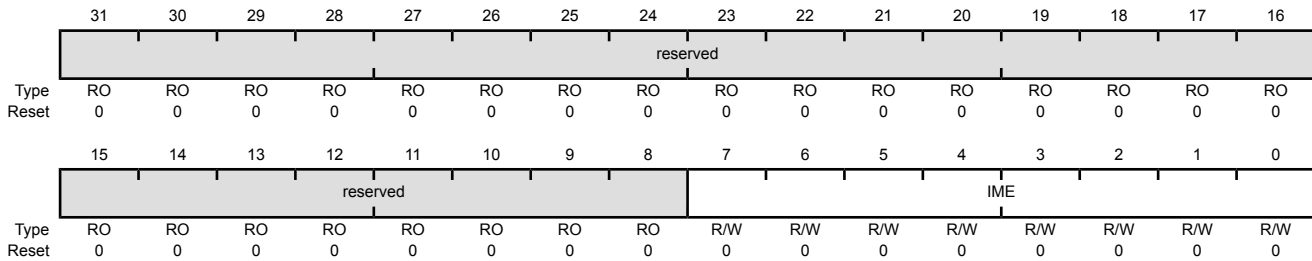
Value	Description
0	Falling edge or Low levels on corresponding pins trigger interrupts.
1	Rising edge or High levels on corresponding pins trigger interrupts.

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
 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	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 184). 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
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								RIS							
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

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

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

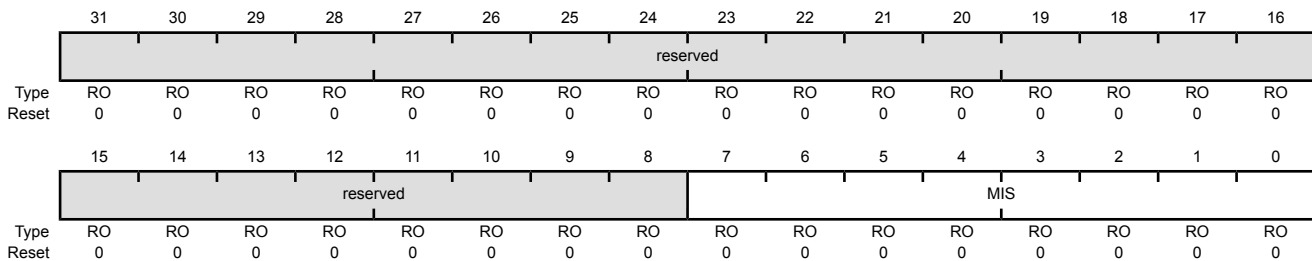
Register 8: GPIO Masked Interrupt Status (GPIOMIS), offset 0x418

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								IC							
Type	RO	RO	RO	RO	RO	RO	RO	RO	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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	IC	W1C	0x00	GPIO Interrupt Clear

The **IC** values are defined as follows:

Value	Description
0	Corresponding interrupt is unaffected.
1	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 GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is currently provided for the five JTAG/SWD pins (PB7 and PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 188) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 198) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 199) have been set to 1.

Important: All GPIO pins are tri-stated by default (**GPIOAFSEL=0**, **GIODEN=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**, **GIODEN=1** and **GPIOPUR=1**). A Power-On-Reset (\overline{POR}) or asserting \overline{RST} puts both groups of pins back to their default state.

Caution – 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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								AFSEL							
Type	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	-	-	-	-	-	-	-	-

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.

Bit/Field	Name	Type	Reset	Description
7:0	AFSEL	R/W	-	GPIO Alternate Function Select The AFSEL values are defined as follows: Value Description 0 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 (P_{B7} and $P_{C[3:0]}$). 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DRV2							
Type	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	1	1	1	1	1	1	1	1

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	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DRV4							
Type	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	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	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DRV8							
Type	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	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	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 197). 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 cleared. If open drain is selected while the GPIO is configured as an input, the GPIO will remain an input and the open-drain selection has no effect until the GPIO is changed to an output.

When using the I²C module, in addition to configuring the pin to open drain, the **GPIO Alternate Function Select (GPIOAFSEL)** register bits for the I²C clock and data pins should be set to 1 (see examples in “Initialization and Configuration” on page 175).

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
 Offset 0x50C
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								ODE							
Type	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	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 195).

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: 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 0x510
 Type R/W, reset -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PUE							
Type	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	-	-	-	-	-	-	-	-

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 194).

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

Offset 0x514

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PDE							
Type	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

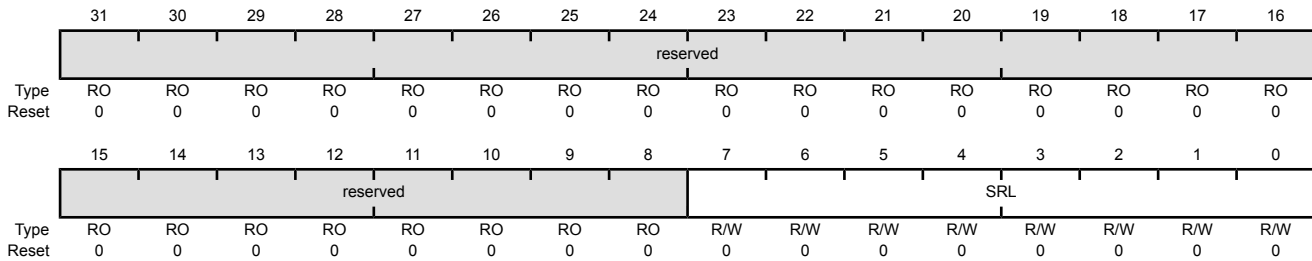
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PDE	R/W	0x00	Pad Weak Pull-Down Enable A write 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 192).

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.
1	Slew rate control enabled.

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

Note: Pins configured as digital inputs are Schmitt-triggered.

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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DEN							
Type	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	-	-	-	-	-	-	-	-

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

- 0 Digital functions disabled.
- 1 Digital functions enabled.

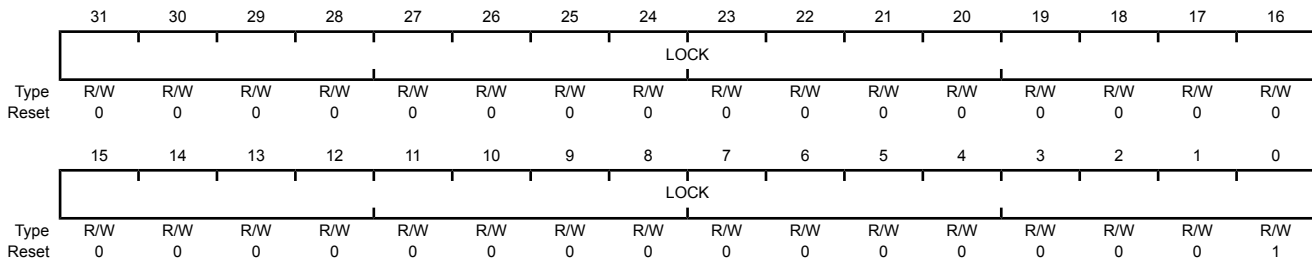
Note: The default reset value for the **GPIODEN**, **GPIOPUR**, and **GPIODEN** registers are 0x0000.0000 for all GPIO pins, with the exception of the five JTAG/SWD pins (**PB7** and **PC[3:0]**). These five pins default to JTAG/SWD functionality. Because of this, the default reset value of these registers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

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

The **GPIOLOCK** register enables write access to the **GPIOCR** register (see page 199). Writing 0x1ACC.E551 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 0x00000000.

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: 0x4002.6000
 GPIO Port H base: 0x4002.7000
 Offset 0x520
 Type R/W, reset 0x0000.0001



Bit/Field	Name	Type	Reset	Description
31:0	LOCK	R/W	0x0000.0001	GPIO Lock

A write of the value 0x1ACC.E551 unlocks the **GPIO Commit (GPIOCR)** register for write access.

A write of any other value or a write to the **GPIOCR** register 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 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 are 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 are ignored if the **GPIOLOCK** register is locked.

Important: This register is designed to prevent accidental programming of the 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 the corresponding 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 -

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CR							
Type	RO	RO	RO	RO	RO	RO	RO	RO	-	-	-	-	-	-	-	-
Reset	0	0	0	0	0	0	0	0	-	-	-	-	-	-	-	-

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.

Bit/Field	Name	Type	Reset	Description
7:0	CR	-	-	<p>GPIO Commit</p> <p>On a bit-wise basis, any bit set allows the corresponding GPIOAFSEL bit to be set to its alternate function.</p> <p>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.</p> <p>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-committable. 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.</p>

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)

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID4							
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

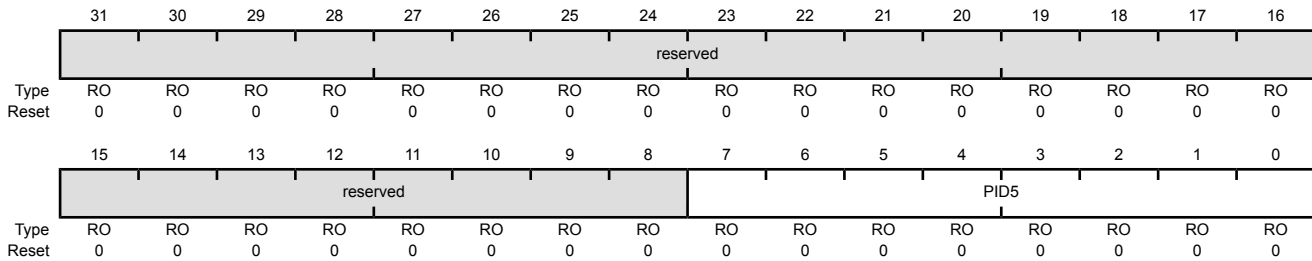
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID4	RO	0x00	GPIO Peripheral ID Register[7:0]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID6							
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

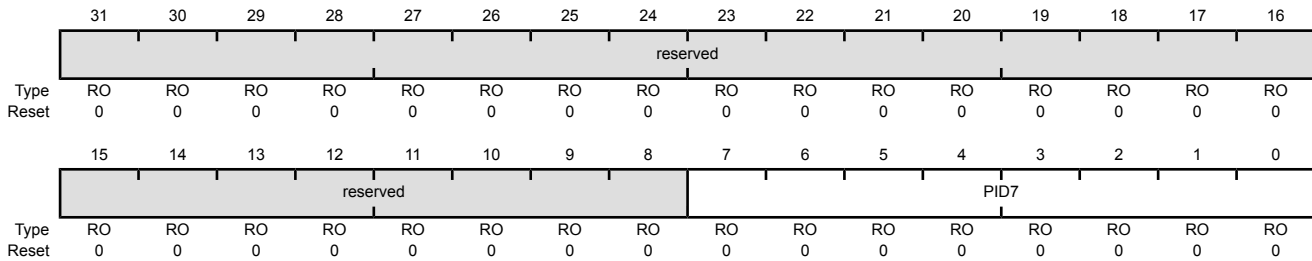
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID6	RO	0x00	GPIO Peripheral ID Register[23:16]

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 0xFE0

Type RO, reset 0x0000.0061

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID0							
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	1	1	0	0	0	0	1

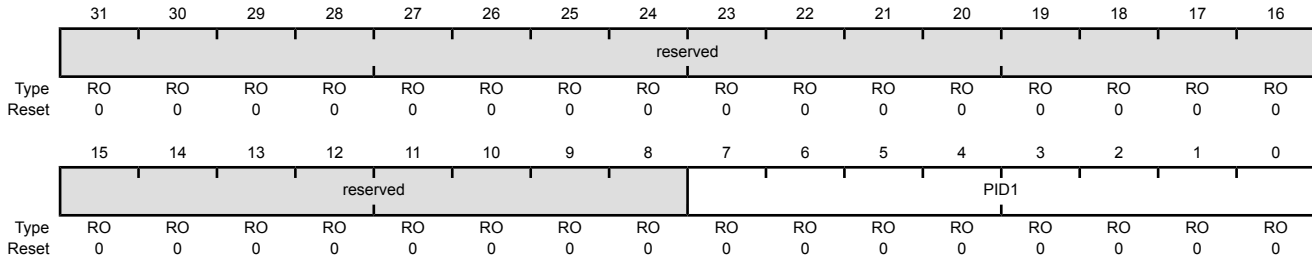
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID0	RO	0x61	GPIO Peripheral ID Register[7:0] Can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

Offset 0xFE8

Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID2							
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	1	1	0	0

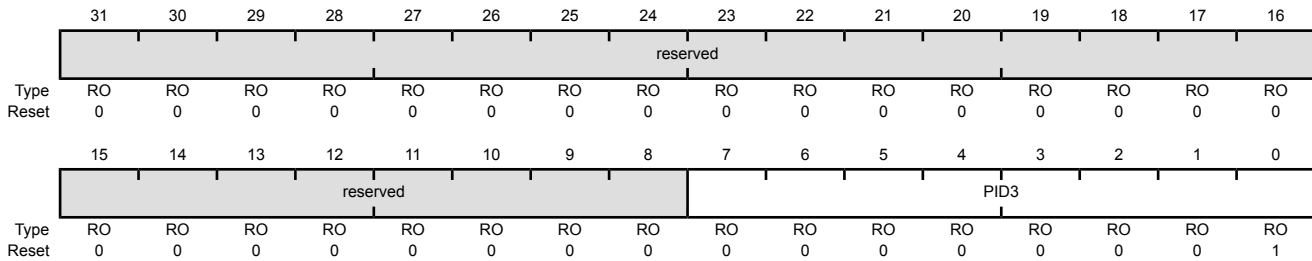
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID2	RO	0x18	GPIO Peripheral ID Register[23:16] 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
 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	GPIO Peripheral ID Register[31:24] Can be used by software to identify the presence of this peripheral.

Register 29: GPIO PrimeCell Identification 0 (GPIOCellID0), offset 0xFF0

The **GPIOCellID0**, **GPIOCellID1**, **GPIOCellID2**, and **GPIOCellID3** 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 (GPIOCellID0)

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID0							
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	1	1	0	1

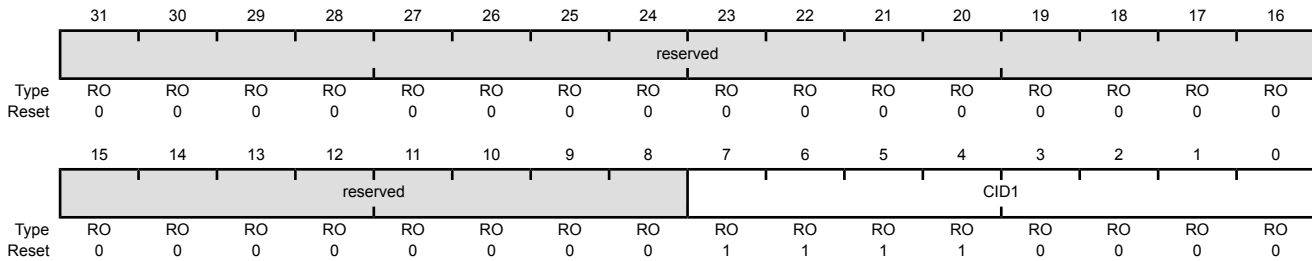
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID0	RO	0x0D	GPIO PrimeCell ID Register[7:0] Provides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

Register 30: GPIO PrimeCell Identification 1 (GPIOCellID1), offset 0xFF4

The **GPIOCellID0**, **GPIOCellID1**, **GPIOCellID2**, and **GPIOCellID3** 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 (GPIOCellID1)

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] Provides software a standard cross-peripheral identification system.

Register 31: GPIO PrimeCell Identification 2 (GPIOCellID2), offset 0xFF8

The **GPIOCellID0**, **GPIOCellID1**, **GPIOCellID2**, and **GPIOCellID3** 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 (GPIOCellID2)

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID2							
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	1	1

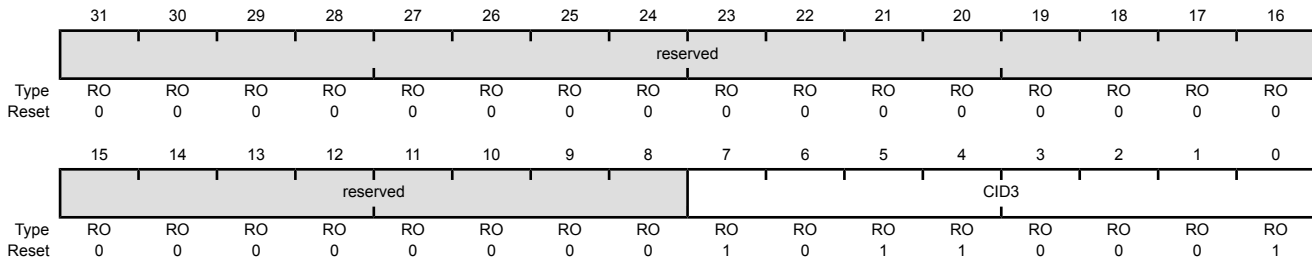
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID2	RO	0x05	GPIO PrimeCell ID Register[23:16]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

Register 32: GPIO PrimeCell Identification 3 (GPIOCellID3), offset 0xFFC

The **GPIOCellID0**, **GPIOCellID1**, **GPIOCellID2**, and **GPIOCellID3** 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 (GPIOCellID3)

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 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	GPIO PrimeCell ID Register[31:24] Provides software a standard cross-peripheral identification system.

10 General-Purpose Timers

Programmable timers can be used to count or time external events that drive the Timer input pins. The Stellaris® General-Purpose Timer Module (GPTM) contains four GPTM blocks (Timer0, Timer1, Timer 2, and Timer 3). 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).

The GPT Module is one timing resource available on the Stellaris® microcontrollers. Other timer resources include the System Timer (SysTick) (see “System Timer (SysTick)” on page 43) and the PWM timer in the PWM module (see “PWM Timer” on page 401).

The General-Purpose Timers provide the following features:

- Four 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
 - 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
 - User-enabled stalling when the controller asserts CPU Halt flag during debug
- 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

10.1 Block Diagram

Note: In Figure 10-1 on page 214, the specific CCP pins available depend on the Stellaris® device. See Table 10-1 on page 214 for the available CCPs.

Figure 10-1. GPTM Module Block Diagram

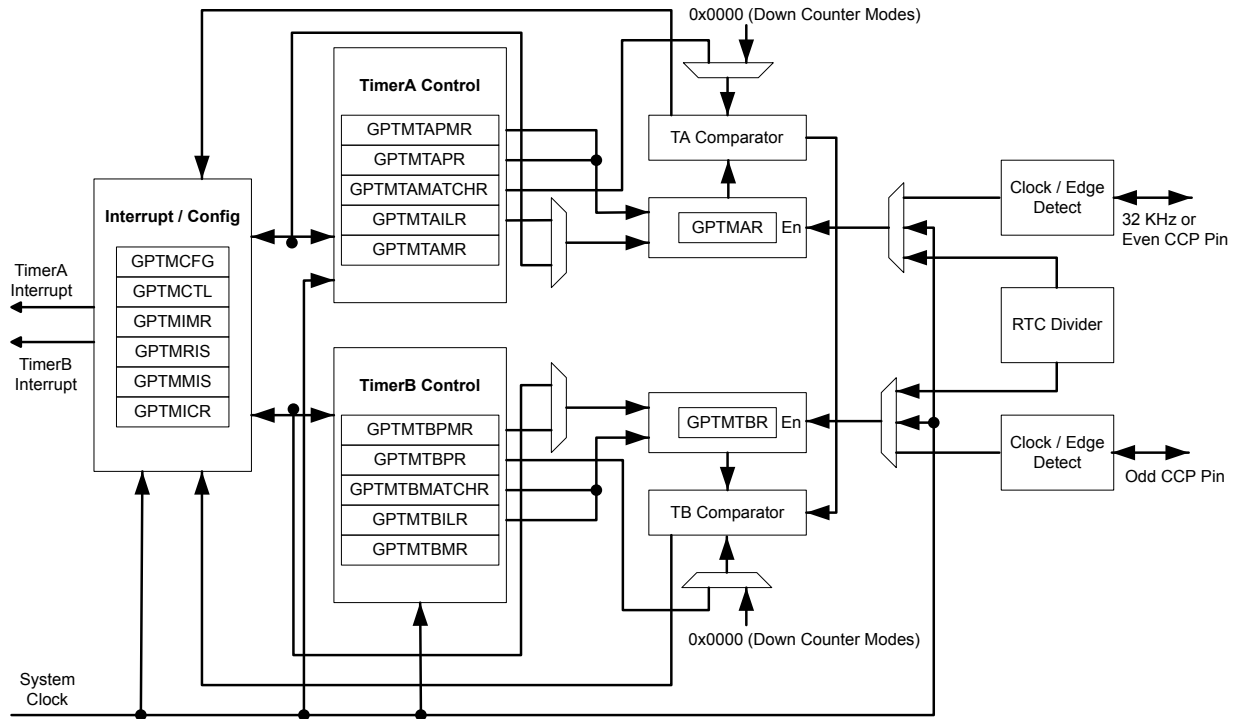


Table 10-1. Available CCP Pins

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	CCP4	-
	TimerB	-	CCP5
Timer 3	TimerA	CCP6	-
	TimerB	-	CCP7

10.2 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 225), the **GPTM TimerA Mode (GPTMTAMR)** register (see page 226), and the **GPTM TimerB Mode**

(**GPTMTBMR**) register (see page 228). 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.

10.2.1 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 (GPTMTAILR)** register (see page 239) and the **GPTM TimerB Interval Load (GPTMTBILR)** register (see page 240). The prescale counters are initialized to 0x00: the **GPTM TimerA Prescale (GPTMTAPR)** register (see page 243) and the **GPTM TimerB Prescale (GPTMTBPR)** register (see page 244).

10.2.2 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 239
- **GPTM TimerB Interval Load (GPTMTBILR)** register [15:0], see page 240
- **GPTM TimerA (GPTMTAR)** register [15:0], see page 247
- **GPTM TimerB (GPTMTBR)** register [15:0], see page 248

