ECCN 5E002 TSPA - Technology / Software Publicly Available.

CC430 Family

User's Guide

Literature Number: SLAU259E
May 2009–Revised January 2013

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Copyright © 2009–2013, Texas Instruments Incorporated

Contents

Preface **33**

1 System Resets, Interrupts, and Operating Modes, System Control Module (SYS) **35**

1.1 System Control Module (SYS) Introduction 36
1.2 System Reset and Initialization 36
1.2.1 Device Initial Conditions After System Reset 38
1.3 Interrupts 38
1.3.1 (Non)Maskable Interrupts (NMIs) 39
1.3.2 SNMI Timing 40
1.3.3 Maskable Interrupts 41
1.3.4 Interrupt Processing 41
1.3.5 Interrupt Nesting 42
1.3.6 Interrupt Vectors 42
1.3.7 SYS Interrupt Vector Generators 43
1.4 Operating Modes 44
1.4.1 Entering and Exiting Low-Power Modes LPM0 Through LPM4 47
1.4.2 Entering and Exiting Low-Power Modes LPMx.5 47
1.4.3 Extended Time in Low-Power Modes 48
1.5 Principles for Low-Power Applications 50
1.6 Connection of Unused Pins 50
1.7 Reset Pin (**RST**/NMI) Configuration 51
1.8 Configuring JTAG pins 51
1.9 Boot Code 51
1.10 Bootstrap Loader (BSL) 51
1.11 Memory Map – Uses and Abilities 53
1.11.1 Vacant Memory Space 53
1.11.2 JTAG Lock Mechanism via the Electronic Fuse 53
1.12 JTAG Mailbox (JMB) System 54
1.12.1 JMB Configuration 54
1.12.2 JMBOUT0 and JMBOUT1 Outgoing Mailbox 54
1.12.3 JMBIN0 and JMBIN1 Incoming Mailbox 54
1.12.4 JMB NMI Usage 55
1.13 Device Descriptor Table 55
1.13.1 Identifying Device Type 56
1.13.2 TLV Descriptors 57
1.13.3 Peripheral Discovery Descriptor 58
1.13.4 CRC Computation 62
1.13.5 Calibration Values 63
1.14 SFR Registers 65
1.14.1 SFRIE1 Register 66
1.14.2 SFRIFG1 Register 67
1.14.3 SFRRPCR Register 69
1.15 SYS Registers 70
1.15.1 SYSCTL Register 71
1.15.2 SYSBSLC Register 72
1.15.3 SYSJMBC Register 73

Copyright © 2009–2013, Texas Instruments Incorporated

1.15.4 SYSJMBI0 Register 74
1.15.5 SYSJMBI1 Register 74
1.15.6 SYSJMBO0 Register 75
1.15.7 SYSJMBO1 Register 75
1.15.8 SYSUNIV Register 76
1.15.9 SYSSNIV Register 77
1.15.10 SYSRSTIV Register 78
1.15.11 SYSBERRIV Register 79

2 Power Management Module and Supply Voltage Supervisor 80
2.1 Power Management Module (PMM) Introduction 81
2.2 PMM Operation 83
2.2.1 V_{CORE} and the Regulator 83
2.2.2 Supply Voltage Supervisor and Monitor 83
2.2.3 Supply Voltage Supervisor and Monitor - Power-Up 89
2.2.4 Increasing V_{CORE} to Support Higher MCLK Frequencies 89
2.2.5 Decreasing V_{CORE} for Power Optimization 91
2.2.6 Transition From LPM3 and LPM4 Modes to AM 91
2.2.7 LPM3.5 and LPM4.5 91
2.2.8 Brownout Reset (BOR), Software BOR, Software POR 91
2.2.9 SVS and SVM Performance Modes and Wakeup Times 92
2.2.10 PMM Interrupts 95
2.2.11 Port I/O Control 95
2.2.12 Supply Voltage Monitor Output (SVMOUT, Optional) 95
2.3 PMM Registers 96
2.3.1 PMMCTL0 Register 97
2.3.2 PMMCTL1 Register 98
2.3.3 SVSMHCTL Register 99
2.3.4 SVSMLCTL Register 100
2.3.5 SVSMIO Register 101
2.3.6 PMMIFG Register 102
2.3.7 PMMRIE Register 104
2.3.8 PM5CTL0 Register 105

3 Unified Clock System (UCS) 106
3.1 Unified Clock System (UCS) Introduction 107
3.2 UCS Operation 109
3.2.1 UCS Module Features for Low-Power Applications 109
3.2.2 Internal Very-Low-Power Low-Frequency Oscillator (VLO) 109
3.2.3 Internal Trimmed Low-Frequency Reference Oscillator (REFO) 110
3.2.4 XT1 Oscillator 110
3.2.5 RF XT2 Oscillator 111
3.2.6 Digitally-Controlled Oscillator (DCO) 111
3.2.7 Frequency Locked Loop (FLL) 111
3.2.8 DCO Modulator 112
3.2.9 Disabling FLL Hardware and Modulator 113
3.2.10 FLL Operation From Low-Power Modes 113
3.2.11 Operation From Low-Power Modes, Requested by Peripheral Modules 113
3.2.12 UCS Module Fail-Safe Operation 115
3.2.13 Synchronization of Clock Signals 118
3.3 Module Oscillator (MODOSC) 119
3.3.1 MODOSC Operation 119
3.4 UCS Module Registers 120
3.4.1 UCSCTL0 Register 121
3.4.2 UCSCTL1 Register 122

Copyright © 2009–2013, Texas Instruments Incorporated

3.4.3 UCSCTL2 Register 123
3.4.4 UCSCTL3 Register 124
3.4.5 UCSCTL4 Register 125
3.4.6 UCSCTL5 Register 126
3.4.7 UCSCTL6 Register 128
3.4.8 UCSCTL7 Register 129
3.4.9 UCSCTL8 Register 130
3.4.10 UCSCTL9 Register 131

4 CPUX 132

4.1 MSP430X CPU (CPUX) Introduction 133
4.2 Interrupts 135
4.3 CPU Registers 136
4.3.1 Program Counter (PC) 136
4.3.2 Stack Pointer (SP) 136
4.3.3 Status Register (SR) 138
4.3.4 Constant Generator Registers (CG1 and CG2) 139
4.3.5 General-Purpose Registers (R4 –R15) 140
4.4 Addressing Modes 142
4.4.1 Register Mode 143
4.4.2 Indexed Mode 144
4.4.3 Symbolic Mode 148
4.4.4 Absolute Mode 153
4.4.5 Indirect Register Mode 155
4.4.6 Indirect Autoincrement Mode 156
4.4.7 Immediate Mode 157
4.5 MSP430 and MSP430X Instructions 159
4.5.1 MSP430 Instructions 159
4.5.2 MSP430X Extended Instructions 164
4.6 Instruction Set Description 175
4.6.1 Extended Instruction Binary Descriptions 176
4.6.2 MSP430 Instructions 178
4.6.3 Extended Instructions 230
4.6.4 Address Instructions 273

5 Flash Memory Controller 288

5.1 Flash Memory Introduction 289
5.2 Flash Memory Segmentation 290
5.2.1 Segment A 291
5.3 Flash Memory Operation 292
5.3.1 Erasing Flash Memory 292
5.3.2 Writing Flash Memory 296
5.3.3 Flash Memory Access During Write or Erase 303
5.3.4 Stopping Write or Erase Cycle 304
5.3.5 Checking Flash Memory 304
5.3.6 Configuring and Accessing the Flash Memory Controller 305
5.3.7 Flash Memory Controller Interrupts 305
5.3.8 Programming Flash Memory Devices 306
5.4 FCTL Registers 307
5.4.1 FCTL1 Register 308
5.4.2 FCTL3 Register 309
5.4.3 FCTL4 Register 310
5.4.4 SFRIE1 Register 311

6 RAM Controller (RAMCTL) 312

6.1 RAM Controller (RAMCTL) Introduction 313

Copyright © 2009–2013, Texas Instruments Incorporated

6.2 RAMCTL Operation ... 313
6.3 RAMCTL Registers ... 314
6.3.1 RCCTL0 Register ... 315
7 Direct Memory Access (DMA) Controller Module ... 316
7.1 Direct Memory Access (DMA) Introduction ... 317
7.2 DMA Operation ... 319
7.2.1 DMA Addressing Modes ... 319
7.2.2 DMA Transfer Modes ... 319
7.2.3 Initiating DMA Transfers ... 325
7.2.4 Halting Executing Instructions for DMA Transfers ... 325
7.2.5 Stopping DMA Transfers ... 326
7.2.6 DMA Channel Priorities ... 326
7.2.7 DMA Transfer Cycle Time ... 327
7.2.8 Using DMA With System Interrupts ... 327
7.2.9 DMA Controller Interrupts ... 327
7.2.10 Using the USCI_B I^2C Module With the DMA Controller ... 329
7.2.11 Using ADC12 With the DMA Controller ... 329
7.2.12 Using DAC12 With the DMA Controller ... 329
7.3 DMA Registers ... 330
7.3.1 DMACTL0 Register ... 332
7.3.2 DMACTL1 Register ... 333
7.3.3 DMACTL2 Register ... 334
7.3.4 DMACTL3 Register ... 335
7.3.5 DMACTL4 Register ... 336
7.3.6 DMAxCTL Register ... 337
7.3.7 DMAxSA Register ... 339
7.3.8 DMAxDA Register ... 340
7.3.9 DMAxSZ Register ... 341
7.3.10 DMAIV Register ... 342
8 Digital I/O Module ... 343
8.1 Digital I/O Introduction ... 344
8.2 Digital I/O Operation ... 345
8.2.1 Input Registers (PxIN) ... 345
8.2.2 Output Registers (PxOUT) ... 345
8.2.3 Direction Registers (PxDIR) ... 345
8.2.4 Pullup or Pulldown Resistor Enable Registers (PxREN) ... 345
8.2.5 Output Drive Strength Registers (PxDS) ... 346
8.2.6 Function Select Registers (PxSEL) ... 346
8.2.7 Port Interrupts ... 346
8.2.8 Configuring Unused Port Pins ... 348
8.3 I/O Configuration and LPMx.5 Low-Power Modes ... 348
8.4 Digital I/O Registers ... 350
8.4.1 P1IV Register ... 356
8.4.2 P2IV Register ... 357
8.4.3 P1IES Register ... 358
8.4.4 P1IE Register ... 358
8.4.5 P1IFG Register ... 358
8.4.6 P2IES Register ... 359
8.4.7 P2IE Register ... 359
8.4.8 P2IFG Register ... 359
8.4.9 PxIN Register ... 360
8.4.10 PxOUT Register ... 360
8.4.11 PxDIR Register ... 360