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]
```

10.2.2.1 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 226), 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 230), 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 triggers when it reaches the 0x000.0000 state. The GPTM sets the **TATORIS** bit in the **GPTM Raw Interrupt Status (GPTMRIS)** register (see page 235), and holds it until it is cleared by writing the **GPTM Interrupt Clear (GPTMICR)** register (see page 237). If the time-out interrupt is enabled in the **GPTM Interrupt**

Mask (GPTIMR) register (see page 233), the GPTM also sets the `TATOMIS` bit in the **GPTM Masked Interrupt Status (GPTMMIS)** register (see page 236).

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.

If the `TASTALL` bit in the **GPTMCTL** register is asserted, the timer freezes counting until the signal is deasserted.

10.2.2.2 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 241) 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 in the **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 `RTCRES` 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**.

10.2.3 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 225). 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.

10.2.3.1 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 `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 `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 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.

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.

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 50-MHz clock with $Tc=20$ ns (clock period).

Table 10-2. 16-Bit Timer With Prescaler Configurations

Prescale	#Clock (T c) ^a	Max Time	Units
00000000	1	1.3107	mS
00000001	2	2.6214	mS
00000010	3	3.9321	mS
-----	--	--	--
11111100	254	332.9229	mS
11111110	255	334.2336	mS
11111111	256	335.5443	mS

a. Tc is the clock period.

10.2.3.2 16-Bit Input Edge Count Mode

Note: For rising-edge detection, the input signal must be High for at least two system clock periods following the rising edge. Similarly, for falling-edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.

Note: The prescaler is not available in 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.

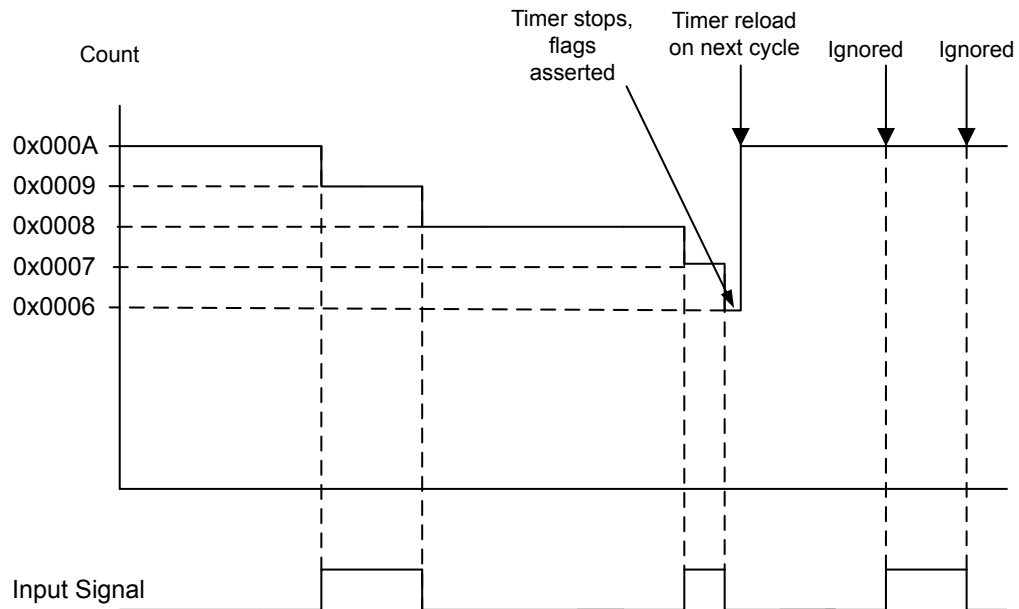
When software writes the $TnEN$ bit in the **GPTM Control (GPTMCTL)** register, the timer is enabled for event capture. Each input event on the CCP pin decrements the counter by 1 until the event count matches **GPTMTnMATCHR**. When the counts match, the GPTM asserts the $CnMRIS$ bit in the **GPTMRIS** 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 10-2 on page 218 shows how input edge count mode works. In this case, the timer start value is set to **GPTMTnILR** = 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.

Figure 10-2. 16-Bit Input Edge Count Mode Example



10.2.3.3 16-Bit Input Edge Time Mode

Note: For rising-edge detection, the input signal must be High for at least two system clock periods following the rising edge. Similarly, for falling edge detection, the input signal must be Low for at least two system clock periods following the falling edge. Based on this criteria, the maximum input frequency for edge detection is 1/4 of the system frequency.

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 either rising or falling edges, but not both. 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.

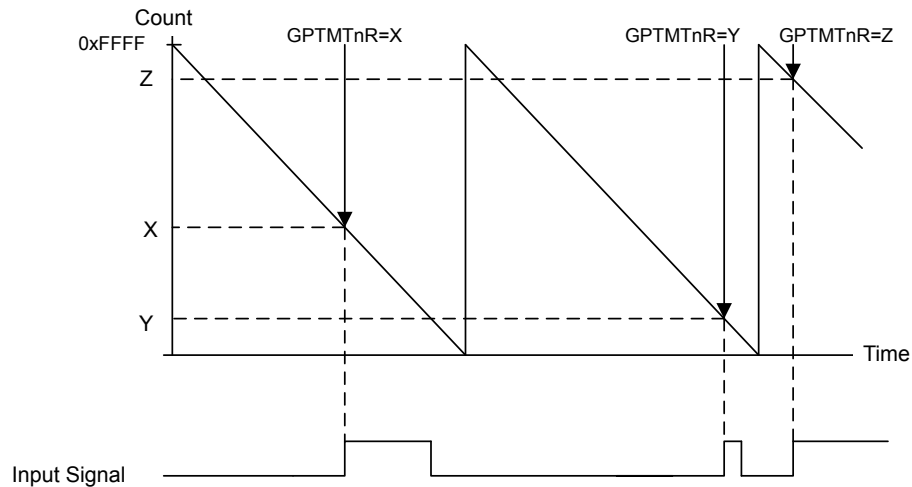
When software writes the **TnEN** bit in the **GPTMCTL** register, the timer is enabled for event capture. When the selected input event is detected, the current **Tn** counter value is captured in the **GPTMTnR** register and is available to be read by the controller. The GPTM then asserts the **CnERIS** 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 **TnEN** bit is cleared. When the timer reaches the 0x0000 state, it is reloaded with the value from the **GPTMnILR** register.

Figure 10-3 on page 219 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 10-3. 16-Bit Input Edge Time Mode Example



10.2.3.4 16-Bit PWM Mode

Note: The prescaler is not available in 16-Bit PWM mode.

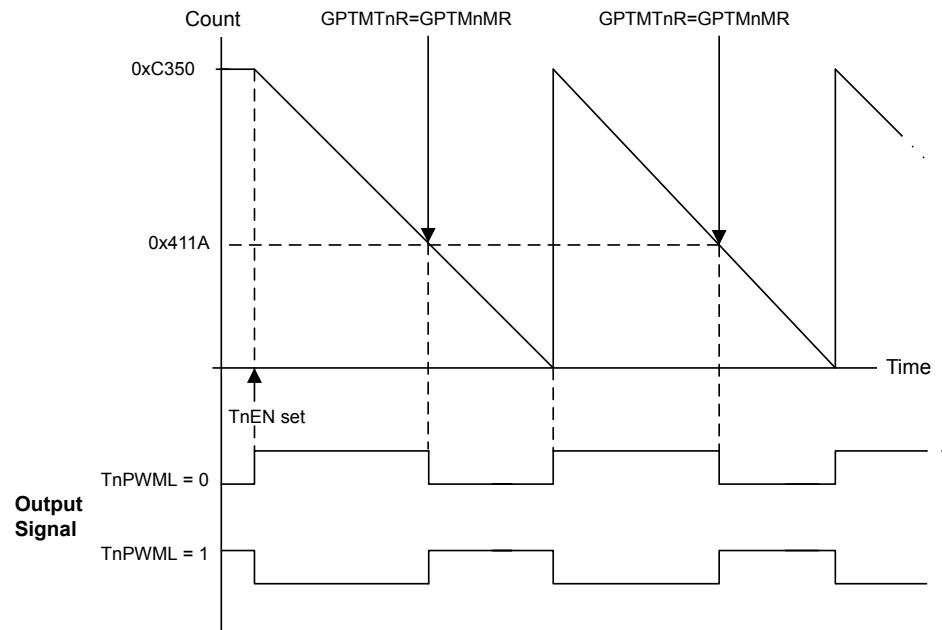
The GPTM supports a simple PWM generation mode. In PWM mode, the timer is configured as a down-counter with a start value (and thus period) defined by **GPTMTnILR**. PWM mode is enabled with the **GPTMTnMR** register by setting the $TnAMS$ bit to 0x1, the $TnCMR$ bit to 0x0, and the $TnMR$ field to 0x2.

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 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 10-4 on page 220 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 **GPTMnILR**=0xC350 and the match value is **GPTMnMR**=0x411A.

Figure 10-4. 16-Bit PWM Mode Example



10.3 Initialization and Configuration

To use the general-purpose timers, the peripheral clock must be enabled by setting the `TIMER0`, `TIMER1`, `TIMER2`, and `TIMER3` bits in the `RCGC1` register.

This section shows module initialization and configuration examples for each of the supported timer modes.

10.3.1 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)**.
5. 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 221. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

10.3.2 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:

1. Ensure the timer is disabled (the `TAEN` bit is cleared) before making any changes.
2. Write the **GPTM Configuration Register (GPTMCFG)** with a value of `0x1`.
3. Write the desired match value to the **GPTM TimerA Match Register (GPTMTAMATCHR)**.
4. Set/clear the `RTCEN` bit in the **GPTM Control Register (GPTMCTL)** as desired.
5. If interrupts are required, set the `RTCIM` bit in the **GPTM Interrupt Mask Register (GPTMIMR)**.
6. 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.

10.3.3 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`.
3. 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.
4. If a prescaler is to be used, write the prescale value to the **GPTM Timern Prescale Register (GPTMTnPR)**.
5. Load the start value into the **GPTM Timer Interval Load Register (GPTMTnILR)**.
6. If interrupts are required, set the `TnTOIM` 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)**.

In One-Shot mode, the timer stops counting after step 8 on page 221. To re-enable the timer, repeat the sequence. A timer configured in Periodic mode does not stop counting after it times out.

10.3.4 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.
4. 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.
6. 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 $TnEN$ 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 $TnEN$ bit is cleared and repeat step 4 on page 222 through step 9 on page 222.

10.3.5 16-Bit Input Edge Timing Mode

A timer is configured to Input Edge Timing 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 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.
5. 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.
7. Set the $TnEN$ bit in the **GPTM Control (GPTMCTL)** register to enable the timer and start counting.
8. Poll the $CnERIS$ 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**

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.

10.3.6 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 $TnEVENT$ field of the **GPTM Control (GPTMCTL)** register.
5. 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.
7. 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.

10.4 Register Map

Table 10-3 on page 223 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
- Timer3: 0x4003.3000

Table 10-3. Timers Register Map

Offset	Name	Type	Reset	Description	See page
0x000	GPTMCFG	R/W	0x0000.0000	GPTM Configuration	225
0x004	GPTMTAMR	R/W	0x0000.0000	GPTM TimerA Mode	226
0x008	GPTMTBMR	R/W	0x0000.0000	GPTM TimerB Mode	228
0x00C	GPTMCTL	R/W	0x0000.0000	GPTM Control	230
0x018	GPTMIMR	R/W	0x0000.0000	GPTM Interrupt Mask	233
0x01C	GPTMRIS	RO	0x0000.0000	GPTM Raw Interrupt Status	235

Table 10-3. Timers Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x020	GPTMMIS	RO	0x0000.0000	GPTM Masked Interrupt Status	236
0x024	GPTMICR	W1C	0x0000.0000	GPTM Interrupt Clear	237
0x028	GPTMTAILR	R/W	0xFFFF.FFFF	GPTM TimerA Interval Load	239
0x02C	GPTMTBILR	R/W	0x0000.FFFF	GPTM TimerB Interval Load	240
0x030	GPTMTAMATCHR	R/W	0xFFFF.FFFF	GPTM TimerA Match	241
0x034	GPTMTBMATCHR	R/W	0x0000.FFFF	GPTM TimerB Match	242
0x038	GPTMTAPR	R/W	0x0000.0000	GPTM TimerA Prescale	243
0x03C	GPTMTBPR	R/W	0x0000.0000	GPTM TimerB Prescale	244
0x040	GPTMTAPMR	R/W	0x0000.0000	GPTM TimerA Prescale Match	245
0x044	GPTMTBPMR	R/W	0x0000.0000	GPTM TimerB Prescale Match	246
0x048	GPTMTAR	RO	0xFFFF.FFFF	GPTM TimerA	247
0x04C	GPTMTBR	RO	0x0000.FFFF	GPTM TimerB	248

10.5 Register Descriptions

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

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
 Timer3 base: 0x4003.3000
 Offset 0x000
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													GPTMCFG			
Type	RO	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	0

Bit/Field	Name	Type	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.
2:0	GPTMCFG	R/W	0x0	GPTM Configuration

The GPTMCFG values are defined as follows:

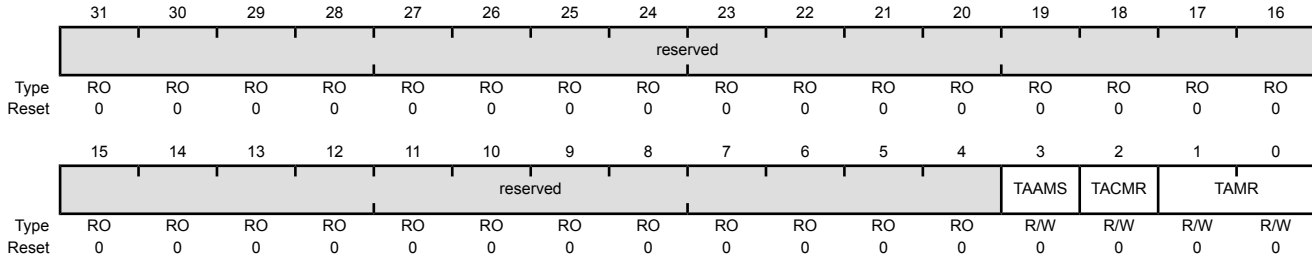
Value	Description
0x0	32-bit timer configuration.
0x1	32-bit real-time clock (RTC) counter configuration.
0x2	Reserved
0x3	Reserved
0x4-0x7	16-bit timer configuration, function is controlled by bits 1:0 of GPTMTAMR and GPTMTBMR .

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)

Timer0 base: 0x4003.0000
 Timer1 base: 0x4003.1000
 Timer2 base: 0x4003.2000
 Timer3 base: 0x4003.3000
 Offset 0x004
 Type R/W, reset 0x0000.0000



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	TAAMS	R/W	0	GPTM TimerA Alternate Mode Select The TAAMS values are defined as follows: Value Description 0 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.
2	TACMR	R/W	0	GPTM TimerA Capture Mode The TACMR values are defined as follows: Value Description 0 Edge-Count mode 1 Edge-Time mode

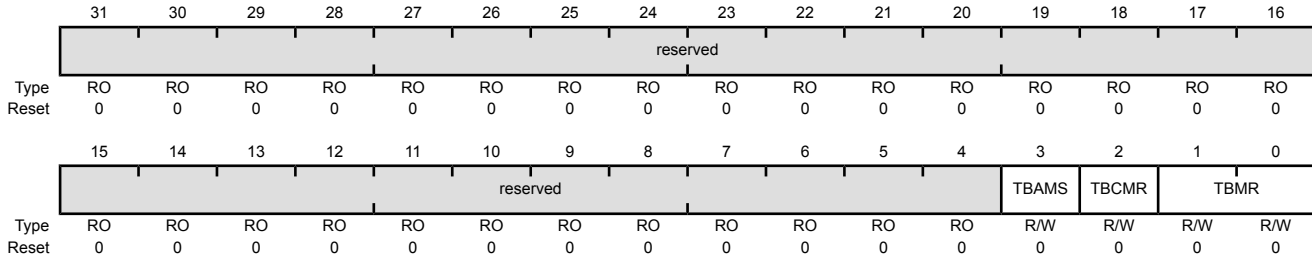
Bit/Field	Name	Type	Reset	Description										
1:0	TAMR	R/W	0x0	<p>GPTM TimerA Mode</p> <p>The TAMR values are defined as follows:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0</td><td>Reserved</td></tr><tr><td>0x1</td><td>One-Shot Timer mode</td></tr><tr><td>0x2</td><td>Periodic Timer mode</td></tr><tr><td>0x3</td><td>Capture mode</td></tr></tbody></table> <p>The Timer mode is based on the timer configuration defined by bits 2:0 in the GPTMCFG register (16-or 32-bit).</p> <p>In 16-bit timer configuration, TAMR controls the 16-bit timer modes for TimerA.</p> <p>In 32-bit timer configuration, this register controls the mode and the contents of GPTMTBMR are ignored.</p>	Value	Description	0x0	Reserved	0x1	One-Shot Timer mode	0x2	Periodic Timer mode	0x3	Capture mode
Value	Description													
0x0	Reserved													
0x1	One-Shot Timer mode													
0x2	Periodic Timer mode													
0x3	Capture mode													

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 0x2.

GPTM TimerB Mode (GPTMTBMR)

Timer0 base: 0x4003.0000
 Timer1 base: 0x4003.1000
 Timer2 base: 0x4003.2000
 Timer3 base: 0x4003.3000
 Offset 0x008
 Type R/W, reset 0x0000.0000



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 The TBAMS values are defined as follows: Value Description 0 Capture mode is enabled. 1 PWM mode is enabled. Note: To enable PWM mode, you must also clear the TBCMR bit and set the TBMR field to 0x2.
2	TBCMR	R/W	0	GPTM TimerB Capture Mode The TBCMR values are defined as follows: Value Description 0 Edge-Count mode 1 Edge-Time mode

Bit/Field	Name	Type	Reset	Description										
1:0	TBMR	R/W	0x0	<p>GPTM TimerB Mode</p> <p>The TBMR values are defined as follows:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0</td><td>Reserved</td></tr><tr><td>0x1</td><td>One-Shot Timer mode</td></tr><tr><td>0x2</td><td>Periodic Timer mode</td></tr><tr><td>0x3</td><td>Capture mode</td></tr></tbody></table> <p>The timer mode is based on the timer configuration defined by bits 2:0 in the GPTMCFG register.</p> <p>In 16-bit timer configuration, these bits control the 16-bit timer modes for TimerB.</p> <p>In 32-bit timer configuration, this register's contents are ignored and GPTMTAMR is used.</p>	Value	Description	0x0	Reserved	0x1	One-Shot Timer mode	0x2	Periodic Timer mode	0x3	Capture mode
Value	Description													
0x0	Reserved													
0x1	One-Shot Timer mode													
0x2	Periodic Timer mode													
0x3	Capture mode													

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.

GPTM Control (GPTMCTL)

Timer0 base: 0x4003.0000
 Timer1 base: 0x4003.1000
 Timer2 base: 0x4003.2000
 Timer3 base: 0x4003.3000
 Offset 0x00C
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved	TBPWML	reserved	reserved	TBEVENT	TBSTALL	TBEN	reserved	TAPWML	reserved	RTCEN	TAEVENT	TASTALL	TAEN		
Type	RO	R/W	RO	RO	R/W	R/W	R/W	R/W	RO	R/W	RO	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	Type	Reset	Description
31:15	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.
14	TBPWML	R/W	0	GPTM TimerB PWM Output Level The TBPWML values are defined as follows: Value Description 0 Output is unaffected. 1 Output is inverted.
13:12	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.
11:10	TBEVENT	R/W	0x0	GPTM TimerB Event Mode The TBEVENT values are defined as follows: Value Description 0x0 Positive edge 0x1 Negative edge 0x2 Reserved 0x3 Both edges
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.

Bit/Field	Name	Type	Reset	Description										
8	TBEN	R/W	0	<p>GPTM TimerB Enable</p> <p>The TBEN values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>TimerB is disabled.</td> </tr> <tr> <td>1</td> <td>TimerB is enabled and begins counting or the capture logic is enabled based on the GPTMCFG register.</td> </tr> </tbody> </table>	Value	Description	0	TimerB is disabled.	1	TimerB is enabled and begins counting or the capture logic is enabled based on the GPTMCFG register.				
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	<p>GPTM TimerA PWM Output Level</p> <p>The TAPWML values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Output is unaffected.</td> </tr> <tr> <td>1</td> <td>Output is inverted.</td> </tr> </tbody> </table>	Value	Description	0	Output is unaffected.	1	Output is inverted.				
Value	Description													
0	Output is unaffected.													
1	Output is inverted.													
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	RTCEN	R/W	0	<p>GPTM RTC Enable</p> <p>The RTCEN values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>RTC counting is disabled.</td> </tr> <tr> <td>1</td> <td>RTC counting is enabled.</td> </tr> </tbody> </table>	Value	Description	0	RTC counting is disabled.	1	RTC counting is enabled.				
Value	Description													
0	RTC counting is disabled.													
1	RTC counting is enabled.													
3:2	TAEVENT	R/W	0x0	<p>GPTM TimerA Event Mode</p> <p>The TAEVENT values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Positive edge</td> </tr> <tr> <td>0x1</td> <td>Negative edge</td> </tr> <tr> <td>0x2</td> <td>Reserved</td> </tr> <tr> <td>0x3</td> <td>Both edges</td> </tr> </tbody> </table>	Value	Description	0x0	Positive edge	0x1	Negative edge	0x2	Reserved	0x3	Both edges
Value	Description													
0x0	Positive edge													
0x1	Negative edge													
0x2	Reserved													
0x3	Both edges													
1	TASTALL	R/W	0	<p>GPTM TimerA Stall Enable</p> <p>The TASTALL values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>TimerA stalling is disabled.</td> </tr> <tr> <td>1</td> <td>TimerA stalling is enabled.</td> </tr> </tbody> </table>	Value	Description	0	TimerA stalling is disabled.	1	TimerA stalling is enabled.				
Value	Description													
0	TimerA stalling is disabled.													
1	TimerA stalling is enabled.													

Bit/Field	Name	Type	Reset	Description
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
 Timer3 base: 0x4003.3000
 Offset 0x018
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved					CBEIM	CBMIM	TBTOIM	reserved					RTCIM	CAEIM	CAMIM	TATOIM
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	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						
31:11	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	CBEIM	R/W	0	GPTM CaptureB Event Interrupt Mask The CBEIM values are defined as follows: <table border="0" style="margin-left: 20px;"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Interrupt is disabled.</td> </tr> <tr> <td>1</td> <td>Interrupt is enabled.</td> </tr> </table>	Value	Description	0	Interrupt is disabled.	1	Interrupt is enabled.
Value	Description									
0	Interrupt is disabled.									
1	Interrupt is enabled.									
9	CBMIM	R/W	0	GPTM CaptureB Match Interrupt Mask The CBMIM values are defined as follows: <table border="0" style="margin-left: 20px;"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Interrupt is disabled.</td> </tr> <tr> <td>1</td> <td>Interrupt is enabled.</td> </tr> </table>	Value	Description	0	Interrupt is disabled.	1	Interrupt is enabled.
Value	Description									
0	Interrupt is disabled.									
1	Interrupt is enabled.									
8	TBTOIM	R/W	0	GPTM TimerB Time-Out Interrupt Mask The TBTOIM values are defined as follows: <table border="0" style="margin-left: 20px;"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Interrupt is disabled.</td> </tr> <tr> <td>1</td> <td>Interrupt is enabled.</td> </tr> </table>	Value	Description	0	Interrupt is disabled.	1	Interrupt is enabled.
Value	Description									
0	Interrupt is disabled.									
1	Interrupt is enabled.									
7:4	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.						

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.

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
 Timer3 base: 0x4003.3000
 Offset 0x01C
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved					CBERIS	CBMRIS	TBTORIS	reserved				RTCRIS	CAERIS	CAMRIS	TATORIS
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

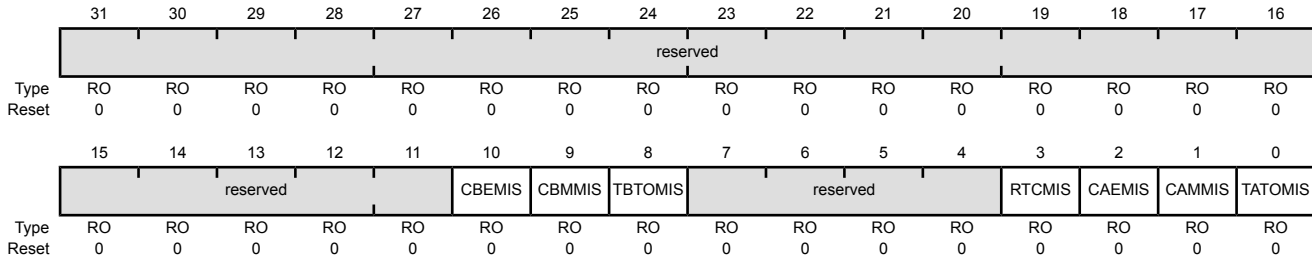
Bit/Field	Name	Type	Reset	Description
31:11	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
10	CBERIS	RO	0	GPTM CaptureB Event Raw Interrupt This is the CaptureB Event interrupt status prior to masking.
9	CBMRIS	RO	0	GPTM CaptureB Match Raw Interrupt This is the CaptureB Match interrupt status prior to masking.
8	TBTORIS	RO	0	GPTM TimerB Time-Out Raw Interrupt This is the TimerB time-out interrupt status prior to masking.
7:4	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	RTCRIS	RO	0	GPTM RTC Raw Interrupt This is the RTC Event interrupt status prior to masking.
2	CAERIS	RO	0	GPTM CaptureA Event Raw Interrupt This is the CaptureA Event interrupt status prior to masking.
1	CAMRIS	RO	0	GPTM CaptureA Match Raw Interrupt This is the CaptureA Match interrupt status prior to masking.
0	TATORIS	RO	0	GPTM TimerA Time-Out Raw Interrupt This the TimerA time-out interrupt status prior to masking.

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
 Timer3 base: 0x4003.3000
 Offset 0x020
 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	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	CBEMIS	RO	0	GPTM CaptureB Event Masked Interrupt This is the CaptureB event interrupt status after masking.
9	CBMMIS	RO	0	GPTM CaptureB Match Masked Interrupt This is the CaptureB match interrupt status after masking.
8	TBTOMIS	RO	0	GPTM TimerB Time-Out Masked Interrupt This is the TimerB time-out interrupt status after masking.
7:4	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	RTCMIS	RO	0	GPTM RTC Masked Interrupt This is the RTC event interrupt status after masking.
2	CAEMIS	RO	0	GPTM CaptureA Event Masked Interrupt This is the CaptureA event interrupt status after masking.
1	CAMMIS	RO	0	GPTM CaptureA Match Masked Interrupt This is the CaptureA match interrupt status after masking.
0	TATOMIS	RO	0	GPTM TimerA Time-Out Masked Interrupt This is the TimerA time-out interrupt status after masking.

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
 Timer3 base: 0x4003.3000
 Offset 0x024
 Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved						CBECINT	CBMCINT	TBTOCINT	reserved				RTCCINT	CAECINT	CAMCINT	TATOCINT
Type	RO	RO	RO	RO	RO		W1C	W1C	W1C	RO	RO	RO	RO	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	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	CBECINT	W1C	0	GPTM CaptureB Event Interrupt Clear The CBECINT values are defined as follows: Value Description 0 The interrupt is unaffected. 1 The interrupt is cleared.
9	CBMCINT	W1C	0	GPTM CaptureB Match Interrupt Clear The CBMCINT values are defined as follows: Value Description 0 The interrupt is unaffected. 1 The interrupt is cleared.
8	TBTOCINT	W1C	0	GPTM TimerB Time-Out Interrupt Clear The TBTOCINT values are defined as follows: Value Description 0 The interrupt is unaffected. 1 The interrupt is cleared.
7:4	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.

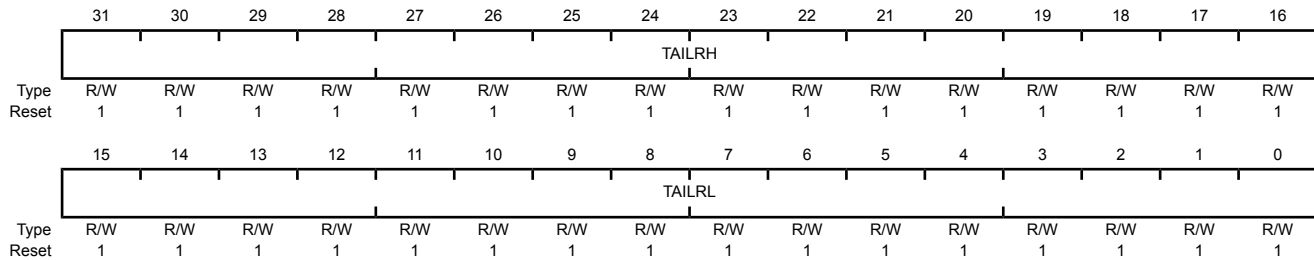
Bit/Field	Name	Type	Reset	Description
3	RTCCINT	W1C	0	GPTM RTC Interrupt Clear The RTCCINT values are defined as follows: Value Description 0 The interrupt is unaffected. 1 The interrupt is cleared.
2	CAECINT	W1C	0	GPTM CaptureA Event Interrupt Clear The CAECINT values are defined as follows: Value Description 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. 1 The interrupt is cleared.

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
 Timer3 base: 0x4003.3000
 Offset 0x028
 Type R/W, reset 0xFFFF.FFFF



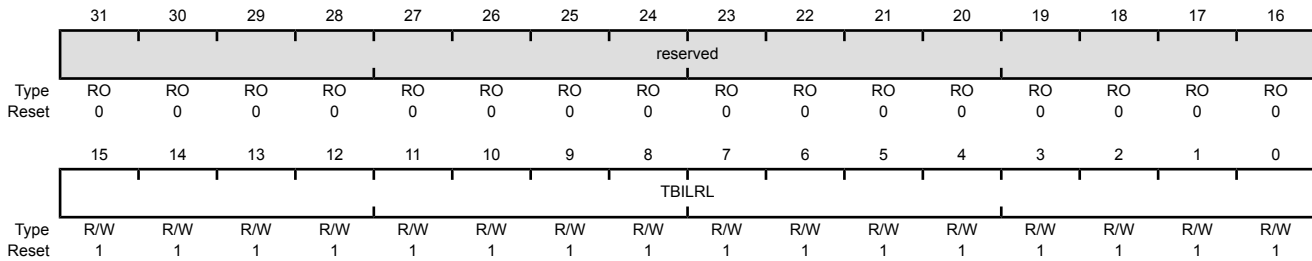
Bit/Field	Name	Type	Reset	Description
31:16	TAILRH	R/W	0xFFFF	GPTM TimerA Interval Load Register High When configured for 32-bit mode via the GPTMCFG register, the GPTM TimerB Interval Load (GPTMTBILR) register loads this value on a write. A read returns the current value of GPTMTBILR . In 16-bit mode, this field reads as 0 and does not have an effect on the state of GPTMTBILR .
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 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
 Timer3 base: 0x4003.3000
 Offset 0x02C
 Type R/W, reset 0x0000.FFFF



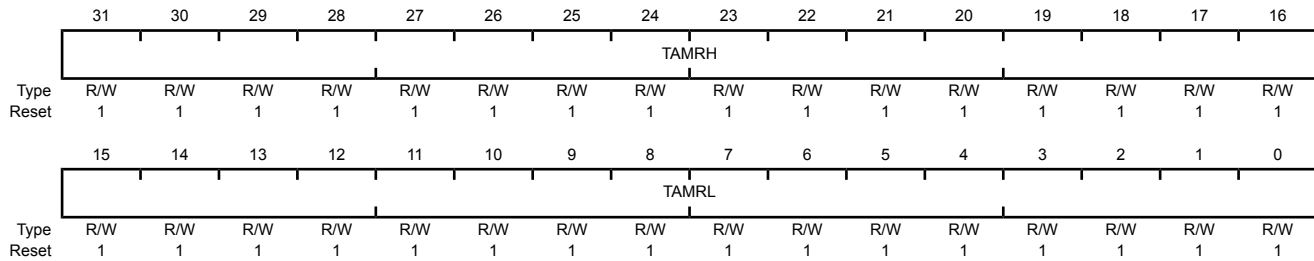
Bit/Field	Name	Type	Reset	Description
31:16	reserved	RO	0x0000	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
15:0	TBILRL	R/W	0xFFFF	GPTM TimerB Interval Load Register 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
 Timer3 base: 0x4003.3000
 Offset 0x030
 Type R/W, reset 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:16	TAMRH	R/W	0xFFFF	<p>GPTM TimerA Match Register High</p> <p>When configured for 32-bit Real-Time Clock (RTC) mode via the GPTMCFG register, this value is compared to the upper half of GPTMTAR, to determine match events.</p> <p>In 16-bit mode, this field reads as 0 and does not have an effect on the state of GPTMTBMATCHR.</p>
15:0	TAMRL	R/W	0xFFFF	<p>GPTM TimerA Match Register Low</p> <p>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.</p> <p>When configured for PWM mode, this value along with GPTMTAILR, determines the duty cycle of the output PWM signal.</p> <p>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.</p>

Register 12: GPTM TimerB Match (GPTMTBMATCHR), offset 0x034

This register is used in 16-bit PWM and Input Edge Count modes.

GPTM TimerB Match (GPTMTBMATCHR)

Timer0 base: 0x4003.0000
 Timer1 base: 0x4003.1000
 Timer2 base: 0x4003.2000
 Timer3 base: 0x4003.3000
 Offset 0x034
 Type R/W, reset 0x0000.FFFF

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	TBMRL															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	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.
15:0	TBMRL	R/W	0xFFFF	GPTM TimerB Match Register Low 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
 Timer3 base: 0x4003.3000
 Offset 0x038
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								TAPSR							
Type	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

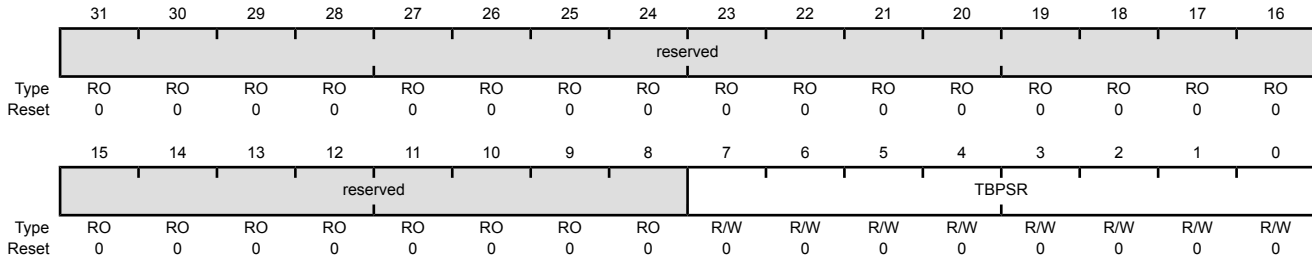
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TAPSR	R/W	0x00	GPTM TimerA Prescale The register loads this value on a write. A read returns the current value of the register. Refer to Table 10-2 on page 217 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
 Timer3 base: 0x4003.3000
 Offset 0x03C
 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	TBPSR	R/W	0x00	GPTM TimerB Prescale The register loads this value on a write. A read returns the current value of this register. Refer to Table 10-2 on page 217 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
 Timer3 base: 0x4003.3000
 Offset 0x040
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								TAPSMR							
Type	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

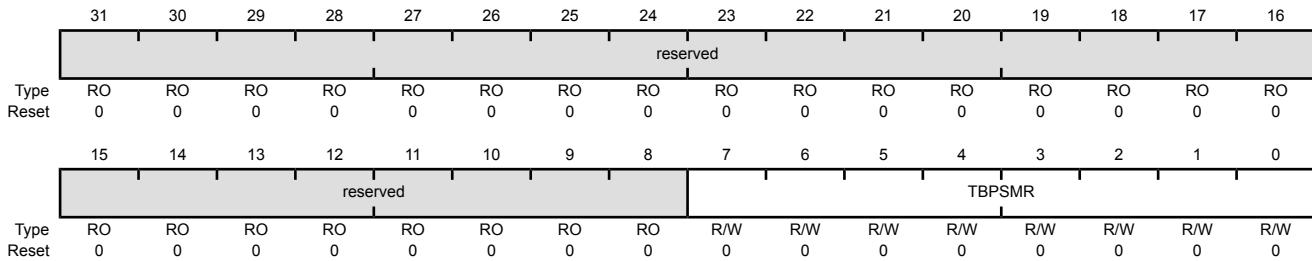
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TAPSMR	R/W	0x00	GPTM TimerA Prescale Match This 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
 Timer3 base: 0x4003.3000
 Offset 0x044
 Type R/W, reset 0x0000.0000



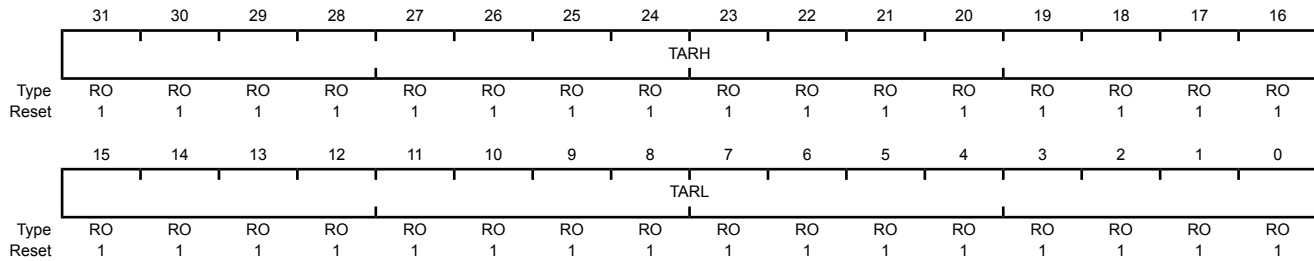
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	TBPSMR	R/W	0x00	GPTM TimerB Prescale Match This value is used alongside GPTMTBMATCHR to detect timer match events while using a prescaler.

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
 Timer3 base: 0x4003.3000
 Offset 0x048
 Type RO, reset 0xFFFF.FFFF



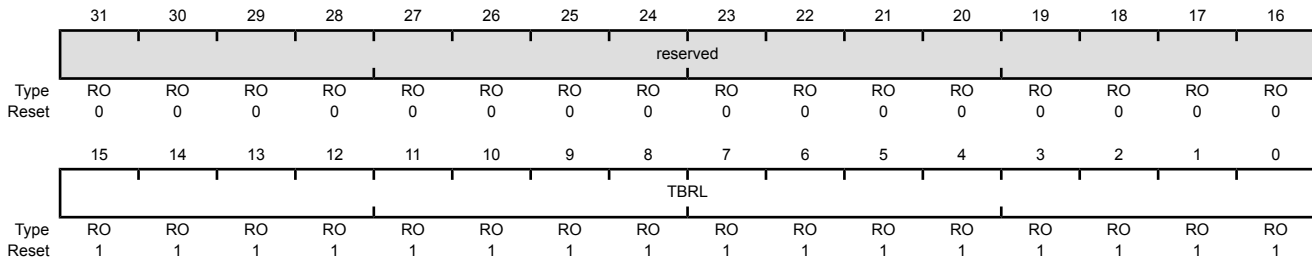
Bit/Field	Name	Type	Reset	Description
31:16	TARH	RO	0xFFFF	GPTM TimerA Register High If the GPTMCFG is in a 32-bit mode, TimerB value is read. If the GPTMCFG is in a 16-bit mode, this is read as zero.
15:0	TARL	RO	0xFFFF	GPTM TimerA Register Low A read returns the current value of the GPTM TimerA Count Register , except in Input Edge Count mode, when it returns the timestamp from the last edge event.

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
 Timer3 base: 0x4003.3000
 Offset 0x04C
 Type RO, reset 0x0000.FFFF



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	TBRL	RO	0xFFFF	GPTM TimerB

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.

11 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.

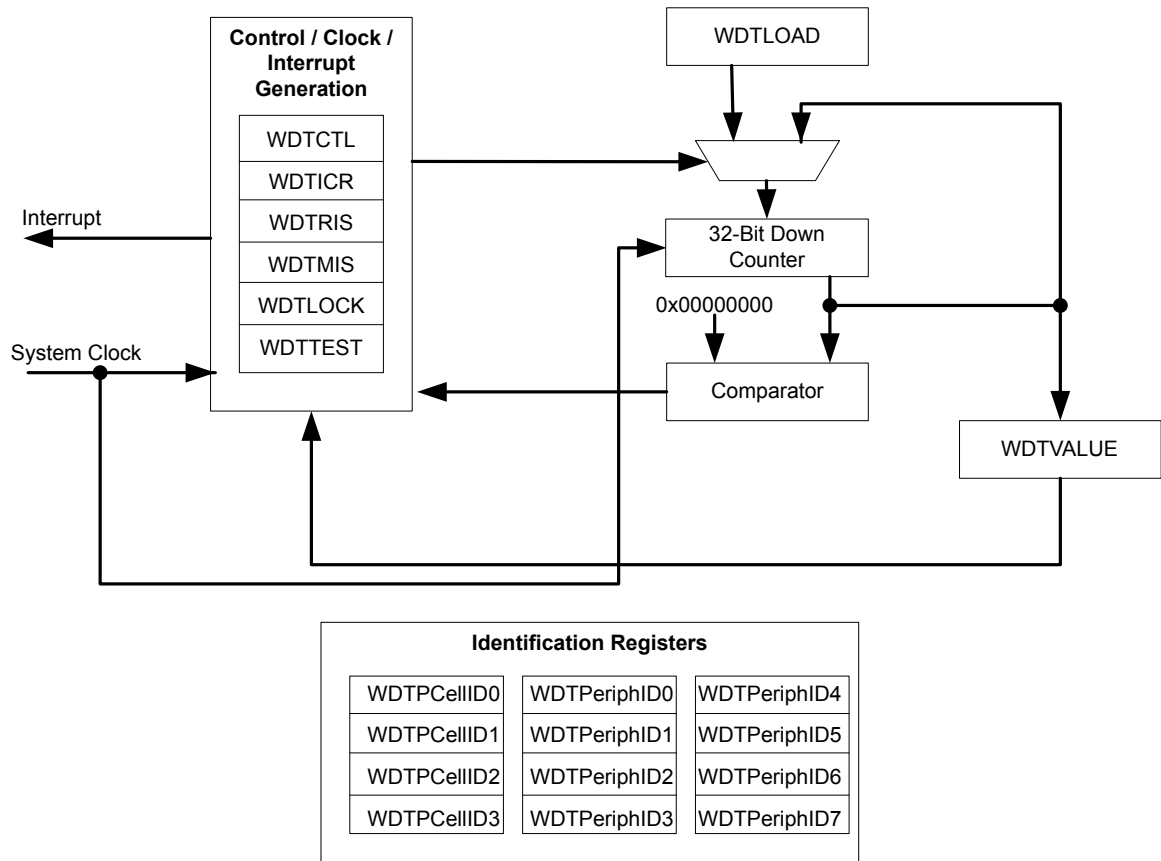
The Stellaris® Watchdog Timer module has the following features:

- 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

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.

11.1 Block Diagram

Figure 11-1. WDT Module Block Diagram



11.2 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.

11.3 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.
2. If the Watchdog is configured to trigger system resets, set the **RESEN** bit in the **WDTCTL** register.
3. 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.

11.4 Register Map

Table 11-1 on page 251 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 11-1. Watchdog Timer Register Map

Offset	Name	Type	Reset	Description	See page
0x000	WDTLOAD	R/W	0xFFFF.FFFF	Watchdog Load	253
0x004	WDTVALUE	RO	0xFFFF.FFFF	Watchdog Value	254
0x008	WDTCTL	R/W	0x0000.0000	Watchdog Control	255
0x00C	WDTICR	WO	-	Watchdog Interrupt Clear	256
0x010	WDTRIS	RO	0x0000.0000	Watchdog Raw Interrupt Status	257
0x014	WDTMIS	RO	0x0000.0000	Watchdog Masked Interrupt Status	258
0x418	WDTTEST	R/W	0x0000.0000	Watchdog Test	259
0xC00	WDTLOCK	R/W	0x0000.0000	Watchdog Lock	260
0xFD0	WDTPeriphID4	RO	0x0000.0000	Watchdog Peripheral Identification 4	261
0xFD4	WDTPeriphID5	RO	0x0000.0000	Watchdog Peripheral Identification 5	262
0xFD8	WDTPeriphID6	RO	0x0000.0000	Watchdog Peripheral Identification 6	263
0xFDC	WDTPeriphID7	RO	0x0000.0000	Watchdog Peripheral Identification 7	264
0xFE0	WDTPeriphID0	RO	0x0000.0005	Watchdog Peripheral Identification 0	265
0xFE4	WDTPeriphID1	RO	0x0000.0018	Watchdog Peripheral Identification 1	266
0xFE8	WDTPeriphID2	RO	0x0000.0018	Watchdog Peripheral Identification 2	267

Table 11-1. Watchdog Timer Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0xFEC	WDTPeriphID3	RO	0x0000.0001	Watchdog Peripheral Identification 3	268
0xFF0	WDTPCellID0	RO	0x0000.000D	Watchdog PrimeCell Identification 0	269
0xFF4	WDTPCellID1	RO	0x0000.00F0	Watchdog PrimeCell Identification 1	270
0xFF8	WDTPCellID2	RO	0x0000.0005	Watchdog PrimeCell Identification 2	271
0xFFC	WDTPCellID3	RO	0x0000.00B1	Watchdog PrimeCell Identification 3	272

11.5 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.FFFF

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

Bit/Field	Name	Type	Reset	Description
31:0	WDTLoad	R/W	0xFFFF.FFFF	Watchdog Load Value

Register 2: Watchdog Value (WDTVALUE), offset 0x004

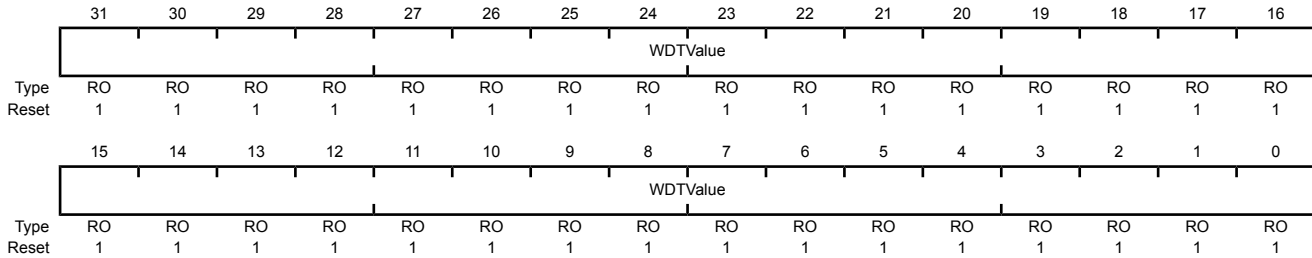
This register contains the current count value of the timer.

Watchdog Value (WDTVALUE)

Base 0x4000.0000

Offset 0x004

Type RO, reset 0xFFFF.FFFF



Bit/Field	Name	Type	Reset	Description
31:0	WDTValue	RO	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved														RESEN	INTEN
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	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						
31:2	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: <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled.</td> </tr> <tr> <td>1</td> <td>Enable the Watchdog module reset output.</td> </tr> </tbody> </table>	Value	Description	0	Disabled.	1	Enable the Watchdog module reset output.
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: <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Interrupt event disabled (once this bit is set, it can only be cleared by a hardware reset).</td> </tr> <tr> <td>1</td> <td>Interrupt event enabled. Once enabled, all writes are ignored.</td> </tr> </tbody> </table>	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.
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

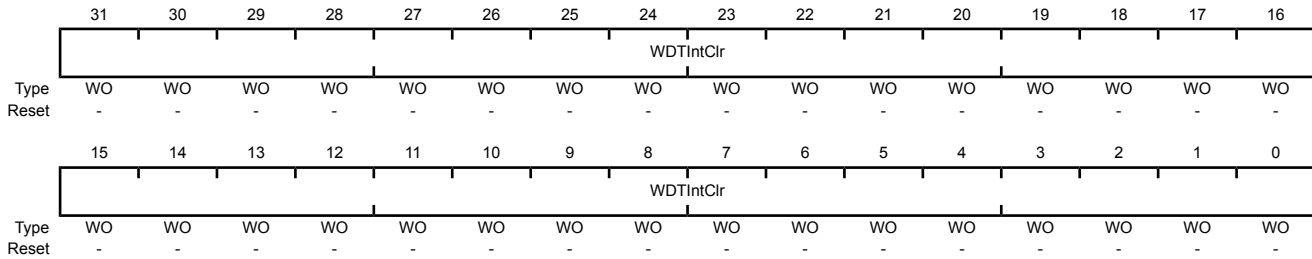
This register is the interrupt clear register. A write of any value to this register clears the Watchdog interrupt and reloads the 32-bit counter from the **WDTLOAD** register. Value for a read or reset is indeterminate.

Watchdog Interrupt Clear (WDTICR)

Base 0x4000.0000

Offset 0x00C

Type WO, reset -



Bit/Field	Name	Type	Reset	Description
31:0	WDTIntClr	WO	-	Watchdog Interrupt Clear

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															WDTRIS
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

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															
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

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	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

This register provides user-enabled stalling when the microcontroller asserts the CPU halt flag during debug.

Watchdog Test (WDTTEST)

Base 0x4000.0000
Offset 0x418
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved							STALL	reserved							
Type	RO	RO	RO	RO	RO	RO	RO	R/W	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

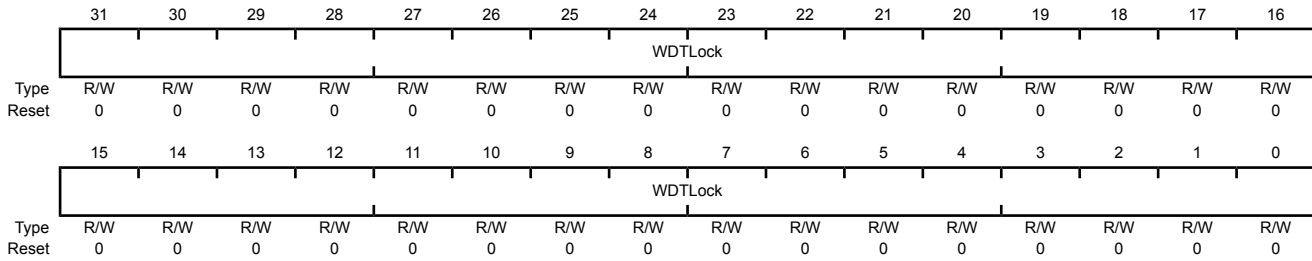
Bit/Field	Name	Type	Reset	Description
31:9	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
8	STALL	R/W	0	Watchdog Stall Enable When set to 1, if the Stellaris® microcontroller is stopped with a debugger, the watchdog timer stops counting. Once the microcontroller is restarted, the watchdog timer resumes counting.
7:0	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 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



Bit/Field	Name	Type	Reset	Description
-----------	------	------	-------	-------------

31:0	WDTLock	R/W	0x0000	Watchdog Lock
------	---------	-----	--------	---------------

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)

Base 0x4000.0000

Offset 0xFD0

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID4							
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

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	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID5							
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

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	WDT Peripheral ID Register[15:8]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID6							
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

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	WDT Peripheral ID Register[23:16]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID7							
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

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	WDT Peripheral ID Register[31:24]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID0							
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	1	0	1

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	0x05	Watchdog Peripheral ID Register[7:0]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID1							
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	1	1	0	0	0

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	0x18	Watchdog Peripheral ID Register[15:8]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID2							
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	1	1	0	0	0

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	Watchdog Peripheral ID Register[23:16]

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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID3							
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	1

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	Watchdog Peripheral ID Register[31:24]

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)

Base 0x4000.0000

Offset 0xFF0

Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID0							
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	1	1	0	1

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	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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID1							
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	1	1	1	1	0	0	0	0

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)

Base 0x4000.0000

Offset 0xFF8

Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID2							
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	1	0	1

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	Watchdog PrimeCell ID Register[23:16]

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)

Base 0x4000.0000
 Offset 0xFFC
 Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID3							
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	1	0	1	1	0	0	0	1

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	Watchdog PrimeCell ID Register[31:24]

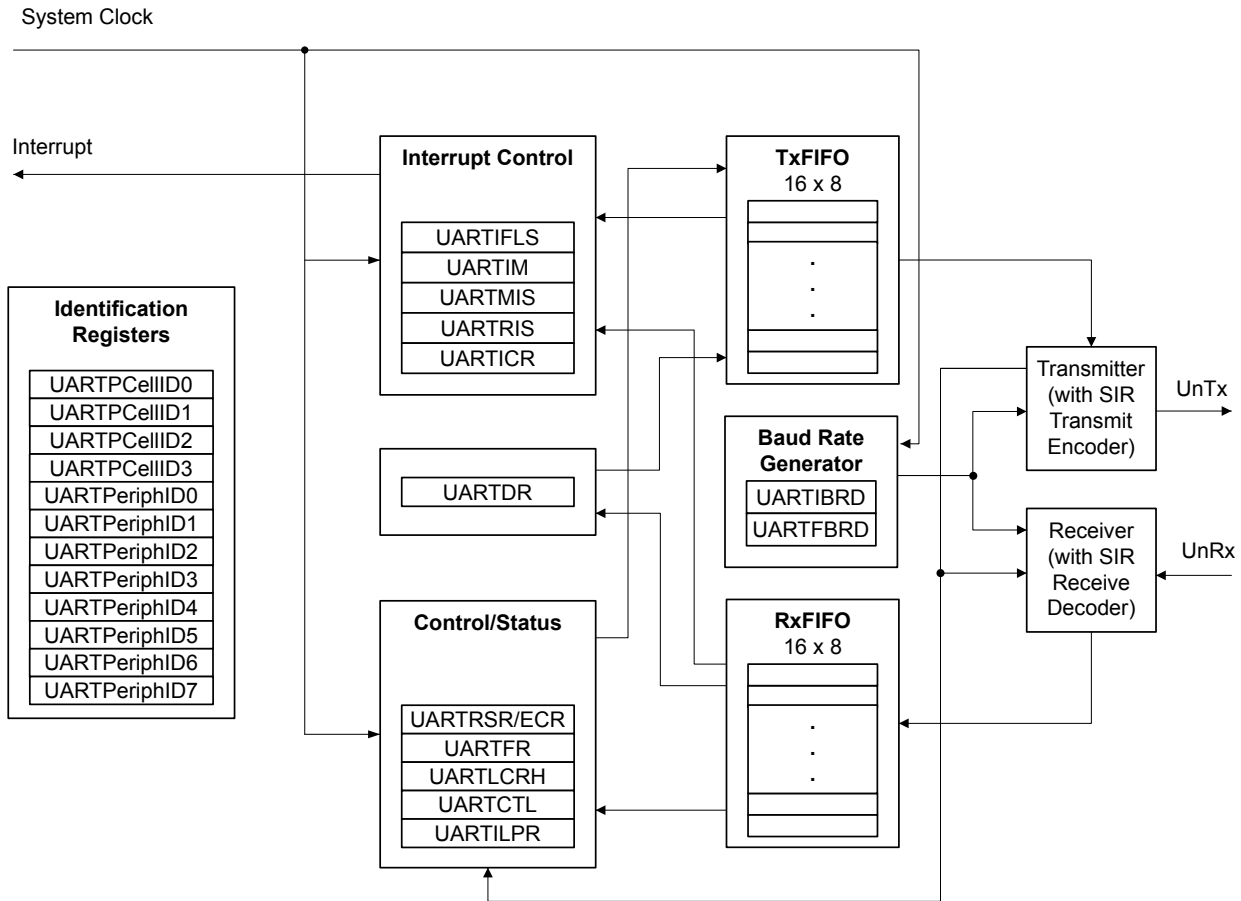
12 Universal Asynchronous Receivers/Transmitters (UARTs)

Each Stellaris® Universal Asynchronous Receiver/Transmitter (UART) has the following features:

- Three fully programmable 16C550-type UARTs with IrDA support
- Separate 16x8 transmit (TX) and receive (RX) FIFOs to reduce CPU interrupt service loading
- Programmable baud-rate generator allowing speeds up to 3.125 Mbps
- 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
- 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 μ s) bit durations
 - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration

12.1 Block Diagram

Figure 12-1. UART Module Block Diagram



12.2 Functional Description

Each Stellaris® 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 292). 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.

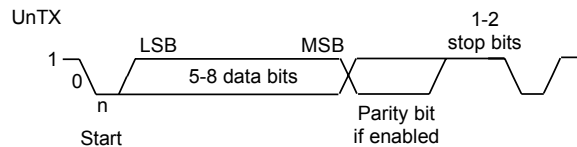
12.2.1 Transmit/Receive Logic

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 12-2 on page 275 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 12-2. UART Character Frame



12.2.2 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 288) and the 6-bit fractional part is loaded with the **UART Fractional Baud-Rate Divisor (UARTFBRD)** register (see page 289). 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.)

$$BRD = BRDI + BRDF = \text{UARTSysClk} / (16 * \text{Baud Rate})$$

where *UARTSysClk* is the system clock connected to the UART.

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:

$$\text{UARTFBRD}[\text{DIVFRAC}] = \text{integer}(\text{BRDF} * 64 + 0.5)$$

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 290), 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

12.2.3 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 285) 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 274).

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 (UARTSR)** register (see page 283). 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.

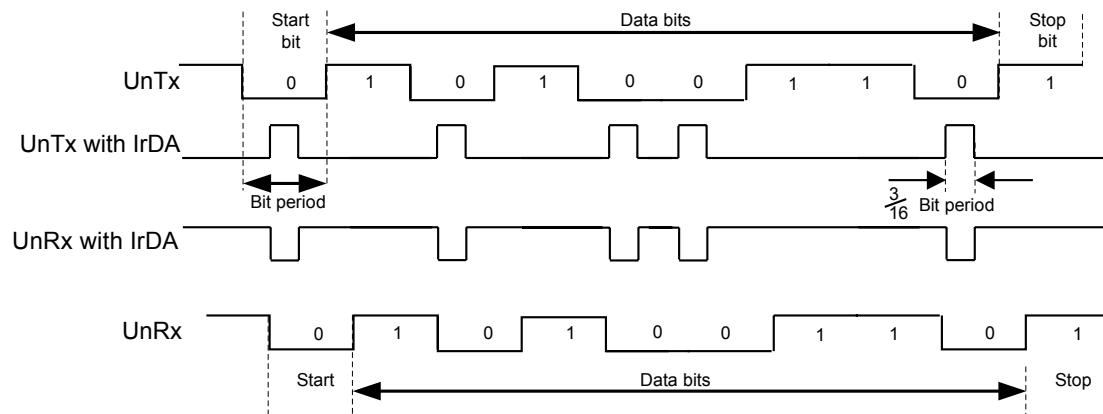
12.2.4 Serial IR (SIR)

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 μ s, assuming a nominal 1.8432 MHz frequency) by changing the appropriate bit in the **UARTCR** register. See page 287 for more information on IrDA low-power pulse-duration configuration.

Figure 12-3 on page 277 shows the UART transmit and receive signals, with and without IrDA modulation.

Figure 12-3. IrDA Data Modulation



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.

If the application does not require the use of the `UnRx` signal, the GPIO pin that has the `UnRx` signal as an alternate function must be configured as the `UnRx` signal and pulled High.

12.2.5 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 281). 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 290).

FIFO status can be monitored via the **UART Flag (UARTFR)** register (see page 285) 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 294). Both FIFOs can be individually configured to trigger interrupts at different levels. Available configurations include 1/8, 1/4, 1/2, 3/4, and 7/8. For example, if the 1/4 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 1/2 mark.

12.2.6 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 299).

The interrupt events that can trigger a controller-level interrupt are defined in the **UART Interrupt Mask (UARTIM)** register (see page 296) 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 298).

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 300).

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.

12.2.7 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 292). In loopback mode, data transmitted on `UnTx` is received on the `UnRx` input.

12.2.8 IrDA SIR block

The IrDA SIR block contains an IrDA serial IR (SIR) protocol encoder/decoder. When enabled, the SIR block uses the `UnTx` and `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.

12.3 Initialization and Configuration

To use the UARTs, the peripheral clock must be enabled by setting the `UART0`, `UART1`, or `UART2` bits in the **RCGC1** register.

This section discusses the steps that are required to use a UART module. For this example, the UART clock is assumed to be 20 MHz and the desired UART configuration is:

- 115200 baud rate

- Data length of 8 bits
- One stop bit
- 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 275, the BRD can be calculated:

$$\text{BRD} = 20,000,000 / (16 * 115,200) = 10.8507$$

which means that the **DIVINT** field of the **UARTIBRD** register (see page 288) should be set to 10. The value to be loaded into the **UARTFBRD** register (see page 289) is calculated by the equation:

$$\text{UARTFBRD}[\text{DIVFRAC}] = \text{integer}(0.8507 * 64 + 0.5) = 54$$

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.
2. Write the integer portion of the BRD to the **UARTIBRD** register.
3. Write the fractional portion of the BRD to the **UARTFBRD** register.
4. 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.

12.4 Register Map

Table 12-1 on page 279 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
- UART1: 0x4000.D000
- UART2: 0x4000.E000

Note: The UART must be disabled (see the **UARTEN** bit in the **UARTCTL** register on page 292) 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 12-1. UART Register Map

Offset	Name	Type	Reset	Description	See page
0x000	UARTDR	R/W	0x0000.0000	UART Data	281
0x004	UARTSR/UARTECR	R/W	0x0000.0000	UART Receive Status/Error Clear	283
0x018	UARTFR	RO	0x0000.0090	UART Flag	285

Table 12-1. UART Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x020	UARTILPR	R/W	0x0000.0000	UART IrDA Low-Power Register	287
0x024	UARTIBRD	R/W	0x0000.0000	UART Integer Baud-Rate Divisor	288
0x028	UARTFBRD	R/W	0x0000.0000	UART Fractional Baud-Rate Divisor	289
0x02C	UARTLCRH	R/W	0x0000.0000	UART Line Control	290
0x030	UARTCTL	R/W	0x0000.0300	UART Control	292
0x034	UARTIFLS	R/W	0x0000.0012	UART Interrupt FIFO Level Select	294
0x038	UARTIM	R/W	0x0000.0000	UART Interrupt Mask	296
0x03C	UARTRIS	RO	0x0000.000F	UART Raw Interrupt Status	298
0x040	UARTMIS	RO	0x0000.0000	UART Masked Interrupt Status	299
0x044	UARTICR	W1C	0x0000.0000	UART Interrupt Clear	300
0xFD0	UARTPeriphID4	RO	0x0000.0000	UART Peripheral Identification 4	302
0xFD4	UARTPeriphID5	RO	0x0000.0000	UART Peripheral Identification 5	303
0xFD8	UARTPeriphID6	RO	0x0000.0000	UART Peripheral Identification 6	304
0xFDC	UARTPeriphID7	RO	0x0000.0000	UART Peripheral Identification 7	305
0xFE0	UARTPeriphID0	RO	0x0000.0011	UART Peripheral Identification 0	306
0xFE4	UARTPeriphID1	RO	0x0000.0000	UART Peripheral Identification 1	307
0xFE8	UARTPeriphID2	RO	0x0000.0018	UART Peripheral Identification 2	308
0xFEC	UARTPeriphID3	RO	0x0000.0001	UART Peripheral Identification 3	309
0xFF0	UARTPCellID0	RO	0x0000.000D	UART PrimeCell Identification 0	310
0xFF4	UARTPCellID1	RO	0x0000.00F0	UART PrimeCell Identification 1	311
0xFF8	UARTPCellID2	RO	0x0000.0005	UART PrimeCell Identification 2	312
0xFFC	UARTPCellID3	RO	0x0000.00B1	UART PrimeCell Identification 3	313

12.5 Register Descriptions

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

Register 1: UART Data (UARTDR), offset 0x000

Important: Use caution when reading this register. Performing a read may change bit status.

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x000
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				OE	BE	PE	FE	DATA							
Type	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	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 0 There has been no data loss due to a FIFO overrun. 1 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 FIFO. The next character is only enabled after the received data input goes to a 1 (marking state) and the next valid start bit is received.

Bit/Field	Name	Type	Reset	Description
9	PE	RO	0	<p>UART Parity Error</p> <p>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.</p> <p>In FIFO mode, this error is associated with the character at the top of the FIFO.</p>
8	FE	RO	0	<p>UART Framing Error</p> <p>This bit is set to 1 when the received character does not have a valid stop bit (a valid stop bit is 1).</p>
7:0	DATA	R/W	0	<p>Data Transmitted or Received</p> <p>When written, the data that is to be transmitted via the UART. When read, the data that was received by the UART.</p>

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.

Reads

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0x004

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													OE	BE	PE	FE
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	

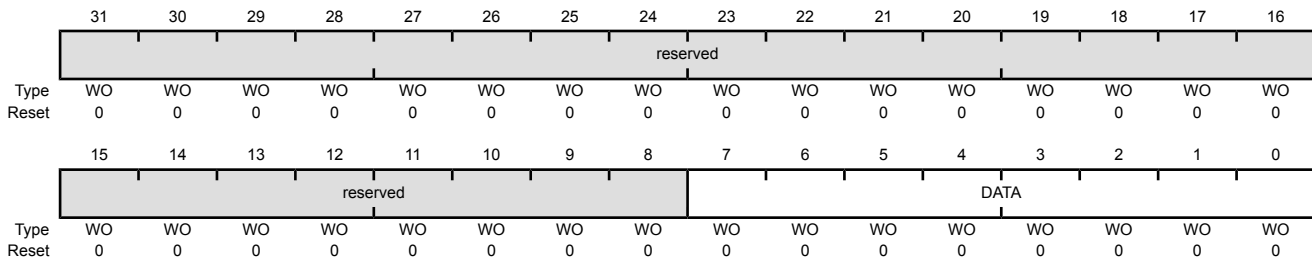
Bit/Field	Name	Type	Reset	Description
31:4	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
3	OE	RO	0	<p>UART Overrun Error</p> <p>When this bit is set to 1, data is received and the FIFO is already full. This bit is cleared to 0 by a write to UARTECR.</p> <p>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.</p>
2	BE	RO	0	<p>UART Break Error</p> <p>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).</p> <p>This bit is cleared to 0 by a write to UARTECR.</p> <p>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.</p>

Bit/Field	Name	Type	Reset	Description
1	PE	RO	0	<p>UART Parity Error</p> <p>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.</p> <p>This bit is cleared to 0 by a write to UARTECR.</p>
0	FE	RO	0	<p>UART Framing Error</p> <p>This bit is set to 1 when the received character does not have a valid stop bit (a valid stop bit is 1).</p> <p>This bit is cleared to 0 by a write to UARTECR.</p> <p>In FIFO mode, this error is associated with the character at the top of the FIFO.</p>

Writes

UART Receive Status/Error Clear (UARTRSR/UARTECR)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x004
 Type WO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	WO	0	<p>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.</p>
7:0	DATA	WO	0	<p>Error Clear</p> <p>A write to this register of any data clears the framing, parity, break, and overrun flags.</p>

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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0x018

Type RO, reset 0x0000.0090

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								TXFE	RXFF	TXFF	RXFE	BUSY	reserved		
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	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	<p>UART Transmit FIFO Empty</p> <p>The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register.</p> <p>If the FIFO is disabled (FEN is 0), this bit is set when the transmit holding register is empty.</p> <p>If the FIFO is enabled (FEN is 1), this bit is set when the transmit FIFO is empty.</p>
6	RXFF	RO	0	<p>UART Receive FIFO Full</p> <p>The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register.</p> <p>If the FIFO is disabled, this bit is set when the receive holding register is full.</p> <p>If the FIFO is enabled, this bit is set when the receive FIFO is full.</p>
5	TXFF	RO	0	<p>UART Transmit FIFO Full</p> <p>The meaning of this bit depends on the state of the FEN bit in the UARTLCRH register.</p> <p>If the FIFO is disabled, this bit is set when the transmit holding register is full.</p> <p>If the FIFO is enabled, this bit is set when the transmit FIFO is full.</p>

Bit/Field	Name	Type	Reset	Description
4	RXFE	RO	1	<p>UART Receive FIFO Empty</p> <p>The meaning of this bit depends on the state of the <code>FEN</code> bit in the UARTLCRH register.</p> <p>If the FIFO is disabled, this bit is set when the receive holding register is empty.</p> <p>If the FIFO is enabled, this bit is set when the receive FIFO is empty.</p>
3	BUSY	RO	0	<p>UART Busy</p> <p>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.</p> <p>This bit is set as soon as the transmit FIFO becomes non-empty (regardless of whether UART is enabled).</p>
2:0	reserved	RO	0	<p>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.</p>

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 derive the low-power SIR pulse width clock by dividing down the system clock (SysClk). All the bits are cleared to 0 when reset.

The internal $F_{IrLPBaud16}$ clock is generated by dividing down SysClk according to the low-power divisor value written to **UARTILPR**. The duration of SIR pulses generated when low-power mode is enabled is three times the period of the $F_{IrLPBaud16}$ clock. The low-power divisor value is calculated as follows:

$$ILPDVSR = \text{SysClk} / F_{IrLPBaud16}$$

where $F_{IrLPBaud16}$ is nominally 1.8432 MHz.

You must choose the divisor so that $1.42 \text{ MHz} < F_{IrLPBaud16} < 2.12 \text{ MHz}$, which results in a low-power pulse duration of 1.41–2.11 μs (three times the period of $F_{IrLPBaud16}$). The minimum frequency of $F_{IrLPBaud16}$ ensures that pulses less than one period of $F_{IrLPBaud16}$ are rejected, but that pulses greater than 1.4 μs are accepted as valid pulses.

Note: Zero is an illegal value. Programming a zero value results in no $F_{IrLPBaud16}$ pulses being generated.

UART IrDA Low-Power Register (UARTILPR)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x020
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								ILPDVSR							
Type	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	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.

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 275 for configuration details.

UART Integer Baud-Rate Divisor (UARTIBRD)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x024
 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:16	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.
15:0	DIVINT	R/W	0x0000	Integer Baud-Rate Divisor

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 275 for configuration details.

UART Fractional Baud-Rate Divisor (UARTFBRD)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x028
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved											DIVFRAC				
Type	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/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.
5:0	DIVFRAC	R/W	0x000	Fractional Baud-Rate Divisor

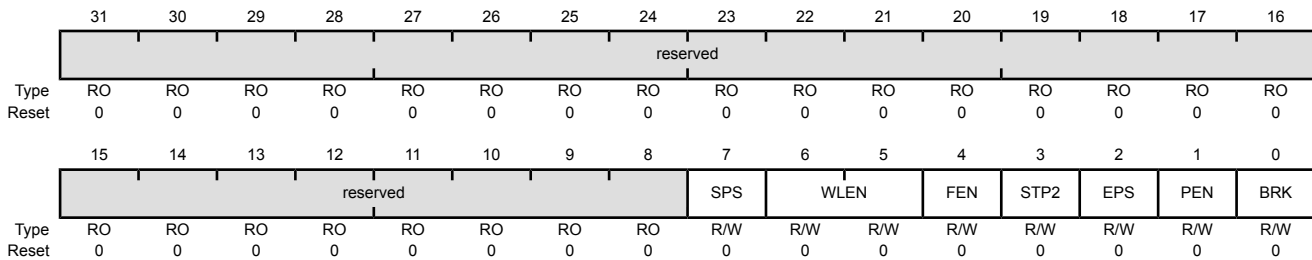
Register 7: UART Line Control (UARTLCRH), offset 0x02C

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x02C
 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	SPS	R/W	0	UART Stick Parity Select When bits 1, 2, and 7 of UARTLCRH 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.

Bit/Field	Name	Type	Reset	Description
3	STP2	R/W	0	<p>UART Two Stop Bits Select</p> <p>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.</p>
2	EPS	R/W	0	<p>UART Even Parity Select</p> <p>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.</p> <p>When cleared to 0, then odd parity is performed, which checks for an odd number of 1s.</p> <p>This bit has no effect when parity is disabled by the <code>PEN</code> bit.</p>
1	PEN	R/W	0	<p>UART Parity Enable</p> <p>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.</p>
0	BRK	R/W	0	<p>UART Send Break</p> <p>If this bit is set to 1, a Low level is continually output on the <code>U_nTX</code> 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.</p>

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.

Note: The **UARTCTL** register should not be changed while the UART is enabled or else the results are unpredictable. The following sequence is recommended for making changes to the **UARTCTL** register.

1. Disable the UART.
2. Wait for the end of transmission or reception of the current character.
3. Flush the transmit FIFO by disabling bit 4 (**FEN**) in the line control register (**UARTLCRH**).
4. Reprogram the control register.
5. Enable the UART.

UART Control (UARTCTL)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x030
 Type R/W, reset 0x0000.0300

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved						RXE	TXE	LBE	reserved				SIRLP	SIREN	UARTEN
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:10	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.
9	RXE	R/W	1	<p>UART Receive Enable</p> <p>If this bit is set to 1, the receive section of the UART is enabled. When the UART is disabled in the middle of a receive, it completes the current character before stopping.</p> <p>Note: To enable reception, the UARTEN bit must also be set.</p>

Bit/Field	Name	Type	Reset	Description
8	TXE	R/W	1	<p>UART Transmit Enable</p> <p>If this bit is set to 1, the transmit section of the UART is enabled. When the UART is disabled in the middle of a transmission, it completes the current character before stopping.</p> <p>Note: To enable transmission, the <code>UARTEN</code> bit must also be set.</p>
7	LBE	R/W	0	<p>UART Loop Back Enable</p> <p>If this bit is set to 1, the <code>UnTX</code> path is fed through the <code>UnRX</code> path.</p>
6:3	reserved	RO	0	<p>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.</p>
2	SIRLP	R/W	0	<p>UART SIR Low Power Mode</p> <p>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 <code>IrLPBaud16</code> input signal, regardless of the selected bit rate. Setting this bit uses less power, but might reduce transmission distances. See page 287 for more information.</p>
1	SIREN	R/W	0	<p>UART SIR Enable</p> <p>If this bit is set to 1, the IrDA SIR block is enabled, and the UART will transmit and receive data using SIR protocol.</p>
0	UARTEN	R/W	0	<p>UART Enable</p> <p>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.</p>

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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 \geq 1/8 full
0x1	RX FIFO \geq 1/4 full
0x2	RX FIFO \geq 1/2 full (default)
0x3	RX FIFO \geq 3/4 full
0x4	RX FIFO \geq 7/8 full

0x5-0x7 Reserved

Bit/Field	Name	Type	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 \leq 1/8 full 0x1 TX FIFO \leq 1/4 full 0x2 TX FIFO \leq 1/2 full (default) 0x3 TX FIFO \leq 3/4 full 0x4 TX FIFO \leq 7/8 full 0x5-0x7 Reserved

Register 10: UART Interrupt Mask (UARTIM), offset 0x038

The **UARTIM** register is the interrupt mask set/clear register.

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.

UART Interrupt Mask (UARTIM)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x038
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved						OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM	reserved		
Type	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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
31:11	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	OEIM	R/W	0	UART Overrun Error Interrupt Mask On a read, the current mask for the OEIM interrupt is returned. Setting this bit to 1 promotes the OEIM interrupt to the interrupt controller.
9	BEIM	R/W	0	UART Break Error Interrupt Mask On a read, the current mask for the BEIM interrupt is returned. Setting this bit to 1 promotes the BEIM interrupt to the interrupt controller.
8	PEIM	R/W	0	UART Parity Error Interrupt Mask On a read, the current mask for the PEIM interrupt is returned. Setting this bit to 1 promotes the PEIM interrupt to the interrupt controller.
7	FEIM	R/W	0	UART Framing Error Interrupt Mask On a read, the current mask for the FEIM interrupt is returned. Setting this bit to 1 promotes the FEIM interrupt to the interrupt controller.
6	RTIM	R/W	0	UART Receive Time-Out Interrupt Mask On a read, the current mask for the RTIM interrupt is returned. Setting this bit to 1 promotes the RTIM interrupt to the interrupt controller.
5	TXIM	R/W	0	UART Transmit Interrupt Mask On a read, the current mask for the TXIM interrupt is returned. Setting this bit to 1 promotes the TXIM interrupt to the interrupt controller.

Bit/Field	Name	Type	Reset	Description
4	RXIM	R/W	0	UART Receive Interrupt Mask On a read, the current mask for the <code>RXIM</code> interrupt is returned. Setting this bit to 1 promotes the <code>RXIM</code> 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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x03C
 Type RO, reset 0x0000.000F

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS	reserved				
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	1	1	1	1

Bit/Field	Name	Type	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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x040
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS	reserved				
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

Bit/Field	Name	Type	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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0x044
 Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved					OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC	reserved			
Type	RO	RO	RO	RO	RO	W1C	W1C	W1C	W1C	W1C	W1C	W1C	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
31:11	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.

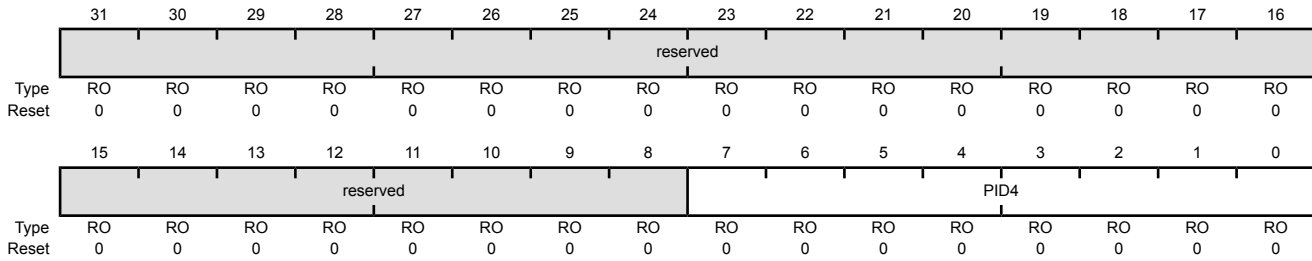
Bit/Field	Name	Type	Reset	Description
7	FEIC	W1C	0	Framing Error Interrupt Clear The FEIC values are defined as follows: Value Description 0 No effect on the interrupt. 1 Clears interrupt.
6	RTIC	W1C	0	Receive Time-Out Interrupt Clear The RTIC values are defined as follows: Value Description 0 No effect on the interrupt. 1 Clears interrupt.
5	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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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] Can be used by software to identify the presence of this peripheral.

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0xFD4
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID5							
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

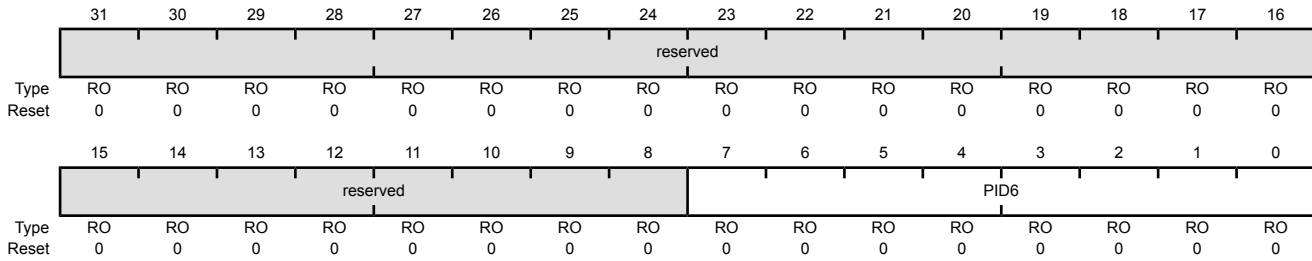
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID5	RO	0x0000	UART Peripheral ID Register[15:8] 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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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] Can be used by software to identify the presence of this peripheral.

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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0xFDC

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID7							
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

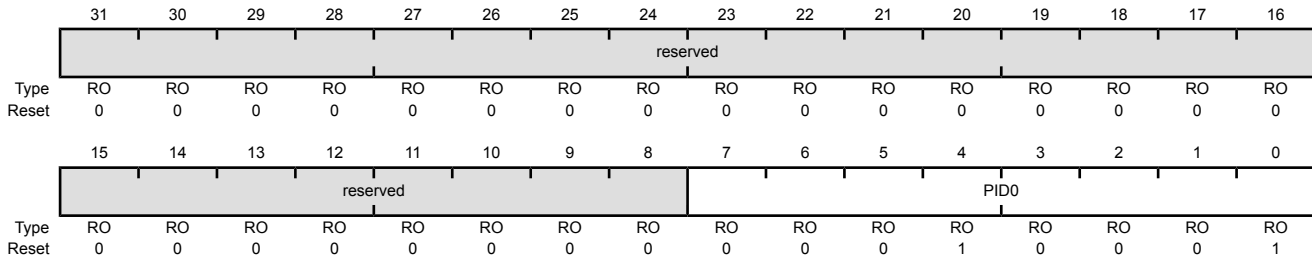
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID7	RO	0x0000	UART Peripheral ID Register[31:24] Can be used by software to identify the presence of this peripheral.

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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] Can be used by software to identify the presence of this peripheral.

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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0xFE4

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID1							
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

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	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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 Offset 0xFE8
 Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID2							
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	1	1	0	0	0

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	UART Peripheral ID Register[23:16] Can be used by software to identify the presence of this peripheral.

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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0xFEC

Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID3							
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	1

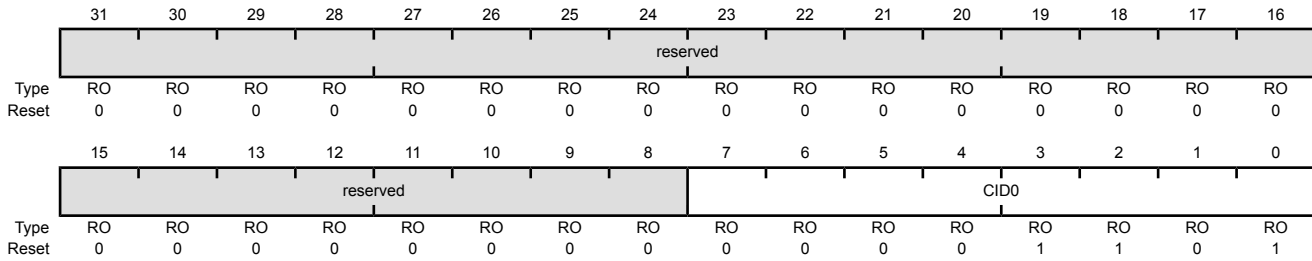
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PID3	RO	0x01	UART Peripheral ID Register[31:24]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

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
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0xFF4

Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID1							
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	1	1	1	1	0	0	0	0

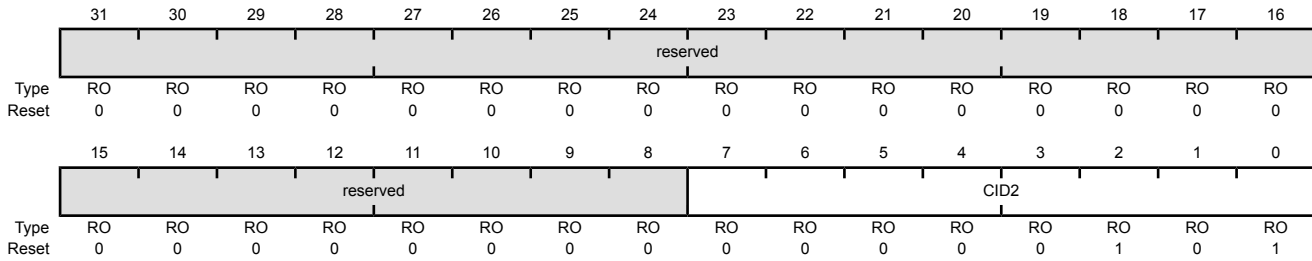
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	CID1	RO	0xF0	UART PrimeCell ID Register[15:8] Provides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

Register 24: UART PrimeCell Identification 2 (UARTPCelIID2), offset 0xFF8

The **UARTPCelIIDn** registers are hard-coded and the fields within the registers determine the reset values.

UART PrimeCell Identification 2 (UARTPCelIID2)

UART0 base: 0x4000.C000
 UART1 base: 0x4000.D000
 UART2 base: 0x4000.E000
 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

UART1 base: 0x4000.D000

UART2 base: 0x4000.E000

Offset 0xFFC

Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID3							
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	1	0	1	1	0	0	0	1

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.

13 Synchronous Serial Interface (SSI)

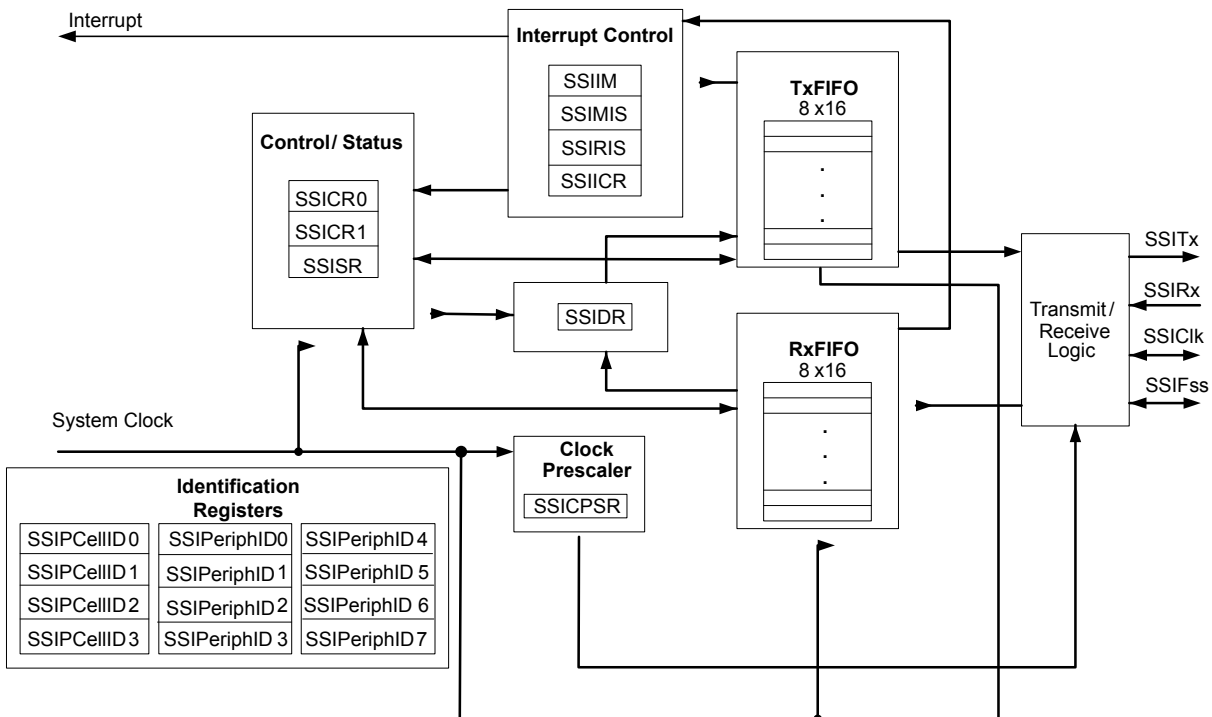
The Stellaris[®] microcontroller includes two Synchronous Serial Interface (SSI) modules. Each SSI is a master or slave interface for synchronous serial communication with peripheral devices that have either Freescale SPI, MICROWIRE, or Texas Instruments synchronous serial interfaces.

Each Stellaris[®] SSI module has the following features:

- Two SSI modules, each with 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

13.1 Block Diagram

Figure 13-1. SSI Module Block Diagram



13.2 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.

13.2.1 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 input clock (FSysClk). 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 333). 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 326).

The frequency of the output clock SSIClk is defined by:

$$SSIClk = F_{SysClk} / (CPSDVSR * (1 + SCR))$$

Note: For master mode, the system clock must be at least two times faster than the SSIClk. For slave mode, the system clock must be at least 12 times faster than the SSIClk.

See “Synchronous Serial Interface (SSI)” on page 494 to view SSI timing parameters.

13.2.2 FIFO Operation

13.2.2.1 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 330), 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.

In slave mode, the SSI transmits data each time it receives data from the master. It is possible that the slave may be required to transmit when the transmit FIFO has no valid data. In this situation, whatever data in the FIFO that the TX FIFO pointer points to is transmitted even though it is not valid. The TX FIFO pointer is not incremented and upon request, the same invalid data is transferred unless valid data has been written to the FIFO.

13.2.2.2 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.

13.2.3 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

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 (SSIM)** register (see page 334). 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 336 and page 337, respectively).

13.2.4 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 (*SSIClk*) is held inactive while the SSI is idle, and *SSIClk* transitions at the programmed frequency only during active transmission or reception of data. The idle state of *SSIClk* is utilized to provide a receive timeout indication that occurs when the receive FIFO still contains data after a timeout period.

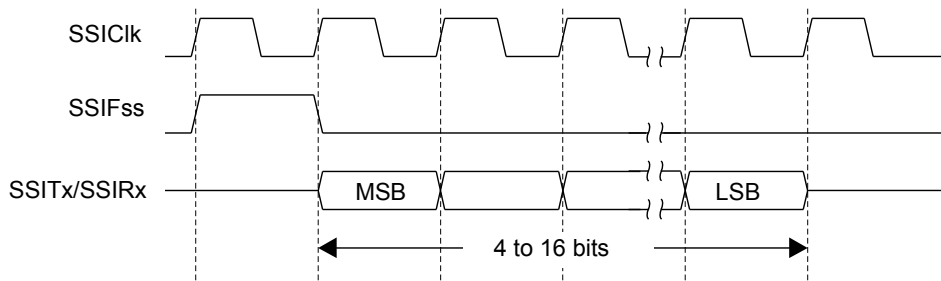
For Freescale SPI and MICROWIRE frame formats, the serial frame (*SSIFSS*) pin is active Low, and is asserted (pulled down) during the entire transmission of the frame.

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 *SSIClk*, 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.

13.2.4.1 Texas Instruments Synchronous Serial Frame Format

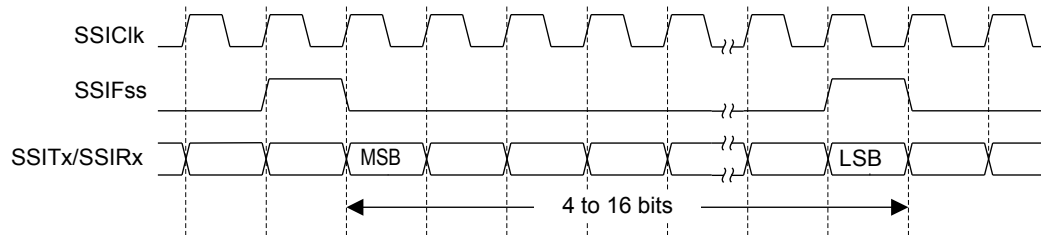
Figure 13-2 on page 317 shows the Texas Instruments synchronous serial frame format for a single transmitted frame.

Figure 13-2. TI Synchronous Serial Frame Format (Single Transfer)

In this mode, *SSIClk* and *SSIFss* are forced Low, and the transmit data line *SSITx* is tristated whenever the SSI is idle. Once the bottom entry of the transmit FIFO contains data, *SSIFss* is pulsed High for one *SSIClk* period. The value to be transmitted is also transferred from the transmit FIFO to the serial shifter of the transmit logic. On the next rising edge of *SSIClk*, the MSB of the 4 to 16-bit data frame is shifted out on the *SSITx* pin. Likewise, the MSB of the received data is shifted onto the *SSIRx* 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 13-3 on page 317 shows the Texas Instruments synchronous serial frame format when back-to-back frames are transmitted.

Figure 13-3. TI Synchronous Serial Frame Format (Continuous Transfer)

13.2.4.2 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.

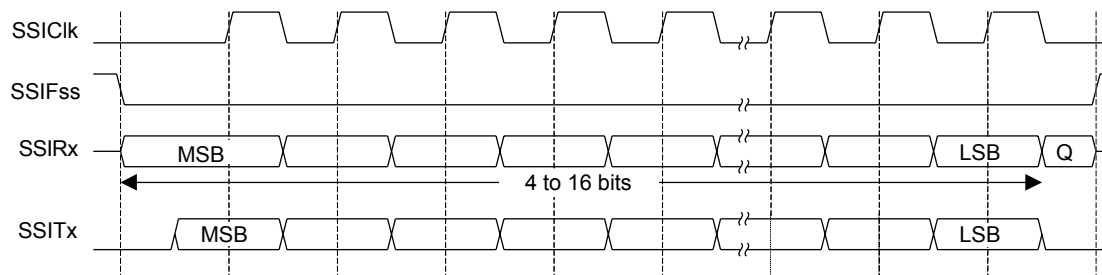
SPH Phase Control Bit

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.

13.2.4.3 Freescale SPI Frame Format with SPO=0 and SPH=0

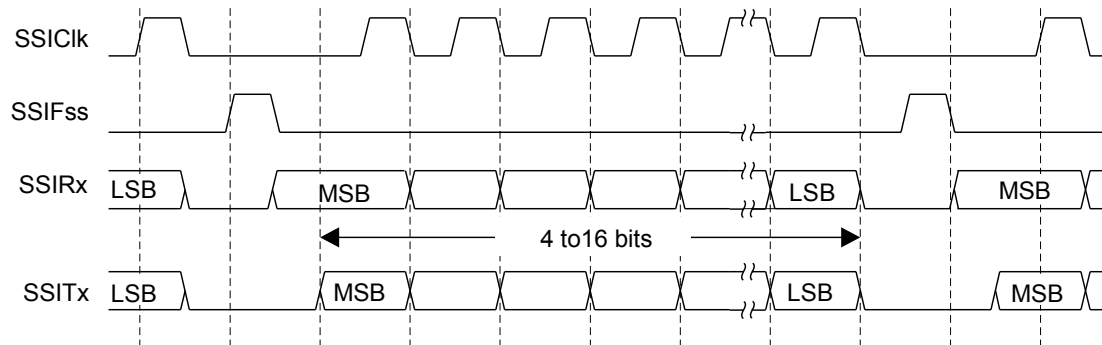
Single and continuous transmission signal sequences for Freescale SPI format with *SPO*=0 and *SPH*=0 are shown in Figure 13-4 on page 318 and Figure 13-5 on page 318.

Figure 13-4. Freescale SPI Format (Single Transfer) with SPO=0 and SPH=0



Note: Q is undefined.

Figure 13-5. Freescale SPI Format (Continuous Transfer) with SPO=0 and SPH=0



In this configuration, during idle periods:

- SSIClk 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 $SSIClk$ period later, valid master data is transferred to the $SSITx$ pin. Now that both the master and slave data have been set, the $SSIClk$ master clock pin goes High after one further half $SSIClk$ period.

The data is now captured on the rising and propagated on the falling edges of the $SSIClk$ signal.

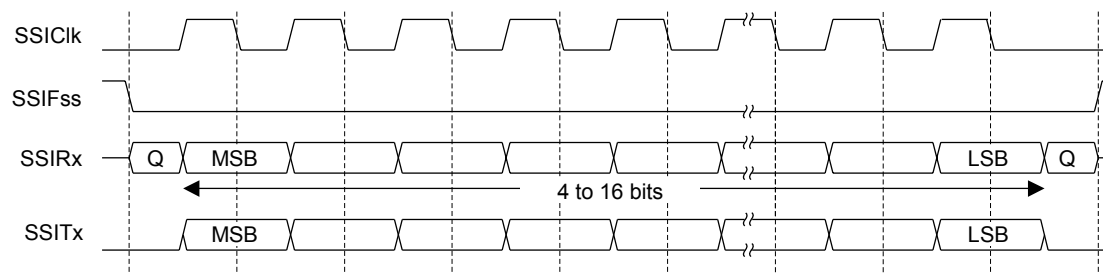
In the case of a single word transmission, after all bits of the data word have been transferred, the $SSIFss$ line is returned to its idle High state one $SSIClk$ period after the last bit has been captured.

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.

13.2.4.4 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 13-6 on page 319, which covers both single and continuous transfers.

Figure 13-6. Freescale SPI Frame Format with $SPO=0$ and $SPH=1$



Note: Q is undefined.

In this configuration, during idle periods:

- $SSIClk$ 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 $SSIClk$ signal.

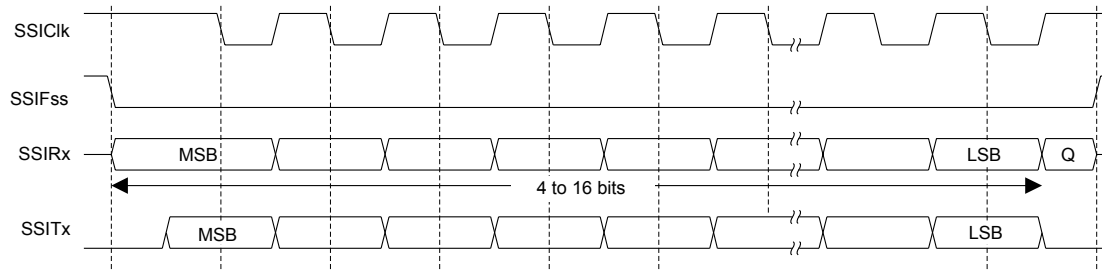
In the case of a single word transfer, after all bits have been transferred, the $SSIFss$ line is returned to its idle High state one $SSIClk$ period after the last bit has been captured.

For continuous back-to-back transfers, the $SSIF_{SS}$ pin is held Low between successive data words and termination is the same as that of the single word transfer.

13.2.4.5 Freescale SPI Frame Format with SPO=1 and SPH=0

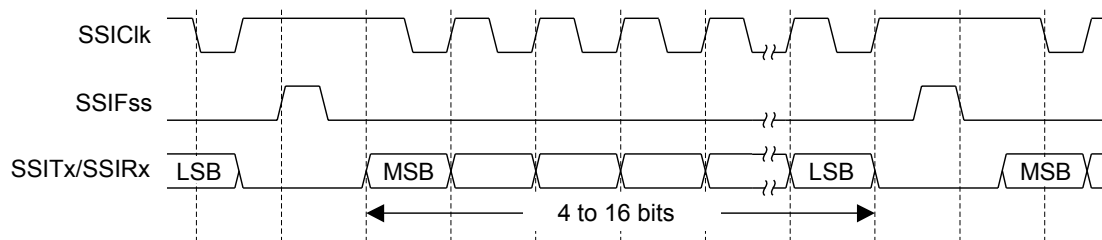
Single and continuous transmission signal sequences for Freescale SPI format with SPO=1 and SPH=0 are shown in Figure 13-7 on page 320 and Figure 13-8 on page 320.

Figure 13-7. Freescale SPI Frame Format (Single Transfer) with SPO=1 and SPH=0



Note: Q is undefined.

Figure 13-8. Freescale SPI Frame Format (Continuous Transfer) with SPO=1 and SPH=0



In this configuration, during idle periods:

- $SSIClk$ is forced High
- $SSIF_{SS}$ 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 $SSIF_{SS}$ 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 $SSITx$ line. Now that both the master and slave data have been set, the $SSIClk$ master clock pin becomes Low after one further half $SSIClk$ period. This means that data is captured on the falling edges and propagated on the rising edges of the $SSIClk$ signal.

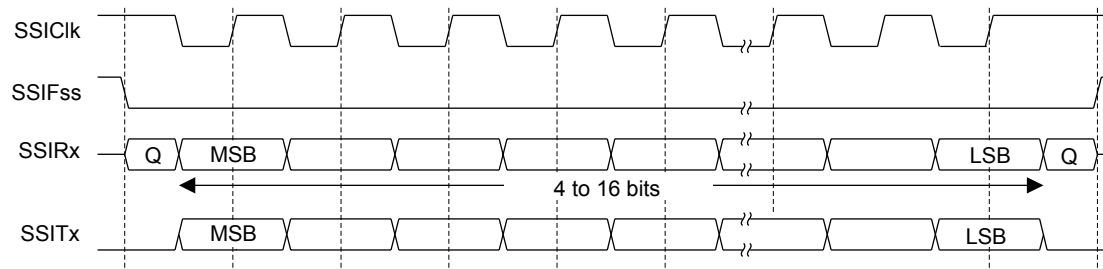
In the case of a single word transmission, after all bits of the data word are transferred, the $SSIF_{SS}$ line is returned to its idle High state one $SSIClk$ period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the $SSIF_{SS}$ 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 $SSIF_{SS}$ pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the $SSIF_{SS}$ pin is returned to its idle state one $SSIClk$ period after the last bit has been captured.

13.2.4.6 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 13-9 on page 321, which covers both single and continuous transfers.

Figure 13-9. Freescale SPI Frame Format with $SPO=1$ and $SPH=1$



Note: Q is undefined.

In this configuration, during idle periods:

- $SSIClk$ is forced High
- $SSIF_{SS}$ 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 $SSIF_{SS}$ 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.

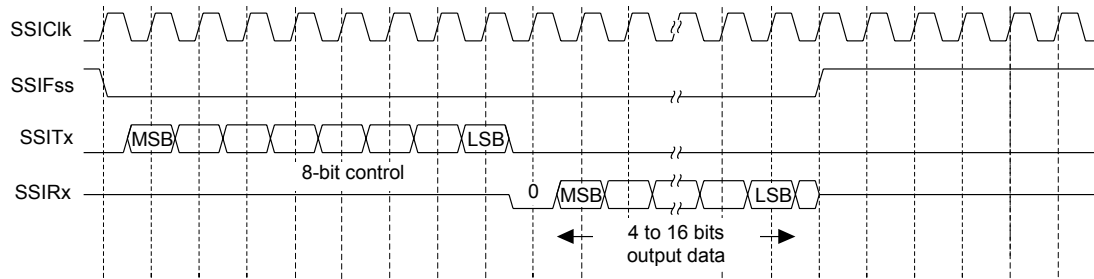
After all bits have been transferred, in the case of a single word transmission, the $SSIF_{SS}$ line is returned to its idle high state one $SSIClk$ period after the last bit has been captured.

For continuous back-to-back transmissions, the $SSIF_{SS}$ 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 $SSIF_{SS}$ pin is held Low between successive data words and termination is the same as that of the single word transfer.

13.2.4.7 MICROWIRE Frame Format

Figure 13-10 on page 322 shows the MICROWIRE frame format, again for a single frame. Figure 13-11 on page 323 shows the same format when back-to-back frames are transmitted.

Figure 13-10. 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:

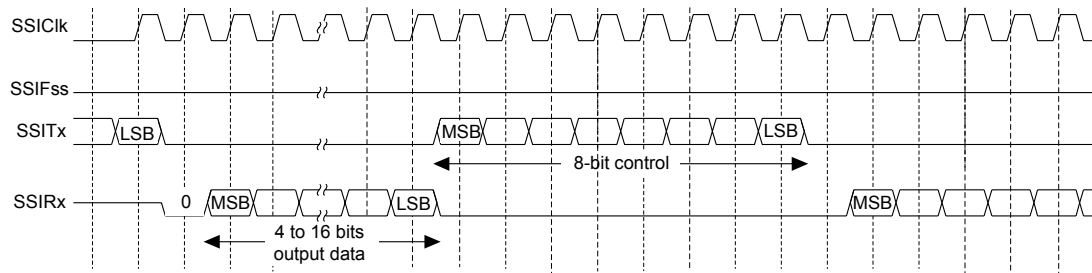
- SSIClk is forced Low
- SSIFss is forced High
- The transmit data line SSITx is arbitrarily forced Low

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 SSITx pin. SSIFss remains Low for the duration of the frame transmission. The SSIRx 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.

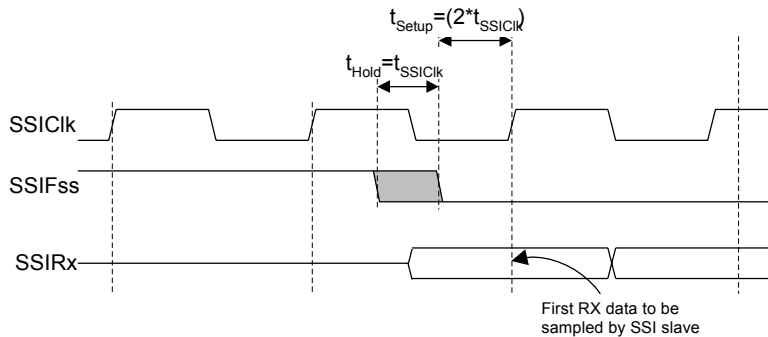
Note: The off-chip slave device can tristate the receive line either on the falling edge of SSIClk after the LSB has been latched by the receive shifter, or when the SSIFss pin goes High.

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.

Figure 13-11. 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 13-12 on page 323 illustrates these setup and hold time requirements. With respect to the `SSIClk` rising edge on which the first bit of receive data is to be sampled by the SSI slave, `SSIFss` must have a setup of at least two times the period of `SSIClk` on which the SSI operates. With respect to the `SSIClk` rising edge previous to this edge, `SSIFss` must have a hold of at least one `SSIClk` period.

Figure 13-12. MICROWIRE Frame Format, SSIFss Input Setup and Hold Requirements

13.3 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.
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:

$$F_{SSIClk} = F_{SysClk} / (CPSDVSR * (1 + SCR))$$

$$1 \times 10^6 = 20 \times 10^6 / (CPSDVSR * (1 + SCR))$$

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.
4. 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.

13.4 Register Map

Table 13-1 on page 324 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
- SSI1: 0x4000.9000

Note: The SSI must be disabled (see the SSE bit in the **SSICR1** register) before any of the control registers are reprogrammed.

Table 13-1. SSI Register Map

Offset	Name	Type	Reset	Description	See page
0x000	SSICR0	R/W	0x0000.0000	SSI Control 0	326

Table 13-1. SSI Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x004	SSICR1	R/W	0x0000.0000	SSI Control 1	328
0x008	SSIDR	R/W	0x0000.0000	SSI Data	330
0x00C	SSISR	RO	0x0000.0003	SSI Status	331
0x010	SSICPSR	R/W	0x0000.0000	SSI Clock Prescale	333
0x014	SSIIM	R/W	0x0000.0000	SSI Interrupt Mask	334
0x018	SSIRIS	RO	0x0000.0008	SSI Raw Interrupt Status	336
0x01C	SSIMIS	RO	0x0000.0000	SSI Masked Interrupt Status	337
0x020	SSIICR	W1C	0x0000.0000	SSI Interrupt Clear	338
0xFD0	SSIPeriphID4	RO	0x0000.0000	SSI Peripheral Identification 4	339
0xFD4	SSIPeriphID5	RO	0x0000.0000	SSI Peripheral Identification 5	340
0xFD8	SSIPeriphID6	RO	0x0000.0000	SSI Peripheral Identification 6	341
0xFDC	SSIPeriphID7	RO	0x0000.0000	SSI Peripheral Identification 7	342
0xFE0	SSIPeriphID0	RO	0x0000.0022	SSI Peripheral Identification 0	343
0xFE4	SSIPeriphID1	RO	0x0000.0000	SSI Peripheral Identification 1	344
0xFE8	SSIPeriphID2	RO	0x0000.0018	SSI Peripheral Identification 2	345
0xFEC	SSIPeriphID3	RO	0x0000.0001	SSI Peripheral Identification 3	346
0xFF0	SSIPCellIID0	RO	0x0000.000D	SSI PrimeCell Identification 0	347
0xFF4	SSIPCellIID1	RO	0x0000.00F0	SSI PrimeCell Identification 1	348
0xFF8	SSIPCellIID2	RO	0x0000.0005	SSI PrimeCell Identification 2	349
0xFFC	SSIPCellIID3	RO	0x0000.00B1	SSI PrimeCell Identification 3	350

13.5 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
 SSI1 base: 0x4000.9000
 Offset 0x000
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	SCR								SPH	SPO	FRF		DSS			
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	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.
15:8	SCR	R/W	0x0000	SSI Serial Clock Rate The value <i>SCR</i> is used to generate the transmit and receive bit rate of the SSI. The bit rate is: $BR = FSSIClk / (CPSDVSr * (1 + SCR))$ where <i>CPSDVSr</i> is an even value from 2-254 programmed in the SSICPSR register, and <i>SCR</i> is a value from 0-255.
7	SPH	R/W	0	SSI Serial Clock Phase This bit is only applicable to the Freescale SPI Format. The <i>SPH</i> 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 <i>SPH</i> bit is 0, data is captured on the first clock edge transition. If <i>SPH</i> is 1, data is captured on the second clock edge transition.
6	SPO	R/W	0	SSI Serial Clock Polarity This bit is only applicable to the Freescale SPI Format. When the <i>SPO</i> bit is 0, it produces a steady state Low value on the <i>SSIClk</i> pin. If <i>SPO</i> is 1, a steady state High value is placed on the <i>SSIClk</i> pin when data is not being transferred.