Copyright © 2009–2013, Texas Instruments Incorporated

8.4.12 PxREN Register 361
8.4.13 PxDS Register 361
8.4.14 PxSEL Register 361

9 Port Mapping Controller 362
9.1 Port Mapping Controller Introduction 363
9.2 Port Mapping Controller Operation 363
9.2.1 Access 363
9.2.2 Mapping 363
9.3 Port Mapping Controller Registers 365
9.3.1 PMAPKEYID Register 366
9.3.2 PMAPCTL Register 366
9.3.3 PxMAPy Register 366

10 Cyclic Redundancy Check (CRC) Module 367
10.1 Cyclic Redundancy Check (CRC) Module Introduction 368
10.2 CRC Standard and Bit Order 368
10.3 CRC Checksum Generation 369
10.3.1 CRC Implementation 369
10.3.2 Assembler Examples 370
10.4 CRC Registers 372
10.4.1 CRCDI Register 373
10.4.2 CRCDIRB Register 373
10.4.3 CRCINIRES Register 374
10.4.4 CRCRESR Register 374

11 AES Accelerator 375
11.1 AES Accelerator Introduction 376
11.2 AES Accelerator Operation 377
11.2.1 Encryption 378
11.2.2 Decryption 379
11.2.3 Decryption Key Generation 380
11.2.4 Using the AES Accelerator With Low-Power Modes 381
11.2.5 AES Accelerator Interrupts 381
11.2.6 Implementing Block Cipher Modes 381
11.3 AES_ACCEL Registers 382
11.3.1 AESACTL0 Register 383
11.3.2 AESACTL1 Register 384
11.3.3 AESASTAT Register 385
11.3.4 AESAKEY Register 386
11.3.5 AESADIN Register 387
11.3.6 AESADOUT Register 387
11.3.7 AESAXDIN Register 388
11.3.8 AESAXIN Register 388

12 Watchdog Timer (WDT_A) 389
12.1 WDT_A Introduction 390
12.2 WDT_A Operation 392
12.2.1 Watchdog Timer Counter (WDTCNT) 392
12.2.2 Watchdog Mode 392
12.2.3 Interval Timer Mode 392
12.2.4 Watchdog Timer Interrupts 392
12.2.5 Clock Fail-Safe Feature 393
12.2.6 Operation in Low-Power Modes 393
12.2.7 Software Examples 393
12.3 WDT_A Registers 394

Copyright © 2009–2013, Texas Instruments Incorporated

12.3.1 WDTCTL Register 395

13 Timer_A 396

13.1 Timer_A Introduction 397
13.2 Timer_A Operation 399
13.2.1 16-Bit Timer Counter 399
13.2.2 Starting the Timer 399
13.2.3 Timer Mode Control 400
13.2.4 Capture/Compare Blocks 403
13.2.5 Output Unit 405
13.2.6 Timer_A Interrupts 409
13.3 Timer_A Registers 411
13.3.1 TAxCTL Register 412
13.3.2 TAxR Register 413
13.3.3 TAxCCTLn Register 414
13.3.4 TAxCCRn Register 416
13.3.5 TAxIV Register 416
13.3.6 TAxEX0 Register 417

14 Real-Time Clock (RTC) Overview 418

14.1 RTC Overview 418

15 Real-Time Clock (RTC_A) 419

15.1 RTC_A Introduction 420
15.2 RTC_A Operation 422
15.2.1 Counter Mode 422
15.2.2 Calendar Mode 422
15.2.3 Real-Time Clock Interrupts 424
15.2.4 Real-Time Clock Calibration 426
15.3 RTC_A Registers 428
15.3.1 RTCCTL0 Register 430
15.3.2 RTCCTL1 Register 431
15.3.3 RTCCTL2 Register 432
15.3.4 RTCCTL3 Register 432
15.3.5 RTCNT1 Register 433
15.3.6 RTCNT2 Register 433
15.3.7 RTCNT3 Register 433
15.3.8 RTCNT4 Register 433
15.3.9 RTCSEC Register – Calendar Mode With Hexadecimal Format 434
15.3.10 RTCSEC Register – Calendar Mode With BCD Format 434
15.3.11 RTCMIN Register – Calendar Mode With Hexadecimal Format 435
15.3.12 RTCMIN Register – Calendar Mode With BCD Format 435
15.3.13 RTCHOUR Register – Calendar Mode With Hexadecimal Format 436
15.3.14 RTCHOUR Register – Calendar Mode With BCD Format 436
15.3.15 RTCDOW Register – Calendar Mode 437
15.3.16 RTCDAY Register – Calendar Mode With Hexadecimal Format 437
15.3.17 RTCDAY Register – Calendar Mode With BCD Format 437
15.3.18 RTCMON Register – Calendar Mode With Hexadecimal Format 438
15.3.19 RTCMON Register – Calendar Mode With BCD Format 438
15.3.20 RTCYEARL Register – Calendar Mode With Hexadecimal Format 439
15.3.21 RTCYEARL Register – Calendar Mode With BCD Format 439
15.3.22 RTCYEARH Register – Calendar Mode With Hexadecimal Format 440
15.3.23 RTCYEARH Register – Calendar Mode With BCD Format 440
15.3.24 RTCAMIN Register – Calendar Mode With Hexadecimal Format 441
15.3.25 RTCAMIN Register – Calendar Mode With BCD Format 441
15.3.26 RTCAHOUR Register – Calendar Mode With Hexadecimal Format 442

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.27 RTCAHOUR Register – Calendar Mode With BCD Format 442
15.3.28 RTCADOW Register 443
15.3.29 RTCADAY Register – Calendar Mode With Hexadecimal Format 443
15.3.30 RTCADAY Register – Calendar Mode With BCD Format 443
15.3.31 RTCPS0CTL Register 444
15.3.32 RTCPS1CTL Register 445
15.3.33 RT0PS Register 446
15.3.34 RT1PS Register 446
15.3.35 RTCIV Register 446

16 Real-Time Clock D (RTC_D) 447

16.1 Real-Time Clock RTC_D Introduction 448
16.2 RTC_D Operation 450
16.2.1 Counter Mode 450
16.2.2 Calendar Mode 450
16.2.3 Real-Time Clock Interrupts 453
16.2.4 Real-Time Clock Calibration 454
16.2.5 Real-Time Clock Operation in LPMx.5 Low-Power Mode 456
16.3 RTC_D Registers 457
16.3.1 RTCCTL0 Register 459
16.3.2 RTCCTL1 Register 460
16.3.3 RTCCTL2 Register 461
16.3.4 RTCCTL3 Register 461
16.3.5 RTCNT1 Register 462
16.3.6 RTCNT2 Register 462
16.3.7 RTCNT3 Register 462
16.3.8 RTCNT4 Register 462
16.3.9 RTCSEC Register – Hexadecimal Format 463
16.3.10 RTCSEC Register – BCD Format 463
16.3.11 RTCMIN Register – Hexadecimal Format 464
16.3.12 RTCMIN Register – BCD Format 464
16.3.13 RTCHOUR Register – Hexadecimal Format 465
16.3.14 RTCHOUR Register – BCD Format 465
16.3.15 RTCDOW Register – Calendar Mode 466
16.3.16 RTCDAY Register – Hexadecimal Format 466
16.3.17 RTCDAY Register – BCD Format 466
16.3.18 RTCMON Register – Hexadecimal Format 467
16.3.19 RTCMON Register – BCD Format 467
16.3.20 RTCYEAR Register – Calendar Mode With Hexadecimal Format 468
16.3.21 RTCYEAR Register – Calendar Mode With BCD Format 468
16.3.22 RTCAMIN Register – Hexadecimal Format 469
16.3.23 RTCAMIN Register – BCD Format 469
16.3.24 RTCAHOUR Register – Hexadecimal Format 470
16.3.25 RTCAHOUR Register – BCD Format 470
16.3.26 RTCADOW Register 471
16.3.27 RTCADAY Register – Hexadecimal Format 471
16.3.28 RTCADAY Register – BCD Format 471
16.3.29 RTCPS0CTL Register 472
16.3.30 RTCPS1CTL Register 473
16.3.31 RTCPS0 Register 474
16.3.32 RTCPS1 Register 474
16.3.33 RTCIV Register 475
16.3.34 BIN2BCD Register 476
16.3.35 BCD2BIN Register 476

Copyright © 2009–2013, Texas Instruments Incorporated

17 32-Bit Hardware Multiplier (MPY32) 477
17.1 32-Bit Hardware Multiplier (MPY32) Introduction 478
17.2 MPY32 Operation 480
17.2.1 Operand Registers 481
17.2.2 Result Registers 482
17.2.3 Software Examples 483
17.2.4 Fractional Numbers 484
17.2.5 Putting It All Together 487
17.2.6 Indirect Addressing of Result Registers 490
17.2.7 Using Interrupts 490
17.2.8 Using DMA 491
17.3 MPY32 Registers 492
17.3.1 MPY32CTL0 Register 494
18 REF 495
18.1 REF Introduction 496
18.2 Principle of Operation 498
18.2.1 Low-Power Operation 498
18.2.2 REFCTL 499
18.2.3 Reference System Requests 500
18.3 REF Registers 502
18.3.1 REFCTL0 Register (offset = 00h) [reset = 0080h] 503
19 ADC10_A 505
19.1 ADC10_A Introduction 506
19.2 ADC10_A Operation 508
19.2.1 10-Bit ADC Core 508
19.2.2 ADC10_A Inputs and Multiplexer 508
19.2.3 Voltage Reference Generator 509
19.2.4 Auto Power Down 509
19.2.5 Sample and Conversion Timing 509
19.2.6 Conversion Result 511
19.2.7 ADC10_A Conversion Modes 511
19.2.8 Window Comparator 516
19.2.9 Using the Integrated Temperature Sensor 517
19.2.10 ADC10_A Grounding and Noise Considerations 518
19.2.11 ADC10_A Interrupts 518
19.3 ADC10_A Registers 520
19.3.1 ADC10CTL0 Register 521
19.3.2 ADC10CTL1 Register 522
19.3.3 ADC10CTL2 Register 524
19.3.4 ADC10MEM0 Register 525
19.3.5 ADC10MEM0 Register, 2s-Complement Format 525
19.3.6 ADC10MCTL0 Register 526
19.3.7 ADC10HI Register 527
19.3.8 ADC10HI Register, 2s-Complement Format 527
19.3.9 ADC10LO Register 528
19.3.10 ADC10LO Register, 2s-Complement Format 528
19.3.11 ADC10IE Register 529
19.3.12 ADC10IFG Register 530
19.3.13 ADC10IV Register 531
20 ADC12_A 532
20.1 ADC12_A Introduction 533
20.2 ADC12_A Operation 536
20.2.1 12-Bit ADC Core 536