Bit/Field	Name	Type	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 Instruments Synchronous Serial Frame Format 0x2 MICROWIRE Frame Format 0x3 Reserved
3:0	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
 SSI1 base: 0x4000.9000
 Offset 0x004
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved												SOD	MS	SSE	LBM	
Type	RO	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	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	SOD	R/W	0	<p>SSI Slave Mode Output Disable</p> <p>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.</p>
---	-----	-----	---	---

The SOD values are defined as follows:

Value	Description
0	SSI can drive SSITx output in Slave Output mode.
1	SSI must not drive the SSITx output in Slave mode.

2	MS	R/W	0	<p>SSI Master/Slave Select</p> <p>This bit selects Master or Slave mode and can be modified only when SSI is disabled ($SSE=0$).</p>
---	----	-----	---	---

The MS values are defined as follows:

Value	Description
0	Device configured as a master.
1	Device configured as a slave.

Bit/Field	Name	Type	Reset	Description						
1	SSE	R/W	0	<p>SSI Synchronous Serial Port Enable</p> <p>Setting this bit enables SSI operation.</p> <p>The <code>SSE</code> values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>SSI operation disabled.</td> </tr> <tr> <td>1</td> <td>SSI operation enabled.</td> </tr> </tbody> </table> <p>Note: This bit must be set to 0 before any control registers are reprogrammed.</p>	Value	Description	0	SSI operation disabled.	1	SSI operation enabled.
Value	Description									
0	SSI operation disabled.									
1	SSI operation enabled.									
0	LBM	R/W	0	<p>SSI Loopback Mode</p> <p>Setting this bit enables Loopback Test mode.</p> <p>The <code>LBM</code> values are defined as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Normal serial port operation enabled.</td> </tr> <tr> <td>1</td> <td>Output of the transmit serial shift register is connected internally to the input of the receive serial shift register.</td> </tr> </tbody> </table>	Value	Description	0	Normal serial port operation enabled.	1	Output of the transmit serial shift register is connected internally to the input of the receive serial shift register.
Value	Description									
0	Normal serial port operation enabled.									
1	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

Important: Use caution when reading this register. Performing a read may change bit status.

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).

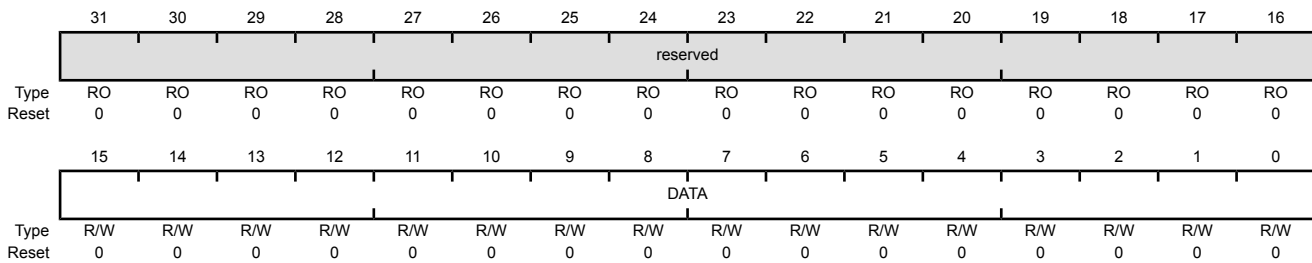
When **SSIDR** is written to, the entry in the transmit FIFO (pointed to by the write pointer) is written to. Data values are removed from the transmit FIFO one value at a time by the transmit logic. It is loaded into the transmit serial shifter, then serially shifted out onto the **SSITx** pin at the 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)

SSI0 base: 0x4000.8000
 SSI1 base: 0x4000.9000
 Offset 0x008
 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	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.

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

SSI1 base: 0x4000.9000

Offset 0x00C

Type RO, reset 0x0000.0003

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved												BSY	RFF	RNE	TNF	TFE
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	1	1	

Bit/Field	Name	Type	Reset	Description
31:5	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.
4	BSY	RO	0	SSI Busy Bit The BSY values are defined as follows: Value Description 0 SSI is idle. 1 SSI is currently transmitting and/or receiving a frame, or the transmit FIFO is not empty.
3	RFF	RO	0	SSI Receive FIFO Full The RFF values are defined as follows: Value Description 0 Receive FIFO is not full. 1 Receive FIFO is full.
2	RNE	RO	0	SSI Receive FIFO Not Empty The RNE values are defined as follows: Value Description 0 Receive FIFO is empty. 1 Receive FIFO is not empty.

Bit/Field	Name	Type	Reset	Description
1	TNF	RO	1	SSI Transmit FIFO Not Full The TNF values are defined as follows: Value Description 0 Transmit FIFO is full. 1 Transmit FIFO is not full.
0	TFE	RO	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
 SSI1 base: 0x4000.9000
 Offset 0x010
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CPSDVSR							
Type	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	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 <code>SSIClk</code> . 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
 SSI1 base: 0x4000.9000
 Offset 0x014
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													TXIM	RXIM	RTIM	RORIM
Type	RO	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	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	TXIM	R/W	0	SSI Transmit FIFO Interrupt Mask The TXIM values are defined as follows: Value Description 0 TX FIFO half-full or less condition interrupt is masked. 1 TX FIFO half-full or less condition interrupt is not masked.
2	RXIM	R/W	0	SSI Receive FIFO Interrupt Mask The RXIM values are defined as follows: Value Description 0 RX FIFO half-full or more condition interrupt is masked. 1 RX FIFO half-full or more condition interrupt is not masked.
1	RTIM	R/W	0	SSI Receive Time-Out Interrupt Mask The RTIM values are defined as follows: Value Description 0 RX FIFO time-out interrupt is masked. 1 RX FIFO time-out interrupt is not masked.

Bit/Field	Name	Type	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.

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
 SSI1 base: 0x4000.9000
 Offset 0x018
 Type RO, reset 0x0000.0008

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													TXRIS	RXRIS	RTRIS	RORRIS
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	1	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	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
 SSI1 base: 0x4000.9000
 Offset 0x01C
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													TXMIS	RXMIS	RTMIS	RORMIS
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	

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
 SSI1 base: 0x4000.9000
 Offset 0x020
 Type W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved														RTIC	RORIC	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	W1C	W1C
Reset	0	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	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 The RORIC values are defined as follows: Value Description 0 No effect on interrupt. 1 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

SSI1 base: 0x4000.9000

Offset 0xFD0

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID4							
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

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.

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
 SSI1 base: 0x4000.9000
 Offset 0xFD4
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID5							
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

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

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

SSI1 base: 0x4000.9000

Offset 0xFD8

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID6							
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

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

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
 SSI1 base: 0x4000.9000
 Offset 0xFDC
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID7							
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

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

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

SSI1 base: 0x4000.9000

Offset 0xFE0

Type RO, reset 0x0000.0022

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID0							
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	1	0	0	0	1	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.
7:0	PID0	RO	0x22	SSI Peripheral ID Register[7:0] Can be used by software to identify the presence of this peripheral.

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
 SSI1 base: 0x4000.9000
 Offset 0xFE4
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID1							
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

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

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
 SSI1 base: 0x4000.9000
 Offset 0xFE8
 Type RO, reset 0x0000.0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID2							
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	1	1	0	0

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	SSI Peripheral ID Register [23:16] Can be used by software to identify the presence of this peripheral.

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
 SSI1 base: 0x4000.9000
 Offset 0xFEC
 Type RO, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								PID3							
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	1

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

Register 18: SSI PrimeCell Identification 0 (SSIPCellID0), offset 0xFF0

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 0 (SSIPCellID0)

SSI0 base: 0x4000.8000
 SSI1 base: 0x4000.9000
 Offset 0xFF0
 Type RO, reset 0x0000.000D

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID0							
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	1	1	0	1

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 **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 1 (SSIPCellID1)

SSI0 base: 0x4000.8000
 SSI1 base: 0x4000.9000
 Offset 0xFF4
 Type RO, reset 0x0000.00F0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID1							
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	1	1	1	1	0	0	0	0

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 (SSIPCellID2)

SSI0 base: 0x4000.8000

SSI1 base: 0x4000.9000

Offset 0xFF8

Type RO, reset 0x0000.0005

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID2							
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	1	0	1

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	SSI PrimeCell ID Register [23:16] Provides software a standard cross-peripheral identification system.

Register 21: SSI PrimeCell Identification 3 (SSIPCellID3), offset 0xFFC

The **SSIPCellIDn** registers are hard-coded, and the fields within the register determine the reset value.

SSI PrimeCell Identification 3 (SSIPCellID3)

SSI0 base: 0x4000.8000
 SSI1 base: 0x4000.9000
 Offset 0xFFC
 Type RO, reset 0x0000.00B1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								CID3							
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	1	0	1	1	0	0	0	1

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.

14 Inter-Integrated Circuit (I²C) Interface

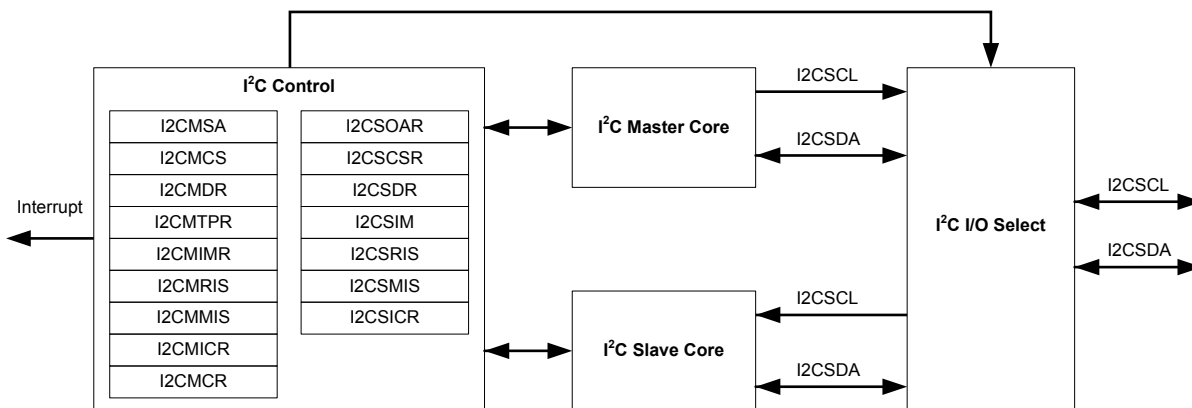
The Inter-Integrated Circuit (I²C) bus provides bi-directional data transfer through a two-wire design (a serial data line SDA and a serial clock line SCL), and interfaces to external I²C devices such as serial memory (RAMs and ROMs), networking devices, LCDs, tone generators, and so on. The I²C bus may also be used for system testing and diagnostic purposes in product development and manufacture. The LM3S1960 microcontroller includes two I²C modules, providing the ability to interact (both send and receive) with other I²C devices on the bus.

The Stellaris® I²C interface has the following features:

- Two I²C modules, each with the following features:
 - Devices on the I²C bus can be designated as either a master or a slave
 - Supports both sending and receiving data as either a master or a slave
 - Supports simultaneous master and slave operation
 - Four I²C modes
 - Master transmit
 - Master receive
 - Slave transmit
 - Slave receive
 - Two transmission speeds: Standard (100 Kbps) and Fast (400 Kbps)
 - Master and slave interrupt generation
 - Master generates interrupts when a transmit or receive operation completes (or aborts due to an error)
 - Slave generates interrupts when data has been sent or requested by a master
 - Master with arbitration and clock synchronization, multimaster support, and 7-bit addressing mode

14.1 Block Diagram

Figure 14-1. I²C Block Diagram

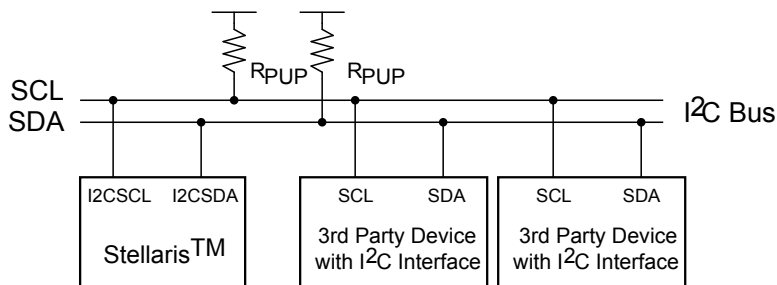


14.2 Functional Description

Each I²C 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 I²C bus configuration is shown in Figure 14-2 on page 352.

See “Inter-Integrated Circuit (I²C) Interface” on page 495 for I²C timing diagrams.

Figure 14-2. I²C Bus Configuration



14.2.1 I²C Bus Functional Overview

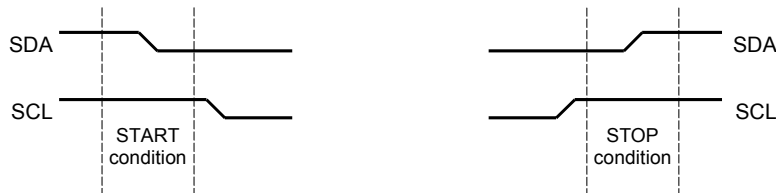
The I²C bus uses only two signals: SDA and SCL, named I2CSDA and I2CSCL on Stellaris[®] microcontrollers. SDA is the bi-directional serial data line and SCL is the bi-directional serial clock line. The bus is considered idle when both lines are High.

Every transaction on the I²C 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 353) 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.

14.2.1.1 START and STOP Conditions

The protocol of the I²C 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 14-3 on page 353.

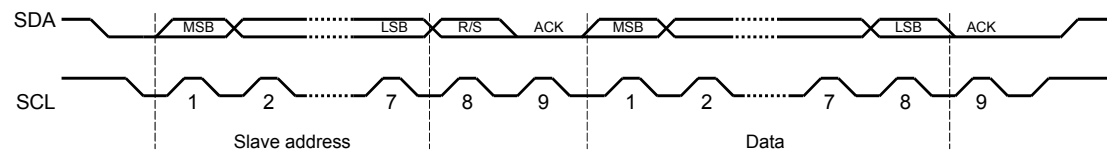
Figure 14-3. START and STOP Conditions



14.2.1.2 Data Format with 7-Bit Address

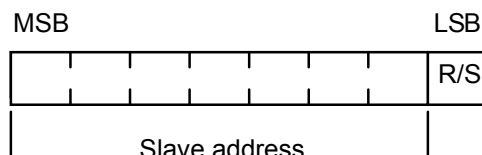
Data transfers follow the format shown in Figure 14-4 on page 353. After the START condition, a slave address is sent. This address is 7-bits long followed by an eighth bit, which is a data direction bit (R/S bit in the I2CMSA register). A zero indicates a transmit operation (send), and a one indicates 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.

Figure 14-4. Complete Data Transfer with a 7-Bit Address



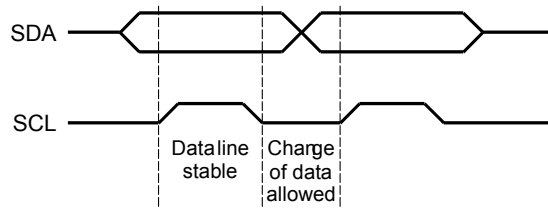
The first seven bits of the first byte make up the slave address (see Figure 14-5 on page 353). The eighth bit determines the direction of the message. A zero in the R/S position of the first byte means that the master will write (send) data to the selected slave, and a one in this position means that the master will receive data from the slave.

Figure 14-5. R/S Bit in First Byte



14.2.1.3 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 14-6 on page 354).

Figure 14-6. Data Validity During Bit Transfer on the I²C Bus

14.2.1.4 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 353.

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.

14.2.1.5 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.

14.2.2 Available Speed Modes

The I²C 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 **I²C Master Timer Period (I2CMTPR)** register (see page 372).

The I²C clock period is calculated as follows:

$$\text{SCL_PERIOD} = 2 * (1 + \text{TIMER_PRD}) * (\text{SCL_LP} + \text{SCL_HP}) * \text{CLK_PRD}$$

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 14-1 on page 355 gives examples of timer period, system clock, and speed mode (Standard or Fast).

Table 14-1. Examples of I²C 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
33 MHz	0x10	97.1 Kbps	0x04	330 Kbps
40 MHz	0x13	100 Kbps	0x04	400 Kbps
50 MHz	0x18	100 Kbps	0x06	357 Kbps

14.2.3 Interrupts

The I²C 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 I²C master and I²C slave modules. While both modules can generate interrupts for multiple conditions, only a single interrupt signal is sent to the interrupt controller.

14.2.3.1 I²C Master Interrupts

The I²C master module generates an interrupt when a transaction completes (either transmit or receive), or when an error occurs during a transaction. To enable the I²C master interrupt, software must write a '1' to the **I²C Master Interrupt Mask (I2CMIMR)** register. When an interrupt condition is met, software must check the **ERROR** bit in the **I²C 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 **I²C Master Interrupt Clear (I2CMICR)** register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the **I²C Master Raw Interrupt Status (I2CMRIS)** register.

14.2.3.2 I²C Slave Interrupts

The slave module can generate an interrupt when data has been received or requested. This interrupt is enabled by writing a 1 to the `DATAIM` bit in the **I²C Slave Interrupt Mask (I2CSIMR)** register. Software determines whether the module should write (transmit) or read (receive) data from the **I²C Slave Data (I2CSDR)** register, by checking the `RREQ` and `TREQ` bits of the **I²C 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 `DATAIC` bit in the **I²C Slave Interrupt Clear (I2CSICR)** register.

If the application doesn't require the use of interrupts, the raw interrupt status is always visible via the **I²C Slave Raw Interrupt Status (I2CSRIS)** register.

14.2.4 Loopback Operation

The I²C modules can be placed into an internal loopback mode for diagnostic or debug work. This is accomplished by setting the `LPBK` bit in the **I²C Master Configuration (I2CMCR)** register. In loopback mode, the SDA and SCL signals from the master and slave modules are tied together.

14.2.5 Command Sequence Flow Charts

This section details the steps required to perform the various I²C transfer types in both master and slave mode.

14.2.5.1 I²C Master Command Sequences

The figures that follow show the command sequences available for the I²C master.

Figure 14-7. Master Single SEND

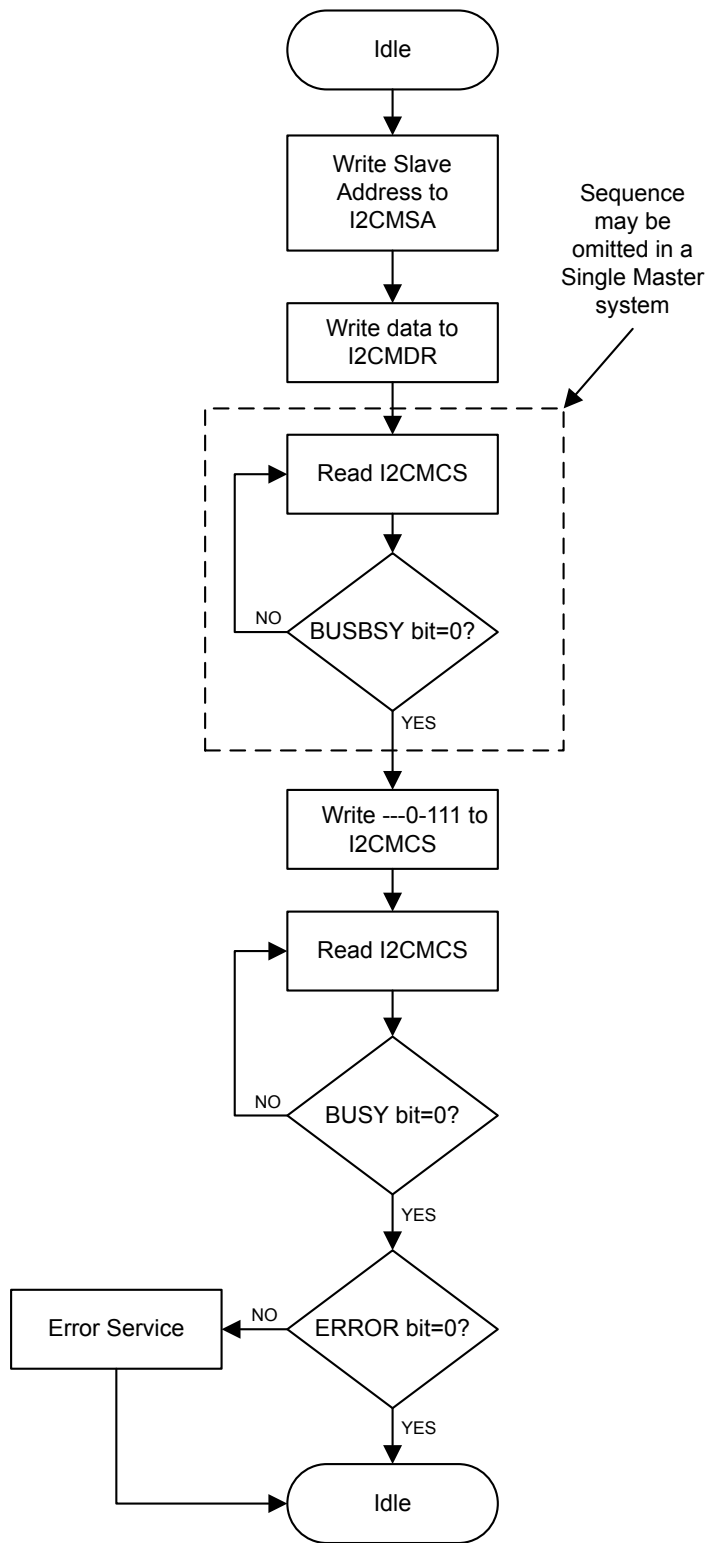


Figure 14-8. Master Single RECEIVE

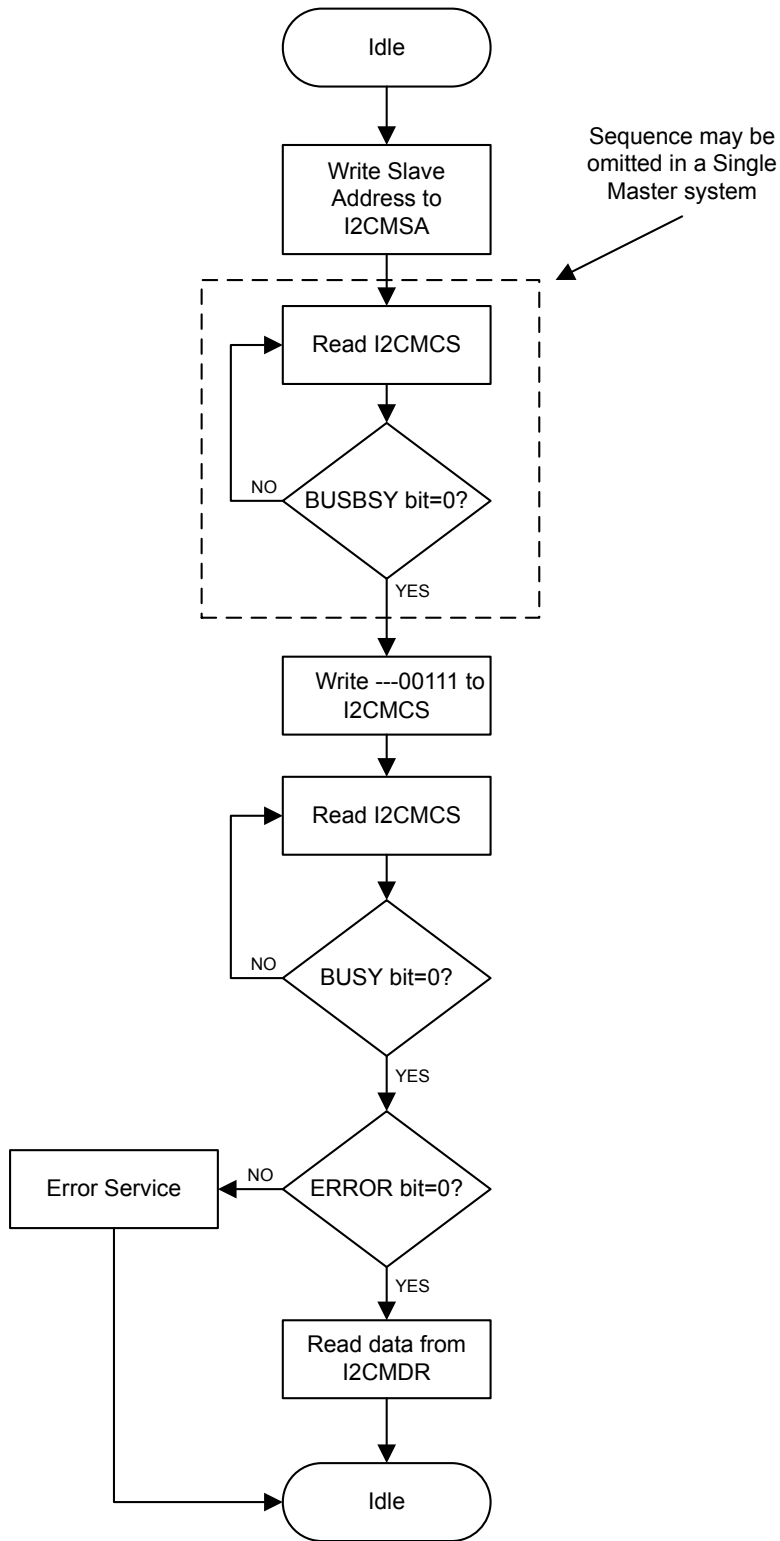


Figure 14-9. Master Burst SEND

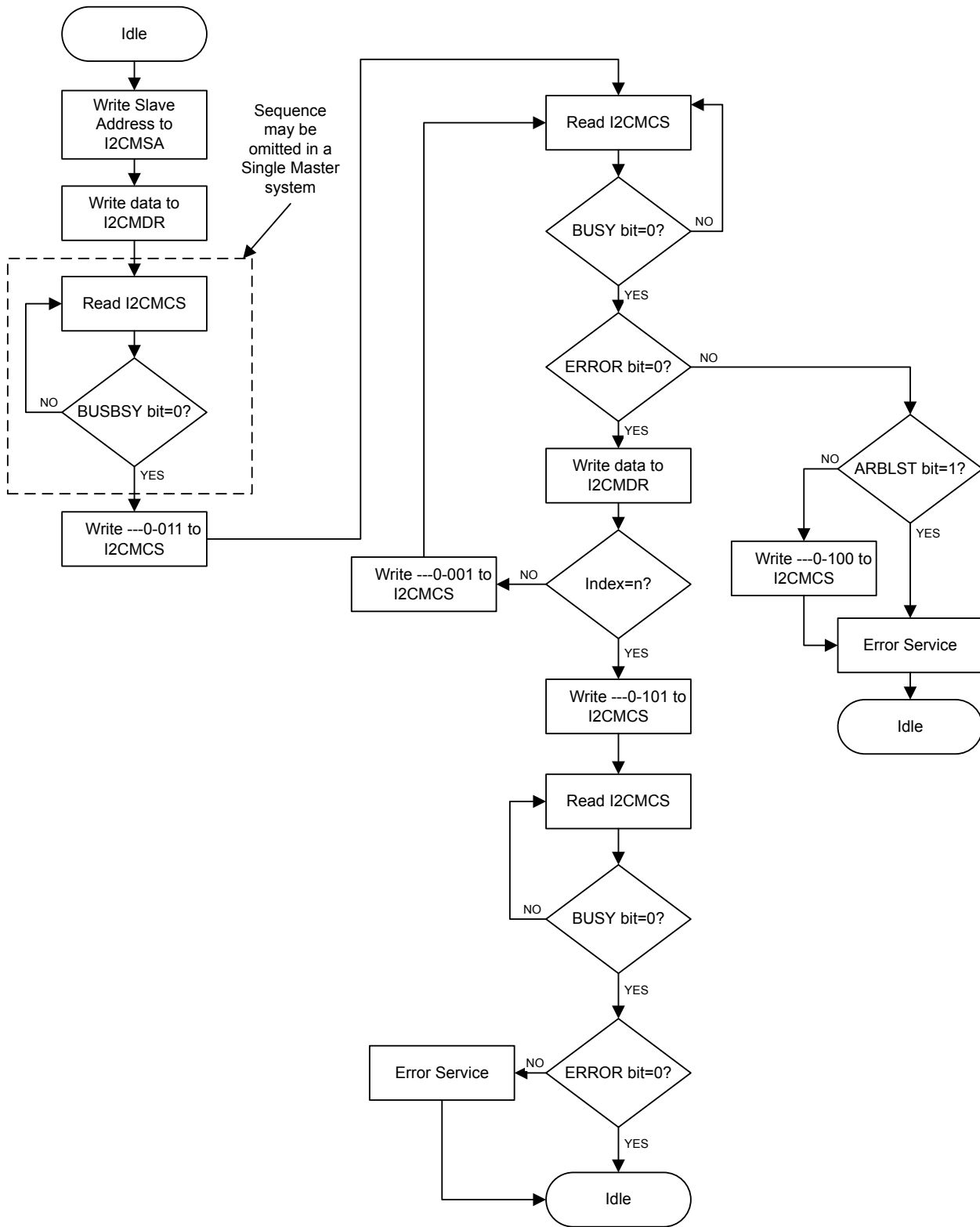


Figure 14-10. Master Burst RECEIVE

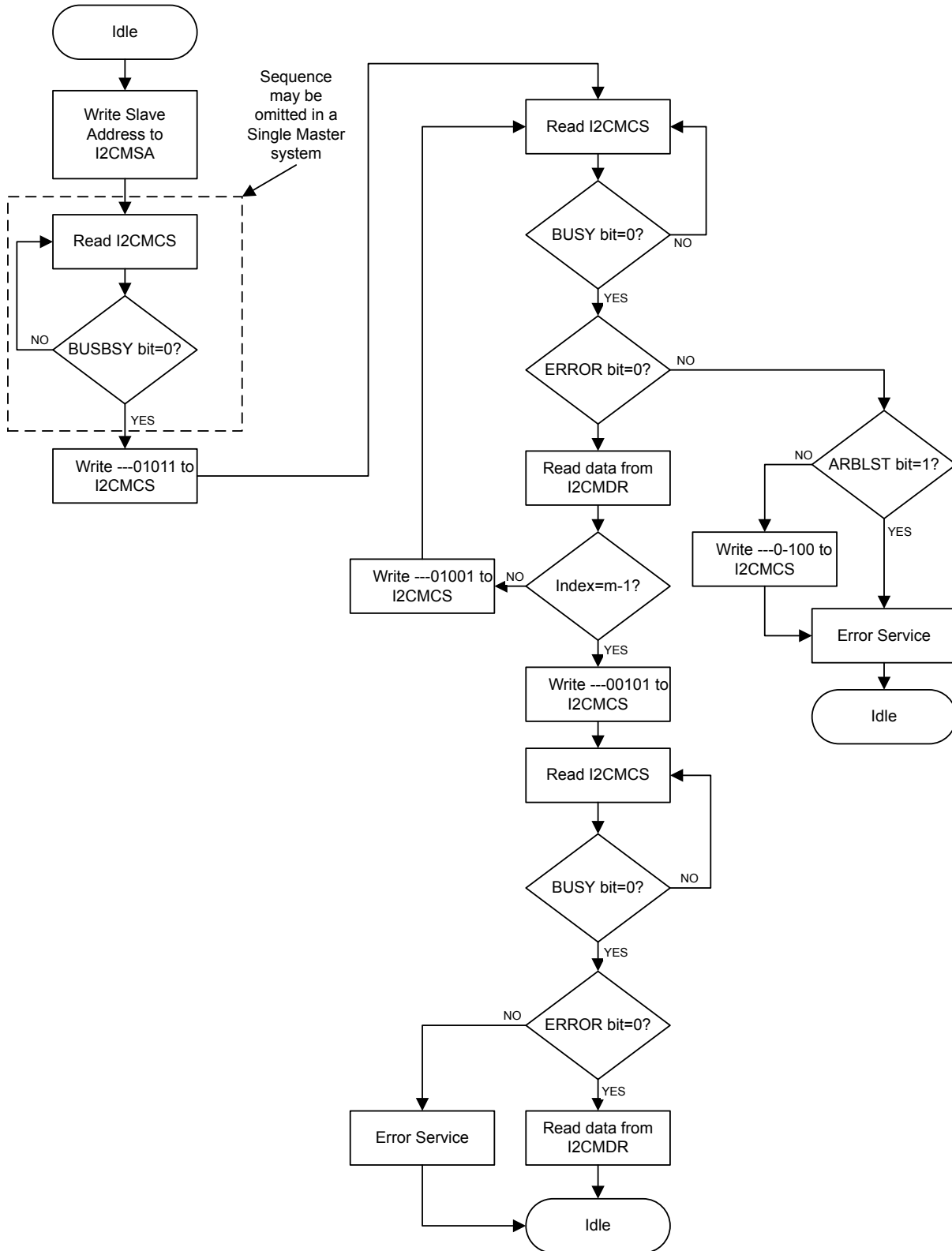


Figure 14-11. Master Burst RECEIVE after Burst SEND

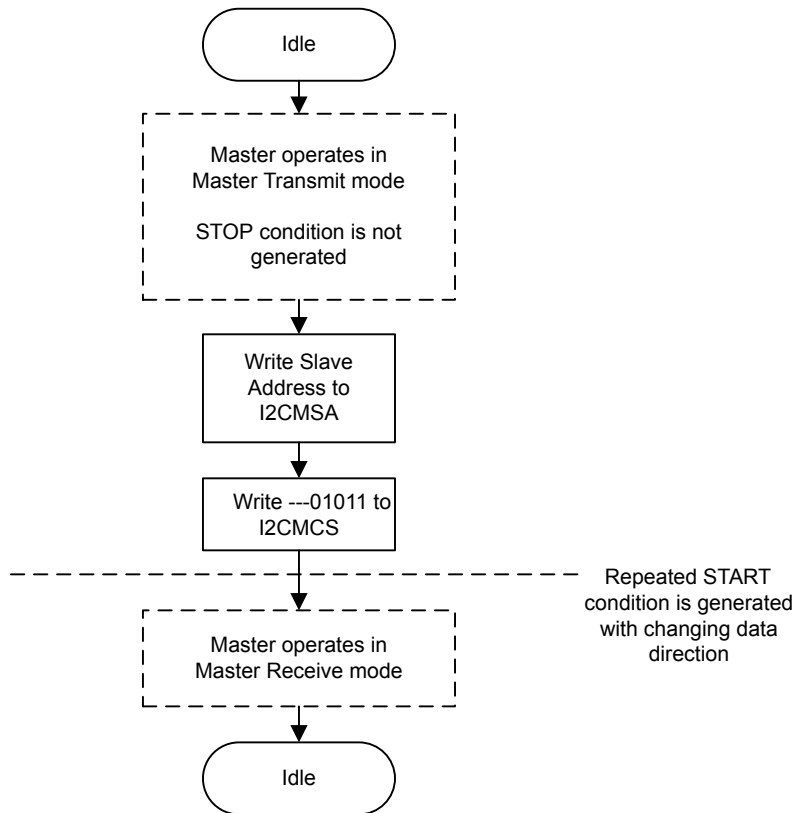
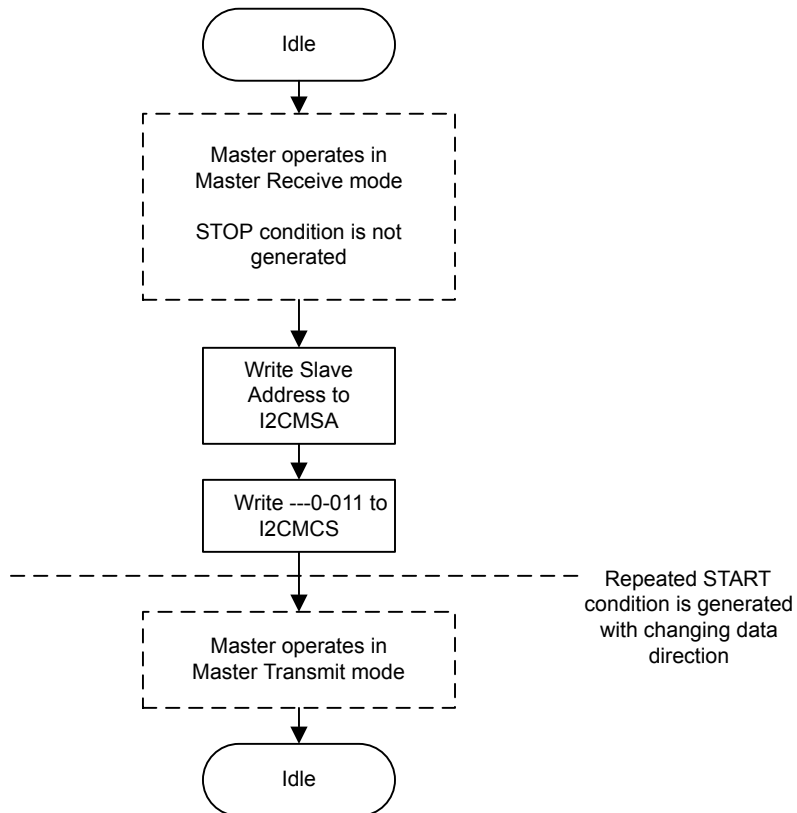


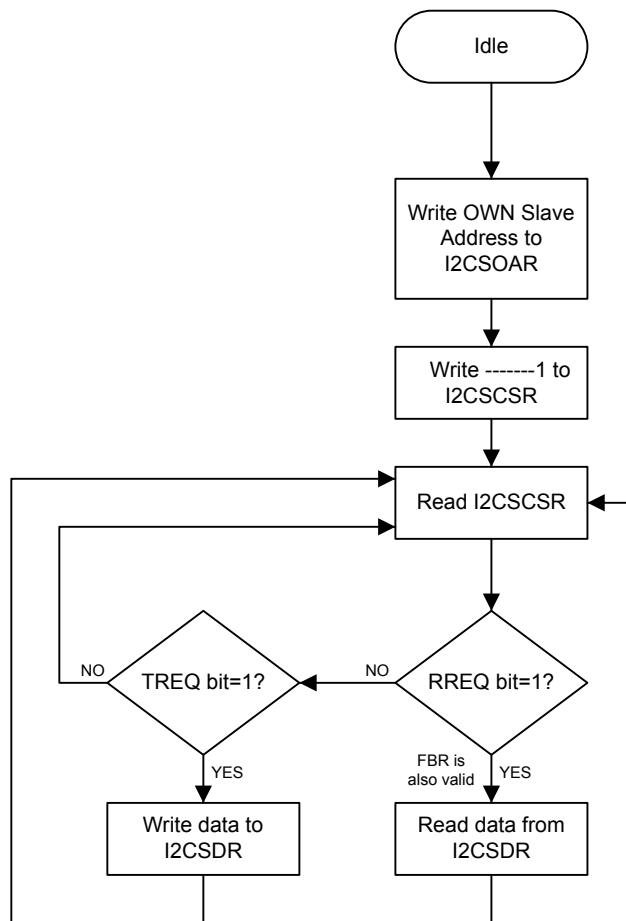
Figure 14-12. Master Burst SEND after Burst RECEIVE



14.2.5.2 I²C Slave Command Sequences

Figure 14-13 on page 363 presents the command sequence available for the I²C slave.

Figure 14-13. Slave Command Sequence



14.3 Initialization and Configuration

The following example shows how to configure the I²C module to send a single byte as a master. This assumes the system clock is 20 MHz.

1. Enable the I²C clock by writing a value of 0x0000.1000 to the **RCGC1** register in the System Control module.
2. 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 I²C 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:

$$\text{TPR} = (\text{System Clock} / (2 * (\text{SCL_LP} + \text{SCL_HP}) * \text{SCL_CLK})) - 1;$$

$$\text{TPR} = (20\text{MHz} / (2 * (6 + 4) * 100000)) - 1;$$

$$\text{TPR} = 9$$

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.
7. 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.

14.4 Register Map

Table 14-2 on page 364 lists the I²C registers. All addresses given are relative to the I²C base addresses for the master and slave:

- I²C Master 0: 0x4002.0000
- I²C Slave 0: 0x4002.0800
- I²C Master 1: 0x4002.1000
- I²C Slave 1: 0x4002.1800

Table 14-2. Inter-Integrated Circuit (I²C) Interface Register Map

Offset	Name	Type	Reset	Description	See page
I²C Master					
0x000	I2CMSA	R/W	0x0000.0000	I2C Master Slave Address	366
0x004	I2CMCS	R/W	0x0000.0000	I2C Master Control/Status	367
0x008	I2CMDR	R/W	0x0000.0000	I2C Master Data	371
0x00C	I2CMTPR	R/W	0x0000.0001	I2C Master Timer Period	372
0x010	I2CMIMR	R/W	0x0000.0000	I2C Master Interrupt Mask	373
0x014	I2CMRIS	RO	0x0000.0000	I2C Master Raw Interrupt Status	374
0x018	I2CMMIS	RO	0x0000.0000	I2C Master Masked Interrupt Status	375
0x01C	I2CMICR	WO	0x0000.0000	I2C Master Interrupt Clear	376
0x020	I2CMCR	R/W	0x0000.0000	I2C Master Configuration	377
I²C Slave					
0x000	I2CSOAR	R/W	0x0000.0000	I2C Slave Own Address	379
0x004	I2CSCSR	RO	0x0000.0000	I2C Slave Control/Status	380
0x008	I2CSDR	R/W	0x0000.0000	I2C Slave Data	382

Table 14-2. Inter-Integrated Circuit (I²C) Interface Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x00C	I2CSIMR	R/W	0x0000.0000	I2C Slave Interrupt Mask	383
0x010	I2CSRIS	RO	0x0000.0000	I2C Slave Raw Interrupt Status	384
0x014	I2CSMIS	RO	0x0000.0000	I2C Slave Masked Interrupt Status	385
0x018	I2CSICR	WO	0x0000.0000	I2C Slave Interrupt Clear	386

14.5 Register Descriptions (I²C Master)

The remainder of this section lists and describes the I²C master registers, in numerical order by address offset. See also “Register Descriptions (I²C Slave)” on page 378.

Register 1: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x000

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								SA							R/S
Type	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	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 ² 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). Value Description 0 Send. 1 Receive.

Register 2: I²C 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 I²C 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 **I²C 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 I²C module operates in Master receiver mode, the ACK bit must be set normally to logic 1. This causes the I²C bus controller to send an acknowledge automatically after each byte. This bit must be reset when the I²C bus controller requires no further data to be sent from the slave transmitter.

Reads

I2C Master Control/Status (I2CMCS)

I2C Master 0 base: 0x4002.0000

I2C Master 1 base: 0x4002.1000

Offset 0x004

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved										BUSBSY	IDLE	ARBLST	DATAACK	ADRACK	ERROR	BUSY
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	

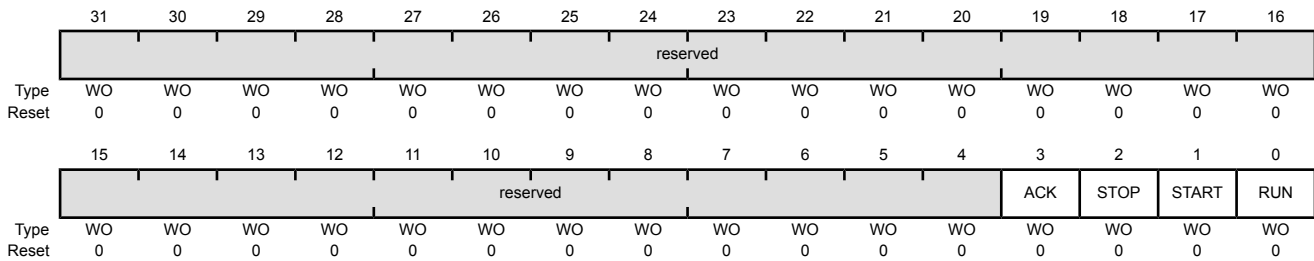
Bit/Field	Name	Type	Reset	Description
31:7	reserved	RO	0x00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
6	BUSBSY	RO	0	Bus Busy This bit specifies the state of the I ² C 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 ² C Idle This bit specifies the I ² C 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	Type	Reset	Description
3	DATAACK	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 ² C Busy This bit specifies the state of the controller. If set, the controller is busy; otherwise, the controller is idle. When the <code>BUSY</code> bit is set, the other status bits are not valid.

Writes

I2C Master Control/Status (I2CMCS)

I2C Master 0 base: 0x4002.0000
 I2C Master 1 base: 0x4002.1000
 Offset 0x004
 Type WO, reset 0x0000.0000



Bit/Field	Name	Type	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 14-3 on page 369.
2	STOP	WO	0	Generate STOP When set, causes the generation of the STOP condition. See field decoding in Table 14-3 on page 369.

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 14-3 on page 369.
0	RUN	WO	0	I ² C Master Enable When set, allows the master to send or receive data. See field decoding in Table 14-3 on page 369.

Table 14-3. Write Field Decoding for I2CMCS[3:0] Field (Sheet 1 of 3)

Current State	I2CMSA[0]	I2CMCS[3:0]				Description
	R/S	ACK	STOP	START	RUN	
Idle	0	X ^a	0	1	1	START condition followed by SEND (master goes to the Master Transmit state).
	0	X	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 combinations not listed are non-operations.					
Master Transmit	X	X	0	0	1	SEND operation (master remains in Master Transmit state).
	X	X	1	0	0	STOP condition (master goes to Idle state).
	X	X	1	0	1	SEND followed by STOP condition (master goes to Idle state).
	0	X	0	1	1	Repeated START condition followed by a SEND (master remains in Master Transmit state).
	0	X	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 combinations not listed are non-operations.					

Table 14-3. Write Field Decoding for I2CMCS[3:0] Field (Sheet 1 of 3) (continued)

Current State	I2CMSA[0]	I2CMCS[3:0]				Description
	R/S	ACK	STOP	START	RUN	
Master Receive	X	0	0	0	1	RECEIVE operation with negative ACK (master remains in Master Receive state).
	X	X	1	0	0	STOP condition (master goes to Idle state). ^b
	X	0	1	0	1	RECEIVE followed by STOP condition (master goes to Idle state).
	X	1	0	0	1	RECEIVE operation (master remains in Master Receive state).
	X	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	X	0	1	1	Repeated START condition followed by SEND (master goes to Master Transmit state).
	0	X	1	1	1	Repeated START condition followed by SEND and STOP condition (master goes to Idle state).
	All other combinations not listed are non-operations.					

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: I²C Master Data (I2CMDR), offset 0x008

Important: Use caution when reading this register. Performing a read may change bit status.

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

I2C Master 1 base: 0x4002.1000

Offset 0x008

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DATA							
Type	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	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	Data Transferred Data transferred during transaction.

Register 4: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x00C

Type R/W, reset 0x0000.0001

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								TPR							
Type	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	1

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.

$$SCL_PRD = 2 * (1 + TPR) * (SCL_LP + SCL_HP) * CLK_PRD$$

where:

- SCL_PRD is the SCL line period (I²C clock).
- TPR is the Timer Period register value (range of 1 to 255).
- SCL_LP is the SCL Low period (fixed at 6).
- SCL_HP is the SCL High period (fixed at 4).

Register 5: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x010

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															IM	
Type	RO	RO	RO	RO	RO	RO	RO	RO	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	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	IM	R/W	0	Interrupt 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: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x014

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															RIS
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

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 ² C master block. If set, an interrupt is pending; otherwise, an interrupt is not pending.

Register 7: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x018

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															MIS
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

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 ² C master block. If set, an interrupt was signaled; otherwise, an interrupt has not been generated since the bit was last cleared.

Register 8: I²C Master Interrupt Clear (I2CMICR), offset 0x01C

This register clears the raw interrupt.

I2C Master Interrupt Clear (I2CMICR)

I2C Master 0 base: 0x4002.0000

I2C Master 1 base: 0x4002.1000

Offset 0x01C

Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															IC	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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	IC	WO	0	<p>Interrupt Clear</p> <p>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.</p>

Register 9: I²C 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

I2C Master 1 base: 0x4002.1000

Offset 0x020

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved											SFE	MFE	reserved			LPBK
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	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
31:6	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 ² 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 ² 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 ² 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.

14.6 Register Descriptions (I²C Slave)

The remainder of this section lists and describes the I²C slave registers, in numerical order by address offset. See also "Register Descriptions (I²C Master)" on page 365.

Register 10: I²C Slave Own Address (I2CSOAR), offset 0x000

This register consists of seven address bits that identify the Stellaris® I²C device on the I²C bus.

I²C Slave Own Address (I2CSOAR)

I²C Slave 0 base: 0x4002.0800

I²C Slave 1 base: 0x4002.1800

Offset 0x000

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved									OAR						
Type	RO	RO	RO	RO	RO	RO	RO	RO	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

Bit/Field	Name	Type	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.
6:0	OAR	R/W	0x00	I ² C Slave Own Address This field specifies bits A6 through A0 of the slave address.

Register 11: I²C Slave Control/Status (I2CCSR), offset 0x004

This register accesses one control bit when written, and three status bits when read.

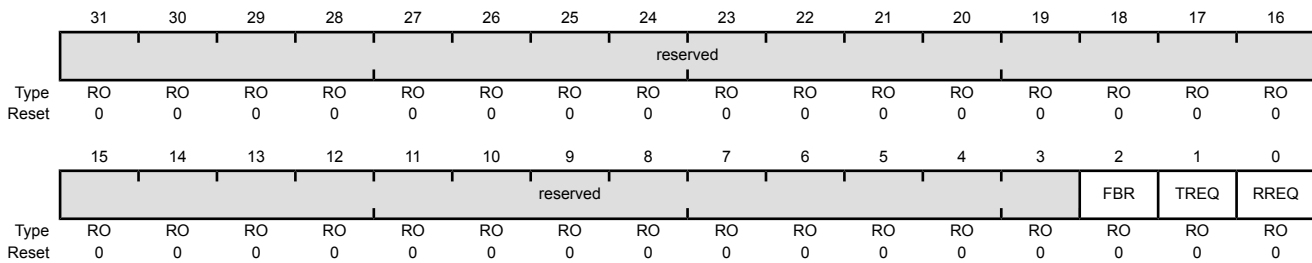
The read-only Status register consists of three bits: the **FBR**, **RREQ**, and **TREQ** bits. The **First Byte Received (FBR)** bit is set only after the Stellaris[®] device detects its own slave address and receives the first data byte from the I²C master. The **Receive Request (RREQ)** bit indicates 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 RREQ bit. The **Transmit Request (TREQ)** bit indicates that the Stellaris[®] I²C device is addressed as a Slave Transmitter. Write one data byte into the **I²C 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[®] I²C slave operation.

Reads

I2C Slave Control/Status (I2CCSR)

I2C Slave 0 base: 0x4002.0800
 I2C Slave 1 base: 0x4002.1800
 Offset 0x004
 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	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	FBR	RO	0	<p>First Byte Received</p> <p>Indicates that the first byte following the slave's own address is received. This bit is only valid when the RREQ bit is set, and is automatically cleared when data has been read from the I2CSDR register.</p> <p>Note: This bit is not used for slave transmit operations.</p>
1	TREQ	RO	0	<p>Transmit Request</p> <p>This bit specifies the state of the I²C slave with regards to outstanding transmit requests. If set, the I²C 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.</p>

Bit/Field	Name	Type	Reset	Description
0	RREQ	RO	0	Receive Request This bit specifies the status of the I ² C slave with regards to outstanding receive requests. If set, the I ² C unit has outstanding receive data from the I ² C master and uses clock stretching to delay the master until the data has been read from the I2CSDR register. Otherwise, no receive data is outstanding.

Writes

I2C Slave Control/Status (I2CSCSR)

I2C Slave 0 base: 0x4002.0800

I2C Slave 1 base: 0x4002.1800

Offset 0x004

Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															DA	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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	DA	WO	0	Device Active Value Description 0 Disables the I ² C slave operation. 1 Enables the I ² C slave operation.

Register 12: I²C Slave Data (I2CSDR), offset 0x008

Important: Use caution when reading this register. Performing a read may change bit status.

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

I2C Slave 1 base: 0x4002.1800

Offset 0x008

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved								DATA							
Type	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	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	0x0	Data for Transfer This field contains the data for transfer during a slave receive or transmit operation.

Register 13: I²C 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

I2C Slave 1 base: 0x4002.1800

Offset 0x00C

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															DATAIM
Type	RO	RO	RO	RO	RO	RO	RO	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

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	DATAIM	R/W	0	Data Interrupt Mask This bit controls whether the raw interrupt for data received and data requested 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: I²C 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

I2C Slave 1 base: 0x4002.1800

Offset 0x010

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															DATARIS
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

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	DATARIS	RO	0	Data Raw Interrupt Status This bit specifies the raw interrupt state for data received and data requested (prior to masking) of the I ² C slave block. If set, an interrupt is pending; otherwise, an interrupt is not pending.

Register 15: I²C 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

I2C Slave 1 base: 0x4002.1800

Offset 0x014

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															DATAMIS
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

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	DATAMIS	RO	0	Data Masked Interrupt Status This bit specifies the interrupt state for data received and data requested (after masking) of the I ² C slave block. If set, an interrupt was signaled; otherwise, an interrupt has not been generated since the bit was last cleared.

Register 16: I²C Slave Interrupt Clear (I2CSICR), offset 0x018

This register clears the raw interrupt. A read of this register returns no meaningful data.

I2C Slave Interrupt Clear (I2CSICR)

I2C Slave 0 base: 0x4002.0800

I2C Slave 1 base: 0x4002.1800

Offset 0x018

Type WO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															DATAIC	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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	DATAIC	WO	0	Data Interrupt Clear This bit controls the clearing of the raw interrupt for data received and data requested. When set, it clears the <code>DATARIS</code> interrupt bit; otherwise, it has no effect on the <code>DATARIS</code> bit value.

15 Analog Comparators

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

Note: Not all comparators have the option to drive an output pin.

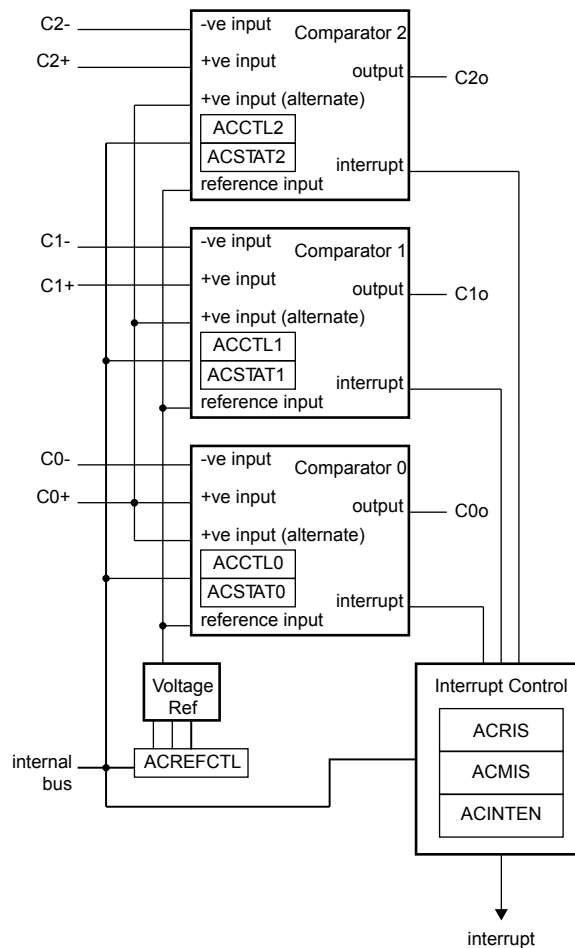
The comparator can provide its output to a device pin, acting as a replacement for an analog comparator on the board, or it can be used to signal the application via interrupts to cause it to start capturing a sample sequence.

The Stellaris® Analog Comparators module has the following features:

- 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
- 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

15.1 Block Diagram

Figure 15-1. Analog Comparator Module Block Diagram



15.2 Functional Description

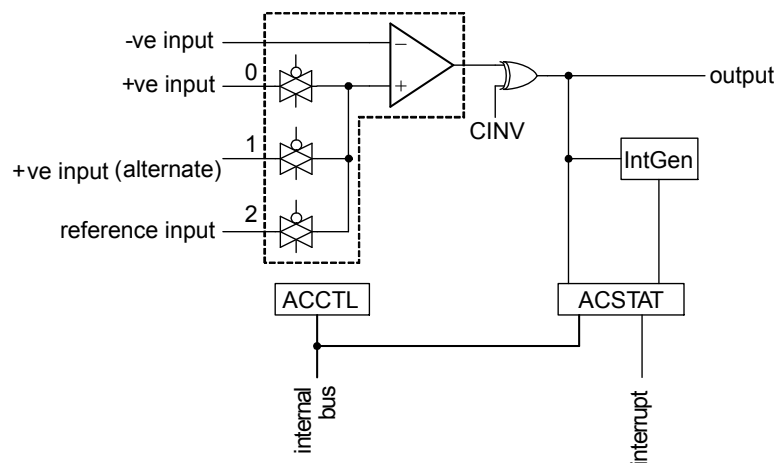
Important: It is recommended that the Digital-Input enable (the `GPIOEN` bit in the GPIO module) for the analog input pin be disabled to prevent excessive current draw from the I/O pads.

The comparator compares the V_{IN-} and V_{IN+} inputs to produce an output, V_{OUT} .

$$V_{IN-} < V_{IN+}, V_{OUT} = 1$$

$$V_{IN-} > V_{IN+}, V_{OUT} = 0$$

As shown in Figure 15-2 on page 389, the input source for V_{IN-} is an external input. In addition to an external input, input sources for V_{IN+} can be the +ve input of comparator 0 or an internal reference.

Figure 15-2. Structure of Comparator Unit

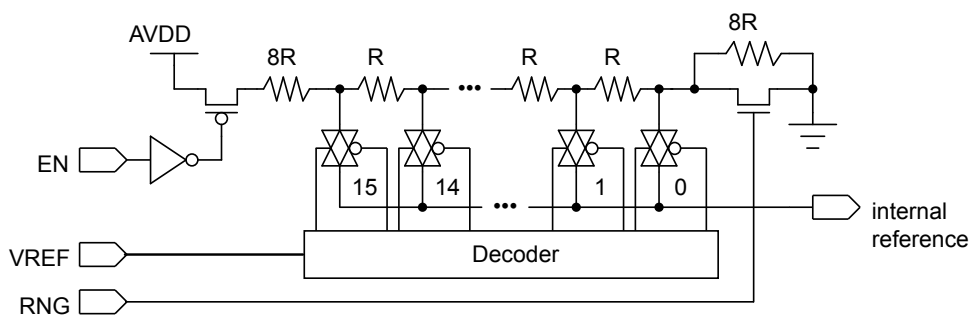
A comparator is configured through two status/control registers (**ACCTL** and **ACSTAT**). The internal reference is configured through one control register (**ACREFCTL**). Interrupt status and control is configured through three registers (**ACMIS**, **ACRIS**, and **ACINTEN**).

Typically, the comparator output is used internally to generate controller interrupts. It may also be used to drive an external pin.

Important: The **ASRCP** bits in the **ACCTLn** register must be set before using the analog comparators.

15.2.1 Internal Reference Programming

The structure of the internal reference is shown in Figure 15-3 on page 389. This is controlled by a single configuration register (**ACREFCTL**). Table 15-1 on page 389 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 15-1. 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	RNG=X	0 V (GND) for any value of VREF; however, it is recommended that RNG=1 and VREF=0 for the least noisy ground reference.

Table 15-1. Internal Reference Voltage and ACREFCTL Field Values (continued)

ACREFCTL Register		Output Reference Voltage Based on VREF Field Value
EN Bit Value	RNG Bit Value	
EN=1	RNG=0	<p>Total resistance in ladder is 31 R.</p> $V_{REF} = AV_{DD} \times \frac{R_{VREF}}{R_T}$ $V_{REF} = AV_{DD} \times \frac{(V_{REF} + 8)}{31}$ $V_{REF} = 0.85 + 0.106 \times V_{REF}$ <p>The range of internal reference in this mode is 0.85-2.448 V.</p>