Copyright © 2009–2013, Texas Instruments Incorporated

20.2.2 ADC12_A Inputs and Multiplexer 536
20.2.3 Voltage Reference Generator 537
20.2.4 Auto Power Down 538
20.2.5 Sample and Conversion Timing 538
20.2.6 Conversion Memory 540
20.2.7 ADC12_A Conversion Modes 540
20.2.8 Using the Integrated Temperature Sensor 546
20.2.9 ADC12_A Grounding and Noise Considerations 547
20.2.10 ADC12_A Interrupts 548
20.3 ADC12_A Registers 550
20.3.1 ADC12CTL0 Register 552
20.3.2 ADC12CTL1 Register 554
20.3.3 ADC12CTL2 Register 555
20.3.4 ADC12MEMx Register 556
20.3.5 ADC12MCTLx Register 557
20.3.6 ADC12IE Register 558
20.3.7 ADC12IFG Register 560
20.3.8 ADC12IV Register 562

21 Comp_B 563

21.1 Comp_B Introduction 564
21.2 Comp_B Operation 565
21.2.1 Comparator 565
21.2.2 Analog Input Switches 565
21.2.3 Port Logic 565
21.2.4 Input Short Switch 565
21.2.5 Output Filter 566
21.2.6 Reference Voltage Generator 567
21.2.7 Comp_B, Port Disable Register CBPD 568
21.2.8 Comp_B Interrupts 568
21.2.9 Comp_B Used to Measure Resistive Elements 568
21.3 Comp_B Registers 570
21.3.1 CBCTL0 Register 571
21.3.2 CBCTL1 Register 572
21.3.3 CBCTL2 Register 574
21.3.4 CBCTL3 Register 575
21.3.5 CBINT Register 577
21.3.6 CBIV Register 578

22 Universal Serial Communication Interface – UART Mode 579

22.1 Universal Serial Communication Interface (USCI) Overview 580
22.2 USCI Introduction – UART Mode 581
22.3 USCI Operation – UART Mode 583
22.3.1 USCI Initialization and Reset 583
22.3.2 Character Format 583
22.3.3 Asynchronous Communication Format 583
22.3.4 Automatic Baud-Rate Detection 586
22.3.5 IrDA Encoding and Decoding 587
22.3.6 Automatic Error Detection 588
22.3.7 USCI Receive Enable 589
22.3.8 USCI Transmit Enable 589
22.3.9 UART Baud-Rate Generation 590
22.3.10 Setting a Baud Rate 592
22.3.11 Transmit Bit Timing 592
22.3.12 Receive Bit Timing 593

Copyright © 2009–2013, Texas Instruments Incorporated

22.3.13 Typical Baud Rates and Errors 594
22.3.14 Using the USCI Module in UART Mode With Low-Power Modes 597
22.3.15 USCI Interrupts 597
22.4 USCI_A UART Mode Registers 599
22.4.1 UCAxCTL0 Register 600
22.4.2 UCAxCTL1 Register 601
22.4.3 UCAxBR0 Register 602
22.4.4 UCAxBR1 Register 602
22.4.5 UCAxMCTL Register 602
22.4.6 UCAxSTAT Register 603
22.4.7 UCAxRXBUF Register 604
22.4.8 UCAxTXBUF Register 604
22.4.9 UCAxIRTCTL Register 605
22.4.10 UCAxIRRCTL Register 605
22.4.11 UCAxABCTL Register 606
22.4.12 UCAxIE Register 607
22.4.13 UCAxIFG Register 607
22.4.14 UCAxIV Register 608
23 Universal Serial Communication Interface – SPI Mode 609
23.1 Universal Serial Communication Interface (USCI) Overview 610
23.2 USCI Introduction – SPI Mode 611
23.3 USCI Operation – SPI Mode 613
23.3.1 USCI Initialization and Reset 613
23.3.2 Character Format 613
23.3.3 Master Mode 614
23.3.4 Slave Mode 615
23.3.5 SPI Enable 615
23.3.6 Serial Clock Control 616
23.3.7 Using the SPI Mode With Low-Power Modes 616
23.3.8 SPI Interrupts 617
23.4 USCI_A SPI Mode Registers 618
23.4.1 UCAxCTL0 Register 619
23.4.2 UCAxCTL1 Register 620
23.4.3 UCAxBR0 Register 621
23.4.4 UCAxBR1 Register 621
23.4.5 UCAxMCTL Register 621
23.4.6 UCAxSTAT Register 622
23.4.7 UCAxRXBUF Register 623
23.4.8 UCAxTXBUF Register 623
23.4.9 UCAxIE Register 624
23.4.10 UCAxIFG Register 624
23.4.11 UCAxIV Register 625
23.5 USCI_B SPI Mode Registers 626
23.5.1 UCBxCTL0 Register 627
23.5.2 UCBxCTL1 Register 628
23.5.3 UCBxBR0 Register 629
23.5.4 UCBxBR1 Register 629
23.5.5 UCBxMCTL Register 629
23.5.6 UCBxSTAT Register 630
23.5.7 UCBxRXBUF Register 631
23.5.8 UCBxTXBUF Register 631
23.5.9 UCBxIE Register 632
23.5.10 UCBxIFG Register 632

Copyright © 2009–2013, Texas Instruments Incorporated

23.5.11 UCBxIV Register 633

24 Universal Serial Communication Interface – I²C Mode 634

24.1 Universal Serial Communication Interface (USCI) Overview 635
24.2 USCI Introduction – I²C Mode 636
24.3 USCI Operation – I²C Mode 637
24.3.1 USCI Initialization and Reset 638
24.3.2 I²C Serial Data 638
24.3.3 I²C Addressing Modes 640
24.3.4 I²C Module Operating Modes 641
24.3.5 I²C Clock Generation and Synchronization 652
24.3.6 Using the USCI Module in I²C Mode With Low-Power Modes 653
24.3.7 USCI Interrupts in I²C Mode 653
24.4 USCI_B I2C Mode Registers 656
24.4.1 UCBxCTL0 Register 657
24.4.2 UCBxCTL1 Register 658
24.4.3 UCBxBR0 Register 659
24.4.4 UCBxBR1 Register 659
24.4.5 UCBxSTAT Register 660
24.4.6 UCBxRXBUF Register 661
24.4.7 UCBxTXBUF Register 661
24.4.8 UCBxI2COA Register 662
24.4.9 UCBxI2CSA Register 662
24.4.10 UCBxIE Register 663
24.4.11 UCBxIFG Register 664
24.4.12 UCBxIV Register 665

25 CC1101-Based Radio Module (RF1A) 666

25.1 Radio Module Introduction 667
25.2 Radio Interface Operation 668
25.2.1 Radio Interface 668
25.2.2 Radio Interface Interrupts 674
25.2.3 Radio Core Interrupts 674
25.2.4 Using Radio With Low-Power Modes 675
25.2.5 Radio Interrupt Handling 675
25.2.6 Software Considerations 678
25.3 CC1101-Based Radio Core 680
25.3.1 Differences From CC1101 680
25.3.2 Instruction Set for CC1101-Based Radio Core 681
25.3.3 Radio-Core Operation 684
25.3.4 System Considerations and Guidelines 705
25.3.5 Radio Core Registers 709
25.4 RF1A Registers 732
25.4.1 RF1AIFCTL0 Register 734
25.4.2 RF1AIFCTL1 Register 735
25.4.3 RF1AIFERR Register 736
25.4.4 RF1AIFERRV Register 737
25.4.5 RF1AIFIV Register 737
25.4.6 RF1AIN Register 738
25.4.7 RF1AIFG Register 738
25.4.8 RF1AIES Register 739
25.4.9 RF1AIE Register 739
25.4.10 RF1AIV Register 740

26 LCD_B Controller 741

26.1 LCD_B Controller Introduction 742

Copyright © 2009–2013, Texas Instruments Incorporated

26.2 LCD_B Controller Operation 744
26.2.1 LCD Memory 744
26.2.2 LCD Timing Generation 744
26.2.3 Blanking the LCD 745
26.2.4 LCD Blinking 745
26.2.5 LCD_B Voltage And Bias Generation 746
26.2.6 LCD Outputs 748
26.2.7 LCD_B Interrupts 748
26.2.8 Static Mode 750
26.2.9 2-Mux Mode 753
26.2.10 3-Mux Mode 756
26.2.11 4-Mux Mode 759
26.3 LCD_B Registers 762
26.3.1 LCDBCTL0 Register 765
26.3.2 LCDBCTL1 Register 766
26.3.3 LCDBBLKCTL Register 767
26.3.4 LCDBMEMCTL Register 768
26.3.5 LCDBVCTL Register 769
26.3.6 LCDBPCTL0 Register 771
26.3.7 LCDBPCTL1 Register 771
26.3.8 LCDBPCTL2 Register 772
26.3.9 LCDBPCTL3 Register 772
26.3.10 LCDBCPCTL Register 773
26.3.11 LCDBIV Register 774
27 Embedded Emulation Module (EEM) 775
27.1 Embedded Emulation Module (EEM) Introduction 776
27.2 EEM Building Blocks 778
27.2.1 Triggers 778
27.2.2 Trigger Sequencer 778
27.2.3 State Storage (Internal Trace Buffer) 778
27.2.4 Cycle Counter 778
27.2.5 Clock Control 779
27.3 EEM Configurations 779
Revision History 780

Copyright © 2009–2013, Texas Instruments Incorporated

List of Figures

1-1. BOR/POR/PUC Reset Circuit 37
1-2. Interrupt Priority 39
1-3. NMIs With Reentrance Protection 40
1-4. Interrupt Processing 41
1-5. Return From Interrupt 42
1-6. Operation Modes 45
1-7. Devices Descriptor Table 56
1-8. SFRIE1 Register 66
1-9. SFRIFG1 Register 67
1-10. SFRRPCR Register 69
1-11. SYSCTL Register 71
1-12. SYSBSLC Register 72
1-13. SYSJMBC Register 73
1-14. SYSJMBI0 Register 74
1-15. SYSJMBI1 Register 74
1-16. SYSJMBO0 Register 75
1-17. SYSJMBO1 Register 75
1-18. SYSUNIV Register 76
1-19. SYSSNIV Register 77
1-20. SYSRSTIV Register 78
1-21. SYSBERRIV Register 79
2-1. System Frequency, Supply Voltage, and Core Voltage – See Device-Specific Data Sheet 81
2-2. PMM Block Diagram 82
2-3. Available SVM_H Settings Versus VCORE Settings 85
2-4. High-Side and Low-Side Voltage Failure and Resulting PMM Actions 86
2-5. High-Side SVS and SVM 87
2-6. Low-Side SVS and SVM 88
2-7. PMM Action at Device Power-Up 89
2-8. Changing V_{CORE} and SVM_L and SVS_L Levels 90
2-9. PMMCTL0 Register 97
2-10. PMMCTL1 Register 98
2-11. SVSMHCTL Register 99
2-12. SVSMLCTL Register 100
2-13. SVSMIO Register 101
2-14. PMMIFG Register 102
2-15. PMMRIE Register 104
2-16. PM5CTL0 Register 105
3-1. UCS Block Diagram 108
3-2. Modulator Patterns 112
3-3. Module Request Clock System 114
3-4. Oscillator Fault Logic 117
3-5. Switch MCLK from DCOCLK to XT1CLK 118
3-6. UCSCTL0 Register 121
3-7. UCSCTL1 Register 122
3-8. UCSCTL2 Register 123
3-9. UCSCTL3 Register 124
3-10. UCSCTL4 Register 125

Copyright © 2009–2013, Texas Instruments Incorporated

3-11. UCSCTL5 Register 126
3-12. UCSCTL6 Register 128
3-13. UCSCTL7 Register 129
3-14. UCSCTL8 Register 130
3-15. UCSCTL9 Register 131
4-1. MSP430X CPU Block Diagram 134
4-2. PC Storage on the Stack for Interrupts 135
4-3. Program Counter 136
4-4. PC Storage on the Stack for CALLA 136
4-5. Stack Pointer 137
4-6. Stack Usage 137
4-7. PUSHX.A Format on the Stack 137
4-8. PUSH SP, POP SP Sequence 137
4-9. SR Bits 138
4-10. Register-Byte and Byte-Register Operation 140
4-11. Register-Word Operation 140
4-12. Word-Register Operation 141
4-13. Register – Address-Word Operation 141
4-14. Address-Word – Register Operation 142
4-15. Indexed Mode in Lower 64 KB 144
4-16. Indexed Mode in Upper Memory 145
4-17. Overflow and Underflow for Indexed Mode 146
4-18. Example for Indexed Mode 147
4-19. Symbolic Mode Running in Lower 64 KB 149
4-20. Symbolic Mode Running in Upper Memory 150
4-21. Overflow and Underflow for Symbolic Mode 151
4-22. MSP430 Double-Operand Instruction Format 159
4-23. MSP430 Single-Operand Instructions 160
4-24. Format of Conditional Jump Instructions 161
4-25. Extension Word for Register Modes 164
4-26. Extension Word for Non-Register Modes 164
4-27. Example for Extended Register or Register Instruction 165
4-28. Example for Extended Immediate or Indexed Instruction 166
4-29. Extended Format I Instruction Formats 167
4-30. 20-Bit Addresses in Memory 167
4-31. Extended Format II Instruction Format 168
4-32. PUSHM and POPM Instruction Format 169
4-33. RRCM, RRAM, RRUM, and RLAM Instruction Format 169
4-34. BRA Instruction Format 169
4-35. CALLA Instruction Format 169
4-36. Decrement Overlap 195
4-37. Stack After a RET Instruction 214
4-38. Destination Operand—Arithmetic Shift Left 216
4-39. Destination Operand—Carry Left Shift 217
4-40. Rotate Right Arithmetically RRA.B and RRA.W 218
4-41. Rotate Right Through Carry RRC.B and RRC.W 219
4-42. Swap Bytes in Memory 226
4-43. Swap Bytes in a Register 226
4-44. Rotate Left Arithmetically—RLAM[.W] and RLAM.A 253

Copyright © 2009–2013, Texas Instruments Incorporated

4-45. Destination Operand-Arithmetic Shift Left 254
4-46. Destination Operand-Carry Left Shift 255
4-47. Rotate Right Arithmetically RRAM[.W] and RRAM.A 256
4-48. Rotate Right Arithmetically RRAX(.B,.A) – Register Mode 258
4-49. Rotate Right Arithmetically RRAX(.B,.A) – Non-Register Mode 258
4-50. Rotate Right Through Carry RRCM[.W] and RRCM.A 260
4-51. Rotate Right Through Carry RRCX(.B,.A) – Register Mode 262
4-52. Rotate Right Through Carry RRCX(.B,.A) – Non-Register Mode 262
4-53. Rotate Right Unsigned RRUM[.W] and RRUM.A 263
4-54. Rotate Right Unsigned RRUX(.B,.A) – Register Mode 264
4-55. Swap Bytes SWPBX.A Register Mode 268
4-56. Swap Bytes SWPBX.A In Memory 268
4-57. Swap Bytes SWPBX[.W] Register Mode 269
4-58. Swap Bytes SWPBX[.W] In Memory 269
4-59. Sign Extend SXTX.A 270
4-60. Sign Extend SXTX[.W] 270
5-1. Flash Memory Module Block Diagram 289
5-2. 256-KB Flash Memory Segments Example 290
5-3. Erase Cycle Timing 293
5-4. Erase Cycle From Flash 294
5-5. Erase Cycle From RAM 295
5-6. Byte, Word, and Long-Word Write Timing 296
5-7. Initiating a Byte or Word Write From Flash 297
5-8. Initiating a Byte or Word Write From RAM 298
5-9. Initiating Long-Word Write From Flash 299
5-10. Initiating Long-Word Write from RAM 300
5-11. Block-Write Cycle Timing 301
5-12. Block Write Flow 302
5-13. User-Developed Programming Solution 306
5-14. FCTL1 Register 308
5-15. FCTL3 Register 309
5-16. FCTL4 Register 310
5-17. SFRIE1 Register 311
6-1. RCCTL0 Register 315
7-1. DMA Controller Block Diagram 318
7-2. DMA Addressing Modes 319
7-3. DMA Single Transfer State Diagram 321
7-4. DMA Block Transfer State Diagram 322
7-5. DMA Burst-Block Transfer State Diagram 324
7-6. DMACTL0 Register 332
7-7. DMACTL1 Register 333
7-8. DMACTL2 Register 334
7-9. DMACTL3 Register 335
7-10. DMACTL4 Register 336
7-11. DMAxCTL Register 337
7-12. DMAxSA Register 339
7-13. DMAxDA Register 340
7-14. DMAxSZ Register 341
7-15. DMAIV Register 342

Copyright © 2009–2013, Texas Instruments Incorporated

8-1. P1IV Register 356
8-2. P2IV Register 357
8-3. P1IES Register 358
8-4. P1IE Register 358
8-5. P1IFG Register 358
8-6. P2IES Register 359
8-7. P2IE Register 359
8-8. P2IFG Register 359
8-9. PxIN Register 360
8-10. PxOUT Register 360
8-11. PxDIR Register 360
8-12. PxREN Register 361
8-13. PxDS Register 361
8-14. PxSEL Register 361
9-1. PMAPKEYID Register 366
9-2. PMAPCTL Register 366
9-3. PxMAPy Register 366
10-1. LFSR Implementation of CRC-CCITT Standard, Bit 0 is the MSB of the Result 368
10-2. Implementation of CRC-CCITT Using the CRCDI and CRCINIRES Registers 370
10-3. CRCDI Register 373
10-4. CRCDIRB Register 373
10-5. CRCINIRES Register 374
10-6. CRCRESR Register 374
11-1. AES Accelerator Block Diagram 376
11-2. AES State Array Input and Output 377
11-3. AES-128 Encryption Process 378
11-4. AES-128 Decryption Process using AESOPx = 01 379
11-5. AES-128 Decryption Process using AESOPx = 10 and 11 380
11-6. AESACTL0 Register 383
11-7. AESACTL1 Register 384
11-8. AESASTAT Register 385
11-9. AESAKEY Register 386
11-10. AESADIN Register 387
11-11. AESADOUT Register 387
11-12. AESAXDIN Register 388
11-13. AESAXIN Register 388
12-1. Watchdog Timer Block Diagram 391
12-2. WDTCTL Register 395
13-1. Timer_A Block Diagram 398
13-2. Up Mode 400
13-3. Up Mode Flag Setting 400
13-4. Continuous Mode 401
13-5. Continuous Mode Flag Setting 401
13-6. Continuous Mode Time Intervals 401
13-7. Up/Down Mode 402
13-8. Up/Down Mode Flag Setting 402
13-9. Output Unit in Up/Down Mode 403
13-10. Capture Signal (SCS = 1) 404
13-11. Capture Cycle 404

Copyright © 2009–2013, Texas Instruments Incorporated

13-12. Output Example – Timer in Up Mode 406
13-13. Output Example – Timer in Continuous Mode 407
13-14. Output Example – Timer in Up/Down Mode 408
13-15. Capture/Compare TAxCCR0 Interrupt Flag 409
13-16. TAxCTL Register 412
13-17. TAxR Register 413
13-18. TAxCCTLn Register 414
13-19. TAxCCRn Register 416
13-20. TAxIV Register 416
13-21. TAxEX0 Register 417
15-1. RTC_A 421
15-2. RTCCTL0 Register 430
15-3. RTCCTL1 Register 431
15-4. RTCCTL2 Register 432
15-5. RTCCTL3 Register 432
15-6. RTCNT1 Register 433
15-7. RTCNT2 Register 433
15-8. RTCNT3 Register 433
15-9. RTCNT4 Register 433
15-10. RTCSEC Register 434
15-11. RTCSEC Register 434
15-12. RTCMIN Register 435
15-13. RTCMIN Register 435
15-14. RTCHOUR Register 436
15-15. RTCHOUR Register 436
15-16. RTCDOW Register 437
15-17. RTCDAY Register 437
15-18. RTCDAY Register 437
15-19. RTCMON Register 438
15-20. RTCMON Register 438
15-21. RTCYEARL Register 439
15-22. RTCYEARL Register 439
15-23. RTCYEARH Register 440
15-24. RTCYEARH Register 440
15-25. RTCAMIN Register 441
15-26. RTCAMIN Register 441
15-27. RTCAHOUR Register 442
15-28. RTCAHOUR Register 442
15-29. RTCADOW Register 443
15-30. RTCADAY Register 443
15-31. RTCADAY Register 443
15-32. RTCPS0CTL Register 444
15-33. RTCPS1CTL Register 445
15-34. RT0PS Register 446
15-35. RTPS1 Register 446
15-36. RTCIV Register 446
16-1. RTC_D Block Diagram 449
16-2. RTCCTL0 Register 459
16-3. RTCCTL1 Register 460