	RNG=1	<p>Total resistance in ladder is 23 R.</p> $V_{REF} = AV_{DD} \times \frac{R_{VREF}}{R_T}$ $V_{REF} = AV_{DD} \times \frac{V_{REF}}{23}$ $V_{REF} = 0.143 \times V_{REF}$ <p>The range of internal reference for this mode is 0-2.152 V.</p>

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 C0- as a GPIO input.
3. 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 by writing the **ACCTL0** register with the value of 0x0000.040C.
5. 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 C0- to see the OVAL value change.

15.4 Register Map

Table 15-2 on page 391 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 15-2. Analog Comparators Register Map

Offset	Name	Type	Reset	Description	See page
0x000	ACMIS	R/W1C	0x0000.0000	Analog Comparator Masked Interrupt Status	392
0x004	ACRIS	RO	0x0000.0000	Analog Comparator Raw Interrupt Status	393
0x008	ACINTEN	R/W	0x0000.0000	Analog Comparator Interrupt Enable	394
0x010	ACREFCTL	R/W	0x0000.0000	Analog Comparator Reference Voltage Control	395
0x020	ACSTAT0	RO	0x0000.0000	Analog Comparator Status 0	396
0x024	ACCTL0	R/W	0x0000.0000	Analog Comparator Control 0	397
0x040	ACSTAT1	RO	0x0000.0000	Analog Comparator Status 1	396
0x044	ACCTL1	R/W	0x0000.0000	Analog Comparator Control 1	397
0x060	ACSTAT2	RO	0x0000.0000	Analog Comparator Status 2	396
0x064	ACCTL2	R/W	0x0000.0000	Analog Comparator Control 2	397

15.5 Register Descriptions

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

Register 1: Analog Comparator Masked Interrupt Status (ACMIS), offset 0x000

This register provides a summary of the interrupt status (masked) of the comparator.

Analog Comparator Masked Interrupt Status (ACMIS)

Base 0x4003.C000

Offset 0x000

Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													IN2	IN1	IN0	
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	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 0x004

This register provides a summary of the interrupt status (raw) of the comparator.

Analog Comparator Raw Interrupt Status (ACRIS)

Base 0x4003.C000

Offset 0x004

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved													IN2	IN1	IN0
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

Bit/Field	Name	Type	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 0x008

This register provides the interrupt enable for the comparator.

Analog Comparator Interrupt Enable (ACINTEN)

Base 0x4003.C000
 Offset 0x008
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													IN2	IN1	IN0	
Type	RO	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	0

Bit/Field	Name	Type	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.

Register 4: Analog Comparator Reference Voltage Control (ACREFCTL), offset 0x010

This register specifies whether the resistor ladder is powered on as well as the range and tap.

Analog Comparator Reference Voltage Control (ACREFCTL)

Base 0x4003.C000

Offset 0x010

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved						EN	RNG	reserved				VREF			
Type	RO	RO	RO	RO	RO	RO	R/W	R/W	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
31:10	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.
9	EN	R/W	0	Resistor Ladder Enable The EN bit specifies whether the resistor ladder is powered on. If 0, the resistor ladder is unpowered. If 1, the resistor ladder is connected to the analog V_{DD} . This bit is reset to 0 so that the internal reference consumes the least amount of power if not used and programmed.
8	RNG	R/W	0	Resistor Ladder Range The RNG bit specifies the range of the resistor ladder. If 0, the resistor ladder has a total resistance of 31 R. If 1, the resistor ladder has a total resistance of 23 R.
7:4	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.
3:0	VREF	R/W	0x00	Resistor Ladder Voltage Ref The 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 15-1 on page 389 for some output reference voltage examples.

Register 5: Analog Comparator Status 0 (ACSTAT0), offset 0x020

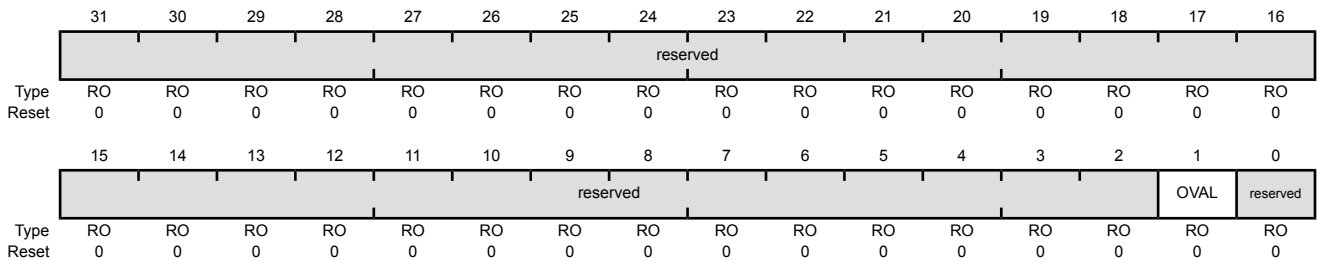
Register 6: Analog Comparator Status 1 (ACSTAT1), offset 0x040

Register 7: Analog Comparator Status 2 (ACSTAT2), offset 0x060

These registers specify the current output value of the comparator.

Analog Comparator Status 0 (ACSTAT0)

Base 0x4003.C000
 Offset 0x020
 Type RO, 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	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.

Register 8: Analog Comparator Control 0 (ACCTL0), offset 0x024**Register 9: Analog Comparator Control 1 (ACCTL1), offset 0x044****Register 10: Analog Comparator Control 2 (ACCTL2), offset 0x064**

These registers configure the comparator's input and output.

Analog Comparator Control 0 (ACCTL0)

Base 0x4003.C000

Offset 0x024

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				ASRCP		reserved				ISLVAL	ISEN		CINV	reserved	
Type	RO	RO	RO	RO	RO	R/W	R/W	RO	RO	RO	RO	R/W	R/W	R/W	R/W	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description										
31:11	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.										
10:9	ASRCP	R/W	0x00	<p>Analog Source Positive</p> <p>The <code>ASRCP</code> field specifies the source of input voltage to the <code>VIN+</code> terminal of the comparator. The encodings for this field are as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Pin value</td> </tr> <tr> <td>0x1</td> <td>Pin value of C0+</td> </tr> <tr> <td>0x2</td> <td>Internal voltage reference</td> </tr> <tr> <td>0x3</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Function	0x0	Pin value	0x1	Pin value of C0+	0x2	Internal voltage reference	0x3	Reserved
Value	Function													
0x0	Pin value													
0x1	Pin value of C0+													
0x2	Internal voltage reference													
0x3	Reserved													
8:5	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	ISLVAL	R/W	0	<p>Interrupt Sense Level Value</p> <p>The <code>ISLVAL</code> bit specifies the sense value of the input that generates an interrupt if in Level Sense mode. If 0, an interrupt is generated if the comparator output is Low. Otherwise, an interrupt is generated if the comparator output is High.</p>										

Bit/Field	Name	Type	Reset	Description										
3:2	ISEN	R/W	0x0	<p>Interrupt Sense</p> <p>The <code>ISEN</code> field specifies the sense of the comparator output that generates an interrupt. The sense conditioning is as follows:</p> <table><thead><tr><th>Value</th><th>Function</th></tr></thead><tbody><tr><td>0x0</td><td>Level sense, see <code>ISLVAL</code></td></tr><tr><td>0x1</td><td>Falling edge</td></tr><tr><td>0x2</td><td>Rising edge</td></tr><tr><td>0x3</td><td>Either edge</td></tr></tbody></table>	Value	Function	0x0	Level sense, see <code>ISLVAL</code>	0x1	Falling edge	0x2	Rising edge	0x3	Either edge
Value	Function													
0x0	Level sense, see <code>ISLVAL</code>													
0x1	Falling edge													
0x2	Rising edge													
0x3	Either edge													
1	CINV	R/W	0	<p>Comparator Output Invert</p> <p>The <code>CINV</code> 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.</p>										
0	reserved	RO	0	<p>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.</p>										

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® PWM module consists of three PWM generator blocks and a control block. 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 (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 blocks are managed by the output control block before being passed to the device pins.

The Stellaris® 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. Three generator blocks can also generate the full six channels of gate controls required by a 3-phase inverter bridge.

Each Stellaris® PWM module has the following features:

- Three PWM generator blocks, each with one 16-bit counter, two PWM comparators, a PWM signal generator, a dead-band generator, and an interrupt selector
- One fault input in hardware to promote low-latency shutdown
- 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

16.1 Block Diagram

Figure 16-1 on page 400 provides the Stellaris[®] PWM module unit diagram and Figure 16-2 on page 401 provides a more detailed diagram of a Stellaris[®] PWM generator. The LM3S1960 controller contains three generator blocks (PWM0, PWM1, and PWM2) and generates six independent PWM signals or three paired PWM signals with dead-band delays inserted.

Figure 16-1. PWM Unit Diagram

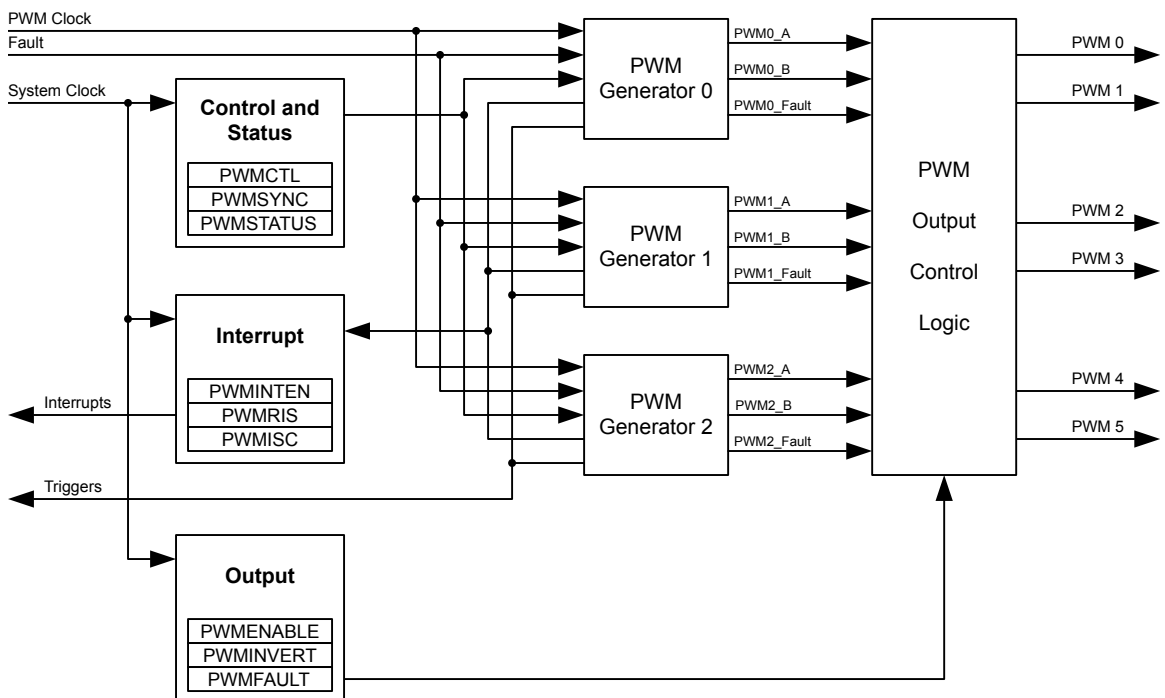
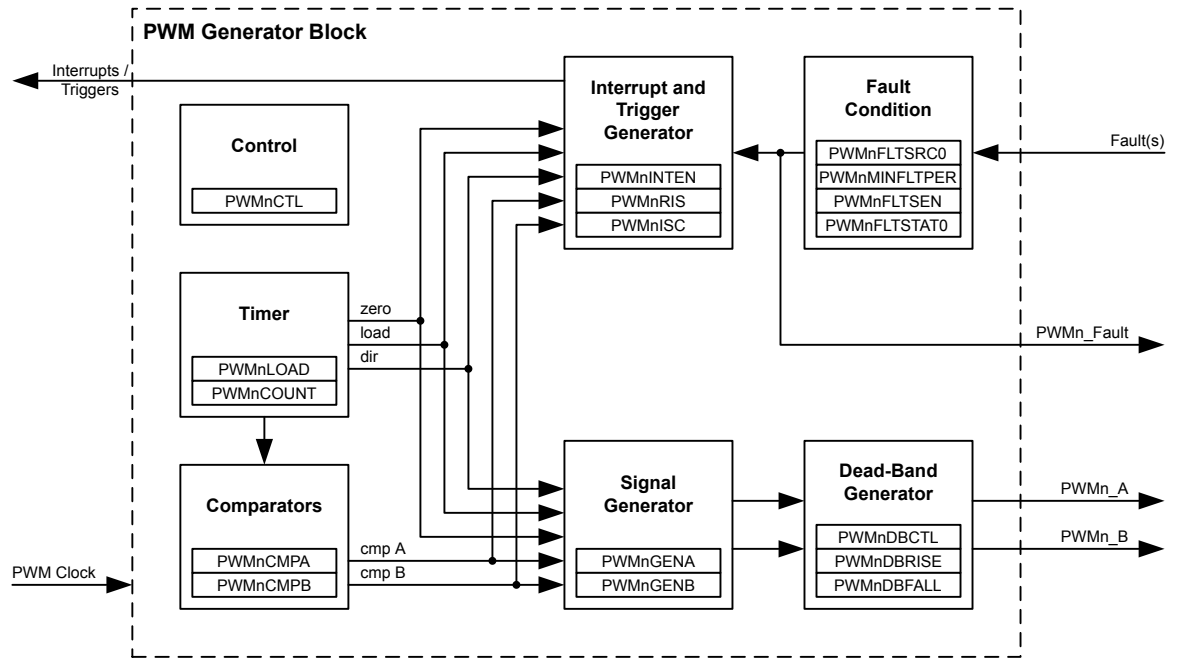


Figure 16-2. PWM Module Block Diagram



16.2 Functional Description

16.2.1 PWM Timer

The timer in each PWM generator 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 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 each 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 16-3 on page 402 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Down mode. Figure 16-4 on page 402 shows the behavior of the counter and the relationship of these pulses when the counter is in Count-Up/Down mode.

Figure 16-3. PWM Count-Down Mode

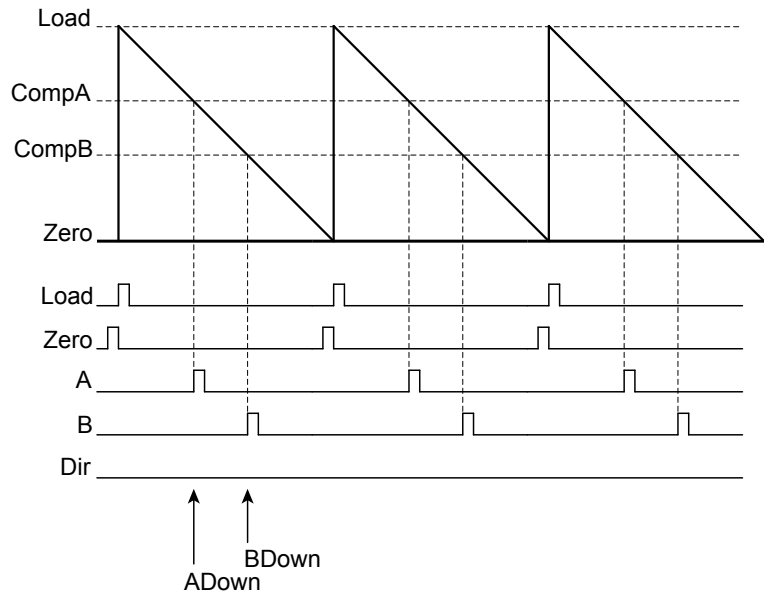
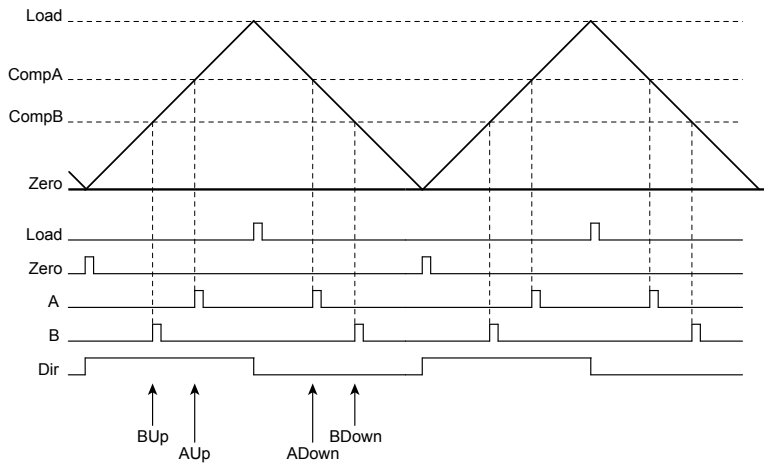


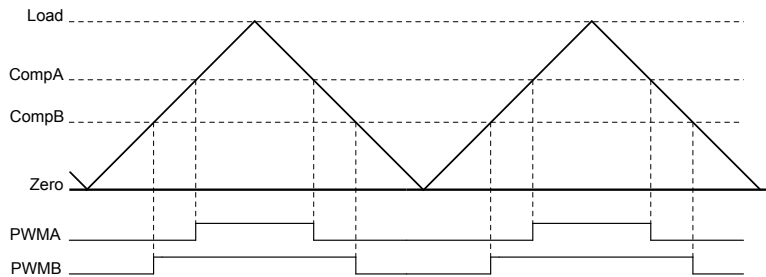
Figure 16-4. PWM Count-Up/Down Mode



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 16-5 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.

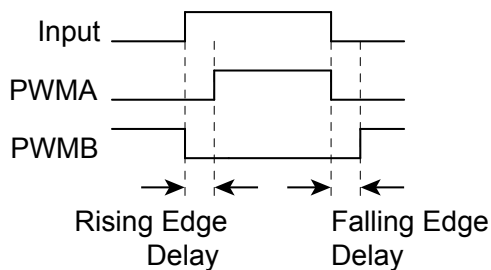
Figure 16-5. PWM Generation Example In Count-Up/Down Mode

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 16-6 on page 403 shows the effect of the dead-band generator on an input PWM signal.

Figure 16-6. 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 synchronously reset any or all of the counters in the PWM generators. If multiple PWM generators are configured with the same counter load value, this can be used to guarantee that they also have the same count value (this does imply that the PWM generators must be configured before they are synchronized). With this, more than two PWM signals can be produced with a known relationship between the edges of those signals since the counters always have the same values.

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 in multiple PWM generators 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. The Update mode of the load and comparator match values can be individually configured in each PWM generator block. It typically makes sense to use the synchronous update mechanism across PWM generator blocks when the timers in those blocks are synchronized, though this is not required in order for this mechanism to function properly.

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 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 each 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.
2. 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.
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).
5. 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.

16.4 Register Map

Table 16-1 on page 405 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	Type	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

Table 16-1. PWM Register Map (continued)

Offset	Name	Type	Reset	Description	See page
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
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
0x080	PWM1CTL	R/W	0x0000.0000	PWM1 Control	417
0x084	PWM1INTEN	R/W	0x0000.0000	PWM1 Interrupt Enable	419
0x088	PWM1RIS	RO	0x0000.0000	PWM1 Raw Interrupt Status	421
0x08C	PWM1ISC	R/W1C	0x0000.0000	PWM1 Interrupt Status and Clear	422
0x090	PWM1LOAD	R/W	0x0000.0000	PWM1 Load	423
0x094	PWM1COUNT	RO	0x0000.0000	PWM1 Counter	424
0x098	PWM1CMPA	R/W	0x0000.0000	PWM1 Compare A	425
0x09C	PWM1CMPB	R/W	0x0000.0000	PWM1 Compare B	426
0x0A0	PWM1GENA	R/W	0x0000.0000	PWM1 Generator A Control	427
0x0A4	PWM1GENB	R/W	0x0000.0000	PWM1 Generator B Control	430
0x0A8	PWM1DBCTL	R/W	0x0000.0000	PWM1 Dead-Band Control	433
0x0AC	PWM1DBRISE	R/W	0x0000.0000	PWM1 Dead-Band Rising-Edge Delay	434

Table 16-1. PWM Register Map (continued)

Offset	Name	Type	Reset	Description	See page
0x0B0	PWM1DBFALL	R/W	0x0000.0000	PWM1 Dead-Band Falling-Edge-Delay	435
0x0C0	PWM2CTL	R/W	0x0000.0000	PWM2 Control	417
0x0C4	PWM2INTEN	R/W	0x0000.0000	PWM2 InterruptEnable	419
0x0C8	PWM2RIS	RO	0x0000.0000	PWM2 Raw Interrupt Status	421
0x0CC	PWM2ISC	R/W1C	0x0000.0000	PWM2 Interrupt Status and Clear	422
0x0D0	PWM2LOAD	R/W	0x0000.0000	PWM2 Load	423
0x0D4	PWM2COUNT	RO	0x0000.0000	PWM2 Counter	424
0x0D8	PWM2CMPA	R/W	0x0000.0000	PWM2 Compare A	425
0x0DC	PWM2CMPB	R/W	0x0000.0000	PWM2 Compare B	426
0x0E0	PWM2GENA	R/W	0x0000.0000	PWM2 Generator A Control	427
0x0E4	PWM2GENB	R/W	0x0000.0000	PWM2 Generator B Control	430
0x0E8	PWM2DBCTL	R/W	0x0000.0000	PWM2 Dead-Band Control	433
0x0EC	PWM2DBRISE	R/W	0x0000.0000	PWM2 Dead-Band Rising-Edge Delay	434
0x0F0	PWM2DBFALL	R/W	0x0000.0000	PWM2 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.

Register 1: PWM Master Control (PWMCTL), offset 0x000

This register provides master control over the PWM generation blocks.

PWM Master Control (PWMCTL)

Base 0x4002.8000
 Offset 0x000
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													GlobalSync2	GlobalSync1	GlobalSync0	
Type	RO	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	0

Bit/Field	Name	Type	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	GlobalSync2	R/W	0	Update PWM Generator 2 Same as GlobalSync0 but for PWM generator 2.
1	GlobalSync1	R/W	0	Update PWM Generator 1 Same as GlobalSync0 but for PWM generator 1.
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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved													Sync2	Sync1	Sync0
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/Field	Name	Type	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	Sync2	R/W	0	Reset Generator 2 Counter Performs a reset of the PWM generator 2 counter.
1	Sync1	R/W	0	Reset Generator 1 Counter Performs a reset of the PWM generator 1 counter.
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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved											PWM5En	PWM4En	PWM3En	PWM2En	PWM1En	PWM0En
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	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	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	PWM5En	R/W	0	PWM5 Output Enable When set, allows the generated PWM5 signal to be passed to the device pin.
4	PWM4En	R/W	0	PWM4 Output Enable When set, allows the generated PWM4 signal to be passed to the device pin.
3	PWM3En	R/W	0	PWM3 Output Enable When set, allows the generated PWM3 signal to be passed to the device pin.
2	PWM2En	R/W	0	PWM2 Output Enable When set, allows the generated PWM2 signal to be passed to the device pin.
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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved										PWM5Inv	PWM4Inv	PWM3Inv	PWM2Inv	PWM1Inv	PWM0Inv
Type	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/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	PWM5Inv	R/W	0	Invert PWM5 Signal When set, the generated PWM5 signal is inverted.
4	PWM4Inv	R/W	0	Invert PWM4 Signal When set, the generated PWM4 signal is inverted.
3	PWM3Inv	R/W	0	Invert PWM3 Signal When set, the generated PWM3 signal is inverted.
2	PWM2Inv	R/W	0	Invert PWM2 Signal When set, the generated PWM2 signal is inverted.
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 inputs and debug events are considered fault conditions. On a fault condition, each PWM signal can 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 occurs 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved											Fault5	Fault4	Fault3	Fault2	Fault1	Fault0
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	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	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	Fault5	R/W	0	PWM5 Fault When set, the PWM5 output signal is driven Low on a fault condition.
4	Fault4	R/W	0	PWM4 Fault When set, the PWM4 output signal is driven Low on a fault condition.
3	Fault3	R/W	0	PWM3 Fault When set, the PWM3 output signal is driven Low on a fault condition.
2	Fault2	R/W	0	PWM2 Fault When set, the PWM2 output signal is driven Low on a fault condition.
1	Fault1	R/W	0	PWM1 Fault When set, the PWM1 output signal is driven Low on a fault condition.
0	Fault0	R/W	0	PWM0 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 generators.

PWM Interrupt Enable (PWMINTEN)

Base 0x4002.8000

Offset 0x014

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved															IntFault	
Type	RO	RO	RO	RO	RO	RO	RO	RO	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	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	reserved												IntPWM2	IntPWM1	IntPWM0		
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/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 set, an interrupt occurs when the fault input is asserted.
15:3	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	IntPWM2	R/W	0	PWM2 Interrupt Enable When set, an interrupt occurs when the PWM generator 2 block asserts an interrupt.
1	IntPWM1	R/W	0	PWM1 Interrupt Enable When set, an interrupt occurs when the PWM generator 1 block asserts an interrupt.
0	IntPWM0	R/W	0	PWM0 Interrupt Enable When set, 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 generators; they are cleared via the interrupt status register in the PWM generator blocks. Bits set to 1 indicate the events that are active; zero bits indicate that the event in question is not active.

PWM Raw Interrupt Status (PWMRIS)

Base 0x4002.8000
 Offset 0x018
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															IntFault
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
	reserved												IntPWM2	IntPWM1	IntPWM0	
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

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	RO	0	Fault Interrupt Asserted Indicates that the fault input is asserting.
15:3	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	IntPWM2	RO	0	PWM2 Interrupt Asserted Indicates that the PWM generator 2 block is asserting its interrupt.
1	IntPWM1	RO	0	PWM1 Interrupt Asserted Indicates that the PWM generator 1 block is asserting its interrupt.
0	IntPWM0	RO	0	PWM0 Interrupt Asserted Indicates that the PWM generator 0 block is asserting its interrupt.

Register 8: PWM Interrupt Status and Clear (PWMISC), offset 0x01C

This register provides a summary of the interrupt status of the individual PWM generator blocks. A bit set to 1 indicates that the corresponding generator block is asserting an interrupt. The individual interrupt status registers in each block 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
Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															IntFault
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C
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												IntPWM2	IntPWM1	IntPWM0	
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

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/W1C	0	Fault Interrupt Asserted Indicates that the fault input is asserting an interrupt.
15:3	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	IntPWM2	RO	0	PWM2 Interrupt Status Indicates if the PWM generator 2 block is asserting an interrupt.
1	IntPWM1	RO	0	PWM1 Interrupt Status Indicates if the PWM generator 1 block is asserting an interrupt.
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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															Fault
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

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	Fault	RO	0	Fault Interrupt Status When set, indicates the fault input is asserted.

Register 10: PWM Control (PWM0CTL), offset 0x040**Register 11: PWM1 Control (PWM1CTL), offset 0x080****Register 12: PWM2 Control (PWM2CTL), offset 0x0C0**

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, the PWM1 block produces the `PWM2` and `PWM3` outputs, and the PWM2 block produces the `PWM4` and `PWM5` outputs.

PWM0 Control (PWM0CTL)

Base 0x4002.8000
Offset 0x040
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved										CmpBUpd	CmpAUpd	LoadUpd	Debug	Mode	Enable
Type	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/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	CmpBUpd	R/W	0	Comparator B Update Mode Same as <code>CmpAUpd</code> but for the comparator B register.
4	CmpAUpd	R/W	0	Comparator A Update Mode The Update mode for the comparator A register. When not set, updates to the register are reflected to the comparator the next time the counter is 0. When set, updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (PWMCTL) register (see page 408).
3	LoadUpd	R/W	0	Load Register Update Mode The Update mode for the load register. When not set, updates to the register are reflected to the counter the next time the counter is 0. When set, updates to the register are delayed until the next time the counter is 0 after a synchronous update has been requested through the PWM Master Control (PWMCTL) register.
2	Debug	R/W	0	Debug Mode The behavior of the counter in Debug mode. When not set, the counter stops running when it next reaches 0, and continues running again when no longer in Debug mode. When set, the counter always runs.

Bit/Field	Name	Type	Reset	Description
1	Mode	R/W	0	Counter Mode The mode for the counter. When not set, the counter counts down from the load value to 0 and then wraps back to the load value (Count-Down mode). When set, 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. When not set, the entire block is disabled and not clocked. When set, the block is enabled and produces PWM signals.

Register 13: PWM0 Interrupt Enable (PWM0INTEN), offset 0x044**Register 14: PWM1 Interrupt Enable (PWM1INTEN), offset 0x084****Register 15: PWM2 InterruptEnable (PWM2INTEN), offset 0x0C4**

These registers control the interrupt generation capabilities of the PWM generators (**PWM0INTEN** controls the PWM generator 0 block, and so on). The events that can cause an interrupt are:

- The counter being equal to the load register
- The counter being equal to zero
- 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
- The counter being equal to the comparator B register while counting down

Any combination of these events can generate either an interrupt.

PWM0 Interrupt Enable (PWM0INTEN)

Base 0x4002.8000
Offset 0x044
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved											IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
Type	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/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/W	0	Interrupt for Counter=Comparator B Down Value Description 1 A raw interrupt occurs when the counter matches the value in the PWMnCMPB register value while counting down. 0 No interrupt.
4	IntCmpBU	R/W	0	Interrupt for Counter=Comparator B Up Value Description 1 A raw interrupt occurs when the counter matches the value in the PWMnCMPB register value while counting up. 0 No interrupt.

Bit/Field	Name	Type	Reset	Description
3	IntCmpAD	R/W	0	<p>Interrupt for Counter=Comparator A Down</p> <p>Value Description</p> <p>1 A raw interrupt occurs when the counter matches the value in the PWMnCMPA register value while counting down.</p> <p>0 No interrupt.</p>
2	IntCmpAU	R/W	0	<p>Interrupt for Counter=Comparator A Up</p> <p>Value Description</p> <p>1 A raw interrupt occurs when the counter matches the value in the PWMnCMPA register value while counting up.</p> <p>0 No interrupt.</p>
1	IntCntLoad	R/W	0	<p>Interrupt for Counter=Load</p> <p>Value Description</p> <p>1 A raw interrupt occurs when the counter matches the value in the PWMnLOAD register value.</p> <p>0 No interrupt.</p>
0	IntCntZero	R/W	0	<p>Interrupt for Counter=0</p> <p>Value Description</p> <p>1 A raw interrupt occurs when the counter is zero.</p> <p>0 No interrupt.</p>

Register 16: PWM0 Raw Interrupt Status (PWM0RIS), offset 0x048**Register 17: PWM1 Raw Interrupt Status (PWM1RIS), offset 0x088****Register 18: PWM2 Raw Interrupt Status (PWM2RIS), offset 0x0C8**

These registers provide the current set of interrupt sources that are asserted, regardless of whether they cause an interrupt to be asserted to the controller (**PWM0RIS** controls the PWM generator 0 block, and so on). Bits set to 1 indicate the latched events that have occurred; bits set to 0 indicate 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	
	reserved																
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	
	reserved											IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
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	

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	RO	0	Comparator B Down Interrupt Status Indicates that the counter has matched the comparator B value while counting down.
4	IntCmpBU	RO	0	Comparator B Up Interrupt Status Indicates that the counter has matched the comparator B value while counting up.
3	IntCmpAD	RO	0	Comparator A Down Interrupt Status Indicates that the counter has matched the comparator A value while counting down.
2	IntCmpAU	RO	0	Comparator A Up Interrupt Status Indicates that the counter has matched the comparator A value while counting up.
1	IntCntLoad	RO	0	Counter=Load Interrupt Status Indicates that the counter has matched the PWMnLOAD register.
0	IntCntZero	RO	0	Counter=0 Interrupt Status Indicates that the counter has matched 0.

Register 19: PWM0 Interrupt Status and Clear (PWM0ISC), offset 0x04C

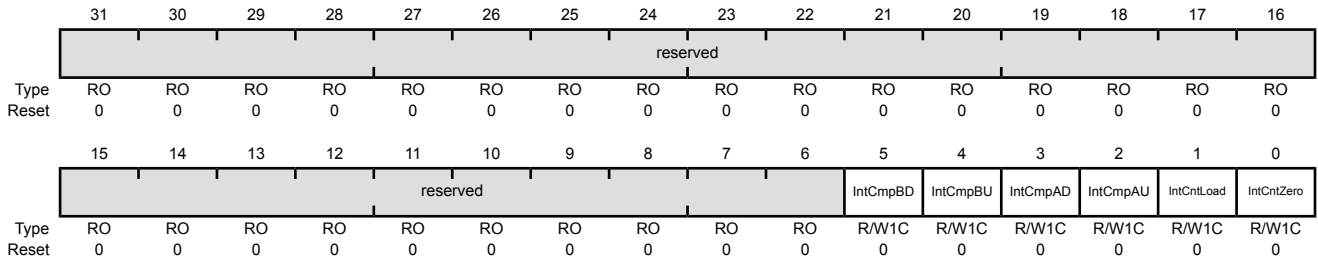
Register 20: PWM1 Interrupt Status and Clear (PWM1ISC), offset 0x08C

Register 21: PWM2 Interrupt Status and Clear (PWM2ISC), offset 0x0CC

These registers provide the current set of interrupt sources that are asserted to the controller (**PWM0ISC** controls the PWM generator 0 block, and so on). Bits set to 1 indicate the latched events that have occurred; bits set to 0 indicate 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)

Base 0x4002.8000
 Offset 0x04C
 Type R/W1C, reset 0x0000.0000



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	Counter=Load Interrupt Indicates that the counter has matched the PWMnLOAD register.
0	IntCntZero	R/W1C	0	Counter=0 Interrupt Indicates that the counter has matched 0.

Register 22: PWM0 Load (PWM0LOAD), offset 0x050**Register 23: PWM1 Load (PWM1LOAD), offset 0x090****Register 24: PWM2 Load (PWM2LOAD), offset 0x0D0**

These registers contain the load value for the PWM counter (**PWM0LOAD** controls the PWM generator 0 block, and so on). 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	Load															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	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	Load	R/W	0	Counter Load Value The counter load value.

Register 25: PWM0 Counter (PWM0COUNT), offset 0x054

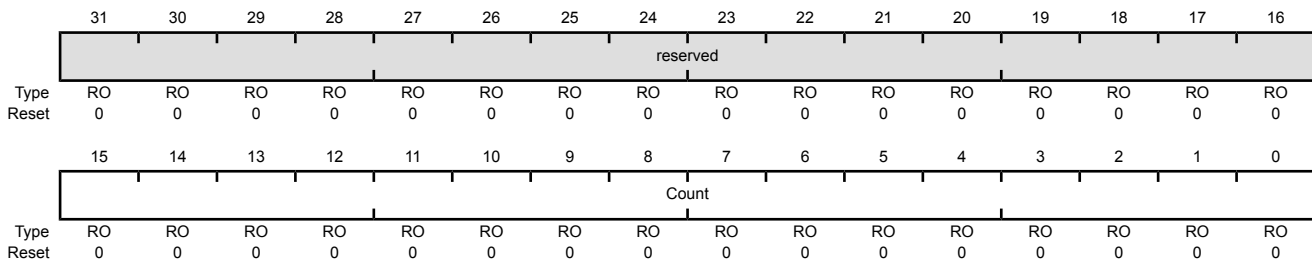
Register 26: PWM1 Counter (PWM1COUNT), offset 0x094

Register 27: PWM2 Counter (PWM2COUNT), offset 0x0D4

These registers contain the current value of the PWM counter (**PWM0COUNT** is the value of the PWM generator 0 block, and so on). 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



Bit/Field	Name	Type	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	Count	RO	0x00	Counter Value The current value of the counter.

Register 28: PWM0 Compare A (PWM0CMPA), offset 0x058**Register 29: PWM1 Compare A (PWM1CMPA), offset 0x098****Register 30: PWM2 Compare A (PWM2CMPA), offset 0x0D8**

These registers contain a value to be compared against the counter (**PWM0CMPA** controls the PWM generator 0 block, and so on). 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), 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

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	CompA															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	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 31: PWM0 Compare B (PWM0CMPB), offset 0x05C

Register 32: PWM1 Compare B (PWM1CMPB), offset 0x09C

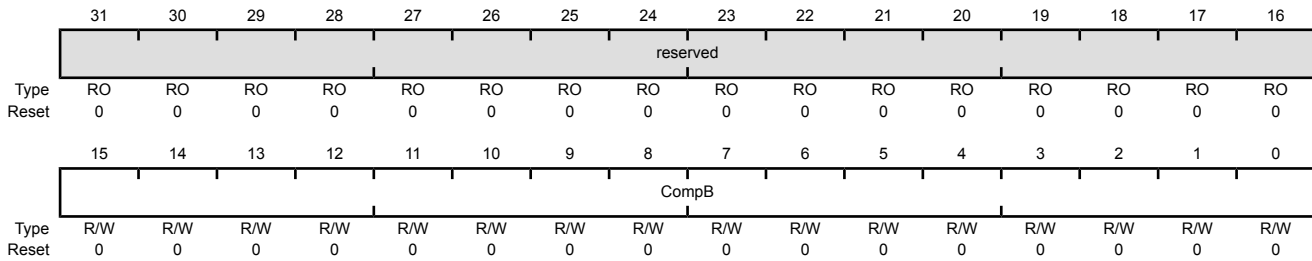
Register 33: PWM2 Compare B (PWM2CMPB), offset 0x0DC

These registers contain a value to be compared against the counter (**PWM0CMPB** controls the PWM generator 0 block, and so on). 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, no pulse is ever output.

If the comparator B update mode is immediate (based on the **CmpBUpd** bit in the **PWMnCTL** register), this 16-bit **CompB** 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
 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	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	CompB	R/W	0x00	Comparator B Value The value to be compared against the counter.

Register 34: PWM0 Generator A Control (PWM0GENA), offset 0x060**Register 35: PWM1 Generator A Control (PWM1GENA), offset 0x0A0****Register 36: PWM2 Generator A Control (PWM2GENA), offset 0x0E0**

These registers control the generation of the PWM_nA 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 (**PWM0GENA** controls the PWM generator 0 block, and so on). 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; **PWM1GENA**, the $PWM1A$ signal; and **PWM2GENA**, the $PWM2A$ 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.

PWM0 Generator A Control (PWM0GENA)

Base 0x4002.8000
Offset 0x060
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	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	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.

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										
9:8	ActCmpBU	R/W	0x0	<p>Action for Comparator B Up</p> <p>The action to be taken when the counter matches comparator B while counting up. Occurs only when the <code>Mode</code> bit in the PWMnCTL register (see page 417) is set to 1.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
Value	Description													
0x0	Do nothing.													
0x1	Invert the output signal.													
0x2	Set the output signal to 0.													
0x3	Set the output signal to 1.													
7:6	ActCmpAD	R/W	0x0	<p>Action for Comparator A Down</p> <p>The action to be taken when the counter matches comparator A while counting down.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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	<p>Action for Comparator A Up</p> <p>The action to be taken when the counter matches comparator A while counting up. Occurs only when the <code>Mode</code> bit in the PWMnCTL register is set to 1.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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	<p>Action for Counter=Load</p> <p>The action to be taken when the counter matches the load value.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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 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.

Register 37: PWM0 Generator B Control (PWM0GENB), offset 0x064

Register 38: PWM1 Generator B Control (PWM1GENB), offset 0x0A4

Register 39: PWM2 Generator B Control (PWM2GENB), offset 0x0E4

These registers control the generation of the PWM_nB 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 (**PWM0GENB** controls the PWM generator 0 block, and so on). 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; **PWM1GENB**, the $PWM1B$ signal; and **PWM2GENB**, the $PWM2B$ 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.

PWM0 Generator B Control (PWM0GENB)

Base 0x4002.8000
 Offset 0x064
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	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	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.

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										
9:8	ActCmpBU	R/W	0x0	<p>Action for Comparator B Up</p> <p>The action to be taken when the counter matches comparator B while counting up. Occurs only when the <code>Mode</code> bit in the PWMnCTL register is set to 1.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
Value	Description													
0x0	Do nothing.													
0x1	Invert the output signal.													
0x2	Set the output signal to 0.													
0x3	Set the output signal to 1.													
7:6	ActCmpAD	R/W	0x0	<p>Action for Comparator A Down</p> <p>The action to be taken when the counter matches comparator A while counting down.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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	<p>Action for Comparator A Up</p> <p>The action to be taken when the counter matches comparator A while counting up. Occurs only when the <code>Mode</code> bit in the PWMnCTL register is set to 1.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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	<p>Action for Counter=Load</p> <p>The action to be taken when the counter matches the load value.</p> <p>The table below defines the effect of the event on the output signal.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0x0</td> <td>Do nothing.</td> </tr> <tr> <td>0x1</td> <td>Invert the output signal.</td> </tr> <tr> <td>0x2</td> <td>Set the output signal to 0.</td> </tr> <tr> <td>0x3</td> <td>Set the output signal to 1.</td> </tr> </tbody> </table>	Value	Description	0x0	Do nothing.	0x1	Invert the output signal.	0x2	Set the output signal to 0.	0x3	Set the output signal to 1.
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.

Register 40: PWM0 Dead-Band Control (PWM0DBCTL), offset 0x068**Register 41: PWM1 Dead-Band Control (PWM1DBCTL), offset 0x0A8****Register 42: PWM2 Dead-Band Control (PWM2DBCTL), offset 0x0E8**

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). In a similar manner, **PWM2** and **PWM3** are produced from the **PWM1A** and **PWM1B** signals, and **PWM4** and **PWM5** are produced from the **PWM2A** and **PWM2B** signals.

PWM0 Dead-Band Control (PWM0DBCTL)

Base 0x4002.8000

Offset 0x068

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved															Enable
Type	RO	RO	RO	RO	RO	RO	RO	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

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 43: PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE), offset 0x06C

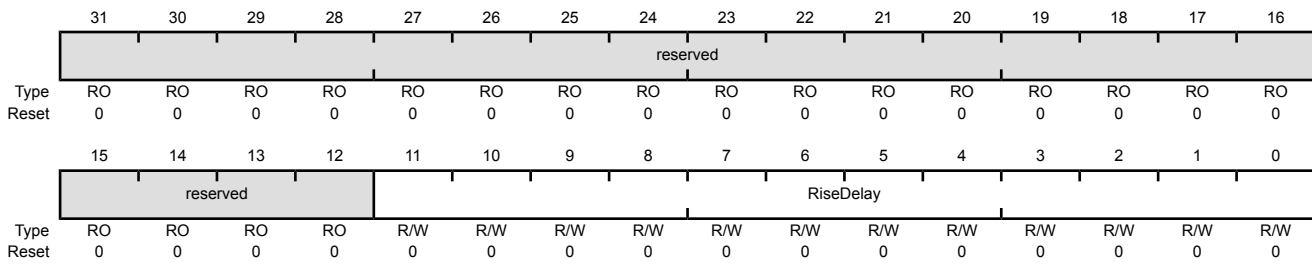
Register 44: PWM1 Dead-Band Rising-Edge Delay (PWM1DBRISE), offset 0x0AC

Register 45: PWM2 Dead-Band Rising-Edge Delay (PWM2DBRISE), offset 0x0EC

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. In a similar manner, **PWM2** is generated from **PWM1A** with its rising edge delayed and **PWM4** is produced from **PWM2A** with its rising edge delayed.

PWM0 Dead-Band Rising-Edge Delay (PWM0DBRISE)

Base 0x4002.8000
 Offset 0x06C
 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.
11:0	RiseDelay	R/W	0	Dead-Band Rise Delay The number of clock ticks to delay the rising edge.

Register 46: PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL), offset 0x070**Register 47: PWM1 Dead-Band Falling-Edge-Delay (PWM1DBFALL), offset 0x0B0****Register 48: PWM2 Dead-Band Falling-Edge-Delay (PWM2DBFALL), offset 0x0F0**

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. In a similar manner, **PWM3** is generated from **PWM1A** with its falling edge delayed and **PWM5** is produced from **PWM2A** with its falling edge delayed.

PWM0 Dead-Band Falling-Edge-Delay (PWM0DBFALL)

Base 0x4002.8000
Offset 0x070
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved				FallDelay											
Type	RO	RO	RO	RO	R/W	R/W	R/W	R/W	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	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.
11:0	FallDelay	R/W	0x00	Dead-Band Fall Delay The number of clock ticks to delay the falling edge.

17 Quadrature Encoder Interface (QEI)

A quadrature encoder, also known as a 2-channel incremental encoder, converts linear displacement into a pulse signal. By monitoring both the number of pulses and the relative phase of the two signals, you can track the position, direction of rotation, and speed. In addition, a third channel, or index signal, can be used to reset the position counter.

The LM3S1960 microcontroller includes two quadrature encoder interface (QEI) modules. Each QEI module interprets the code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

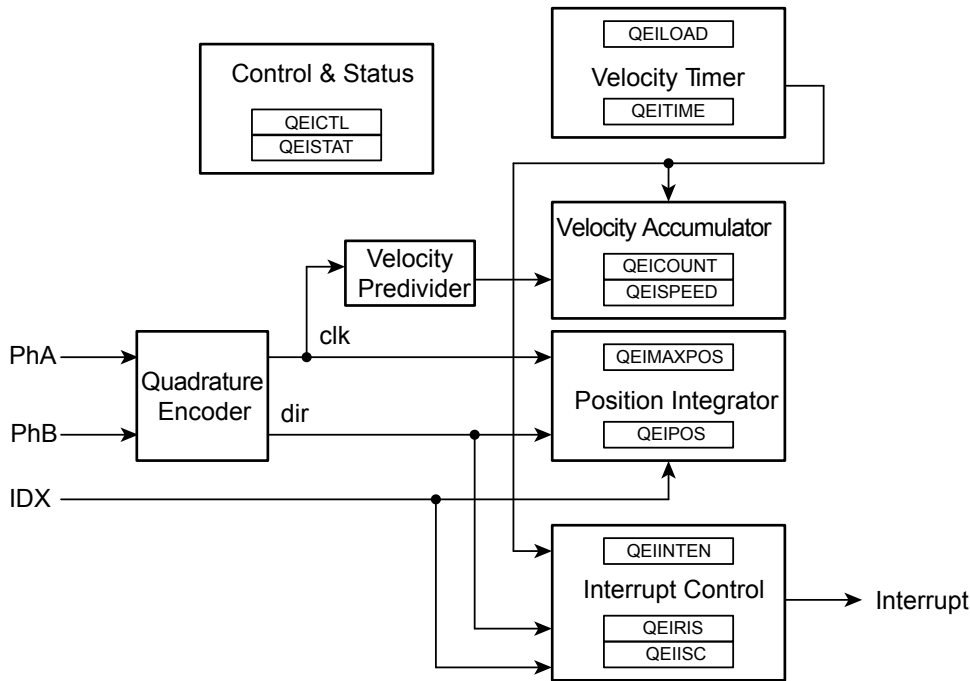