Copyright © 2009–2013, Texas Instruments Incorporated

16-4. RTCCTL2 Register 461
16-5. RTCCTL3 Register 461
16-6. RTCNT1 Register 462
16-7. RTCNT2 Register 462
16-8. RTCNT3 Register 462
16-9. RTCNT4 Register 462
16-10. RTCSEC Register 463
16-11. RTCSEC Register 463
16-12. RTCMIN Register 464
16-13. RTCMIN Register 464
16-14. RTCHOUR Register 465
16-15. RTCHOUR Register 465
16-16. RTCDOW Register 466
16-17. RTCDAY Register 466
16-18. RTCDAY Register 466
16-19. RTCMON Register 467
16-20. RTCMON Register 467
16-21. RTCYEAR Register 468
16-22. RTCYEAR Register 468
16-23. RTCAMIN Register 469
16-24. RTCAMIN Register 469
16-25. RTCAHOUR Register 470
16-26. RTCAHOUR Register 470
16-27. RTCADOW Register 471
16-28. RTCADAY Register 471
16-29. RTCADAY Register 471
16-30. RTCPS0CTL Register 472
16-31. RTCPS1CTL Register 473
16-32. RTCPS0 Register 474
16-33. RTCPS1 Register 474
16-34. RTCIV Register 475
16-35. BIN2BCD Register 476
16-36. BCD2BIN Register 476
17-1. MPY32 Block Diagram 479
17-2. Q15 Format Representation 484
17-3. Q14 Format Representation 484
17-4. Saturation Flow Chart 486
17-5. Multiplication Flow Chart 488
17-6. MPY32CTL0 Register 494
18-1. REF Block Diagram 497
18-2. REFCTL0 Register 503
19-1. ADC10_A Block Diagram 507
19-2. Analog Multiplexer 508
19-3. Extended Sample Mode 510
19-4. Pulse Sample Mode 510
19-5. Analog Input Equivalent Circuit 511
19-6. Single-Channel Single-Conversion Mode 512
19-7. Sequence-of-Channels Mode 513
19-8. Repeat-Single-Channel Mode 514

Copyright © 2009–2013, Texas Instruments Incorporated

19-9. Repeat-Sequence-of-Channels Mode 515
19-10. Typical Temperature Sensor Transfer Function 517
19-11. ADC10_A Grounding and Noise Considerations 518
19-12. ADC10CTL0 Register 521
19-13. ADC10CTL1 Register 522
19-14. ADC10CTL2 Register 524
19-15. ADC10MEM0 Register 525
19-16. ADC10MEM0 Register 525
19-17. ADC10MCTL0 Register 526
19-18. ADC10HI Register 527
19-19. ADC10HI Register 527
19-20. ADC10LO Register 528
19-21. ADC10LO Register 528
19-22. ADC10IE Register 529
19-23. ADC10IFG Register 530
19-24. ADC10IV Register 531
20-1. ADC12_A Block Diagram (Devices With REF Module) 534
20-2. ADC12_A MSP430F54xx (non-A) Block Diagram 535
20-3. Analog Multiplexer 536
20-4. Extended Sample Mode 538
20-5. Pulse Sample Mode 539
20-6. Analog Input Equivalent Circuit 539
20-7. Single-Channel Single-Conversion Mode 541
20-8. Sequence-of-Channels Mode 542
20-9. Repeat-Single-Channel Mode 543
20-10. Repeat-Sequence-of-Channels Mode 544
20-11. Typical Temperature Sensor Transfer Function 546
20-12. ADC12_A Grounding and Noise Considerations 547
20-13. ADC12CTL0 Register 552
20-14. ADC12CTL1 Register 554
20-15. ADC12CTL2 Register 555
20-16. ADC12MEMx Register 556
20-17. ADC12MCTLx Register 557
20-18. ADC12IE Register 558
20-19. ADC12IFG Register 560
20-20. ADC12IV Register 562
21-1. Comp_B Block Diagram 564
21-2. Comp_B Sample-And-Hold 566
21-3. RC-Filter Response at the Output of the Comparator 567
21-4. Reference Generator Block Diagram 567
21-5. Transfer Characteristic and Power Dissipation in a CMOS Inverter/Buffer 568
21-6. Temperature Measurement System 568
21-7. Timing for Temperature Measurement Systems 569
21-8. CBCTL0 Register 571
21-9. CBCTL1 Register 572
21-10. CBCTL2 Register 574
21-11. CBCTL3 Register 575
21-12. CBINT Register 577
21-13. CBIV Register 578

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22-1. USCI_Ax Block Diagram – UART Mode (UCSYNC = 0) 582
22-2. Character Format 583
22-3. Idle-Line Format 584
22-4. Address-Bit Multiprocessor Format 585
22-5. Auto Baud-Rate Detection – Break/Synch Sequence 586
22-6. Auto Baud-Rate Detection – Synch Field 586
22-7. UART vs IrDA Data Format 587
22-8. Glitch Suppression, USCI Receive Not Started 589
22-9. Glitch Suppression, USCI Activated 589
22-10. BITCLK Baud-Rate Timing With UCOS16 = 0 590
22-11. Receive Error 593
22-12. UCAxCTL0 Register 600
22-13. UCAxCTL1 Register 601
22-14. UCAxBR0 Register 602
22-15. UCAxBR1 Register 602
22-16. UCAxMCTL Register 602
22-17. UCAxSTAT Register 603
22-18. UCAxRXBUF Register 604
22-19. UCAxTXBUF Register 604
22-20. UCAxIRTCTL Register 605
22-21. UCAxIRRCTL Register 605
22-22. UCAxABCTL Register 606
22-23. UCAxIE Register 607
22-24. UCAxIFG Register 607
22-25. UCAxIV Register 608
23-1. USCI Block Diagram – SPI Mode 612
23-2. USCI Master and External Slave 614
23-3. USCI Slave and External Master 615
23-4. USCI SPI Timing With UCMSB = 1 616
23-5. UCAxCTL0 Register 619
23-6. UCAxCTL1 Register 620
23-7. UCAxBR0 Register 621
23-8. UCAxBR1 Register 621
23-9. UCAxMCTL Register 621
23-10. UCAxSTAT Register 622
23-11. UCAxRXBUF Register 623
23-12. UCAxTXBUF Register 623
23-13. UCAxIE Register 624
23-14. UCAxIFG Register 624
23-15. UCAxIV Register 625
23-16. UCBxCTL0 Register 627
23-17. UCBxCTL1 Register 628
23-18. UCBxBR0 Register 629
23-19. UCBxBR1 Register 629
23-20. UCBxMCTL Register 629
23-21. UCBxSTAT Register 630
23-22. UCBxRXBUF Register 631
23-23. UCBxTXBUF Register 631
23-24. UCBxIE Register 632

Copyright © 2009–2013, Texas Instruments Incorporated

23-25. UCBxIFG Register 632
23-26. UCBxIV Register 633
24-1. USCI Block Diagram – I²C Mode 637
24-2. I²C Bus Connection Diagram 638
24-3. I²C Module Data Transfer 639
24-4. Bit Transfer on I²C Bus 639
24-5. I²C Module 7-Bit Addressing Format 640
24-6. I²C Module 10-Bit Addressing Format 640
24-7. I²C Module Addressing Format With Repeated START Condition 640
24-8. I²C Time-Line Legend 641
24-9. I²C Slave Transmitter Mode 642
24-10. I²C Slave Receiver Mode 644
24-11. I²C Slave 10-Bit Addressing Mode 645
24-12. I²C Master Transmitter Mode 647
24-13. I²C Master Receiver Mode 649
24-14. I²C Master 10-Bit Addressing Mode 650
24-15. Arbitration Procedure Between Two Master Transmitters 651
24-16. Synchronization of Two I²C Clock Generators During Arbitration 652
24-17. UCBxCTL0 Register 657
24-18. UCBxCTL1 Register 658
24-19. UCBxBR0 Register 659
24-20. UCBxBR1 Register 659
24-21. UCBxSTAT Register 660
24-22. UCBxRXBUF Register 661
24-23. UCBxTXBUF Register 661
24-24. UCBxI2COA Register 662
24-25. UCBxI2CSA Register 662
24-26. UCBxIE Register 663
24-27. UCBxIFG Register 664
24-28. UCBxIV Register 665
25-1. Simplified Block Diagram of the CC1101-Based Radio Module 667
25-2. Radio Interface Overview 668
25-3. Logical Channels Between Radio Interface and Radio Core 669
25-4. Data Whitening in TX Mode 688
25-5. Packet Format 688
25-6. Packet Length Greater Than 255 690
25-7. Complete Radio-Control State Diagram 697
25-8. Event 0 and Event 1 Relationship 699
25-9. FIFO_THR = 13 vs Number of Bytes in FIFO (GDOx_CFG = 0x00 in RX and GDOx_CFG = 0x02 in TX) 702
25-10. Example of FIFOs at Threshold 702
25-11. PA_POWER and PATABLE 704
25-12. Shaping of ASK Signal 704
25-13. Block Diagram of CC1101-based Radio With External Power Amplifier 708
25-14. RF1AIFCTL0 Register 734
25-15. RF1AIFCTL1 Register 735
25-16. RF1AIFERR Register 736
25-17. RF1AIFERRV Register 737
25-18. RF1AIFIV Register 737

Copyright © 2009–2013, Texas Instruments Incorporated

25-19. RF1AIN Register 738
25-20. RF1AIFG Register 738
25-21. RF1AIES Register 739
25-22. RF1AIE Register 739
25-23. RF1AIV Register 740
26-1. LCD_B Controller Block Diagram 743
26-2. LCD Memory - Example for 160 Segments Maximum 744
26-3. Bias Generation 747
26-4. Example Static Waveforms 750
26-5. Static LCD Example (MAB addresses need to be replaced with LCDMx) 751
26-6. Example 2-Mux Waveforms 753
26-7. 2-Mux LCD Example (MAB addresses need to be replaced with LCDMx) 754
26-8. Example 3-Mux Waveforms 756
26-9. 3-Mux LCD Example (MAB addresses need to be replaced with LCDMx) 757
26-10. Example 4-Mux Waveforms 759
26-11. 4-Mux LCD Example (MAB addresses need to be replaced with LCDMx) 760
26-12. LCDBCTL0 Register 765
26-13. LCDBCTL1 Register 766
26-14. LCDBBLKCTL Register 767
26-15. LCDBMEMCTL Register 768
26-16. LCDBVCTL Register 769
26-17. LCDBPCTL0 Register 771
26-18. LCDBPCTL1 Register 771
26-19. LCDBPCTL2 Register 772
26-20. LCDBPCTL3 Register 772
26-21. LCDBCPCTL Register 773
26-22. LCDBIV Register 774
27-1. Large Implementation of EEM 777

Copyright © 2009–2013, Texas Instruments Incorporated

List of Tables

1-1. Interrupt Sources, Flags, and Vectors 42
1-2. Operation Modes 46
1-3. Connection of Unused Pins 50
1-4. Tag Values 57
1-5. Peripheral Discovery Descriptor 58
1-6. Values for Memory Entry 58
1-7. Values for Peripheral Entry 59
1-8. Peripheral IDs 59
1-9. Sample Peripheral Discovery Descriptor 61
1-10. SFR Base Address 65
1-11. SFR Registers 65
1-12. SFRIE1 Register Description 66
1-13. SFRIFG1 Register Description 67
1-14. SFRRPCR Register Description 69
1-15. SYS Base Address 70
1-16. SYS Registers 70
1-17. SYSCTL Register Description 71
1-18. SYSBSLC Register Description 72
1-19. SYSJMBC Register Description 73
1-20. SYSJMBI0 Register Description 74
1-21. SYSJMBI1 Register Description 74
1-22. SYSJMBO0 Register Description 75
1-23. SYSJMBO1 Register Description 75
1-24. SYSUNIV Register Description 76
1-25. SYSSNIV Register Description 77
1-26. SYSRSTIV Register Description 78
1-27. SYSBERRIV Register Description 79
2-1. SVS and SVM Thresholds 84
2-2. Recommended SVS_L Settings 84
2-3. Recommended SVS_H Settings 84
2-4. Available SVS_H and SVM_H Settings Versus V_{CORE} Settings 85
2-5. SVS_L and SVM_L Control Mode Selection 93
2-6. SVS_L Automatic Performance Control 93
2-7. SVS_L Manual Performance Modes 93
2-8. SVM_L Automatic Performance Control 93
2-9. SVM_L Manual Performance Modes 93
2-10. SVS_H and SVM_H Control Mode Selection 94
2-11. SVS_H Automatic Performance Control 94
2-12. SVS_H Manual Performance Modes 94
2-13. SVM_H Automatic Performance Control 94
2-14. SVM_H Manual Performance Modes 94
2-15. PMM Registers 96
2-16. PMMCTL0 Register Description 97
2-17. PMMCTL1 Register Description 98
2-18. SVSMHCTL Register Description 99
2-19. SVSMLCTL Register Description 100
2-20. SVSMIO Register Description 101

Copyright © 2009–2013, Texas Instruments Incorporated

2-21. PMMIFG Register Description 102
2-22. PMMRIE Register Description 104
2-23. PM5CTL0 Register Description 105
3-1. Clock Request System and Power Modes 114
3-2. UCS Registers 120
3-3. UCSCTL0 Register Description 121
3-4. UCSCTL1 Register Description 122
3-5. UCSCTL2 Register Description 123
3-6. UCSCTL3 Register Description 124
3-7. UCSCTL4 Register Description 125
3-8. UCSCTL5 Register Description 126
3-9. UCSCTL6 Register Description 128
3-10. UCSCTL7 Register Description 129
3-11. UCSCTL8 Register Description 130
3-12. UCSCTL9 Register Description 131
4-1. SR Bit Description 138
4-2. Values of Constant Generators CG1, CG2 139
4-3. Source and Destination Addressing 142
4-4. MSP430 Double-Operand Instructions 160
4-5. MSP430 Single-Operand Instructions 160
4-6. Conditional Jump Instructions 161
4-7. Emulated Instructions 161
4-8. Interrupt, Return, and Reset Cycles and Length 162
4-9. MSP430 Format II Instruction Cycles and Length 162
4-10. MSP430 Format I Instructions Cycles and Length 163
4-11. Description of the Extension Word Bits for Register Mode 164
4-12. Description of Extension Word Bits for Non-Register Modes 165
4-13. Extended Double-Operand Instructions 166
4-14. Extended Single-Operand Instructions 168
4-15. Extended Emulated Instructions 170
4-16. Address Instructions, Operate on 20-Bit Register Data 171
4-17. MSP430X Format II Instruction Cycles and Length 172
4-18. MSP430X Format I Instruction Cycles and Length 173
4-19. Address Instruction Cycles and Length 174
4-20. Instruction Map of MSP430X 175
5-1. Supported Simultaneous Code Execution and Flash Operations 292
5-2. Erase Modes 292
5-3. Write Modes 296
5-4. Flash Access While Flash is Busy (BUSY = 1) 303
5-5. FCTL Registers 307
5-6. FCTL1 Register Description 308
5-7. FCTL3 Register Description 309
5-8. FCTL4 Register Description 310
5-9. SFRIE1 Register Description 311
6-1. RAMCTL Registers 314
6-2. RCCTL0 Register Description 315
7-1. DMA Transfer Modes 320
7-2. DMA Trigger Operation 326
7-3. Maximum Single-Transfer DMA Cycle Time 327

Copyright © 2009–2013, Texas Instruments Incorporated

7-4. DMA Registers 330
7-5. DMACTL0 Register Description 332
7-6. DMACTL1 Register Description 333
7-7. DMACTL2 Register Description 334
7-8. DMACTL3 Register Description 335
7-9. DMACTL4 Register Description 336
7-10. DMAxCTL Register Description 337
7-11. DMAxSA Register Description 339
7-12. DMAxDA Register Description 340
7-13. DMAxSZ Register Description 341
7-14. DMAIV Register Description 342
8-1. I/O Configuration 345
8-2. Digital I/O Registers 350
8-3. P1IV Register Description 356
8-4. P2IV Register Description 357
8-5. P1IES Register Description 358
8-6. P1IE Register Description 358
8-7. P1IFG Register Description 358
8-8. P2IES Register Description 359
8-9. P2IE Register Description 359
8-10. P2IFG Register Description 359
8-11. PxIN Register Description 360
8-12. PxOUT Register Description 360
8-13. PxDIR Register Description 360
8-14. PxREN Register Description 361
8-15. PxDS Register Description 361
8-16. PxSEL Register Description 361
9-1. Examples for Port Mapping Mnemonics and Functions 364
9-2. Port Mapping Control Registers 365
9-3. Port Mapping Registers for Port Px – Byte Access 365
9-4. Port Mapping Registers for Port Px – Word Access 365
9-5. PMAPKEYID Register Description 366
9-6. PMAPCTL Register Description 366
9-7. PxMAPy Register Description 366
10-1. CRC Registers 372
10-2. CRCDI Register Description 373
10-3. CRCDIRB Register Description 373
10-4. CRCINIRES Register Description 374
10-5. CRCRESR Register Description 374
11-1. AES_ACCEL Registers 382
11-2. AESACTL0 Register Description 383
11-3. AESACTL1 Register Description 384
11-4. AESASTAT Register Description 385
11-5. AESAKEY Register Description 386
11-6. AESADIN Register Description 387
11-7. AESADOUT Register Description 387
11-8. AESAXDIN Register Description 388
11-9. AESAXIN Register Description 388
12-1. WDT_A Registers 394

Copyright © 2009–2013, Texas Instruments Incorporated

12-2. WDTCTL Register Description 395
13-1. Timer Modes 400
13-2. Output Modes 405
13-3. Timer_A Registers 411
13-4. TAxCTL Register Description 412
13-5. TAxR Register Description 413
13-6. TAxCCTLn Register Description 414
13-7. TAxCCRn Register Description 416
13-8. TAxIV Register Description 416
13-9. TAxEX0 Register Description 417
14-1. RTC Overview 418
15-1. RTC_A Registers 428
15-2. RTCCTL0 Register Description 430
15-3. RTCCTL1 Register Description 431
15-4. RTCCTL2 Register Description 432
15-5. RTCCTL3 Register Description 432
15-6. RTCNT1 Register Description 433
15-7. RTCNT2 Register Description 433
15-8. RTCNT3 Register Description 433
15-9. RTCNT4 Register Description 433
15-10. RTCSEC Register Description 434
15-11. RTCSEC Register Description 434
15-12. RTCMIN Register Description 435
15-13. RTCMIN Register Description 435
15-14. RTCHOUR Register Description 436
15-15. RTCHOUR Register Description 436
15-16. RTCDOW Register Description 437
15-17. RTCDAY Register Description 437
15-18. RTCDAY Register Description 437
15-19. RTCMON Register Description 438
15-20. RTCMON Register Description 438
15-21. RTCYEARL Register Description 439
15-22. RTCYEARL Register Description 439
15-23. RTCYEARH Register Description 440
15-24. RTCYEARH Register Description 440
15-25. RTCAMIN Register Description 441
15-26. RTCAMIN Register Description 441
15-27. RTCAHOUR Register Description 442
15-28. RTCAHOUR Register Description 442
15-29. RTCADOW Register Description 443
15-30. RTCADAY Register Description 443
15-31. RTCADAY Register Description 443
15-32. RTCPS0CTL Register Description 444
15-33. RTCPS1CTL Register Description 445
15-34. RT0PS Register Description 446
15-35. RT1PS Register Description 446
15-36. RTCIV Register Description 446
16-1. RTC_D Registers 457
16-2. RTCCTL0 Register Description 459

Copyright © 2009–2013, Texas Instruments Incorporated

16-3. RTCCTL1 Register Description 460
16-4. RTCCTL2 Register Description 461
16-5. RTCCTL3 Register Description 461
16-6. RTCNT1 Register Description 462
16-7. RTCNT2 Register Description 462
16-8. RTCNT3 Register Description 462
16-9. RTCNT4 Register Description 462
16-10. RTCSEC Register Description 463
16-11. RTCSEC Register Description 463
16-12. RTCMIN Register Description 464
16-13. RTCMIN Register Description 464
16-14. RTCHOUR Register Description 465
16-15. RTCHOUR Register Description 465
16-16. RTCDOW Register Description 466
16-17. RTCDAY Register Description 466
16-18. RTCDAY Register Description 466
16-19. RTCMON Register Description 467
16-20. RTCMON Register Description 467
16-21. RTCYEAR Register Description 468
16-22. RTCYEAR Register Description 468
16-23. RTCAMIN Register Description 469
16-24. RTCAMIN Register Description 469
16-25. RTCAHOUR Register Description 470
16-26. RTCAHOUR Register Description 470
16-27. RTCADOW Register Description 471
16-28. RTCADAY Register Description 471
16-29. RTCADAY Register Description 471
16-30. RTCPS0CTL Register Description 472
16-31. RTCPS1CTL Register Description 473
16-32. RTCPS0 Register Description 474
16-33. RTCPS1 Register Description 474
16-34. RTCIV Register Description 475
16-35. BIN2BCD Register Description 476
16-36. BCD2BIN Register Description 476
17-1. Result Availability (MPYFRAC = 0, MPYSAT = 0) 480
17-2. OP1 Registers 481
17-3. OP2 Registers 481
17-4. SUMEXT and MPYC Contents 482
17-5. Result Availability in Fractional Mode (MPYFRAC = 1, MPYSAT = 0) 485
17-6. Result Availability in Saturation Mode (MPYSAT = 1) 486
17-7. MPY32 Registers 492
17-8. Alternative Registers 493
17-9. MPY32CTL0 Register Description 494
18-1. REF Control of Reference System (REFMSTR = 1) (Default) 499
18-2. Control of Reference System (REFMSTR = 0, ADC12_A only) 500
18-3. REF Registers 502
18-4. REFCTL0 Register Description 503
19-1. Conversion Mode Summary 511
19-2. ADC10_A Registers 520