Each Stellaris[®] quadrature encoder has the following features:

- Two QEI modules, each with the following features:
 - Position integrator that tracks the encoder position
 - Velocity capture using built-in timer
 - The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 12.5 MHz for a 50-MHz system)
 - Interrupt generation on:
 - Index pulse
 - Velocity-timer expiration
 - Direction change
 - Quadrature error detection

17.1 Block Diagram

Figure 17-1 on page 437 provides a block diagram of a Stellaris[®] QEI module.

Figure 17-1. QEI Block Diagram



17.2 Functional Description

The QEI module interprets the two-bit gray code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

The position integrator and velocity capture can be independently enabled, though the position integrator must be enabled before the velocity capture can be enabled. The two phase signals, PhA and PhB , can be swapped before being interpreted by the QEI module to change the meaning of forward and backward, and to correct for miswiring of the system. Alternatively, the phase signals can be interpreted as a clock and direction signal as output by some encoders.

The QEI module supports two modes of signal operation: quadrature phase mode and clock/direction mode. In quadrature phase mode, the encoder produces two clocks that are 90 degrees out of phase; the edge relationship is used to determine the direction of rotation. In clock/direction mode, the encoder produces a clock signal to indicate steps and a direction signal to indicate the direction of rotation. This mode is determined by the $SigMode$ bit of the **QEI Control (QEICTL)** register (see page 441).

When the QEI module is set to use the quadrature phase mode ($SigMode$ bit equals zero), the capture mode for the position integrator can be set to update the position counter on every edge of the PhA signal or to update on every edge of both PhA and PhB . Updating the position counter on every PhA and PhB provides more positional resolution at the cost of less range in the positional counter.

When edges on PhA lead edges on PhB , the position counter is incremented. When edges on PhB lead edges on PhA , the position counter is decremented. When a rising and falling edge pair is seen on one of the phases without any edges on the other, the direction of rotation has changed.

The positional counter is automatically reset on one of two conditions: sensing the index pulse or reaching the maximum position value. Which mode is determined by the `ResMode` bit of the **QEI Control (QEICTL)** register.

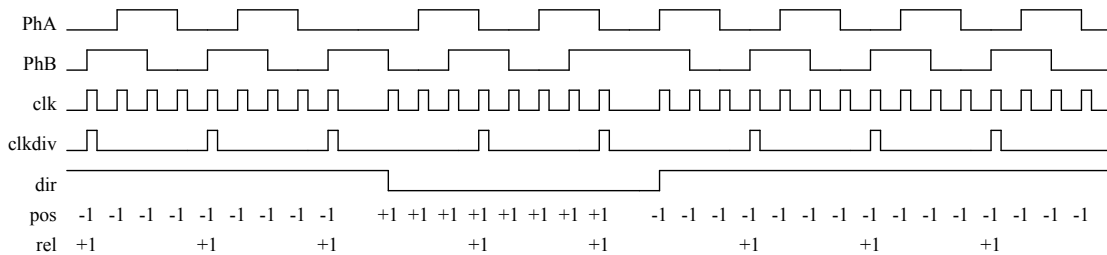
When `ResMode` is 1, the positional counter is reset when the index pulse is sensed. This limits the positional counter to the values `[0:N-1]`, where `N` is the number of phase edges in a full revolution of the encoder wheel. The **QEIMAXPOS** register must be programmed with `N-1` so that the reverse direction from position 0 can move the position counter to `N-1`. In this mode, the position register contains the absolute position of the encoder relative to the index (or home) position once an index pulse has been seen.

When `ResMode` is 0, the positional counter is constrained to the range `[0:M]`, where `M` is the programmable maximum value. The index pulse is ignored by the positional counter in this mode.

The velocity capture has a configurable timer and a count register. It counts the number of phase edges (using the same configuration as for the position integrator) in a given time period. The edge count from the previous time period is available to the controller via the **QEISPEED** register, while the edge count for the current time period is being accumulated in the **QEICOUNT** register. As soon as the current time period is complete, the total number of edges counted in that time period is made available in the **QEISPEED** register (losing the previous value), the **QEICOUNT** is reset to 0, and counting commences on a new time period. The number of edges counted in a given time period is directly proportional to the velocity of the encoder.

Figure 17-2 on page 438 shows how the Stellaris® quadrature encoder converts the phase input signals into clock pulses, the direction signal, and how the velocity predivider operates (in Divide by 4 mode).

Figure 17-2. Quadrature Encoder and Velocity Predivider Operation



The period of the timer is configurable by specifying the load value for the timer in the **QEILOAD** register. When the timer reaches zero, an interrupt can be triggered, and the hardware reloads the timer with the **QEILOAD** value and continues to count down. At lower encoder speeds, a longer timer period is needed to be able to capture enough edges to have a meaningful result. At higher encoder speeds, both a shorter timer period and/or the velocity predivider can be used.

The following equation converts the velocity counter value into an rpm value:

$$\text{rpm} = (\text{clock} * (2 ^ \text{VelDiv}) * \text{Speed} * 60) \div (\text{Load} * \text{ppr} * \text{edges})$$

where:

`clock` is the controller clock rate

`ppr` is the number of pulses per revolution of the physical encoder

`edges` is 2 or 4, based on the capture mode set in the **QEICTL** register (2 for `CapMode` set to 0 and 4 for `CapMode` set to 1)

For example, consider a motor running at 600 rpm. A 2048 pulse per revolution quadrature encoder is attached to the motor, producing 8192 phase edges per revolution. With a velocity predivider of $\div 1$ (`VelDiv` set to 0) and clocking on both `PhA` and `PhB` edges, this results in 81,920 pulses per second (the motor turns 10 times per second). If the timer were clocked at 10,000 Hz, and the load value was 2,500 ($\frac{1}{4}$ of a second), it would count 20,480 pulses per update. Using the above equation:

$$\text{rpm} = (10000 * 1 * 20480 * 60) \div (2500 * 2048 * 4) = 600 \text{ rpm}$$

Now, consider that the motor is sped up to 3000 rpm. This results in 409,600 pulses per second, or 102,400 every $\frac{1}{4}$ of a second. Again, the above equation gives:

$$\text{rpm} = (10000 * 1 * 102400 * 60) \div (2500 * 2048 * 4) = 3000 \text{ rpm}$$

Care must be taken when evaluating this equation since intermediate values may exceed the capacity of a 32-bit integer. In the above examples, the clock is 10,000 and the divider is 2,500; both could be predivided by 100 (at compile time if they are constants) and therefore be 100 and 25. In fact, if they were compile-time constants, they could also be reduced to a simple multiply by 4, cancelled by the $\div 4$ for the edge-count factor.

Important: Reducing constant factors at compile time is the best way to control the intermediate values of this equation, as well as reducing the processing requirement of computing this equation.

The division can be avoided by selecting a timer load value such that the divisor is a power of 2; a simple shift can therefore be done in place of the division. For encoders with a power of 2 pulses per revolution, this is a simple matter of selecting a power of 2 load value. For other encoders, a load value must be selected such that the product is very close to a power of two. For example, a 100 pulse per revolution encoder could use a load value of 82, resulting in 32,800 as the divisor, which is 0.09% above 2^{14} ; in this case a shift by 15 would be an adequate approximation of the divide in most cases. If absolute accuracy were required, the controller's divide instruction could be used.

The QEI module can produce a controller interrupt on several events: phase error, direction change, reception of the index pulse, and expiration of the velocity timer. Standard masking, raw interrupt status, interrupt status, and interrupt clear capabilities are provided.

17.3 Initialization and Configuration

The following example shows how to configure the Quadrature Encoder module to read back an absolute position:

1. Enable the QEI clock by writing a value of 0x0000.0100 to the **RCGC1** register in the System Control module.
2. 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.
4. Configure the quadrature encoder to capture edges on both signals and maintain an absolute position by resetting on index pulses. Using a 1000-line encoder at four edges per line, there are 4000 pulses per revolution; therefore, set the maximum position to 3999 (0xF9F) since the count is zero-based.

- Write the **QEICTL** register with the value of 0x0000.0018.
 - Write the **QEIMAXPOS** register with the value of 0x0000.0F9F.
5. Enable the quadrature encoder by setting bit 0 of the **QEICTL** register.
 6. Delay for some time.
 7. Read the encoder position by reading the **QEIPOS** register value.

17.4 Register Map

Table 17-1 on page 440 lists the QEI registers. The offset listed is a hexadecimal increment to the register's address, relative to the module's base address:

- QEI0: 0x4002.C000
- QEI1: 0x4002.D000

Table 17-1. QEI Register Map

Offset	Name	Type	Reset	Description	See page
0x000	QEICTL	R/W	0x0000.0000	QEI Control	441
0x004	QEISTAT	RO	0x0000.0000	QEI Status	443
0x008	QEIPOS	R/W	0x0000.0000	QEI Position	444
0x00C	QEIMAXPOS	R/W	0x0000.0000	QEI Maximum Position	445
0x010	QEILOAD	R/W	0x0000.0000	QEI Timer Load	446
0x014	QEITIME	RO	0x0000.0000	QEI Timer	447
0x018	QEICOUNT	RO	0x0000.0000	QEI Velocity Counter	448
0x01C	QEISPEED	RO	0x0000.0000	QEI Velocity	449
0x020	QEIINTEN	R/W	0x0000.0000	QEI Interrupt Enable	450
0x024	QEIRIS	RO	0x0000.0000	QEI Raw Interrupt Status	451
0x028	QEIISC	R/W1C	0x0000.0000	QEI Interrupt Status and Clear	452

17.5 Register Descriptions

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

Register 1: QEI Control (QEICTL), offset 0x000

This register contains the configuration of the QEI module. Separate enables are provided for the quadrature encoder and the velocity capture blocks; the quadrature encoder must be enabled in order to capture the velocity, but the velocity does not need to be captured in applications that do not need it. The phase signal interpretation, phase swap, Position Update mode, Position Reset mode, and velocity predivider are all set via this register.

QEI Control (QEICTL)

QEI0 base: 0x4002.C000

QEI1 base: 0x4002.D000

Offset 0x000

Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
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
	reserved			STALLEN	INVI	INVB	INVA	VelDiv			VelEn	ResMode	CapMode	SigMode	Swap	Enable
Type	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	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	Type	Reset	Description																		
31:13	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.																		
12	STALLEN	R/W	0	Stall QEI When set, the QEI stalls when the microcontroller asserts Halt.																		
11	INVI	R/W	0	Invert Index Pulse When set, the input Index Pulse is inverted.																		
10	INVB	R/W	0	Invert PhB When set, the PhB input is inverted.																		
9	INVA	R/W	0	Invert PhA When set, the PhA input is inverted.																		
8:6	VelDiv	R/W	0x0	Predivide Velocity A predivider of the input quadrature pulses before being applied to the QEICOUNT accumulator. This field can be set to the following values: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Predivider</th> </tr> </thead> <tbody> <tr><td>0x0</td><td>÷1</td></tr> <tr><td>0x1</td><td>÷2</td></tr> <tr><td>0x2</td><td>÷4</td></tr> <tr><td>0x3</td><td>÷8</td></tr> <tr><td>0x4</td><td>÷16</td></tr> <tr><td>0x5</td><td>÷32</td></tr> <tr><td>0x6</td><td>÷64</td></tr> <tr><td>0x7</td><td>÷128</td></tr> </tbody> </table>	Value	Predivider	0x0	÷1	0x1	÷2	0x2	÷4	0x3	÷8	0x4	÷16	0x5	÷32	0x6	÷64	0x7	÷128
Value	Predivider																					
0x0	÷1																					
0x1	÷2																					
0x2	÷4																					
0x3	÷8																					
0x4	÷16																					
0x5	÷32																					
0x6	÷64																					
0x7	÷128																					

Bit/Field	Name	Type	Reset	Description
5	VelEn	R/W	0	Capture Velocity When set, enables capture of the velocity of the quadrature encoder.
4	ResMode	R/W	0	Reset Mode The Reset mode for the position counter. When 0, the position counter is reset when it reaches the maximum; when 1, the position counter is reset when the index pulse is captured.
3	CapMode	R/W	0	Capture Mode The Capture mode defines the phase edges that are counted in the position. When 0, only the PhA edges are counted; when 1, the PhA and PhB edges are counted, providing twice the positional resolution but half the range.
2	SigMode	R/W	0	Signal Mode When 1, the PhA and PhB signals are clock and direction; when 0, they are quadrature phase signals.
1	Swap	R/W	0	Swap Signals Swaps the PhA and PhB signals.
0	Enable	R/W	0	Enable QEI Enables the quadrature encoder module.

Register 2: QEI Status (QEISTAT), offset 0x004

This register provides status about the operation of the QEI module.

QEI Status (QEISTAT)

QEI0 base: 0x4002.C000

QEI1 base: 0x4002.D000

Offset 0x004

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved															Direction	Error
Type	RO	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	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	Direction	RO	0	Direction of Rotation Indicates the direction the encoder is rotating. The <code>Direction</code> values are defined as follows: <table border="0"> <tr> <td>Value</td> <td>Description</td> </tr> <tr> <td>0</td> <td>Forward rotation</td> </tr> <tr> <td>1</td> <td>Reverse rotation</td> </tr> </table>	Value	Description	0	Forward rotation	1	Reverse rotation
Value	Description									
0	Forward rotation									
1	Reverse rotation									
0	Error	RO	0	Error Detected Indicates that an error was detected in the gray code sequence (that is, both signals changing at the same time).						

Register 3: QEI Position (QEIP0S), offset 0x008

This register contains the current value of the position integrator. Its value is updated by inputs on the QEI phase inputs, and can be set to a specific value by writing to it.

QEI Position (QEIP0S)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x008
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Position															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Position															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	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	Type	Reset	Description
31:0	Position	R/W	0x00	Current Position Integrator Value The current value of the position integrator.

Register 4: QEI Maximum Position (QEIMAXPOS), offset 0x00C

This register contains the maximum value of the position integrator. When moving forward, the position register resets to zero when it increments past this value. When moving backward, the position register resets to this value when it decrements from zero.

QEI Maximum Position (QEIMAXPOS)

QEI0 base: 0x4002.C000

QEI1 base: 0x4002.D000

Offset 0x00C

Type R/W, reset 0x0000.0000

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

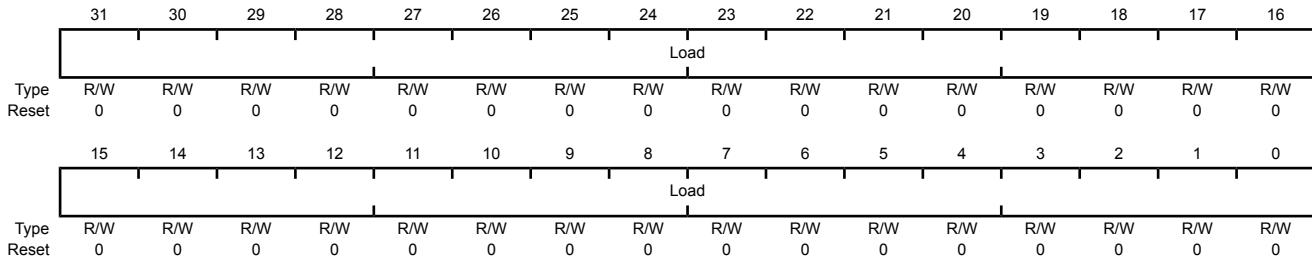
Bit/Field	Name	Type	Reset	Description
31:0	MaxPos	R/W	0x00	Maximum Position Integrator Value The maximum value of the position integrator.

Register 5: QEI Timer Load (QEILOAD), offset 0x010

This register contains the load value for the velocity timer. Since this value is loaded into the timer the clock cycle after the timer is zero, this value should be one less than the number of clocks in the desired period. So, for example, to have 2000 clocks per timer period, this register should contain 1999.

QEI Timer Load (QEILOAD)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x010
 Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:0	Load	R/W	0x00	Velocity Timer Load Value The load value for the velocity timer.

Register 6: QEI Timer (QEITIME), offset 0x014

This register contains the current value of the velocity timer. This counter does not increment when `VelEn` in `QEICTL` is 0.

QEI Timer (QEITIME)

QEI0 base: 0x4002.C000

QEI1 base: 0x4002.D000

Offset 0x014

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Time															
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
	Time															
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

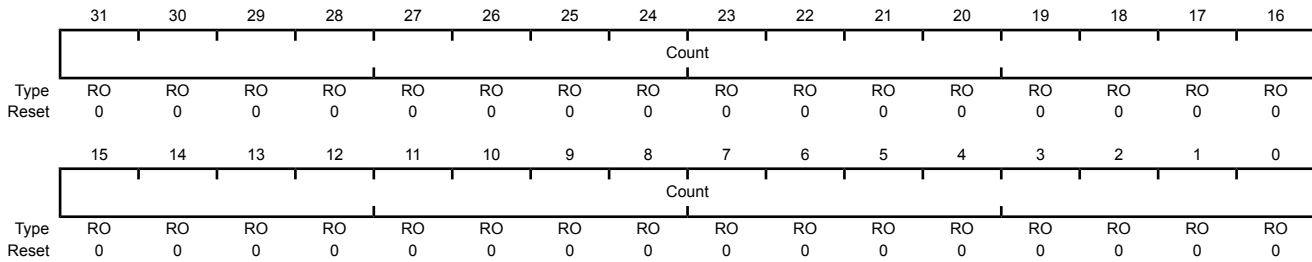
Bit/Field	Name	Type	Reset	Description
31:0	Time	RO	0x00	Velocity Timer Current Value The current value of the velocity timer.

Register 7: QEI Velocity Counter (QEICOUNT), offset 0x018

This register contains the running count of velocity pulses for the current time period. Since this is a running total, the time period to which it applies cannot be known with precision (that is, a read of this register does not necessarily correspond to the time returned by the **QEITIME** register since there is a small window of time between the two reads, during which time either value may have changed). The **QEISPEED** register should be used to determine the actual encoder velocity; this register is provided for information purposes only. This counter does not increment when `VelEn` in **QEICTL** is 0.

QEI Velocity Counter (QEICOUNT)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x018
 Type RO, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:0	Count	RO	0x00	Velocity Pulse Count The running total of encoder pulses during this velocity timer period.

Register 8: QEI Velocity (QEISPEED), offset 0x01C

This register contains the most recently measured velocity of the quadrature encoder. This corresponds to the number of velocity pulses counted in the previous velocity timer period. This register does not update when `VelEn` in `QEICTL` is 0.

QEI Velocity (QEISPEED)

QEI0 base: 0x4002.C000

QEI1 base: 0x4002.D000

Offset 0x01C

Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Speed															
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
	Speed															
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

Bit/Field	Name	Type	Reset	Description
31:0	Speed	RO	0x00	Velocity

The measured speed of the quadrature encoder in pulses per period.

Register 9: QEI Interrupt Enable (QEIINTEN), offset 0x020

This register contains enables for each of the QEI module's interrupts. An interrupt is asserted to the controller if its corresponding bit in this register is set to 1.

QEI Interrupt Enable (QEIINTEN)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x020
 Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													IntError	IntDir	IntTimer	IntIndex
Type	RO	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	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	IntError	R/W	0	Phase Error Interrupt Enable When 1, an interrupt occurs when a phase error is detected.
2	IntDir	R/W	0	Direction Change Interrupt Enable When 1, an interrupt occurs when the direction changes.
1	IntTimer	R/W	0	Timer Expires Interrupt Enable When 1, an interrupt occurs when the velocity timer expires.
0	IntIndex	R/W	0	Index Pulse Detected Interrupt Enable When 1, an interrupt occurs when the index pulse is detected.

Register 10: QEI Raw Interrupt Status (QEIRIS), offset 0x024

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 (this is set through the **QEINTEN** register). Bits set to 1 indicate the latched events that have occurred; a zero bit indicates that the event in question has not occurred.

QEI Raw Interrupt Status (QEIRIS)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x024
 Type RO, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													IntError	IntDir	IntTimer	IntIndex
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	

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	IntError	RO	0	Phase Error Detected Indicates that a phase error was detected.
2	IntDir	RO	0	Direction Change Detected Indicates that the direction has changed.
1	IntTimer	RO	0	Velocity Timer Expired Indicates that the velocity timer has expired.
0	IntIndex	RO	0	Index Pulse Asserted Indicates that the index pulse has occurred.

Register 11: QEI Interrupt Status and Clear (QEIISC), offset 0x028

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 zero bit indicates that the event in question has not occurred. This is a R/W1C register; writing a 1 to a bit position clears the corresponding interrupt reason.

QEI Interrupt Status and Clear (QEIISC)

QEI0 base: 0x4002.C000
 QEI1 base: 0x4002.D000
 Offset 0x028
 Type R/W1C, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
	reserved																
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	
	reserved													IntError	IntDir	IntTimer	IntIndex
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W1C	R/W1C	R/W1C	R/W1C
Reset	0	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	IntError	R/W1C	0	Phase Error Interrupt Indicates that a phase error was detected.
2	IntDir	R/W1C	0	Direction Change Interrupt Indicates that the direction has changed.
1	IntTimer	R/W1C	0	Velocity Timer Expired Interrupt Indicates that the velocity timer has expired.
0	IntIndex	R/W1C	0	Index Pulse Interrupt Indicates that the index pulse has occurred.

18 Pin Diagram

The LM3S1960 microcontroller pin diagrams are shown below.

Figure 18-1. 100-Pin LQFP Package Pin Diagram

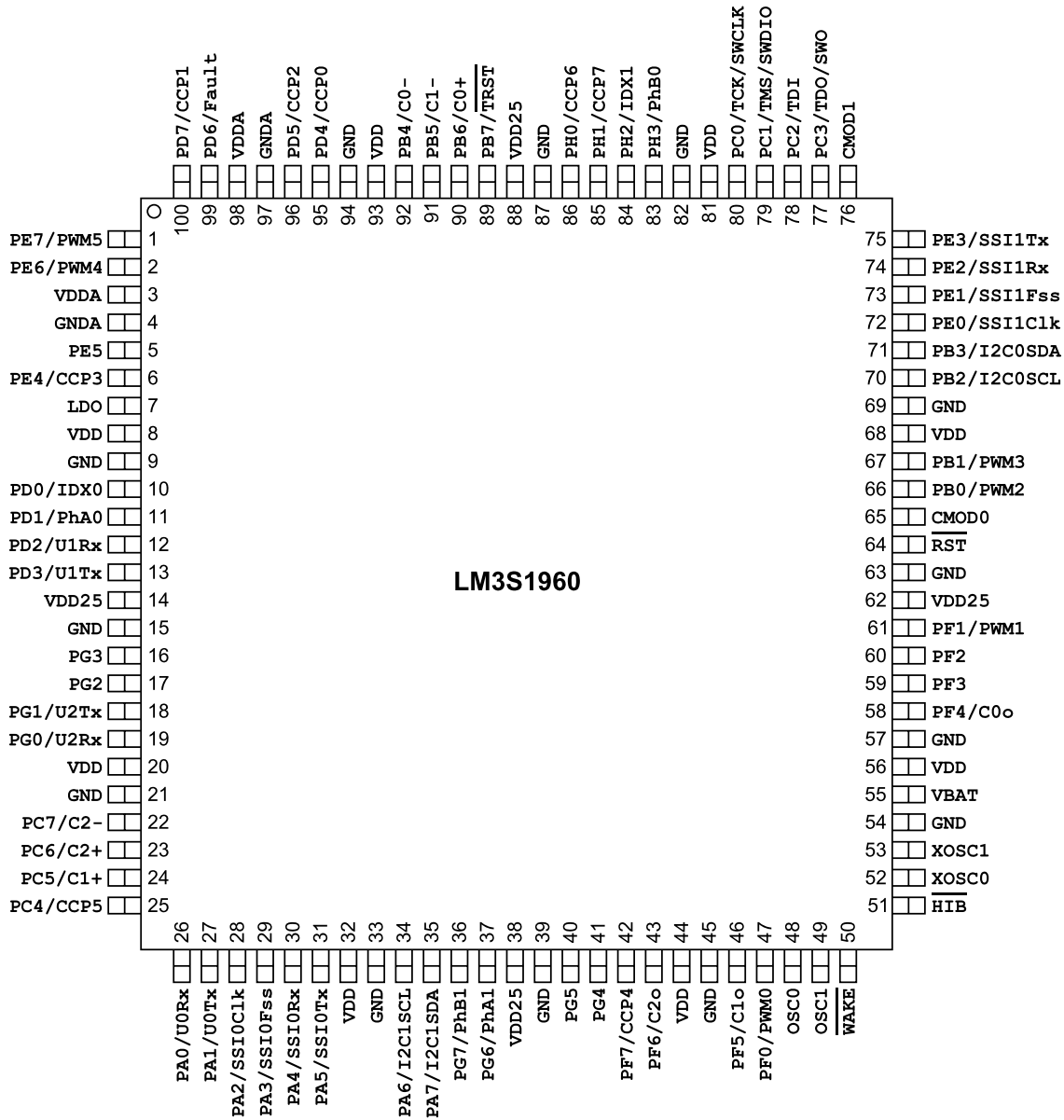
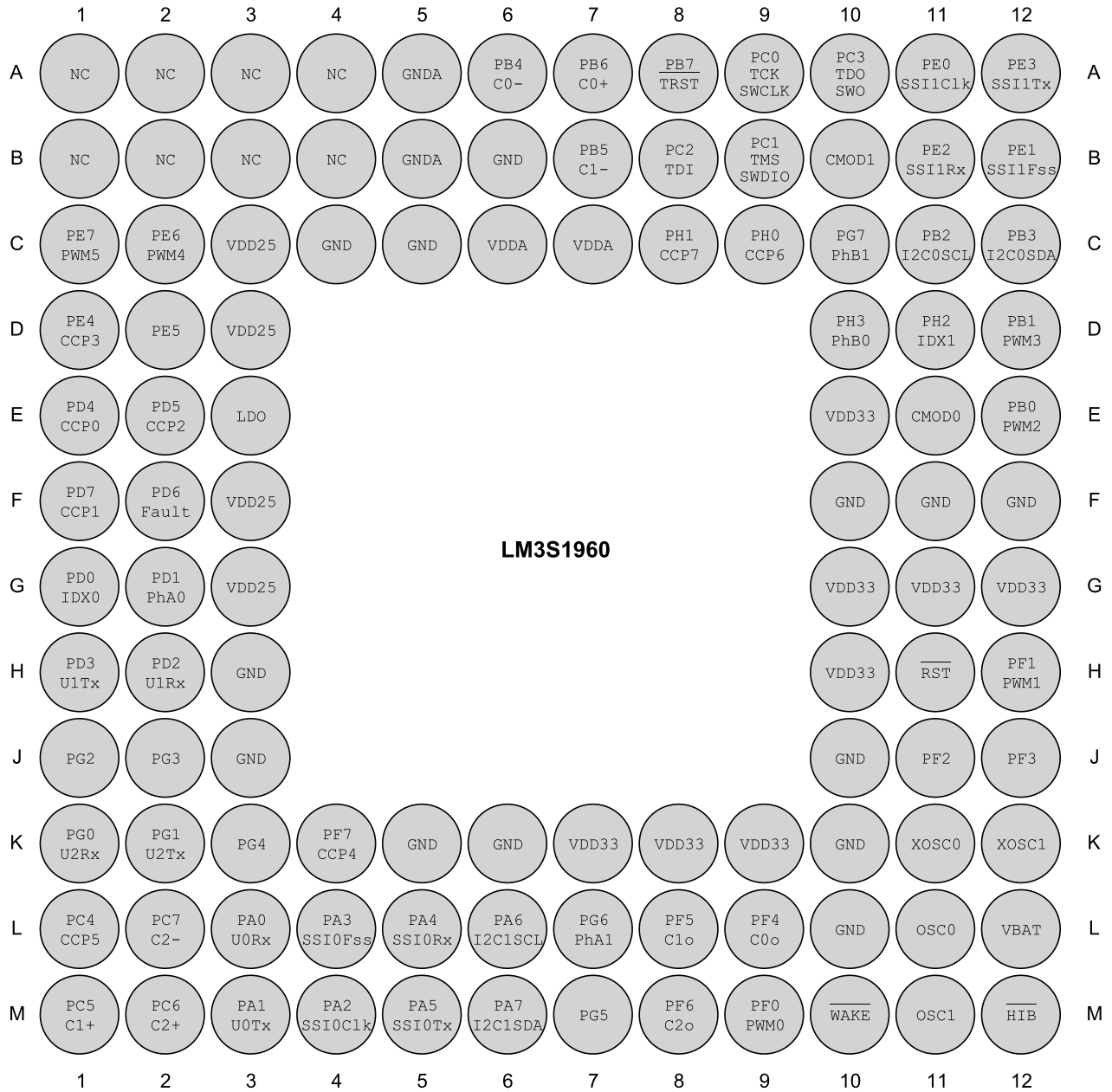


Figure 18-2. 108-Ball BGA Package Pin Diagram (Top View)



19 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 19-1 on page 455 shows the pin-to-signal-name mapping, including functional characteristics of the signals. Table 19-2 on page 459 lists the signals in alphabetical order by signal name.

Table 19-3 on page 464 groups the signals by functionality, except for GPIOs. Table 19-4 on page 466 lists the GPIO pins and their alternate functionality.

19.1 100-Pin LQFP Package Pin Tables

Table 19-1. Signals by Pin Number

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
1	PE7	I/O	TTL	GPIO port E bit 7.
	PWM5	O	TTL	PWM 5.
2	PE6	I/O	TTL	GPIO port E bit 6.
	PWM4	O	TTL	PWM 4.
3	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.
4	GNDA	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
5	PE5	I/O	TTL	GPIO port E bit 5.
6	PE4	I/O	TTL	GPIO port E bit 4.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
7	LDO	-	Power	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).
8	VDD	-	Power	Positive supply for I/O and some logic.
9	GND	-	Power	Ground reference for logic and I/O pins.
10	PD0	I/O	TTL	GPIO port D bit 0.
	IDX0	I	TTL	QE1 module 0 index.
11	PD1	I/O	TTL	GPIO port D bit 1.
	PhA0	I	TTL	QE1 module 0 phase A.
12	PD2	I/O	TTL	GPIO port D bit 2.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
13	PD3	I/O	TTL	GPIO port D bit 3.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.

Table 19-1. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
14	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
15	GND	-	Power	Ground reference for logic and I/O pins.
16	PG3	I/O	TTL	GPIO port G bit 3.
17	PG2	I/O	TTL	GPIO port G bit 2.
18	PG1	I/O	TTL	GPIO port G bit 1.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
19	PG0	I/O	TTL	GPIO port G bit 0.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
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 2 positive input.
24	PC5	I/O	TTL	GPIO port C bit 5.
	C1+	I	Analog	Analog comparator 1 positive input.
25	PC4	I/O	TTL	GPIO port C bit 4.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
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.
	U0Tx	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
28	PA2	I/O	TTL	GPIO port A bit 2.
	SSI0Clk	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.
	SSI0Tx	O	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.
	I2C1SCL	I/O	OD	I ² C module 1 clock.
35	PA7	I/O	TTL	GPIO port A bit 7.
	I2C1SDA	I/O	OD	I ² C module 1 data.
36	PG7	I/O	TTL	GPIO port G bit 7.
	PhB1	I	TTL	QEI module 1 phase B.

Table 19-1. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
37	PG6	I/O	TTL	GPIO port G bit 6.
	PhA1	I	TTL	QEI module 1 phase A.
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	PG5	I/O	TTL	GPIO port G bit 5.
41	PG4	I/O	TTL	GPIO port G bit 4.
42	PF7	I/O	TTL	GPIO port F bit 7.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
43	PF6	I/O	TTL	GPIO port F bit 6.
	C2o	O	TTL	Analog comparator 2 output.
44	VDD	-	Power	Positive supply for I/O and some logic.
45	GND	-	Power	Ground reference for logic and I/O pins.
46	PF5	I/O	TTL	GPIO port F bit 5.
	C1o	O	TTL	Analog comparator 1 output.
47	PF0	I/O	TTL	GPIO port F bit 0.
	PWM0	O	TTL	PWM 0.
48	OSC0	I	Analog	Main oscillator crystal input or an external clock reference input.
49	OSC1	O	Analog	Main oscillator crystal output.
50	WAKE	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
51	HIB	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.
52	XOSC0	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
53	XOSC1	O	Analog	Hibernation module oscillator crystal output.
54	GND	-	Power	Ground reference for logic and I/O pins.
55	VBAT	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
56	VDD	-	Power	Positive supply for I/O and some logic.
57	GND	-	Power	Ground reference for logic and I/O pins.
58	PF4	I/O	TTL	GPIO port F bit 4.
	C0o	O	TTL	Analog comparator 0 output.
59	PF3	I/O	TTL	GPIO port F bit 3.
60	PF2	I/O	TTL	GPIO port F bit 2.
61	PF1	I/O	TTL	GPIO port F bit 1.
	PWM1	O	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.

Table 19-1. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
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.
	PWM2	O	TTL	PWM 2.
67	PB1	I/O	TTL	GPIO port B bit 1.
	PWM3	O	TTL	PWM 3.
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	I ² C module 0 clock.
71	PB3	I/O	TTL	GPIO port B bit 3.
	I2C0SDA	I/O	OD	I ² C module 0 data.
72	PE0	I/O	TTL	GPIO port E bit 0.
	SSI1Clk	I/O	TTL	SSI module 1 clock.
73	PE1	I/O	TTL	GPIO port E bit 1.
	SSI1Fss	I/O	TTL	SSI module 1 frame.
74	PE2	I/O	TTL	GPIO port E bit 2.
	SSI1Rx	I	TTL	SSI module 1 receive.
75	PE3	I/O	TTL	GPIO port E bit 3.
	SSI1Tx	O	TTL	SSI module 1 transmit.
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	O	TTL	JTAG TDO and SWO.
	SWO	O	TTL	JTAG TDO and SWO.
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	PH3	I/O	TTL	GPIO port H bit 3.
	PhB0	I	TTL	QEI module 0 phase B.
84	PH2	I/O	TTL	GPIO port H bit 2.
	IDX1	I	TTL	QEI module 1 index.
85	PH1	I/O	TTL	GPIO port H bit 1.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.

Table 19-1. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
86	PH0	I/O	TTL	GPIO port H bit 0.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
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 TRST.
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.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
96	PD5	I/O	TTL	GPIO port D bit 5.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
97	GNDA	-	Power	The ground reference for the analog circuits (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.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-2. Signals by Signal Name

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
C0+	90	I	Analog	Analog comparator 0 positive input.
C0-	92	I	Analog	Analog comparator 0 negative input.
C0o	58	O	TTL	Analog comparator 0 output.
C1+	24	I	Analog	Analog comparator 1 positive input.
C1-	91	I	Analog	Analog comparator 1 negative input.
C1o	46	O	TTL	Analog comparator 1 output.
C2+	23	I	Analog	Analog comparator 2 positive input.
C2-	22	I	Analog	Analog comparator 2 negative input.
C2o	43	O	TTL	Analog comparator 2 output.

Table 19-2. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
CCP0	95	I/O	TTL	Capture/Compare/PWM 0.
CCP1	100	I/O	TTL	Capture/Compare/PWM 1.
CCP2	96	I/O	TTL	Capture/Compare/PWM 2.
CCP3	6	I/O	TTL	Capture/Compare/PWM 3.
CCP4	42	I/O	TTL	Capture/Compare/PWM 4.
CCP5	25	I/O	TTL	Capture/Compare/PWM 5.
CCP6	86	I/O	TTL	Capture/Compare/PWM 6.
CCP7	85	I/O	TTL	Capture/Compare/PWM 7.
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 15 21 33 39 45 54 57 63 69 82 87 94	-	Power	Ground reference for logic and I/O pins.
GNDA	4 97	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
HIB	51	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.
I2C0SCL	70	I/O	OD	I ² C module 0 clock.
I2C0SDA	71	I/O	OD	I ² C module 0 data.
I2C1SCL	34	I/O	OD	I ² C module 1 clock.
I2C1SDA	35	I/O	OD	I ² C module 1 data.
IDX0	10	I	TTL	QEI module 0 index.
IDX1	84	I	TTL	QEI module 1 index.
LDO	7	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 μ 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).
OSC0	48	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	49	O	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.

Table 19-2. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
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.
PA7	35	I/O	TTL	GPIO port A bit 7.
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.
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.
PE2	74	I/O	TTL	GPIO port E bit 2.
PE3	75	I/O	TTL	GPIO port E bit 3.
PE4	6	I/O	TTL	GPIO port E bit 4.
PE5	5	I/O	TTL	GPIO port E bit 5.
PE6	2	I/O	TTL	GPIO port E bit 6.
PE7	1	I/O	TTL	GPIO port E bit 7.
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.

Table 19-2. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
PF3	59	I/O	TTL	GPIO port F bit 3.
PF4	58	I/O	TTL	GPIO port F bit 4.
PF5	46	I/O	TTL	GPIO port F bit 5.
PF6	43	I/O	TTL	GPIO port F bit 6.
PF7	42	I/O	TTL	GPIO port F bit 7.
PG0	19	I/O	TTL	GPIO port G bit 0.
PG1	18	I/O	TTL	GPIO port G bit 1.
PG2	17	I/O	TTL	GPIO port G bit 2.
PG3	16	I/O	TTL	GPIO port G bit 3.
PG4	41	I/O	TTL	GPIO port G bit 4.
PG5	40	I/O	TTL	GPIO port G bit 5.
PG6	37	I/O	TTL	GPIO port G bit 6.
PG7	36	I/O	TTL	GPIO port G bit 7.
PH0	86	I/O	TTL	GPIO port H bit 0.
PH1	85	I/O	TTL	GPIO port H bit 1.
PH2	84	I/O	TTL	GPIO port H bit 2.
PH3	83	I/O	TTL	GPIO port H bit 3.
PhA0	11	I	TTL	QEI module 0 phase A.
PhA1	37	I	TTL	QEI module 1 phase A.
PhB0	83	I	TTL	QEI module 0 phase B.
PhB1	36	I	TTL	QEI module 1 phase B.
PWM0	47	O	TTL	PWM 0.
PWM1	61	O	TTL	PWM 1.
PWM2	66	O	TTL	PWM 2.
PWM3	67	O	TTL	PWM 3.
PWM4	2	O	TTL	PWM 4.
PWM5	1	O	TTL	PWM 5.
RST	64	I	TTL	System reset input.
SSI0Clk	28	I/O	TTL	SSI module 0 clock.
SSI0Fss	29	I/O	TTL	SSI module 0 frame.
SSI0Rx	30	I	TTL	SSI module 0 receive.
SSI0Tx	31	O	TTL	SSI module 0 transmit.
SSI1Clk	72	I/O	TTL	SSI module 1 clock.
SSI1Fss	73	I/O	TTL	SSI module 1 frame.
SSI1Rx	74	I	TTL	SSI module 1 receive.
SSI1Tx	75	O	TTL	SSI module 1 transmit.
SWCLK	80	I	TTL	JTAG/SWD CLK.
SWDIO	79	I/O	TTL	JTAG TMS and SWDIO.
SWO	77	O	TTL	JTAG TDO and SWO.
TCK	80	I	TTL	JTAG/SWD CLK.
TDI	78	I	TTL	JTAG TDI.

Table 19-2. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
TDO	77	O	TTL	JTAG TDO and SWO.
TMS	79	I/O	TTL	JTAG TMS and SWDIO.
$\overline{\text{TRST}}$	89	I	TTL	JTAG $\overline{\text{TRST}}$.
U0Rx	26	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	27	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1Rx	12	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	13	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
U2Rx	19	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	18	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
VBAT	55	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
VDD	8 20 32 44 56 68 81 93	-	Power	Positive supply for I/O and some logic.
VDD25	14 38 62 88	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
VDDA	3 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.
$\overline{\text{WAKE}}$	50	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	52	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
XOSC1	53	O	Analog	Hibernation module oscillator crystal output.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-3. Signals by Function, Except for GPIO

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
Analog Comparators	C0+	90	I	Analog	Analog comparator 0 positive input.
	C0-	92	I	Analog	Analog comparator 0 negative input.
	C0o	58	O	TTL	Analog comparator 0 output.
	C1+	24	I	Analog	Analog comparator 1 positive input.
	C1-	91	I	Analog	Analog comparator 1 negative input.
	C1o	46	O	TTL	Analog comparator 1 output.
	C2+	23	I	Analog	Analog comparator 2 positive input.
	C2-	22	I	Analog	Analog comparator 2 negative input.
General-Purpose Timers	CCP0	95	I/O	TTL	Capture/Compare/PWM 0.
	CCP1	100	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	96	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	6	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	42	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	25	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	86	I/O	TTL	Capture/Compare/PWM 6.
	CCP7	85	I/O	TTL	Capture/Compare/PWM 7.
Hibernate	HIB	51	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.
	VBAT	55	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
	WAKE	50	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
	XOSC0	52	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
	XOSC1	53	O	Analog	Hibernation module oscillator crystal output.
I2C	I2C0SCL	70	I/O	OD	I ² C module 0 clock.
	I2C0SDA	71	I/O	OD	I ² C module 0 data.
	I2C1SCL	34	I/O	OD	I ² C module 1 clock.
	I2C1SDA	35	I/O	OD	I ² C module 1 data.
JTAG/SWD/SWO	SWCLK	80	I	TTL	JTAG/SWD CLK.
	SWDIO	79	I/O	TTL	JTAG TMS and SWDIO.
	SWO	77	O	TTL	JTAG TDO and SWO.
	TCK	80	I	TTL	JTAG/SWD CLK.
	TDI	78	I	TTL	JTAG TDI.
	TDO	77	O	TTL	JTAG TDO and SWO.
	TMS	79	I/O	TTL	JTAG TMS and SWDIO.
	TRST	89	I	TTL	JTAG $\overline{\text{TRST}}$.

Table 19-3. Signals by Function, Except for GPIO (continued)

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
PWM	Fault	99	I	TTL	PWM Fault.
	PWM0	47	O	TTL	PWM 0.
	PWM1	61	O	TTL	PWM 1.
	PWM2	66	O	TTL	PWM 2.
	PWM3	67	O	TTL	PWM 3.
	PWM4	2	O	TTL	PWM 4.
	PWM5	1	O	TTL	PWM 5.
Power	GND	9 15 21 33 39 45 54 57 63 69 82 87 94	-	Power	Ground reference for logic and I/O pins.
	GNDA	4 97	-	Power	The ground reference for the analog circuits (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 μ 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 20 32 44 56 68 81 93	-	Power	Positive supply for I/O and some logic.
	VDD25	14 38 62 88	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDDA	3 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 19-3. Signals by Function, Except for GPIO (continued)

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
QEI	IDX0	10	I	TTL	QEI module 0 index.
	IDX1	84	I	TTL	QEI module 1 index.
	PhA0	11	I	TTL	QEI module 0 phase A.
	PhA1	37	I	TTL	QEI module 1 phase A.
	PhB0	83	I	TTL	QEI module 0 phase B.
	PhB1	36	I	TTL	QEI module 1 phase B.
SSI	SSI0Clk	28	I/O	TTL	SSI module 0 clock.
	SSI0Fss	29	I/O	TTL	SSI module 0 frame.
	SSI0Rx	30	I	TTL	SSI module 0 receive.
	SSI0Tx	31	O	TTL	SSI module 0 transmit.
	SSI1Clk	72	I/O	TTL	SSI module 1 clock.
	SSI1Fss	73	I/O	TTL	SSI module 1 frame.
	SSI1Rx	74	I	TTL	SSI module 1 receive.
SSI1Tx	75	O	TTL	SSI module 1 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	O	Analog	Main oscillator crystal output.
	RST	64	I	TTL	System reset input.
UART	U0Rx	26	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U0Tx	27	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1Rx	12	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	13	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	19	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	18	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-4. GPIO Pins and Alternate Functions

IO	Pin Number	Multiplexed Function	Multiplexed Function
PA0	26	U0Rx	
PA1	27	U0Tx	
PA2	28	SSI0Clk	
PA3	29	SSI0Fss	
PA4	30	SSI0Rx	
PA5	31	SSI0Tx	

Table 19-4. GPIO Pins and Alternate Functions (continued)

IO	Pin Number	Multiplexed Function	Multiplexed Function
PA6	34	I2C1SCL	
PA7	35	I2C1SDA	
PB0	66	PWM2	
PB1	67	PWM3	
PB2	70	I2C0SCL	
PB3	71	I2C0SDA	
PB4	92	C0-	
PB5	91	C1-	
PB6	90	C0+	
PB7	89	$\overline{\text{TRST}}$	
PC0	80	TCK	SWCLK
PC1	79	TMS	SWDIO
PC2	78	TDI	
PC3	77	TDO	SWO
PC4	25	CCP5	
PC5	24	C1+	
PC6	23	C2+	
PC7	22	C2-	
PD0	10	IDX0	
PD1	11	PhA0	
PD2	12	U1Rx	
PD3	13	U1Tx	
PD4	95	CCP0	
PD5	96	CCP2	
PD6	99	Fault	
PD7	100	CCP1	
PE0	72	SSI1Clk	
PE1	73	SSI1Fss	
PE2	74	SSI1Rx	
PE3	75	SSI1Tx	
PE4	6	CCP3	
PE5	5		
PE6	2	PWM4	
PE7	1	PWM5	
PF0	47	PWM0	
PF1	61	PWM1	
PF2	60		
PF3	59		
PF4	58	C0o	
PF5	46	C1o	
PF6	43	C2o	

Table 19-4. GPIO Pins and Alternate Functions (continued)

IO	Pin Number	Multiplexed Function	Multiplexed Function