Copyright © 2009–2013, Texas Instruments Incorporated

19-3. ADC10CTL0 Register Description 521
19-4. ADC10CTL1 Register Description 522
19-5. ADC10CTL2 Register Description 524
19-6. ADC10MEM0 Register Description 525
19-7. ADC10MEM0 Register Description 525
19-8. ADC10MCTL0 Register Description 526
19-9. ADC10HI Register Description 527
19-10. ADC10HI Register Description 527
19-11. ADC10LO Register Description 528
19-12. ADC10LO Register Description 528
19-13. ADC10IE Register Description 529
19-14. ADC10IFG Register Description 530
19-15. ADC10IV Register Description 531
20-1. ADC12_A Conversion Result Formats 540
20-2. Conversion Mode Summary 540
20-3. ADC12_A Registers 550
20-4. ADC12CTL0 Register Description 552
20-5. ADC12CTL1 Register Description 554
20-6. ADC12CTL2 Register Description 555
20-7. ADC12MEMx Register Description 556
20-8. ADC12MCTLx Register Description 557
20-9. ADC12IE Register Description 558
20-10. ADC12IFG Register Description 560
20-11. ADC12IV Register Description 562
21-1. Comp_B Registers 570
21-2. CBCTL0 Register Description 571
21-3. CBCTL1 Register Description 572
21-4. CBCTL2 Register Description 574
21-5. CBCTL3 Register Description 575
21-6. CBINT Register Description 577
21-7. CBIV Register Description 578
22-1. Receive Error Conditions 588
22-2. BITCLK Modulation Pattern 590
22-3. BITCLK16 Modulation Pattern 591
22-4. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 0 594
22-5. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 1 596
22-6. USCI_A UART Mode Registers 599
22-7. UCAxCTL0 Register Description 600
22-8. UCAxCTL1 Register Description 601
22-9. UCAxBR0 Register Description 602
22-10. UCAxBR1 Register Description 602
22-11. UCAxMCTL Register Description 602
22-12. UCAxSTAT Register Description 603
22-13. UCAxRXBUF Register Description 604
22-14. UCAxTXBUF Register Description 604
22-15. UCAxIRTCTL Register Description 605
22-16. UCAxIRRCTL Register Description 605
22-17. UCAxABCTL Register Description 606
22-18. UCAxIE Register Description 607

Copyright © 2009–2013, Texas Instruments Incorporated

22-19. UCAxIFG Register Description 607
22-20. UCAxIV Register Description 608
23-1. UCxSTE Operation 613
23-2. USCI_A SPI Mode Registers 618
23-3. UCAxCTL0 Register Description 619
23-4. UCAxCTL1 Register Description 620
23-5. UCAxBR0 Register Description 621
23-6. UCAxBR1 Register Description 621
23-7. UCAxMCTL Register Description 621
23-8. UCAxSTAT Register Description 622
23-9. UCAxRXBUF Register Description 623
23-10. UCAxTXBUF Register Description 623
23-11. UCAxIE Register Description 624
23-12. UCAxIFG Register Description 624
23-13. UCAxIV Register Description 625
23-14. USCI_B SPI Mode Registers 626
23-15. UCBxCTL0 Register Description 627
23-16. UCBxCTL1 Register Description 628
23-17. UCBxBR0 Register Description 629
23-18. UCBxBR1 Register Description 629
23-19. UCBxMCTL Register Description 629
23-20. UCBxSTAT Register Description 630
23-21. UCBxRXBUF Register Description 631
23-22. UCBxTXBUF Register Description 631
23-23. UCBxIE Register Description 632
23-24. UCBxIFG Register Description 632
23-25. UCBxIV Register Description 633
24-1. I^2C State Change Interrupt Flags 654
24-2. USCI_B Registers 656
24-3. UCBxCTL0 Register Description 657
24-4. UCBxCTL1 Register Description 658
24-5. UCBxBR0 Register Description 659
24-6. UCBxBR1 Register Description 659
24-7. UCBxSTAT Register Description 660
24-8. UCBxRXBUF Register Description 661
24-9. UCBxTXBUF Register Description 661
24-10. UCBxI2COA Register Description 662
24-11. UCBxI2CSA Register Description 662
24-12. UCBxIE Register Description 663
24-13. UCBxIFG Register Description 664
24-14. UCBxIV Register Description 665
25-1. One-Byte Auto-Read Registers 672
25-2. Two-Byte (One-Word) Auto-Read Registers 672
25-3. Radio Interface Error Conditions 673
25-4. Radio Interface Interrupt Flags 674
25-5. CC1101 Radio Core Interrupt Mapping 675
25-6. CC1101-Based Radio Core Instruction Set – Command Strobes 682
25-7. CC1101-Based Radio Core Instruction Set 683
25-8. Radio Core Status Byte Summary 684

Copyright © 2009–2013, Texas Instruments Incorporated

25-9. Data Rate Step Size 685
25-10. Channel Filter Bandwidths (kHz) (Assuming a 26-MHz Crystal) 685
25-11. Received Packet Status Byte 1 (First Byte Appended After Data) 687
25-12. Received Packet Status Byte 2 (Second Byte Appended After Data) 687
25-13. Symbol Encoding for 2-FSK and 2-GFSK Modulation 692
25-14. Sync Word Qualifier Mode 693
25-15. Typical RSSI Value in dBm at CS Threshold With Default MAGN_TARGET at 2.4 kBaud, 868 MHz 695
25-16. Typical RSSI Value in dBm at CS Threshold With Default MAGN_TARGET at 250 kBaud, 868 MHz 696
25-17. State Transition Timing 700
25-18. FIFO_THR Settings and the Corresponding FIFO Thresholds 702
25-19. Configuration Registers 710
25-20. Status Registers 711
25-21. GDOx Signal Selection (x = 0, 1, or 2) 712
25-22. RF1A Registers 732
25-23. RF1AIFCTL0 Register Description 734
25-24. RF1AIFCTL1 Register Description 735
25-25. RF1AIFERR Register Description 736
25-26. RF1AIFERRV Register Description 737
25-27. RF1AIFIV Register Description 737
25-28. RF1AIN Register Description 738
25-29. RF1AIFG Register Description 738
25-30. RF1AIES Register Description 739
25-31. RF1AIE Register Description 739
25-32. RF1AIV Register Description 740
26-1. LCD Voltage and Biasing Characteristics 748
26-2. LCD_B Registers 762
26-3. LCD_B Memory Registers 763
26-4. LCD_B Blinking Memory Registers 764
26-5. LCDBCTL0 Register Description 765
26-6. LCDBCTL1 Register Description 766
26-7. LCDBBLKCTL Register Description 767
26-8. LCDBMEMCTL Register Description 768
26-9. LCDBVCTL Register Description 769
26-10. LCDBPCTL0 Register Description 771
26-11. LCDBPCTL1 Register Description 771
26-12. LCDBPCTL2 Register Description 772
26-13. LCDBPCTL3 Register Description 772
26-14. LCDBCPCTL Register Description 773
26-15. LCDBIV Register Description 774
27-1. EEM Configurations 779

Copyright © 2009–2013, Texas Instruments Incorporated

Read This First

About This Manual

This manual describes the modules and peripherals of the CC430 family of devices. Each description presents the module or peripheral in a general sense. Not all features and functions of all modules or peripherals may be present on all devices. In addition, modules or peripherals may differ in their exact implementation between device families, or may not be fully implemented on an individual device or device family.

Pin functions, internal signal connections, and operational parameters differ from device to device. The user should consult the device-specific data sheet for these details.

Related Documentation From Texas Instruments

For related documentation see the web site http://www.ti.com/msp430.

FCC Warning

This equipment is intended for use in a laboratory test environment only. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to subpart J of part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

Notational Conventions

Program examples, are shown in a special typeface.

Glossary

ACLK	Auxiliary Clock; see Section 3.1
ADC	Analog-to-Digital Converter
BOR	Brown-Out Reset; see Section 1.2
BSL	Bootstrap Loader; see www.ti.com/msp430 for application reports
CPU	Central Processing Unit; see Section 4.1
DAC	Digital-to-Analog Converter
DCO	Digitally Controlled Oscillator; see Section 3.2.6
dst	Destination; see Section 4.5
FLL	Frequency Locked Loop; see Section 3.2.7
GIE Modes	General Interrupt Enable; see Section 1.3.3
INT(N/2)	Integer portion of N/2
I/O	Input/Output; see Chapter 8
ISR	Interrupt Service Routine
LSB	Least-Significant Bit
LSD	Least-Significant Digit

Copyright © 2009–2013, Texas Instruments Incorporated

LPM	Low-Power Mode; see Section 1.4; also named PM for Power Mode
MAB	Memory Address Bus
MCLK	Master Clock; see Section 3.1
MDB	Memory Data Bus
MSB	Most-Significant Bit
MSD	Most-Significant Digit
NMI	(Non)-Maskable Interrupt; see Section 1.3.1; also split to UNMI and SNMI
PC	Program Counter; see Section 4.3.1
PM	Power Mode; see Section 1.4
POR	Power-On Reset; see Section 1.2
PUC	Power-Up Clear; see Section 1.2
RAM	Random Access Memory
SCG	System Clock Generator; see Section 4.3.3
SFR	Special Function Register; Section 1.14
SMCLK	Sub-System Master Clock; see Section 3.1
SNMI	System NMI; see Section 1.3.1
SP	Stack Pointer; see Section 4.3.2
SR	Status Register; see Section 4.3.3
src	Source; see Section 4.5
TOS	Top of stack; see Section 4.3.2
UNMI	User NMI; see Section 1.3.1
WDT	Watchdog Timer; see Chapter 12
z16	16-bit address space

Register Bit Conventions

Each register is shown with a key indicating the accessibility of the each individual bit, and the initial condition:

Register Bit Accessibility and Initial Condition

Key	Bit Accessibility
rw	Read/write
r	Read only
r0	Read as 0
r1	Read as 1
w	Write only
w0	Write as 0
w1	Write as 1
(w)	No register bit implemented; writing a 1 results in a pulse. The register bit is always read as 0.
h0	Cleared by hardware
h1	Set by hardware
-0,-1	Condition after PUC
-(0),-(1)	Condition after POR
-[0],-[1]	Condition after BOR
-{0},-{1}	Condition after Brownout

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

System Resets, Interrupts, and Operating Modes, System Control Module (SYS)

The system control module (SYS) is available on all devices. The following list shows the basic feature set of SYS.