PF7	42	CCP4	
PG0	19	U2Rx	
PG1	18	U2Tx	
PG2	17		
PG3	16		
PG4	41		
PG5	40		
PG6	37	PhA1	
PG7	36	PhB1	
PH0	86	CCP6	
PH1	85	CCP7	
PH2	84	IDX1	
PH3	83	PhB0	

19.2 108-Pin BGA Package Pin Tables

Table 19-5. Signals by Pin Number

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
A1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
A2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
A3	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
A4	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
A5	GNDA	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
A6	PB4	I/O	TTL	GPIO port B bit 4.
	C0-	I	Analog	Analog comparator 0 negative input.
A7	PB6	I/O	TTL	GPIO port B bit 6.
	C0+	I	Analog	Analog comparator 0 positive input.
A8	PB7	I/O	TTL	GPIO port B bit 7.
	TRST	I	TTL	JTAG TRST.
A9	PC0	I/O	TTL	GPIO port C bit 0.
	TCK	I	TTL	JTAG/SWD CLK.
	SWCLK	I	TTL	JTAG/SWD CLK.
A10	PC3	I/O	TTL	GPIO port C bit 3.
	TDO	O	TTL	JTAG TDO and SWO.
	SWO	O	TTL	JTAG TDO and SWO.
A11	PE0	I/O	TTL	GPIO port E bit 0.
	SSI1Clk	I/O	TTL	SSI module 1 clock.
A12	PE3	I/O	TTL	GPIO port E bit 3.
	SSI1Tx	O	TTL	SSI module 1 transmit.
B1	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.

Table 19-5. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
B2	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
B3	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
B4	NC	-	-	No connect. Leave the pin electrically unconnected/isolated.
B5	GND _A	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on V _{DD} from affecting the analog functions.
B6	GND	-	Power	Ground reference for logic and I/O pins.
B7	PB5	I/O	TTL	GPIO port B bit 5.
	C1-	I	Analog	Analog comparator 1 negative input.
B8	PC2	I/O	TTL	GPIO port C bit 2.
	TDI	I	TTL	JTAG TDI.
B9	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.
B10	CMOD1	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
B11	PE2	I/O	TTL	GPIO port E bit 2.
	SSI1Rx	I	TTL	SSI module 1 receive.
B12	PE1	I/O	TTL	GPIO port E bit 1.
	SSI1F _{SS}	I/O	TTL	SSI module 1 frame.
C1	PE7	I/O	TTL	GPIO port E bit 7.
	PWM5	O	TTL	PWM 5.
C2	PE6	I/O	TTL	GPIO port E bit 6.
	PWM4	O	TTL	PWM 4.
C3	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
C4	GND	-	Power	Ground reference for logic and I/O pins.
C5	GND	-	Power	Ground reference for logic and I/O pins.
C6	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.
C7	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.
C8	PH1	I/O	TTL	GPIO port H bit 1.
	CCP7	I/O	TTL	Capture/Compare/PWM 7.
C9	PH0	I/O	TTL	GPIO port H bit 0.
	CCP6	I/O	TTL	Capture/Compare/PWM 6.
C10	PG7	I/O	TTL	GPIO port G bit 7.
	PhB1	I	TTL	QEI module 1 phase B.
C11	PB2	I/O	TTL	GPIO port B bit 2.
	I2C0SCL	I/O	OD	I ² C module 0 clock.

Table 19-5. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
C12	PB3	I/O	TTL	GPIO port B bit 3.
	I2C0SDA	I/O	OD	I ² C module 0 data.
D1	PE4	I/O	TTL	GPIO port E bit 4.
	CCP3	I/O	TTL	Capture/Compare/PWM 3.
D2	PE5	I/O	TTL	GPIO port E bit 5.
D3	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
D10	PH3	I/O	TTL	GPIO port H bit 3.
	PhB0	I	TTL	QEI module 0 phase B.
D11	PH2	I/O	TTL	GPIO port H bit 2.
	IDX1	I	TTL	QEI module 1 index.
D12	PB1	I/O	TTL	GPIO port B bit 1.
	PWM3	O	TTL	PWM 3.
E1	PD4	I/O	TTL	GPIO port D bit 4.
	CCP0	I/O	TTL	Capture/Compare/PWM 0.
E2	PD5	I/O	TTL	GPIO port D bit 5.
	CCP2	I/O	TTL	Capture/Compare/PWM 2.
E3	LDO	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 μ 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).
E10	VDD33	-	Power	Positive supply for I/O and some logic.
E11	CMOD0	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
E12	PB0	I/O	TTL	GPIO port B bit 0.
	PWM2	O	TTL	PWM 2.
F1	PD7	I/O	TTL	GPIO port D bit 7.
	CCP1	I/O	TTL	Capture/Compare/PWM 1.
F2	PD6	I/O	TTL	GPIO port D bit 6.
	Fault	I	TTL	PWM Fault.
F3	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
F10	GND	-	Power	Ground reference for logic and I/O pins.
F11	GND	-	Power	Ground reference for logic and I/O pins.
F12	GND	-	Power	Ground reference for logic and I/O pins.
G1	PD0	I/O	TTL	GPIO port D bit 0.
	IDX0	I	TTL	QEI module 0 index.
G2	PD1	I/O	TTL	GPIO port D bit 1.
	PhA0	I	TTL	QEI module 0 phase A.
G3	VDD25	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
G10	VDD33	-	Power	Positive supply for I/O and some logic.
G11	VDD33	-	Power	Positive supply for I/O and some logic.

Table 19-5. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
G12	VDD33	-	Power	Positive supply for I/O and some logic.
H1	PD3	I/O	TTL	GPIO port D bit 3.
	U1Tx	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
H2	PD2	I/O	TTL	GPIO port D bit 2.
	U1Rx	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
H3	GND	-	Power	Ground reference for logic and I/O pins.
H10	VDD33	-	Power	Positive supply for I/O and some logic.
H11	RST	I	TTL	System reset input.
H12	PF1	I/O	TTL	GPIO port F bit 1.
	PWM1	O	TTL	PWM 1.
J1	PG2	I/O	TTL	GPIO port G bit 2.
J2	PG3	I/O	TTL	GPIO port G bit 3.
J3	GND	-	Power	Ground reference for logic and I/O pins.
J10	GND	-	Power	Ground reference for logic and I/O pins.
J11	PF2	I/O	TTL	GPIO port F bit 2.
J12	PF3	I/O	TTL	GPIO port F bit 3.
K1	PG0	I/O	TTL	GPIO port G bit 0.
	U2Rx	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
K2	PG1	I/O	TTL	GPIO port G bit 1.
	U2Tx	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
K3	PG4	I/O	TTL	GPIO port G bit 4.
K4	PF7	I/O	TTL	GPIO port F bit 7.
	CCP4	I/O	TTL	Capture/Compare/PWM 4.
K5	GND	-	Power	Ground reference for logic and I/O pins.
K6	GND	-	Power	Ground reference for logic and I/O pins.
K7	VDD33	-	Power	Positive supply for I/O and some logic.
K8	VDD33	-	Power	Positive supply for I/O and some logic.
K9	VDD33	-	Power	Positive supply for I/O and some logic.
K10	GND	-	Power	Ground reference for logic and I/O pins.
K11	XOSC0	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
K12	XOSC1	O	Analog	Hibernation module oscillator crystal output.
L1	PC4	I/O	TTL	GPIO port C bit 4.
	CCP5	I/O	TTL	Capture/Compare/PWM 5.
L2	PC7	I/O	TTL	GPIO port C bit 7.
	C2-	I	Analog	Analog comparator 2 negative input.

Table 19-5. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
L3	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.
L4	PA3	I/O	TTL	GPIO port A bit 3.
	SSI0Fss	I/O	TTL	SSI module 0 frame.
L5	PA4	I/O	TTL	GPIO port A bit 4.
	SSI0Rx	I	TTL	SSI module 0 receive.
L6	PA6	I/O	TTL	GPIO port A bit 6.
	I2C1SCL	I/O	OD	I ² C module 1 clock.
L7	PG6	I/O	TTL	GPIO port G bit 6.
	PhA1	I	TTL	QEI module 1 phase A.
L8	PF5	I/O	TTL	GPIO port F bit 5.
	C1o	O	TTL	Analog comparator 1 output.
L9	PF4	I/O	TTL	GPIO port F bit 4.
	C0o	O	TTL	Analog comparator 0 output.
L10	GND	-	Power	Ground reference for logic and I/O pins.
L11	OSC0	I	Analog	Main oscillator crystal input or an external clock reference input.
L12	VBAT	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
M1	PC5	I/O	TTL	GPIO port C bit 5.
	C1+	I	Analog	Analog comparator 1 positive input.
M2	PC6	I/O	TTL	GPIO port C bit 6.
	C2+	I	Analog	Analog comparator 2 positive input.
M3	PA1	I/O	TTL	GPIO port A bit 1.
	U0Tx	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
M4	PA2	I/O	TTL	GPIO port A bit 2.
	SSI0Clk	I/O	TTL	SSI module 0 clock.
M5	PA5	I/O	TTL	GPIO port A bit 5.
	SSI0Tx	O	TTL	SSI module 0 transmit.
M6	PA7	I/O	TTL	GPIO port A bit 7.
	I2C1SDA	I/O	OD	I ² C module 1 data.
M7	PG5	I/O	TTL	GPIO port G bit 5.
M8	PF6	I/O	TTL	GPIO port F bit 6.
	C2o	O	TTL	Analog comparator 2 output.
M9	PF0	I/O	TTL	GPIO port F bit 0.
	PWM0	O	TTL	PWM 0.
M10	WAKE	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
M11	OSC1	O	Analog	Main oscillator crystal output.

Table 19-5. Signals by Pin Number (continued)

Pin Number	Pin Name	Pin Type	Buffer Type ^a	Description
M12	$\overline{\text{HIB}}$	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-6. Signals by Signal Name

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
C0+	A7	I	Analog	Analog comparator 0 positive input.
C0-	A6	I	Analog	Analog comparator 0 negative input.
C0o	L9	O	TTL	Analog comparator 0 output.
C1+	M1	I	Analog	Analog comparator 1 positive input.
C1-	B7	I	Analog	Analog comparator 1 negative input.
C1o	L8	O	TTL	Analog comparator 1 output.
C2+	M2	I	Analog	Analog comparator 2 positive input.
C2-	L2	I	Analog	Analog comparator 2 negative input.
C2o	M8	O	TTL	Analog comparator 2 output.
CCP0	E1	I/O	TTL	Capture/Compare/PWM 0.
CCP1	F1	I/O	TTL	Capture/Compare/PWM 1.
CCP2	E2	I/O	TTL	Capture/Compare/PWM 2.
CCP3	D1	I/O	TTL	Capture/Compare/PWM 3.
CCP4	K4	I/O	TTL	Capture/Compare/PWM 4.
CCP5	L1	I/O	TTL	Capture/Compare/PWM 5.
CCP6	C9	I/O	TTL	Capture/Compare/PWM 6.
CCP7	C8	I/O	TTL	Capture/Compare/PWM 7.
CMOD0	E11	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
CMOD1	B10	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
Fault	F2	I	TTL	PWM Fault.
GND	B6 C4 C5 F10 F11 F12 H3 J3 J10 K5 K6 K10 L10	-	Power	Ground reference for logic and I/O pins.
GNDA	A5 B5	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
$\overline{\text{HIB}}$	M12	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.

Table 19-6. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
I2C0SCL	C11	I/O	OD	I ² C module 0 clock.
I2C0SDA	C12	I/O	OD	I ² C module 0 data.
I2C1SCL	L6	I/O	OD	I ² C module 1 clock.
I2C1SDA	M6	I/O	OD	I ² C module 1 data.
IDX0	G1	I	TTL	QEI module 0 index.
IDX1	D11	I	TTL	QEI module 1 index.
LDO	E3	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 μ 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	A1 A2 A3 A4 B1 B2 B3 B4	-	-	No connect. Leave the pin electrically unconnected/isolated.
OSC0	L11	I	Analog	Main oscillator crystal input or an external clock reference input.
OSC1	M11	O	Analog	Main oscillator crystal output.
PA0	L3	I/O	TTL	GPIO port A bit 0.
PA1	M3	I/O	TTL	GPIO port A bit 1.
PA2	M4	I/O	TTL	GPIO port A bit 2.
PA3	L4	I/O	TTL	GPIO port A bit 3.
PA4	L5	I/O	TTL	GPIO port A bit 4.
PA5	M5	I/O	TTL	GPIO port A bit 5.
PA6	L6	I/O	TTL	GPIO port A bit 6.
PA7	M6	I/O	TTL	GPIO port A bit 7.
PB0	E12	I/O	TTL	GPIO port B bit 0.
PB1	D12	I/O	TTL	GPIO port B bit 1.
PB2	C11	I/O	TTL	GPIO port B bit 2.
PB3	C12	I/O	TTL	GPIO port B bit 3.
PB4	A6	I/O	TTL	GPIO port B bit 4.
PB5	B7	I/O	TTL	GPIO port B bit 5.
PB6	A7	I/O	TTL	GPIO port B bit 6.
PB7	A8	I/O	TTL	GPIO port B bit 7.
PC0	A9	I/O	TTL	GPIO port C bit 0.
PC1	B9	I/O	TTL	GPIO port C bit 1.
PC2	B8	I/O	TTL	GPIO port C bit 2.
PC3	A10	I/O	TTL	GPIO port C bit 3.
PC4	L1	I/O	TTL	GPIO port C bit 4.
PC5	M1	I/O	TTL	GPIO port C bit 5.

Table 19-6. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
PC6	M2	I/O	TTL	GPIO port C bit 6.
PC7	L2	I/O	TTL	GPIO port C bit 7.
PD0	G1	I/O	TTL	GPIO port D bit 0.
PD1	G2	I/O	TTL	GPIO port D bit 1.
PD2	H2	I/O	TTL	GPIO port D bit 2.
PD3	H1	I/O	TTL	GPIO port D bit 3.
PD4	E1	I/O	TTL	GPIO port D bit 4.
PD5	E2	I/O	TTL	GPIO port D bit 5.
PD6	F2	I/O	TTL	GPIO port D bit 6.
PD7	F1	I/O	TTL	GPIO port D bit 7.
PE0	A11	I/O	TTL	GPIO port E bit 0.
PE1	B12	I/O	TTL	GPIO port E bit 1.
PE2	B11	I/O	TTL	GPIO port E bit 2.
PE3	A12	I/O	TTL	GPIO port E bit 3.
PE4	D1	I/O	TTL	GPIO port E bit 4.
PE5	D2	I/O	TTL	GPIO port E bit 5.
PE6	C2	I/O	TTL	GPIO port E bit 6.
PE7	C1	I/O	TTL	GPIO port E bit 7.
PF0	M9	I/O	TTL	GPIO port F bit 0.
PF1	H12	I/O	TTL	GPIO port F bit 1.
PF2	J11	I/O	TTL	GPIO port F bit 2.
PF3	J12	I/O	TTL	GPIO port F bit 3.
PF4	L9	I/O	TTL	GPIO port F bit 4.
PF5	L8	I/O	TTL	GPIO port F bit 5.
PF6	M8	I/O	TTL	GPIO port F bit 6.
PF7	K4	I/O	TTL	GPIO port F bit 7.
PG0	K1	I/O	TTL	GPIO port G bit 0.
PG1	K2	I/O	TTL	GPIO port G bit 1.
PG2	J1	I/O	TTL	GPIO port G bit 2.
PG3	J2	I/O	TTL	GPIO port G bit 3.
PG4	K3	I/O	TTL	GPIO port G bit 4.
PG5	M7	I/O	TTL	GPIO port G bit 5.
PG6	L7	I/O	TTL	GPIO port G bit 6.
PG7	C10	I/O	TTL	GPIO port G bit 7.
PH0	C9	I/O	TTL	GPIO port H bit 0.
PH1	C8	I/O	TTL	GPIO port H bit 1.
PH2	D11	I/O	TTL	GPIO port H bit 2.
PH3	D10	I/O	TTL	GPIO port H bit 3.
PhA0	G2	I	TTL	QEI module 0 phase A.
PhA1	L7	I	TTL	QEI module 1 phase A.
PhB0	D10	I	TTL	QEI module 0 phase B.

Table 19-6. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
PhB1	C10	I	TTL	QEI module 1 phase B.
PWM0	M9	O	TTL	PWM 0.
PWM1	H12	O	TTL	PWM 1.
PWM2	E12	O	TTL	PWM 2.
PWM3	D12	O	TTL	PWM 3.
PWM4	C2	O	TTL	PWM 4.
PWM5	C1	O	TTL	PWM 5.
$\overline{\text{RST}}$	H11	I	TTL	System reset input.
SSI0Clk	M4	I/O	TTL	SSI module 0 clock.
SSI0Fss	L4	I/O	TTL	SSI module 0 frame.
SSI0Rx	L5	I	TTL	SSI module 0 receive.
SSI0Tx	M5	O	TTL	SSI module 0 transmit.
SSI1Clk	A11	I/O	TTL	SSI module 1 clock.
SSI1Fss	B12	I/O	TTL	SSI module 1 frame.
SSI1Rx	B11	I	TTL	SSI module 1 receive.
SSI1Tx	A12	O	TTL	SSI module 1 transmit.
SWCLK	A9	I	TTL	JTAG/SWD CLK.
SWDIO	B9	I/O	TTL	JTAG TMS and SWDIO.
SWO	A10	O	TTL	JTAG TDO and SWO.
TCK	A9	I	TTL	JTAG/SWD CLK.
TDI	B8	I	TTL	JTAG TDI.
TDO	A10	O	TTL	JTAG TDO and SWO.
TMS	B9	I/O	TTL	JTAG TMS and SWDIO.
$\overline{\text{TRST}}$	A8	I	TTL	JTAG $\overline{\text{TRST}}$.
U0Rx	L3	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
U0Tx	M3	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
U1Rx	H2	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
U1Tx	H1	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
U2Rx	K1	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
U2Tx	K2	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.
VBAT	L12	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
VDD25	C3 D3 F3 G3	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.

Table 19-6. Signals by Signal Name (continued)

Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
VDD33	E10 G10 G11 G12 H10 K7 K8 K9	-	Power	Positive supply for I/O and some logic.
VDDA	C6 C7	-	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.
WAKE	M10	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
XOSC0	K11	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
XOSC1	K12	O	Analog	Hibernation module oscillator crystal output.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-7. Signals by Function, Except for GPIO

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
Analog Comparators	C0+	A7	I	Analog	Analog comparator 0 positive input.
	C0-	A6	I	Analog	Analog comparator 0 negative input.
	C0o	L9	O	TTL	Analog comparator 0 output.
	C1+	M1	I	Analog	Analog comparator 1 positive input.
	C1-	B7	I	Analog	Analog comparator 1 negative input.
	C1o	L8	O	TTL	Analog comparator 1 output.
	C2+	M2	I	Analog	Analog comparator 2 positive input.
	C2-	L2	I	Analog	Analog comparator 2 negative input.
	C2o	M8	O	TTL	Analog comparator 2 output.
General-Purpose Timers	CCP0	E1	I/O	TTL	Capture/Compare/PWM 0.
	CCP1	F1	I/O	TTL	Capture/Compare/PWM 1.
	CCP2	E2	I/O	TTL	Capture/Compare/PWM 2.
	CCP3	D1	I/O	TTL	Capture/Compare/PWM 3.
	CCP4	K4	I/O	TTL	Capture/Compare/PWM 4.
	CCP5	L1	I/O	TTL	Capture/Compare/PWM 5.
	CCP6	C9	I/O	TTL	Capture/Compare/PWM 6.
	CCP7	C8	I/O	TTL	Capture/Compare/PWM 7.

Table 19-7. Signals by Function, Except for GPIO (continued)

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
Hibernate	HIB	M12	O	OD	An open-drain output with internal pull-up that indicates the processor is in Hibernate mode.
	VBAT	L12	-	Power	Power source for the Hibernation module. It is normally connected to the positive terminal of a battery and serves as the battery backup/Hibernation module power-source supply.
	WAKE	M10	I	TTL	An external input that brings the processor out of Hibernate mode when asserted.
	XOSC0	K11	I	Analog	Hibernation module oscillator crystal input or an external clock reference input. Note that this is either a 4.194304-MHz crystal or a 32.768-kHz oscillator for the Hibernation module RTC. See the CLKSEL bit in the HIBCTL register.
	XOSC1	K12	O	Analog	Hibernation module oscillator crystal output.
I2C	I2C0SCL	C11	I/O	OD	I ² C module 0 clock.
	I2C0SDA	C12	I/O	OD	I ² C module 0 data.
	I2C1SCL	L6	I/O	OD	I ² C module 1 clock.
	I2C1SDA	M6	I/O	OD	I ² C module 1 data.
JTAG/SWD/SWO	SWCLK	A9	I	TTL	JTAG/SWD CLK.
	SWDIO	B9	I/O	TTL	JTAG TMS and SWDIO.
	SWO	A10	O	TTL	JTAG TDO and SWO.
	TCK	A9	I	TTL	JTAG/SWD CLK.
	TDI	B8	I	TTL	JTAG TDI.
	TDO	A10	O	TTL	JTAG TDO and SWO.
	TMS	B9	I/O	TTL	JTAG TMS and SWDIO.
	TRST	A8	I	TTL	JTAG TRST.
PWM	Fault	F2	I	TTL	PWM Fault.
	PWM0	M9	O	TTL	PWM 0.
	PWM1	H12	O	TTL	PWM 1.
	PWM2	E12	O	TTL	PWM 2.
	PWM3	D12	O	TTL	PWM 3.
	PWM4	C2	O	TTL	PWM 4.
	PWM5	C1	O	TTL	PWM 5.

Table 19-7. Signals by Function, Except for GPIO (continued)

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
Power	GND	B6 C4 C5 F10 F11 F12 H3 J3 J10 K5 K6 K10 L10	-	Power	Ground reference for logic and I/O pins.
	GNDA	A5 B5	-	Power	The ground reference for the analog circuits (Analog Comparators, etc.). These are separated from GND to minimize the electrical noise contained on VDD from affecting the analog functions.
	LDO	E3	-	Power	Low drop-out regulator output voltage. This pin requires an external capacitor between the pin and GND of 1 μ 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).
	VDD25	C3 D3 F3 G3	-	Power	Positive supply for most of the logic function, including the processor core and most peripherals.
	VDD33	E10 G10 G11 G12 H10 K7 K8 K9	-	Power	Positive supply for I/O and some logic.
	VDDA	C6 C7	-	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.
QEI	IDX0	G1	I	TTL	QEI module 0 index.
	IDX1	D11	I	TTL	QEI module 1 index.
	PhA0	G2	I	TTL	QEI module 0 phase A.
	PhA1	L7	I	TTL	QEI module 1 phase A.
	PhB0	D10	I	TTL	QEI module 0 phase B.
	PhB1	C10	I	TTL	QEI module 1 phase B.

Table 19-7. Signals by Function, Except for GPIO (continued)

Function	Pin Name	Pin Number	Pin Type	Buffer Type ^a	Description
SSI	SSI0Clk	M4	I/O	TTL	SSI module 0 clock.
	SSI0Fss	L4	I/O	TTL	SSI module 0 frame.
	SSI0Rx	L5	I	TTL	SSI module 0 receive.
	SSI0Tx	M5	O	TTL	SSI module 0 transmit.
	SSI1Clk	A11	I/O	TTL	SSI module 1 clock.
	SSI1Fss	B12	I/O	TTL	SSI module 1 frame.
	SSI1Rx	B11	I	TTL	SSI module 1 receive.
	SSI1Tx	A12	O	TTL	SSI module 1 transmit.
System Control & Clocks	CMOD0	E11	I/O	TTL	CPU Mode bit 0. Input must be set to logic 0 (grounded); other encodings reserved.
	CMOD1	B10	I/O	TTL	CPU Mode bit 1. Input must be set to logic 0 (grounded); other encodings reserved.
	OSC0	L11	I	Analog	Main oscillator crystal input or an external clock reference input.
	OSC1	M11	O	Analog	Main oscillator crystal output.
	RST	H11	I	TTL	System reset input.
UART	U0Rx	L3	I	TTL	UART module 0 receive. When in IrDA mode, this signal has IrDA modulation.
	U0Tx	M3	O	TTL	UART module 0 transmit. When in IrDA mode, this signal has IrDA modulation.
	U1Rx	H2	I	TTL	UART module 1 receive. When in IrDA mode, this signal has IrDA modulation.
	U1Tx	H1	O	TTL	UART module 1 transmit. When in IrDA mode, this signal has IrDA modulation.
	U2Rx	K1	I	TTL	UART module 2 receive. When in IrDA mode, this signal has IrDA modulation.
	U2Tx	K2	O	TTL	UART module 2 transmit. When in IrDA mode, this signal has IrDA modulation.

a. The TTL designation indicates the pin has TTL-compatible voltage levels.

Table 19-8. GPIO Pins and Alternate Functions

IO	Pin Number	Multiplexed Function	Multiplexed Function
PA0	L3	U0Rx	
PA1	M3	U0Tx	
PA2	M4	SSI0Clk	
PA3	L4	SSI0Fss	
PA4	L5	SSI0Rx	
PA5	M5	SSI0Tx	
PA6	L6	I2C1SCL	
PA7	M6	I2C1SDA	
PB0	E12	PWM2	
PB1	D12	PWM3	
PB2	C11	I2C0SCL	
PB3	C12	I2C0SDA	

Table 19-8. GPIO Pins and Alternate Functions (continued)

IO	Pin Number	Multiplexed Function	Multiplexed Function
PB4	A6	C0-	
PB5	B7	C1-	
PB6	A7	C0+	
PB7	A8	TRST	
PC0	A9	TCK	SWCLK
PC1	B9	TMS	SWDIO
PC2	B8	TDI	
PC3	A10	TDO	SWO
PC4	L1	CCP5	
PC5	M1	C1+	
PC6	M2	C2+	
PC7	L2	C2-	
PD0	G1	IDX0	
PD1	G2	PhA0	
PD2	H2	U1Rx	
PD3	H1	U1Tx	
PD4	E1	CCP0	
PD5	E2	CCP2	
PD6	F2	Fault	
PD7	F1	CCP1	
PE0	A11	SSI1Clk	
PE1	B12	SSI1Fss	
PE2	B11	SSI1Rx	
PE3	A12	SSI1Tx	
PE4	D1	CCP3	
PE5	D2		
PE6	C2	PWM4	
PE7	C1	PWM5	
PF0	M9	PWM0	
PF1	H12	PWM1	
PF2	J11		
PF3	J12		
PF4	L9	C0o	
PF5	L8	C1o	
PF6	M8	C2o	
PF7	K4	CCP4	
PG0	K1	U2Rx	
PG1	K2	U2Tx	
PG2	J1		
PG3	J2		
PG4	K3		

Table 19-8. GPIO Pins and Alternate Functions (continued)

IO	Pin Number	Multiplexed Function	Multiplexed Function
PG5	M7		
PG6	L7	PhA1	
PG7	C10	PhB1	
PH0	C9	CCP6	
PH1	C8	CCP7	
PH2	D11	IDX1	
PH3	D10	PhB0	

20 Operating Characteristics

Table 20-1. Temperature Characteristics

Characteristic ^a	Symbol	Value	Unit
Industrial operating temperature range	T_A	-40 to +85	°C
Extended operating temperature range	T_A	-40 to +105	°C

a. Maximum storage temperature is 150°C.

Table 20-2. Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance (junction to ambient) ^a	Θ_{JA}	34	°C/W
Average junction temperature ^b	T_J	$T_A + (P_{AVG} \cdot \Theta_{JA})$	°C

a. Junction to ambient thermal resistance Θ_{JA} numbers are determined by a package simulator.

b. Power dissipation is a function of temperature.

21 Electrical Characteristics

21.1 DC Characteristics

21.1.1 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 21-1. Maximum Ratings

Characteristic ^a	Symbol	Value		Unit
		Min	Max	
I/O supply voltage (V_{DD})	V_{DD}	0	4	V
Core supply voltage (V_{DD25})	V_{DD25}	0	3	V
Analog supply voltage (V_{DDA})	V_{DDA}	0	4	V
Battery supply voltage (V_{BAT})	V_{BAT}	0	4	V
Input voltage	V_{IN}	-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 V_{DD}).

21.1.2 Recommended DC Operating Conditions

For special high-current applications, the GPIO output buffers may be used with the following restrictions. With the GPIO pins configured as 8-mA output drivers, a total of four GPIO outputs may be used to sink current loads up to 18 mA each. At 18-mA sink current loading, the V_{OL} value is specified as 1.2 V. The high-current GPIO package pins must be selected such that there are only a maximum of two per side of the physical package or BGA pin group with the total number of high-current GPIO outputs not exceeding four for the entire package.

Table 21-2. Recommended DC Operating Conditions

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{DD}	I/O supply voltage	3.0	3.3	3.6	V
V_{DD25}	Core supply voltage	2.25	2.5	2.75	V
V_{DDA}	Analog supply voltage	3.0	3.3	3.6	V
V_{BAT}	Battery supply voltage	2.3	3.0	3.6	V
V_{IH}	High-level input voltage	2.0	-	5.0	V
V_{IL}	Low-level input voltage	-0.3	-	1.3	V
V_{OH}^a	High-level output voltage	2.4	-	-	V
V_{OL}^a	Low-level output voltage	-	-	0.4	V

Table 21-2. Recommended DC Operating Conditions (continued)

Parameter	Parameter Name	Min	Nom	Max	Unit
I_{OH}	High-level source current, $V_{OH}=2.4$ V				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
I_{OL}	Low-level sink current, $V_{OL}=0.4$ V				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA

a. V_{OL} and V_{OH} shift to 1.2 V when using high-current GPIOs.

21.1.3 On-Chip Low Drop-Out (LDO) Regulator Characteristics

Table 21-3. LDO Regulator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
V_{LDOOUT}	Programmable internal (logic) power supply output value	2.25	2.5	2.75	V
	Output voltage accuracy	-	2%	-	%
t_{PON}	Power-on time	-	-	100	μ s
t_{ON}	Time on	-	-	200	μ s
t_{OFF}	Time off	-	-	100	μ s
V_{STEP}	Step programming incremental voltage	-	50	-	mV
C_{LDO}	External filter capacitor size for internal power supply	1.0	-	3.0	μ F

21.1.4 GPIO Module Characteristics

Table 21-4. GPIO Module DC Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
R_{GPIOPU}	GPIO internal pull-up resistor	50	-	110	k Ω
R_{GPIOPD}	GPIO internal pull-down resistor	55	-	180	k Ω

21.1.5 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):

- $V_{DD} = 3.3$ V
- $V_{DD25} = 2.50$ V
- $V_{BAT} = 3.0$ V
- $V_{DDA} = 3.3$ V
- Temperature = 25°C

- Clock Source (MOSC) = 3.579545 MHz Crystal Oscillator
- Main oscillator (MOSC) = enabled
- Internal oscillator (IOSC) = disabled

Table 21-5. Detailed Power Specifications

Parameter	Parameter Name	Conditions	3.3 V V_{DD} , V_{DDA}		2.5 V V_{DD25}		3.0 V V_{BAT}		Unit
			Nom	Max	Nom	Max	Nom	Max	
I_{DD_RUN}	Run mode 1 (Flash loop)	$V_{DD25} = 2.50\text{ V}$ Code= while(1){} executed in Flash Peripherals = All ON System Clock = 50 MHz (with PLL)	3	pending ^a	108	pending ^a	0	pending ^a	mA
	Run mode 2 (Flash loop)	$V_{DD25} = 2.50\text{ V}$ Code= while(1){} executed in Flash Peripherals = All OFF System Clock = 50 MHz (with PLL)	0	pending ^a	53	pending ^a	0	pending ^a	mA
	Run mode 1 (SRAM loop)	$V_{DD25} = 2.50\text{ V}$ Code= while(1){} executed in SRAM Peripherals = All ON System Clock = 50 MHz (with PLL)	3	pending ^a	102	pending ^a	0	pending ^a	mA
	Run mode 2 (SRAM loop)	$V_{DD25} = 2.50\text{ V}$ Code= while(1){} executed in SRAM Peripherals = All OFF System Clock = 50 MHz (with PLL)	0	pending ^a	47	pending ^a	0	pending ^a	mA
I_{DD_SLEEP}	Sleep mode	$V_{DD25} = 2.50\text{ V}$ Peripherals = All OFF System Clock = 50 MHz (with PLL)	0	pending ^a	17	pending ^a	0	pending ^a	mA
$I_{DD_DEEPSLEEP}$	Deep-Sleep mode	LDO = 2.25 V Peripherals = All OFF System Clock = IOS30KHZ/64	0.14	pending ^a	0.18	pending ^a	0	pending ^a	mA

Table 21-5. Detailed Power Specifications (continued)

Parameter	Parameter Name	Conditions	3.3 V V_{DD} , V_{DDA}		2.5 V V_{DD25}		3.0 V V_{BAT}		Unit
			Nom	Max	Nom	Max	Nom	Max	
$I_{DD_HIBERNATE}$	Hibernate mode	$V_{BAT} = 3.0\text{ V}$ $V_{DD} = 0\text{ V}$ $V_{DD25} = 0\text{ V}$ $V_{DDA} = 0\text{ V}$ Peripherals = All OFF System Clock = OFF Hibernate Module = 32 kHz	0	0	0	0	16	pending ^a	μA

a. Pending characterization completion.

21.1.6 Flash Memory Characteristics

Table 21-6. Flash Memory Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
PE_{CYC}	Number of guaranteed program/erase cycles before failure ^a	10,000	100,000	-	cycles
T_{RET}	Data retention at average operating temperature of 85°C (industrial) or 105°C (extended)	10	-	-	years
T_{PROG}	Word program time	20	-	-	μs
T_{ERASE}	Page erase time	20	-	-	ms
T_{ME}	Mass erase time	200	-	-	ms

a. A program/erase cycle is defined as switching the bits from 1-> 0 -> 1.

21.1.7 Hibernation

Table 21-7. Hibernation Module DC Characteristics

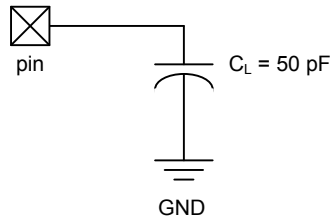
Parameter	Parameter Name	Value	Unit
V_{LOWBAT}	Low battery detect voltage	2.35	V
R_{WAKEPU}	\overline{WAKE} internal pull-up resistor	200	k Ω

21.2 AC Characteristics

21.2.1 Load Conditions

Unless otherwise specified, the following conditions are true for all timing measurements. Timing measurements are for 4-mA drive strength.

Figure 21-1. Load Conditions



21.2.2 Clocks

Table 21-8. Phase Locked Loop (PLL) Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
$f_{\text{ref_crystal}}$	Crystal reference ^a	3.579545	-	8.192	MHz
$f_{\text{ref_ext}}$	External clock reference ^a	3.579545	-	8.192	MHz
f_{pll}	PLL frequency ^b	-	400	-	MHz
T_{READY}	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.

b. PLL frequency is automatically calculated by the hardware based on the `XTAL` field of the **RCC** register.

Table 21-9 on page 488 shows the actual frequency of the PLL based on the crystal frequency used (defined by the `XTAL` field in the **RCC** register).

Table 21-9. Actual PLL Frequency

<code>XTAL</code>	Crystal Frequency (MHz)	PLL Frequency (MHz)	Error
0x4	3.5795	400.904	0.0023%
0x5	3.6864	398.1312	0.0047%
0x6	4.0	400	-
0x7	4.096	401.408	0.0035%
0x8	4.9152	398.1312	0.0047%
0x9	5.0	400	-
0xA	5.12	399.36	0.0016%
0xB	6.0	400	-
0xC	6.144	399.36	0.0016%
0xD	7.3728	398.1312	0.0047%
0xE	8.0	400	0.0047%
0xF	8.192	398.6773333	0.0033%

Table 21-10. Clock Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
f_{IOSC}	Internal 12 MHz oscillator frequency	8.4	12	15.6	MHz
$f_{\text{IOSC30KHZ}}$	Internal 30 KHz oscillator frequency	15	30	45	KHz
f_{XOSC}	Hibernation module oscillator frequency	-	4.194304	-	MHz
$f_{\text{XOSC_XTAL}}$	Crystal reference for hibernation oscillator	-	4.194304	-	MHz

Table 21-10. Clock Characteristics (continued)

Parameter	Parameter Name	Min	Nom	Max	Unit
f _{XOSC_EXT}	External clock reference for hibernation module	-	32.768	-	KHz
f _{MOSC}	Main oscillator frequency	1	-	8.192	MHz
t _{MOSC_per}	Main oscillator period	125	-	1000	ns
f _{ref_crystal_bypass}	Crystal reference using the main oscillator (PLL in BYPASS mode)	1	-	8.192	MHz
f _{ref_ext_bypass}	External clock reference (PLL in BYPASS mode)	0	-	50	MHz
f _{system_clock}	System clock	0	-	50	MHz

Table 21-11. Crystal Characteristics

Parameter Name	Value				Units
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 (-40°C to 85°C)	±25	±25	±25	±25	ppm
Temperature stability (-40°C to 105°C)	±25	±25	±25	±25	ppm
Motional capacitance (typ)	27.8	37.0	55.6	63.5	pF
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

21.2.3 JTAG and Boundary Scan

Table 21-12. JTAG Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J1	f _{TCK}	TCK operational clock frequency	0	-	10	MHz
J2	t _{TCK}	TCK operational clock period	100	-	-	ns
J3	t _{TCK_LOW}	TCK clock Low time	-	t _{TCK}	-	ns
J4	t _{TCK_HIGH}	TCK clock High time	-	t _{TCK}	-	ns
J5	t _{TCK_R}	TCK rise time	0	-	10	ns
J6	t _{TCK_F}	TCK fall time	0	-	10	ns
J7	t _{TMS_SU}	TMS setup time to TCK rise	20	-	-	ns
J8	t _{TMS_HLD}	TMS hold time from TCK rise	20	-	-	ns
J9	t _{TDI_SU}	TDI setup time to TCK rise	25	-	-	ns
J10	t _{TDI_HLD}	TDI hold time from TCK rise	25	-	-	ns

Table 21-12. JTAG Characteristics (continued)

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
J11	TCK fall to Data Valid from High-Z	2-mA drive	-	23	35	ns
		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
		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
		4-mA drive		7	9	ns
		8-mA drive		6	8	ns
		8-mA drive with slew rate control		7	9	ns
J14	t_{TRST}	TRST assertion time	100	-	-	ns
J15	t_{TRST_SU}	TRST setup time to TCK rise	10	-	-	ns

Figure 21-2. JTAG Test Clock Input Timing

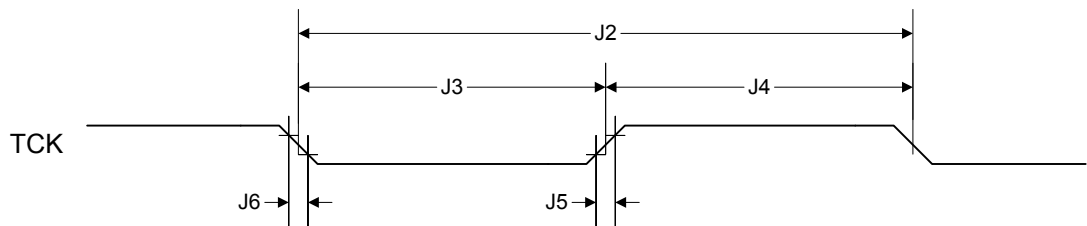


Figure 21-3. JTAG Test Access Port (TAP) Timing

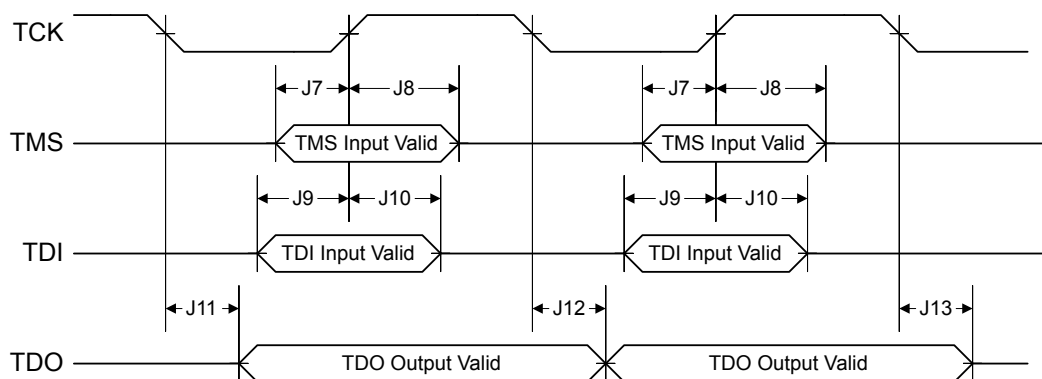
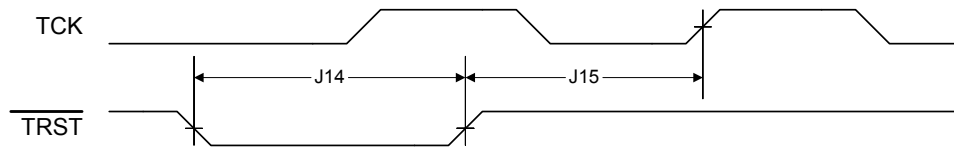


Figure 21-4. JTAG TRST Timing



21.2.4 Reset

Table 21-13. Reset Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
R1	V_{TH}	Reset threshold	-	2.0	-	V
R2	V_{BTH}	Brown-Out threshold	2.85	2.9	2.95	V
R3	T_{POR}	Power-On Reset timeout	-	10	-	ms
R4	T_{BOR}	Brown-Out timeout	-	500	-	μ s
R5	T_{IRPOR}	Internal reset timeout after POR	6	-	11	ms
R6	T_{IRBOR}	Internal reset timeout after BOR ^a	0	-	1	μ s
R7	T_{IRHWR}	Internal reset timeout after hardware reset (\overline{RST} pin)	0	-	1	ms
R8	T_{IRSWR}	Internal reset timeout after software-initiated system reset ^a	2.5	-	20	μ s
R9	T_{IRWDR}	Internal reset timeout after watchdog reset ^a	2.5	-	20	μ s
R10	$T_{VDDRISE}$	Supply voltage (V_{DD}) rise time (0V-3.3V)	-	-	250	ms
R11	T_{MIN}	Minimum \overline{RST} pulse width	2	-	-	μ s

a. $20 * t_{MOSC_per}$

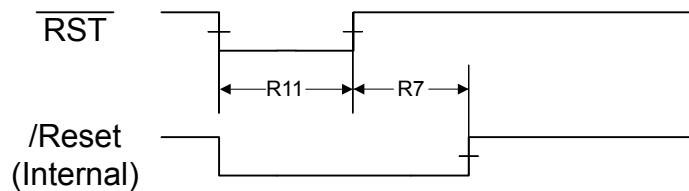
Figure 21-5. External Reset Timing (\overline{RST})

Figure 21-6. Power-On Reset Timing

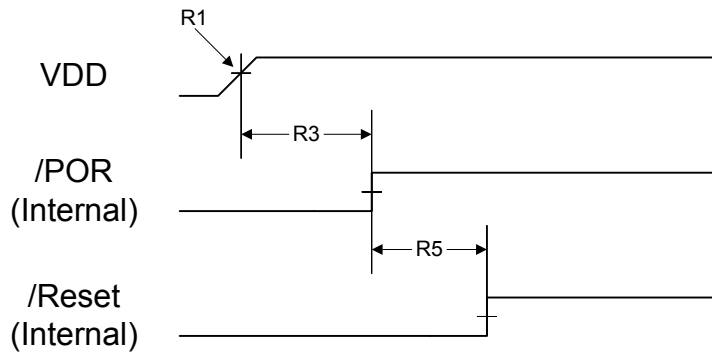


Figure 21-7. Brown-Out Reset Timing

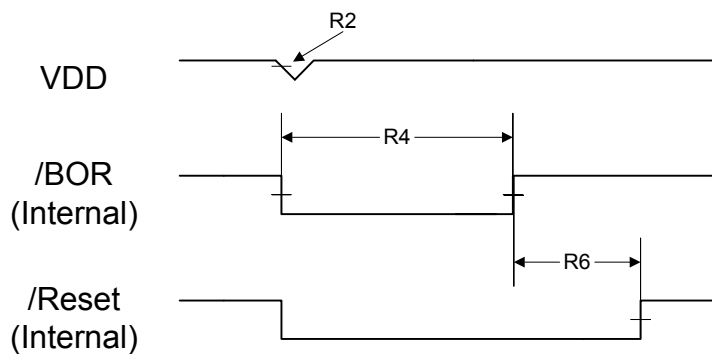


Figure 21-8. Software Reset Timing

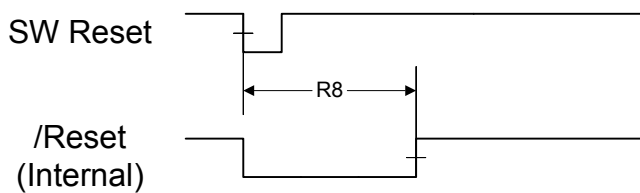
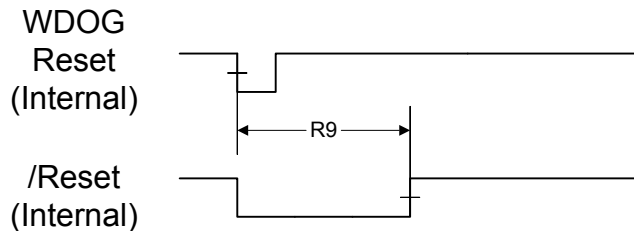


Figure 21-9. Watchdog Reset Timing



21.2.5 Hibernation Module

The Hibernation Module requires special system implementation considerations since it is intended to power-down all other sections of its host device. The system power-supply distribution and interfaces to the device must be driven to 0 V_{DC} or powered down with the same external voltage regulator controlled by $\overline{\text{HIB}}$.

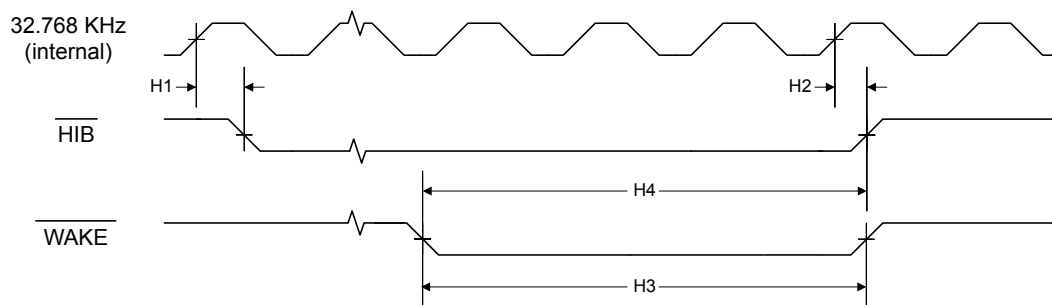
The external voltage regulators controlled by $\overline{\text{HIB}}$ must have a settling time of 250 μs or less.

Table 21-14. Hibernation Module AC Characteristics

Parameter No	Parameter	Parameter Name	Min	Nom	Max	Unit
H1	$t_{\text{HIB_LOW}}$	Internal 32.768 KHz clock reference rising edge to /HIB asserted	-	200	-	μs
H2	$t_{\text{HIB_HIGH}}$	Internal 32.768 KHz clock reference rising edge to /HIB deasserted	-	30	-	μs
H3	$t_{\text{WAKE_ASSERT}}$	/WAKE assertion time	62	-	-	μs
H4	$t_{\text{WAKETOHIB}}$	/WAKE assert to /HIB desassert	62	-	124	μs
H5	$t_{\text{XOSC_SETTLE}}$	XOSC settling time ^a	20	-	-	ms
H6	$t_{\text{HIB_REG_ACCESS}}$	Access time to or from a non-volatile register in HIB module to complete	92	-	-	μs
H7	$t_{\text{HIB_TO_VDD}}$	$\overline{\text{HIB}}$ deassert to VDD and VDD25 at minimum operational level	-	-	250	μs

a. This parameter is highly sensitive to PCB layout and trace lengths, which may make this parameter time longer. Care must be taken in PCB design to minimize trace lengths and RLC (resistance, inductance, capacitance).

Figure 21-10. Hibernation Module Timing



21.2.6 General-Purpose I/O (GPIO)

Note: All GPIOs are 5 V-tolerant.

Table 21-15. GPIO Characteristics

Parameter	Parameter Name	Condition	Min	Nom	Max	Unit
$t_{\text{GPIO R}}$	GPIO Rise Time (from 20% to 80% of V_{DD})	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
$t_{\text{GPIO F}}$	GPIO Fall Time (from 80% to 20% of V_{DD})	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

21.2.7 Synchronous Serial Interface (SSI)

Table 21-16. SSI Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
S1	t_{clk_per}	SSIClk cycle time	2	-	65024	system clocks
S2	t_{clk_high}	SSIClk high time	-	0.5	-	t clk_per
S3	t_{clk_low}	SSIClk low time	-	0.5	-	t clk_per
S4	t_{clkrf}	SSIClk rise/fall time	-	7.4	26	ns
S5	t_{DMd}	Data from master valid delay time	0	-	1	system clocks
S6	t_{DMs}	Data from master setup time	1	-	-	system clocks
S7	t_{DMh}	Data from master hold time	2	-	-	system clocks
S8	t_{DSs}	Data from slave setup time	1	-	-	system clocks
S9	t_{DSH}	Data from slave hold time	2	-	-	system clocks

Figure 21-11. SSI Timing for TI Frame Format (FRF=01), Single Transfer Timing Measurement

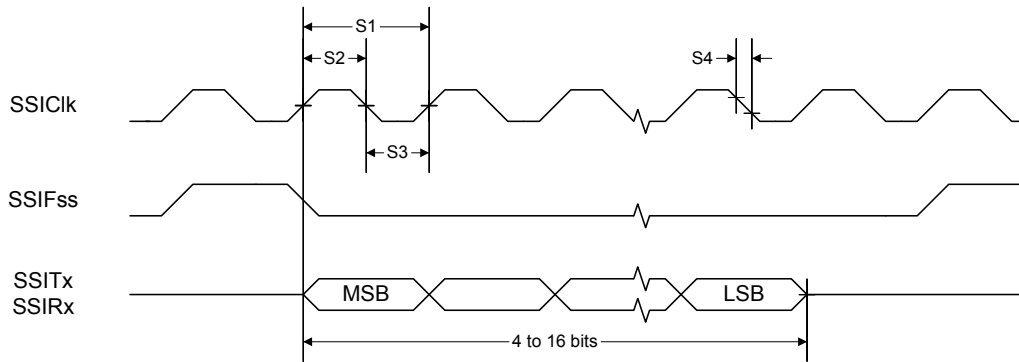


Figure 21-12. SSI Timing for MICROWIRE Frame Format (FRF=10), Single Transfer

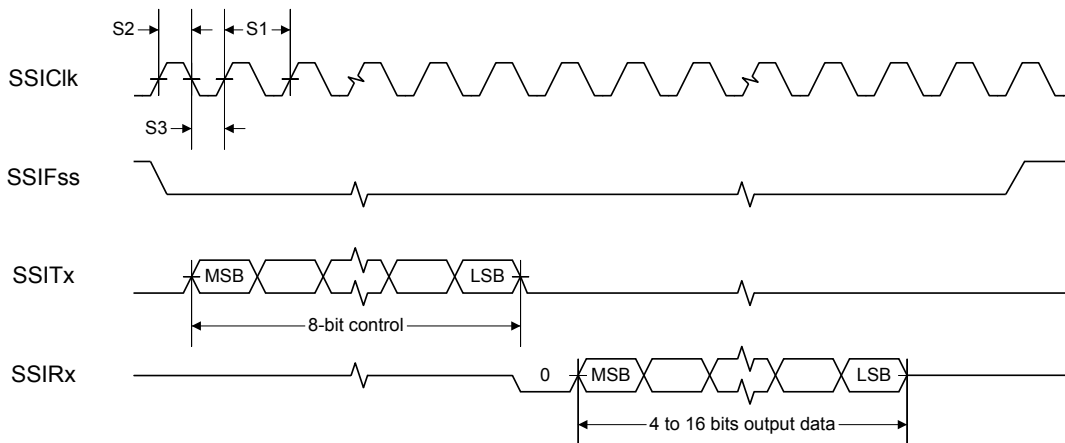
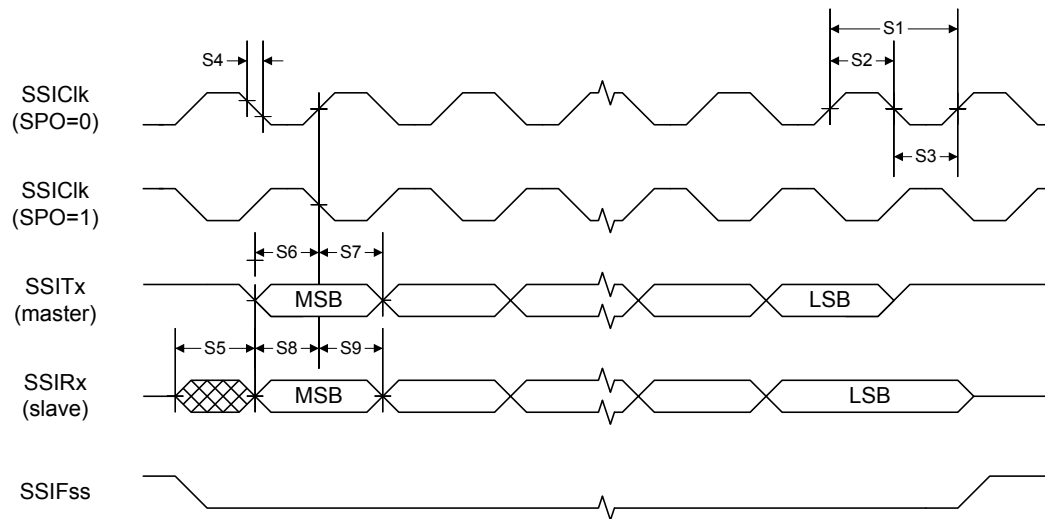


Figure 21-13. SSI Timing for SPI Frame Format (FRF=00), with SPH=1



21.2.8 Inter-Integrated Circuit (I²C) Interface

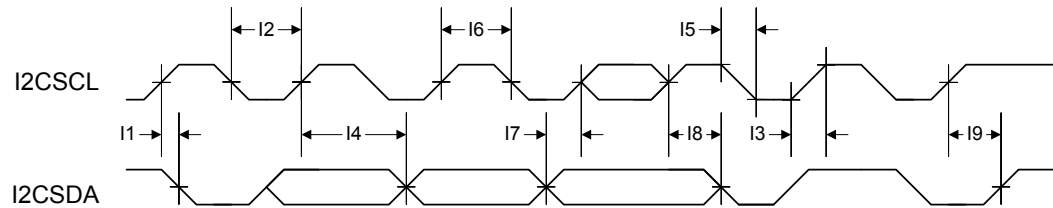
Table 21-17. I²C Characteristics

Parameter No.	Parameter	Parameter Name	Min	Nom	Max	Unit
11 ^a	t_{SCH}	Start condition hold time	36	-	-	system clocks
12 ^a	t_{LP}	Clock Low period	36	-	-	system clocks
13 ^b	t_{SRT}	I ² C _{SCL} /I ² C _{SDA} rise time ($V_{IL}=0.5\text{ V}$ to $V_{IH}=2.4\text{ V}$)	-	-	(see note b)	ns
14 ^a	t_{DH}	Data hold time	2	-	-	system clocks
15 ^c	t_{SFT}	I ² C _{SCL} /I ² C _{SDA} fall time ($V_{IH}=2.4\text{ V}$ to $V_{IL}=0.5\text{ V}$)	-	9	10	ns
16 ^a	t_{HT}	Clock High time	24	-	-	system clocks
17 ^a	t_{DS}	Data setup time	18	-	-	system clocks
18 ^a	t_{SCSR}	Start condition setup time (for repeated start condition only)	36	-	-	system clocks
19 ^a	t_{SCS}	Stop condition setup time	24	-	-	system clocks

a. Values depend on the value programmed into the TPR bit in the I²C Master Timer Period (I2CMTPR) register; a TPR programmed for the maximum I²C_{SCL} frequency (TPR=0x2) results in a minimum output timing as shown in the table above. The I²C interface is designed to scale the actual data transition time to move it to the middle of the I²C_{SCL} 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 I²C_{SCL} and I²C_{SDA} are open-drain-type outputs, which the controller can only actively drive Low, the time I²C_{SCL} or I²C_{SDA} takes to reach a high level depends on external signal capacitance and pull-up resistor values.

c. Specified at a nominal 50 pF load.

Figure 21-14. I²C Timing

21.2.9 Analog Comparator

Table 21-18. Analog Comparator Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
V _{OS}	Input offset voltage	-	±10	±25	mV
V _{CM}	Input common mode voltage range	0	-	V _{DD} -1.5	V
C _{MRR}	Common mode rejection ratio	50	-	-	dB
T _{RT}	Response time	-	-	1	µs
T _{MC}	Comparator mode change to Output Valid	-	-	10	µs

Table 21-19. Analog Comparator Voltage Reference Characteristics

Parameter	Parameter Name	Min	Nom	Max	Unit
R _{HR}	Resolution high range	-	V _{DD} /31	-	LSB
R _{LR}	Resolution low range	-	V _{DD} /23	-	LSB
A _{HR}	Absolute accuracy high range	-	-	±1/2	LSB
A _{LR}	Absolute accuracy low range	-	-	±1/4	LSB

A Serial Flash Loader

A.1 Serial Flash Loader

The Stellaris® 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® device which is calculated as follows:

$$\text{Max Baud Rate} = \text{System Clock Frequency} / 16$$

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 $2 * (20(\text{bits}/\text{sync}) / \text{baud rate} (\text{bits}/\text{sec}))$. 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 316 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.

```
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 $Data[0]+Data[1]+\dots+Data[ucSize-3]$.
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 500).

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

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.

```
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 Control Base 0x400F.E000																															
DID0, type RO, offset 0x000, reset -																															
VER								CLASS																							
MAJOR								MINOR																							
PBORCTL, type R/W, offset 0x030, reset 0x0000.7FFD																															
														BORIOR																	
LDOPTL, type R/W, offset 0x034, reset 0x0000.0000																															
														VADJ																	
RIS, type RO, offset 0x050, reset 0x0000.0000																															
								PLLRIS				BORRIS																			
IMC, type R/W, offset 0x054, reset 0x0000.0000																															
								PLLLIM				BORIM																			
MISC, type R/W1C, offset 0x058, reset 0x0000.0000																															
								PLLLMIS				BORMIS																			
RESC, type R/W, offset 0x05C, reset -																															
										SW	WDT	BOR	POR	EXT																	
RCC, type R/W, offset 0x060, reset 0x078E.3AD1																															
PWRDN				ACG		SYSDIV				USESYS		USEPWMDIV		PWMDIV																	
				BYPASS		XTAL				OSCSRC		IOSCDIS		MOSCDIS																	
PLLCFG, type RO, offset 0x064, reset -																															
F										R																					
RCC2, type R/W, offset 0x070, reset 0x0780.2810																															
USERCC2				SYSDIV2																											
PWRDN2				BYPASS2						OSCSRC2																					
DSLPLCLKCFG, type R/W, offset 0x144, reset 0x0780.0000																															
DSDIVORIDE								DSOSCSRC																							
DID1, type RO, offset 0x004, reset -																															
VER				FAM				PARTNO																							
PINCOUNT								TEMP				PKG		ROHS		QUAL															
DC0, type RO, offset 0x008, reset 0x00FF.007F																															
SRAMSZ								FLASHSZ																							
DC1, type RO, offset 0x010, reset 0x0010.30DF																															
MINSYS								MPU		HIB		PWM		PLL		WDT		SWO		SWD		JTAG									
DC2, type RO, offset 0x014, reset 0x070F.5337																															
I2C1				I2C0				COMP2		COMP1		COMP0		SS1		SS10		TIMER3		TIMER2		TIMER1		TIMER0							
								QE1		QE10								UART2		UART1		UART0									
DC3, type RO, offset 0x018, reset 0xBF0.FFFF																															
32KHZ		CCP5		CCP4		CCP3		CCP2		CCP1		CCP0																			
PMMFAULT		C20		C2PLUS		C2MINUS		C10		C1PLUS		C1MINUS		C00		C0PLUS		C0MINUS		PWM5		PWM4		PWM3		PWM2		PWM1		PWM0	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																								
DC4, type RO, offset 0x01C, reset 0x0000.C0FF																																							
CCP7				CCP6				GPIOH				GPIOG				GPIOF				GPIOE				GPIOD				GPIOC				GPIOB				GPIOA			
RCGC0, type R/W, offset 0x100, reset 0x00000040																																							
PWM																																							
HIB																																							
WDT																																							
SCGC0, type R/W, offset 0x110, reset 0x00000040																																							
PWM																																							
HIB																																							
WDT																																							
DCGC0, type R/W, offset 0x120, reset 0x00000040																																							
PWM																																							
HIB																																							
WDT																																							
RCGC1, type R/W, offset 0x104, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
SCGC1, type R/W, offset 0x114, reset 0x00000000																																							
PWM																																							
HIB																																							
WDT																																							
DCGC1, type R/W, offset 0x124, reset 0x00000000																																							
PWM																																							
HIB																																							
WDT																																							
RCGC2, type R/W, offset 0x108, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
RCGC2, type R/W, offset 0x118, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
RCGC2, type R/W, offset 0x128, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
SRCR0, type R/W, offset 0x040, reset 0x00000000																																							
PWM																																							
HIB																																							
WDT																																							
SRCR1, type R/W, offset 0x044, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
SRCR2, type R/W, offset 0x048, reset 0x00000000																																							
I2C1				I2C0				COMP2				COMP1				COMP0				TIMER3				TIMER2				TIMER1				TIMER0							
I2C1				I2C0				QE1				QE10				SSI1				SSI0				UART2				UART1				UART0							
Hibernation Module																																							
Base 0x400F.C000																																							
HIBRTCC, type RO, offset 0x000, reset 0x0000.0000																																							
RTCC																																							
RTCC																																							
HIBRTCM0, type R/W, offset 0x004, reset 0xFFFF.FFFF																																							
RTCM0																																							
RTCM0																																							
HIBRTCM1, type R/W, offset 0x008, reset 0xFFFF.FFFF																																							
RTCM1																																							
RTCM1																																							
HIBRTCLD, type R/W, offset 0x00C, reset 0xFFFF.FFFF																																							
RTCLD																																							
RTCLD																																							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIBCTL, type R/W, offset 0x010, reset 0x8000.0000															
								VABORT	CLK32EN	LOWBATEN	PINWEN	RTCWEN	CLKSEL	HIBREQ	RTCEN
HIBIM, type R/W, offset 0x014, reset 0x0000.0000															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBRIS, type RO, offset 0x018, reset 0x0000.0000															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBMIS, type RO, offset 0x01C, reset 0x0000.0000															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBIC, type R/W1C, offset 0x020, reset 0x0000.0000															
												EXTW	LOWBAT	RTCALT1	RTCALT0
HIBRTCT, type R/W, offset 0x024, reset 0x0000.7FFF															
TRIM															
HIBDATA, type R/W, offset 0x030-0x12C, reset -															
RTD															
RTD															
Internal Memory															
Flash Memory Control Registers (Flash Control Offset)															
Base 0x400F.D000															
FMA, type R/W, offset 0x000, reset 0x0000.0000															
														OFFSET	
OFFSET															
FMD, type R/W, offset 0x004, reset 0x0000.0000															
DATA															
DATA															
FMC, type R/W, offset 0x008, reset 0x0000.0000															
WRKEY															
												COMT	MERASE	ERASE	WRITE
FCRIS, type RO, offset 0x00C, reset 0x0000.0000															
														PRIS	ARIS
FCIM, type R/W, offset 0x010, reset 0x0000.0000															
														PMASK	AMASK
FCMISC, type R/W1C, offset 0x014, reset 0x0000.0000															
														PMISC	AMISC
Internal Memory															
Flash Memory Protection Registers (System Control Offset)															
Base 0x400F.E000															
USEACL, type R/W, offset 0x140, reset 0x31															
														USEC	
FMPRE0, type R/W, offset 0x130 and 0x200, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_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
FMPPE0, type R/W, offset 0x134 and 0x400, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
USER_DBG, type R/W, offset 0x1D0, reset 0xFFFF.FFFE															
NW				DATA								DATA			
DATA												DBG1		DBG0	
USER_REG0, type R/W, offset 0x1E0, reset 0xFFFF.FFFF															
NW				DATA								DATA			
DATA															
USER_REG1, type R/W, offset 0x1E4, reset 0xFFFF.FFFF															
NW				DATA								DATA			
DATA															
FMPRE1, type R/W, offset 0x204, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE2, type R/W, offset 0x208, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPRE3, type R/W, offset 0x20C, reset 0xFFFF.FFFF															
READ_ENABLE															
READ_ENABLE															
FMPPE1, type R/W, offset 0x404, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE2, type R/W, offset 0x408, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
FMPPE3, type R/W, offset 0x40C, reset 0xFFFF.FFFF															
PROG_ENABLE															
PROG_ENABLE															
General-Purpose Input/Outputs (GPIOs)															
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															
GPIODATA, type R/W, offset 0x000, reset 0x0000.0000															
DATA															
GPIODIR, type R/W, offset 0x400, reset 0x0000.0000															
DIR															
GPIOIS, type R/W, offset 0x404, reset 0x0000.0000															
IS															
GPIOIBE, type R/W, offset 0x408, reset 0x0000.0000															
IBE															
GPIOIEV, type R/W, offset 0x40C, reset 0x0000.0000															
IEV															

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIOIM, type R/W, offset 0x410, reset 0x0000.0000															
												IME			
GPIORIS, type RO, offset 0x414, reset 0x0000.0000															
												RIS			
GPIOMIS, type RO, offset 0x418, reset 0x0000.0000															
												MIS			
GPIOICR, type W1C, offset 0x41C, reset 0x0000.0000															
												IC			
GPIOAFSEL, type R/W, offset 0x420, reset -															
												AFSEL			
GPIODR2R, type R/W, offset 0x500, reset 0x0000.00FF															
												DRV2			
GPIODR4R, type R/W, offset 0x504, reset 0x0000.0000															
												DRV4			
GPIODR8R, type R/W, offset 0x508, reset 0x0000.0000															
												DRV8			
GPIOODR, type R/W, offset 0x50C, reset 0x0000.0000															
												ODE			
GPIOPUR, type R/W, offset 0x510, reset -															
												PUE			
GPIOPDR, type R/W, offset 0x514, reset 0x0000.0000															
												PDE			
GPIOSLR, type R/W, offset 0x518, reset 0x0000.0000															
												SRL			
GPIODEN, type R/W, offset 0x51C, reset -															
												DEN			
GPIOLOCK, type R/W, offset 0x520, reset 0x0000.0001															
												LOCK			
												LOCK			
GPIOCR, type -, offset 0x524, reset -															
												CR			
GPIOPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000															
												PID4			
GPIOPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000															
												PID5			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
GPIOPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000																
												PID6				
GPIOPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000																
												PID7				
GPIOPeriphID0, type RO, offset 0xFE0, reset 0x0000.0061																
												PID0				
GPIOPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000																
												PID1				
GPIOPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018																
												PID2				
GPIOPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001																
												PID3				
GPIOCellID0, type RO, offset 0xFF0, reset 0x0000.000D																
												CID0				
GPIOCellID1, type RO, offset 0xFF4, reset 0x0000.00F0																
												CID1				
GPIOCellID2, type RO, offset 0xFF8, reset 0x0000.0005																
												CID2				
GPIOCellID3, type RO, offset 0xFFC, reset 0x0000.00B1																
												CID3				
General-Purpose Timers																
Timer0 base: 0x4003.0000																
Timer1 base: 0x4003.1000																
Timer2 base: 0x4003.2000																
Timer3 base: 0x4003.3000																
GPTMCFG, type R/W, offset 0x000, reset 0x0000.0000																
												GPTMCFG				
GPTMTAMR, type R/W, offset 0x004, reset 0x0000.0000																
												TAAMS	TACMR	TAMR		
GPTMTBMR, type R/W, offset 0x008, reset 0x0000.0000																
												TBAMS	TBCMR	TBMR		
GPTMCTL, type R/W, offset 0x00C, reset 0x0000.0000																
TBPWML				TBEVENT		TBSTALL	TBEN	TAPWML				RTCEN	TAEVENT		TASTALL	TAEN
GPTMIMR, type R/W, offset 0x018, reset 0x0000.0000																
						CBEIM	CBMIM	TBTOIM					RTCIM	CAEIM	CAMIM	TATOIM
GPTMRIS, type RO, offset 0x01C, reset 0x0000.0000																
						CBERIS	CBMRIS	TBTORIS					RTCRIIS	CAERIS	CAMRIS	TATORIS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
GPTMMIS, type RO, offset 0x020, reset 0x0000.0000																		
								CBEMIS		CBMMIS		TBTOMIS		RTCMIS		CAEMIS	CAMMIS	TATOMIS
GPTMICR, type W1C, offset 0x024, reset 0x0000.0000																		
								CBECINT		CBMCINT		TBTOCINT		RTCCINT		CAECINT	CAMCINT	TATOCINT
GPTMTAILR, type R/W, offset 0x028, reset 0xFFFF.FFFF																		
												TAILRH						
												TAILRL						
GPTMTBILR, type R/W, offset 0x02C, reset 0x0000.FFFF																		
												TBILRL						
GPTMTAMATCHR, type R/W, offset 0x030, reset 0xFFFF.FFFF																		
												TAMRH						
												TAMRL						
GPTMTBMATCHR, type R/W, offset 0x034, reset 0x0000.FFFF																		
												TBMRL						
GPTMTAPR, type R/W, offset 0x038, reset 0x0000.0000																		
												TAPSR						
GPTMTBPR, type R/W, offset 0x03C, reset 0x0000.0000																		
												TBPSR						
GPTMTAPMR, type R/W, offset 0x040, reset 0x0000.0000																		
												TAPSMR						
GPTMTBPMR, type R/W, offset 0x044, reset 0x0000.0000																		
												TBPSMR						
GPTMTAR, type RO, offset 0x048, reset 0xFFFF.FFFF																		
												TARH						
												TARL						
GPTMTBR, type RO, offset 0x04C, reset 0x0000.FFFF																		
												TBRL						
Watchdog Timer																		
Base 0x4000.0000																		
WDTLOAD, type R/W, offset 0x000, reset 0xFFFF.FFFF																		
												WDTLoad						
												WDTLoad						
WDTVALUE, type RO, offset 0x004, reset 0xFFFF.FFFF																		
												WDTValue						
												WDTValue						
WDTCTL, type R/W, offset 0x008, reset 0x0000.0000																		
												RESEN		INTEN				
WDTICR, type WO, offset 0x00C, reset -																		
												WDTIntClr						
												WDTIntClr						
WDTRIS, type RO, offset 0x010, reset 0x0000.0000																		
												WDTRIS						

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTMIS, type RO, offset 0x014, reset 0x0000.0000															
															WDTMIS
WDTEST, type R/W, offset 0x418, reset 0x0000.0000															
															STALL
WDTLOCK, type R/W, offset 0xC00, reset 0x0000.0000															
WDTLock															
WDTLock															
WDTPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000															
															PID4
WDTPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000															
															PID5
WDTPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000															
															PID6
WDTPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000															
															PID7
WDTPeriphID0, type RO, offset 0xFE0, reset 0x0000.0005															
															PID0
WDTPeriphID1, type RO, offset 0xFE4, reset 0x0000.0018															
															PID1
WDTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018															
															PID2
WDTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001															
															PID3
WDTPCellID0, type RO, offset 0xFF0, reset 0x0000.000D															
															CID0
WDTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0															
															CID1
WDTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005															
															CID2
WDTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1															
															CID3
Universal Asynchronous Receivers/Transmitters (UARTs)															
UART0 base: 0x4000.C000															
UART1 base: 0x4000.D000															
UART2 base: 0x4000.E000															
UARTDR, type R/W, offset 0x000, reset 0x0000.0000															
															DATA
				OE	BE	PE	FE								

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		
UARTSR/UARTECR, type RO, offset 0x004, reset 0x0000.0000 (Reads)																	
												OE	BE	PE	FE		
UARTSR/UARTECR, type WO, offset 0x004, reset 0x0000.0000 (Writes)																	
												DATA					
UARTFR, type RO, offset 0x018, reset 0x0000.0090																	
										TXFE	RXFF	TXFF	RXFE	BUSY			
UARTILPR, type R/W, offset 0x020, reset 0x0000.0000																	
												ILPDVSR					
UARTIBRD, type R/W, offset 0x024, reset 0x0000.0000																	
												DIVINT					
UARTFBRD, type R/W, offset 0x028, reset 0x0000.0000																	
												DIVFRAC					
UARTLCRH, type R/W, offset 0x02C, reset 0x0000.0000																	
										SPS	WLEN	FEN	STP2	EPS	PEN	BRK	
UARTCTL, type R/W, offset 0x030, reset 0x0000.0300																	
										RXE	TXE	LBE			SIRLP	SIREN	UARTEN
UARTIFLS, type R/W, offset 0x034, reset 0x0000.0012																	
										RXIFLSEL			TXIFLSEL				
UARTIM, type R/W, offset 0x038, reset 0x0000.0000																	
				OEIM	BEIM	PEIM	FEIM	RTIM	TXIM	RXIM							
UARTRIS, type RO, offset 0x03C, reset 0x0000.000F																	
				OERIS	BERIS	PERIS	FERIS	RTRIS	TXRIS	RXRIS							
UARTMIS, type RO, offset 0x040, reset 0x0000.0000																	
				OEMIS	BEMIS	PEMIS	FEMIS	RTMIS	TXMIS	RXMIS							
UARTICR, type W1C, offset 0x044, reset 0x0000.0000																	
				OEIC	BEIC	PEIC	FEIC	RTIC	TXIC	RXIC							
UARTPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000																	
												PID4					
UARTPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000																	
												PID5					
UARTPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000																	
												PID6					
UARTPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000																	
												PID7					

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
UARTPeriphID0, type RO, offset 0xFE0, reset 0x0000.0011																
												PID0				
UARTPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000																
												PID1				
UARTPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018																
												PID2				
UARTPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001																
												PID3				
UARTPCellID0, type RO, offset 0xFF0, reset 0x0000.000D																
												CID0				
UARTPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0																
												CID1				
UARTPCellID2, type RO, offset 0xFF8, reset 0x0000.0005																
												CID2				
UARTPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1																
												CID3				
Synchronous Serial Interface (SSI)																
SSI0 base: 0x4000.8000																
SSI1 base: 0x4000.9000																
SSICR0, type R/W, offset 0x000, reset 0x0000.0000																
SCR						SPH		SPO		FRF		DSS				
SSICR1, type R/W, offset 0x004, reset 0x0000.0000																
												SOD		MS	SSE	LBM
SSIDR, type R/W, offset 0x008, reset 0x0000.0000																
DATA																
SSISR, type RO, offset 0x00C, reset 0x0000.0003																
												BSY	RFF	RNE	TNF	TFE
SSICPSR, type R/W, offset 0x010, reset 0x0000.0000																
												CPDVDSR				
SSIIM, type R/W, offset 0x014, reset 0x0000.0000																
												TXIM	RXIM	RTIM	RORIM	
SSIRIS, type RO, offset 0x018, reset 0x0000.0008																
												TXRIS	RXRIS	RTRIS	RORRIS	
SSIMIS, type RO, offset 0x01C, reset 0x0000.0000																
												TXMIS	RXMIS	RTMIS	RORMIS	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
SSIICR, type W1C, offset 0x020, reset 0x0000.0000																				
														RTIC	RORIC					
SSIPeriphID4, type RO, offset 0xFD0, reset 0x0000.0000																				
														PID4						
SSIPeriphID5, type RO, offset 0xFD4, reset 0x0000.0000																				
														PID5						
SSIPeriphID6, type RO, offset 0xFD8, reset 0x0000.0000																				
														PID6						
SSIPeriphID7, type RO, offset 0xFDC, reset 0x0000.0000																				
														PID7						
SSIPeriphID0, type RO, offset 0xFE0, reset 0x0000.0022																				
														PID0						
SSIPeriphID1, type RO, offset 0xFE4, reset 0x0000.0000																				
														PID1						
SSIPeriphID2, type RO, offset 0xFE8, reset 0x0000.0018																				
														PID2						
SSIPeriphID3, type RO, offset 0xFEC, reset 0x0000.0001																				
														PID3						
SSIPCellID0, type RO, offset 0xFF0, reset 0x0000.000D																				
														CID0						
SSIPCellID1, type RO, offset 0xFF4, reset 0x0000.00F0																				
														CID1						
SSIPCellID2, type RO, offset 0xFF8, reset 0x0000.0005																				
														CID2						
SSIPCellID3, type RO, offset 0xFFC, reset 0x0000.00B1																				
														CID3						
Inter-Integrated Circuit (I²C) Interface																				
I²C Master																				
I2C Master 0 base: 0x4002.0000																				
I2C Master 1 base: 0x4002.1000																				
I2CMSA, type R/W, offset 0x000, reset 0x0000.0000																				
														SA	R/S					
I2CMCS, type RO, offset 0x004, reset 0x0000.0000 (Reads)																				
														BUSBSY	IDLE	ARBLST	DATAACK	ADRACK	ERROR	BUSY
I2CMCS, type WO, offset 0x004, reset 0x0000.0000 (Writes)																				
														ACK	STOP	START	RUN			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
I2CMRDR, type R/W, offset 0x008, reset 0x0000.0000																
												DATA				
I2CMTPR, type R/W, offset 0x00C, reset 0x0000.0001																
												TPR				
I2CMIMR, type R/W, offset 0x010, reset 0x0000.0000																
															IM	
I2CMRIS, type RO, offset 0x014, reset 0x0000.0000																
															RIS	
I2CMMIS, type RO, offset 0x018, reset 0x0000.0000																
															MIS	
I2CMICR, type WO, offset 0x01C, reset 0x0000.0000																
															IC	
I2CMCR, type R/W, offset 0x020, reset 0x0000.0000																
										SFE	MFE					LPBK
Inter-Integrated Circuit (I²C) Interface																
I²C Slave																
I2C Slave 0 base: 0x4002.0800																
I2C Slave 1 base: 0x4002.1800																
I2CSOAR, type R/W, offset 0x000, reset 0x0000.0000																
												OAR				
I2CSCSR, type RO, offset 0x004, reset 0x0000.0000 (Reads)																
												FBR	TREQ	RREQ		
I2CSCSR, type WO, offset 0x004, reset 0x0000.0000 (Writes)																
															DA	
I2CSDR, type R/W, offset 0x008, reset 0x0000.0000																
												DATA				
I2CSIMR, type R/W, offset 0x00C, reset 0x0000.0000																
															DATAIM	
I2CSRIS, type RO, offset 0x010, reset 0x0000.0000																
															DATARIS	
I2CSMIS, type RO, offset 0x014, reset 0x0000.0000																
															DATAMIS	
I2CSICR, type WO, offset 0x018, reset 0x0000.0000																
															DATAIC	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Analog Comparators															
Base 0x4003.C000															
ACMIS, type R/W1C, offset 0x000, reset 0x0000.0000															
													IN2	IN1	IN0
ACRIS, type RO, offset 0x004, reset 0x0000.0000															
													IN2	IN1	IN0
ACINTEN, type R/W, offset 0x008, reset 0x0000.0000															
													IN2	IN1	IN0
ACREFCTL, type R/W, offset 0x010, reset 0x0000.0000															
						EN	RNG								VREF
ACSTAT0, type RO, offset 0x020, reset 0x0000.0000															
														OVAL	
ACSTAT1, type RO, offset 0x040, reset 0x0000.0000															
														OVAL	
ACSTAT2, type RO, offset 0x060, reset 0x0000.0000															
														OVAL	
ACCTL0, type R/W, offset 0x024, reset 0x0000.0000															
						ASRCP				ISLVAL		ISEN		CINV	
ACCTL1, type R/W, offset 0x044, reset 0x0000.0000															
						ASRCP				ISLVAL		ISEN		CINV	
ACCTL2, type R/W, offset 0x064, reset 0x0000.0000															
						ASRCP				ISLVAL		ISEN		CINV	
Pulse Width Modulator (PWM)															
Base 0x4002.8000															
PWMCTL, type R/W, offset 0x000, reset 0x0000.0000															
													GlobalSync2	GlobalSync1	GlobalSync0
PWMSYNC, type R/W, offset 0x004, reset 0x0000.0000															
													Sync2	Sync1	Sync0
PWMENABLE, type R/W, offset 0x008, reset 0x0000.0000															
										PWM5En	PWM4En	PWM3En	PWM2En	PWM1En	PWM0En
PWMINVERT, type R/W, offset 0x00C, reset 0x0000.0000															
										PWM5Inv	PWM4Inv	PWM3Inv	PWM2Inv	PWM1Inv	PWM0Inv
PWMFAULT, type R/W, offset 0x010, reset 0x0000.0000															
										Fault5	Fault4	Fault3	Fault2	Fault1	Fault0
PWMINTEN, type R/W, offset 0x014, reset 0x0000.0000															
															IntFault
													IntPWM2	IntPWM1	IntPWM0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWMRIS, type RO, offset 0x018, reset 0x0000.0000															
															IntFault
													IntPWM2	IntPWM1	IntPWM0
PWMISC, type R/W1C, offset 0x01C, reset 0x0000.0000															
															IntFault
													IntPWM2	IntPWM1	IntPWM0
PWMSTATUS, type RO, offset 0x020, reset 0x0000.0000															
															Fault
PWM0CTL, type R/W, offset 0x040, reset 0x0000.0000															
										CmpBUpd	CmpAUpd	LoadUpd	Debug	Mode	Enable
PWM1CTL, type R/W, offset 0x080, reset 0x0000.0000															
										CmpBUpd	CmpAUpd	LoadUpd	Debug	Mode	Enable
PWM2CTL, type R/W, offset 0x0C0, reset 0x0000.0000															
										CmpBUpd	CmpAUpd	LoadUpd	Debug	Mode	Enable
PWM0INTEN, type R/W, offset 0x044, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM1INTEN, type R/W, offset 0x084, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM2INTEN, type R/W, offset 0x0C4, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0RIS, type RO, offset 0x048, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM1RIS, type RO, offset 0x088, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM2RIS, type RO, offset 0x0C8, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0ISC, type R/W1C, offset 0x04C, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM1ISC, type R/W1C, offset 0x08C, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM2ISC, type R/W1C, offset 0x0CC, reset 0x0000.0000															
										IntCmpBD	IntCmpBU	IntCmpAD	IntCmpAU	IntCntLoad	IntCntZero
PWM0LOAD, type R/W, offset 0x050, reset 0x0000.0000															
															Load
PWM1LOAD, type R/W, offset 0x090, reset 0x0000.0000															
															Load

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM2LOAD, type R/W, offset 0x0D0, reset 0x0000.0000															
Load															
PWM0COUNT, type RO, offset 0x054, reset 0x0000.0000															
Count															
PWM1COUNT, type RO, offset 0x094, reset 0x0000.0000															
Count															
PWM2COUNT, type RO, offset 0x0D4, reset 0x0000.0000															
Count															
PWM0CMPA, type R/W, offset 0x058, reset 0x0000.0000															
CompA															
PWM1CMPA, type R/W, offset 0x098, reset 0x0000.0000															
CompA															
PWM2CMPA, type R/W, offset 0x0D8, reset 0x0000.0000															
CompA															
PWM0CMPB, type R/W, offset 0x05C, reset 0x0000.0000															
CompB															
PWM1CMPB, type R/W, offset 0x09C, reset 0x0000.0000															
CompB															
PWM2CMPB, type R/W, offset 0x0DC, reset 0x0000.0000															
CompB															
PWM0GENA, type R/W, offset 0x060, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM1GENA, type R/W, offset 0x0A0, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM2GENA, type R/W, offset 0x0E0, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM0GENB, type R/W, offset 0x064, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM1GENB, type R/W, offset 0x0A4, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM2GENB, type R/W, offset 0x0E4, reset 0x0000.0000															
				ActCmpBD		ActCmpBU		ActCmpAD		ActCmpAU		ActLoad		ActZero	
PWM0DBCTL, type R/W, offset 0x068, reset 0x0000.0000															
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										
PWM1DBCTL, type R/W, offset 0x0A8, reset 0x0000.0000																									
															Enable										
PWM2DBCTL, type R/W, offset 0x0E8, reset 0x0000.0000																									
															Enable										
PWM0DBRISE, type R/W, offset 0x06C, reset 0x0000.0000																									
															RiseDelay										
PWM1DBRISE, type R/W, offset 0x0AC, reset 0x0000.0000																									
															RiseDelay										
PWM2DBRISE, type R/W, offset 0x0EC, reset 0x0000.0000																									
															RiseDelay										
PWM0DBFALL, type R/W, offset 0x070, reset 0x0000.0000																									
															FallDelay										
PWM1DBFALL, type R/W, offset 0x0B0, reset 0x0000.0000																									
															FallDelay										
PWM2DBFALL, type R/W, offset 0x0F0, reset 0x0000.0000																									
															FallDelay										
Quadrature Encoder Interface (QEI)																									
QEI0 base: 0x4002.C000																									
QEI1 base: 0x4002.D000																									
QEICTL, type R/W, offset 0x000, reset 0x0000.0000																									
															STALLEN	INVI	INVB	INVA	VelDiv	VelEn	ResMode	CapMode	SigMode	Swap	Enable
QEISTAT, type RO, offset 0x004, reset 0x0000.0000																									
															Direction	Error									
QEIPOS, type R/W, offset 0x008, reset 0x0000.0000																									
															Position										
															Position										
QEIMAXPOS, type R/W, offset 0x00C, reset 0x0000.0000																									
															MaxPos										
															MaxPos										
QEILOAD, type R/W, offset 0x010, reset 0x0000.0000																									
															Load										
															Load										
QEITIME, type RO, offset 0x014, reset 0x0000.0000																									
															Time										
															Time										
QEICOUNT, type RO, offset 0x018, reset 0x0000.0000																									
															Count										
															Count										
QEISPEED, type RO, offset 0x01C, reset 0x0000.0000																									
															Speed										
															Speed										

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
QEINTEN, type R/W, offset 0x020, reset 0x0000.0000																
													IntError	IntDir	IntTimer	IntIndex
QEIRIS, type RO, offset 0x024, reset 0x0000.0000																
													IntError	IntDir	IntTimer	IntIndex
QEIISC, type R/W1C, offset 0x028, reset 0x0000.0000																
													IntError	IntDir	IntTimer	IntIndex

C Ordering and Contact Information

C.1 Ordering Information

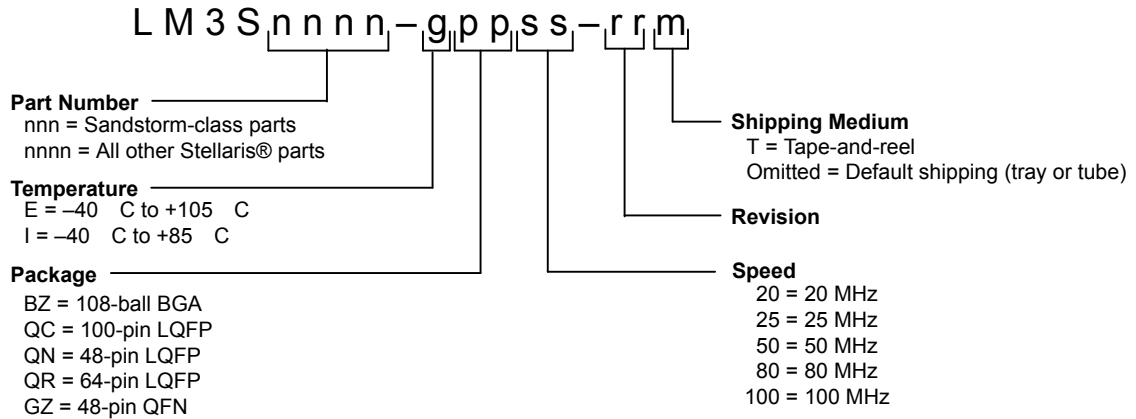


Table C-1. Part Ordering Information

Orderable Part Number	Description
LM3S1960-IBZ50-A2	Stellaris® LM3S1960 Microcontroller Industrial Temperature 108-ball BGA
LM3S1960-IBZ50-A2T	Stellaris® LM3S1960 Microcontroller Industrial Temperature 108-ball BGA Tape-and-reel
LM3S1960-EQC50-A2	Stellaris® LM3S1960 Microcontroller Extended Temperature 100-pin LQFP
LM3S1960-EQC50-A2T	Stellaris® LM3S1960 Microcontroller Extended Temperature 100-pin LQFP Tape-and-reel
LM3S1960-IQC50-A2	Stellaris® LM3S1960 Microcontroller Industrial Temperature 100-pin LQFP
LM3S1960-IQC50-A2T	Stellaris® LM3S1960 Microcontroller Industrial Temperature 100-pin LQFP Tape-and-reel

C.2 Part Markings

The Stellaris® microcontrollers are marked with an identifying number. This code contains the following information:

- The first line indicates the part number. In the example figure below, this is the LM3S6965.
- The first seven characters in the second line indicate the temperature, package, speed, and revision. In the example figure below, this is an Industrial temperature (I), 100-pin LQFP package (QC), 50-MHz (50), revision A2 (A2) device.
- The remaining characters contain internal tracking numbers.



C.3 Kits

The Stellaris[®] 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/kits.html>
- 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/development_kits.html

See the website for the latest tools available, or ask your distributor.

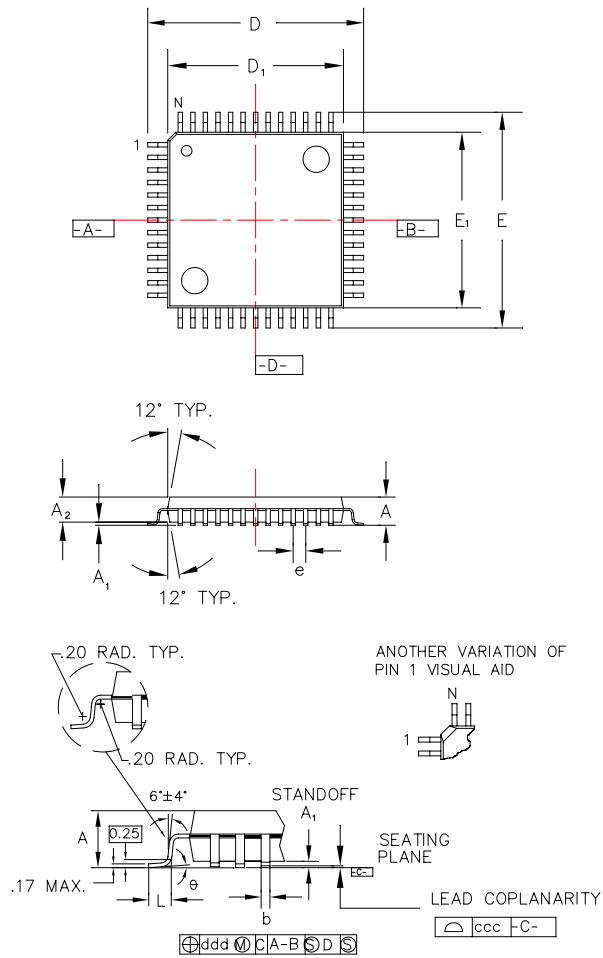
C.4 Support Information

For support on Stellaris[®] products, contact:

support_lmi@ti.com

D Package Information

Figure D-1. 100-Pin LQFP Package

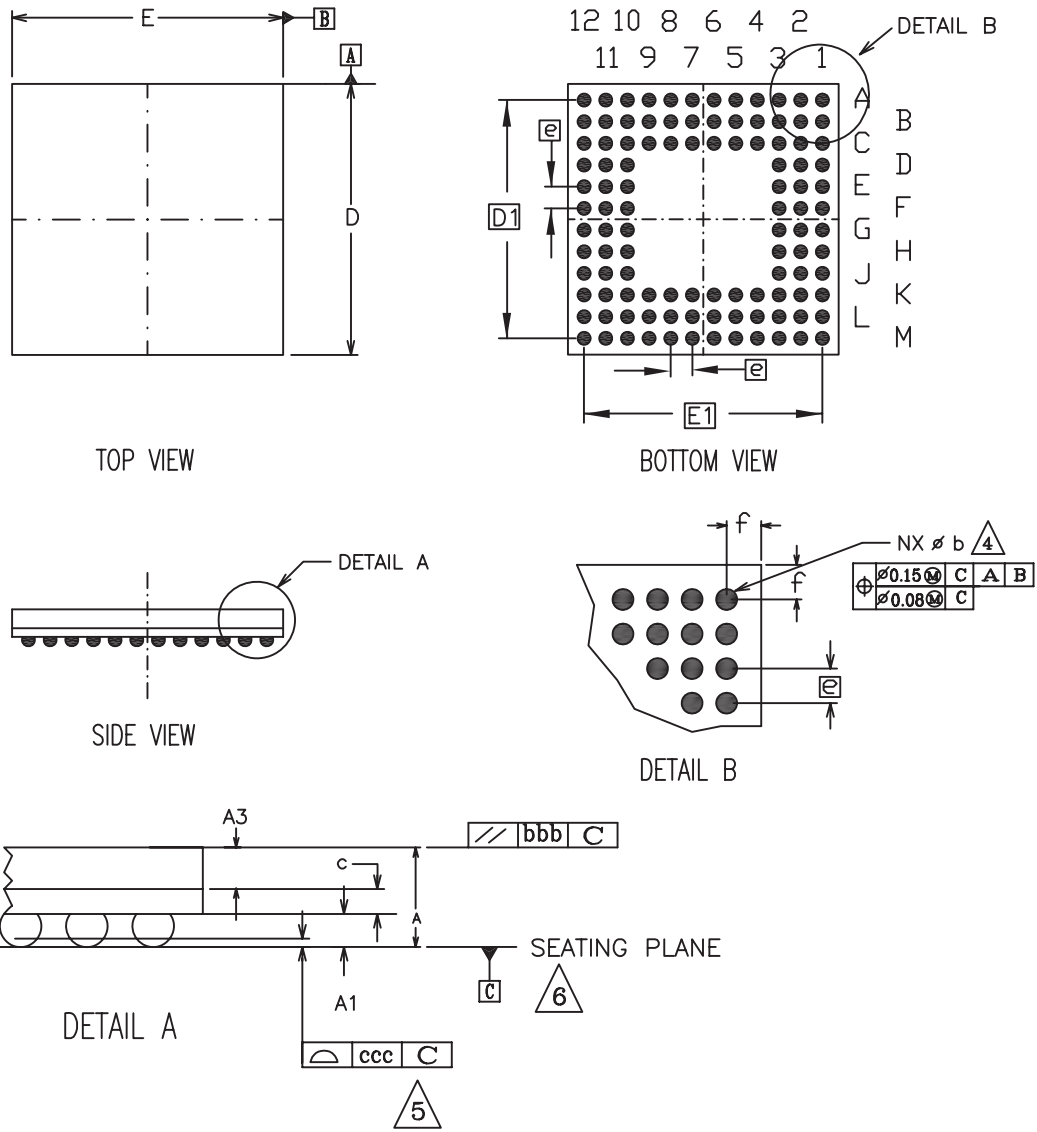


Note: The following notes apply to the package drawing.

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 ₁	-	0.05 Min./0.15 Max.
A ₂	±0.05	1.40
D	±0.20	16.00
D ₁	±0.05	14.00
E	±0.20	16.00
E ₁	±0.05	14.00
L	+0.15/-0.10	0.60
e	Basic	0.50
b	+0.05	0.22
θ	-	0°-7°
ddd	Max.	0.08
ccc	Max.	0.08
JEDEC Reference Drawing		MS-026
Variation Designator		BED

Figure D-2. 108-Ball BGA Package



Note: The following notes apply to the package drawing.

1. ALL DIMENSIONS ARE IN MILLIMETERS.
 2. 'e' REPRESENTS THE BASIC SOLDER BALL GRID PITCH.
 3. 'M' REPRESENTS THE BASIC SOLDER BALL MATRIX SIZE.
AND SYMBOL 'N' IS THE NUMBER OF BALLS AFTER DEPOPULATING.
 4. 'b' IS MEASURABLE AT THE MAXIMUM SOLDER BALL DIAMETER AFTER REFLOW
PARALLEL TO PRIMARY DATUM [C].
 5. DIMENSION 'ccc' IS MEASURED PARALLEL TO PRIMARY DATUM [C].
 6. PRIMARY DATUM [C] AND SEATING PLANE ARE DEFINED BY THE SPHERICAL
CROWNS OF THE SOLDER BALLS.
 7. PACKAGE SURFACE SHALL BE MATTE FINISH CHARMILLES 24 TO 27.
 8. SUBSTRATE MATERIAL BASE IS BT RESIN.
 9. THE OVERALL PACKAGE THICKNESS "A" ALREADY CONSIDERS COLLAPSE BALLS
 10. DIMENSIONING AND TOLERANCING PER ASME Y14.5M 1994.
11. EXCEPT DIMENSION b.

Symbols	MIN	NOM	MAX
A	1.22	1.36	1.50
A1	0.29	0.34	0.39
A3	0.65	0.70	0.75
c	0.28	0.32	0.36
D	9.85	10.00	10.15
D1	8.80 BSC		
E	9.85	10.00	10.15
E1	8.80 BSC		
b	0.43	0.48	0.53
bbb	.20		
ddd	.12		
e	0.80 BSC		
f	-	0.60	-
M	12		
n	108		
REF: JEDEC MO-219F			

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DLP® Products	www.dlp.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
RF/IF and ZigBee® Solutions	www.ti.com/lprf

Applications

Audio	www.ti.com/audio
Automotive	www.ti.com/automotive
Broadband	www.ti.com/broadband
Digital Control	www.ti.com/digitalcontrol
Medical	www.ti.com/medical
Military	www.ti.com/military
Optical Networking	www.ti.com/opticalnetwork
Security	www.ti.com/security
Telephony	www.ti.com/telephony
Video & Imaging	www.ti.com/video
Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2009, Texas Instruments Incorporated