- Brownout reset (BOR) and power on reset (POR) handling
- Power up clear (PUC) handling
- (Non)maskable interrupt (SNMI and UNMI) event source selection and management
- Address decoding
- A user data-exchange mechanism using the JTAG mailbox (JMB)
- Bootstrap loader (BSL) entry mechanism
- Configuration management (device descriptors)
- Provides interrupt vector generators for reset and NMIs

Topic **Page**

1.1 System Control Module (SYS) Introduction 36
1.2 System Reset and Initialization 36
1.3 Interrupts 38
1.4 Operating Modes 44
1.5 Principles for Low-Power Applications 50
1.6 Connection of Unused Pins 50
1.7 Reset Pin ($\overline{\text{RST}}$/NMI) Configuration 51
1.8 Configuring JTAG pins 51
1.9 Boot Code 51
1.10 Bootstrap Loader (BSL) 51
1.11 Memory Map – Uses and Abilities 53
1.12 JTAG Mailbox (JMB) System 54
1.13 Device Descriptor Table 55
1.14 SFR Registers 65
1.15 SYS Registers 70

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.1 System Control Module (SYS) Introduction

SYS is responsible for the interaction between various modules throughout the system. The functions that SYS provides for are not inherent to the modules themselves. Address decoding, bus arbitration, interrupt event consolidation, and reset generation are some examples of the many functions that SYS provides.

1.2 System Reset and Initialization

The system reset circuitry is shown in Figure 1-1 and sources a brownout reset (BOR), a power on reset (POR), and a power up clear (PUC). Different events trigger these reset signals and different initial conditions exist depending on which signal was generated.

A BOR is a device reset. A BOR is only generated by the following events:

- Powering up the device
- A low signal on $\overline{\text{RST}}$/NMI pin when configured in the reset mode
- A wakeup event from LPMx.5 (LPM3.5 or LPM4.5) modes
- A software BOR event

A POR is always generated when a BOR is generated, but a BOR is not generated by a POR. The following events trigger a POR:

- A BOR signal
- A SVS_H and/or SVS_M low condition when enabled (see the PMM chapter for details)
- A SVS_L and/or SVS_L low condition when enabled (see the PMM chapter for details)
- A software POR event

A PUC is always generated when a POR is generated, but a POR is not generated by a PUC. The following events trigger a PUC:

- A POR signal
- Watchdog timer expiration when watchdog mode only (see the WDT_A chapter for details)
- Watchdog timer password violation (see the WDT_A chapter for details)
- A Flash memory password violation (see the Flash Controller chapter for details)
- Power Management Module password violation (see the PMM chapter for details)
- Fetch from peripheral area

NOTE: The number and type of resets available may vary from device to device. See the device-specific data sheet for all reset sources available.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

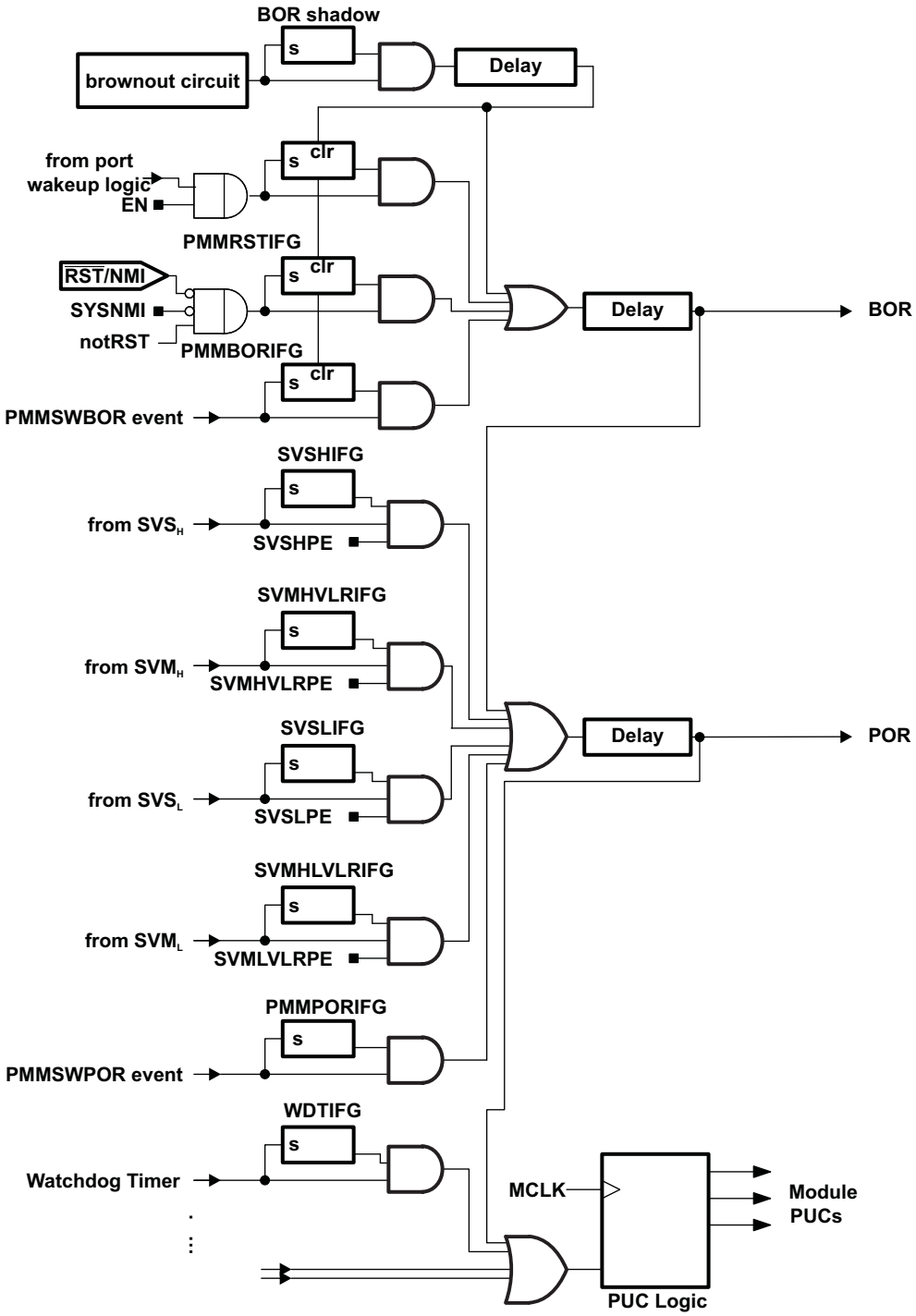


Figure 1-1. BOR/POR/PUC Reset Circuit

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.2.1 Device Initial Conditions After System Reset

After a BOR, the initial device conditions are:

- The $\overline{\text{RST}}$/NMI pin is configured in the reset mode. See Section 1.7 on configuring the $\overline{\text{RST}}$/NMI pin.
- I/O pins are switched to input mode as described in the Digital I/O chapter.
- Other peripheral modules and registers are initialized as described in their respective chapters in this manual.
- Status register (SR) is reset.
- The watchdog timer powers up active in watchdog mode.
- Program counter (PC) is loaded with the boot code address and boot code execution begins at that address. See Section 1.9 for more information regarding the boot code. Upon completion of the boot code, the PC is loaded with the address contained at the SYSRSTIV reset location (0FFFEh).

After a system reset, user software must initialize the device for the application requirements. The following must occur:

- Initialize the stack pointer (SP), typically to the top of RAM.
- Initialize the watchdog to the requirements of the application.
- Configure peripheral modules to the requirements of the application.

NOTE: A device that is unprogrammed or blank is defined as having its reset vector value, residing at memory address FFFEh, equal to FFFFh. Upon system reset of a blank device, the device enters operating mode LPM4 automatically. See Section 1.4 for information on operating modes and Section 1.3.6 for details on interrupt vectors.

NOTE: Some SRAM locations can be modified by the boot code (refer to Section 1.9) after a BOR event. These SRAM locations, when available, are at SRAM locations 01CFAh through 01CFFh and 023FAh through 023FFh.

1.3 Interrupts

The interrupt priorities are fixed and defined by the arrangement of the modules in the connection chain as shown in Figure 1-2. Interrupt priorities determine what interrupt is taken when more than one interrupt is pending simultaneously.

There are three types of interrupts:

- System reset
- (Non)maskable
- Maskable

NOTE: The types of interrupt sources available and their respective priorities can change from device to device. See the device-specific data sheet for all interrupt sources and their priorities.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

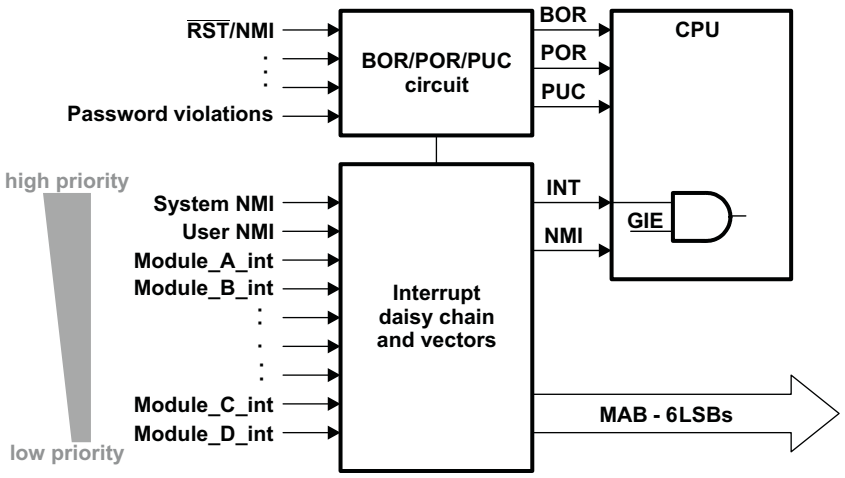


Figure 1-2. Interrupt Priority

1.3.1 (Non)Maskable Interrupts (NMIs)

In general, NMIs are not masked by the general interrupt enable (GIE) bit. The family supports two levels of NMIs — system NMI (SNMI) and user NMI (UNMI). The NMI sources are enabled by individual interrupt enable bits. When an NMI interrupt is accepted, other NMIs of that level are automatically disabled to prevent nesting of consecutive NMIs of the same level. Program execution begins at the address stored in the NMI vector as shown in Table 1-1. To allow software backward compatibility to users of earlier MSP430 families, the software may, but does not need to, reenable NMI sources. The block diagram for NMI sources is shown in Figure 1-3.

A UNMI interrupt can be generated by following sources:

- An edge on the $\overline{\text{RST}}$/NMI pin when configured in NMI mode
- An oscillator fault occurs
- An access violation to the flash memory

A SNMI interrupt can be generated by following sources:

- Power Management Module (PMM) SVM_L/SVM_H supply voltage fault
- PMM high/low side delay expiration
- Vacant memory access
- JTAG mailbox (JMB) event

NOTE: The number and types of NMI sources may vary from device to device. See the device-specific data sheet for all NMI sources available.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.3.2 SNMI Timing

Consecutive SNMIs that occur at a higher rate than they can be handled (interrupt storm) allow the main program to execute one instruction after the SNMI handler is finished with a RETI instruction, before the SNMI handler is executed again. Consecutive SNMIs are not interrupted by UNMIs in this case. This avoids a blocking behavior on high SNMI rates.

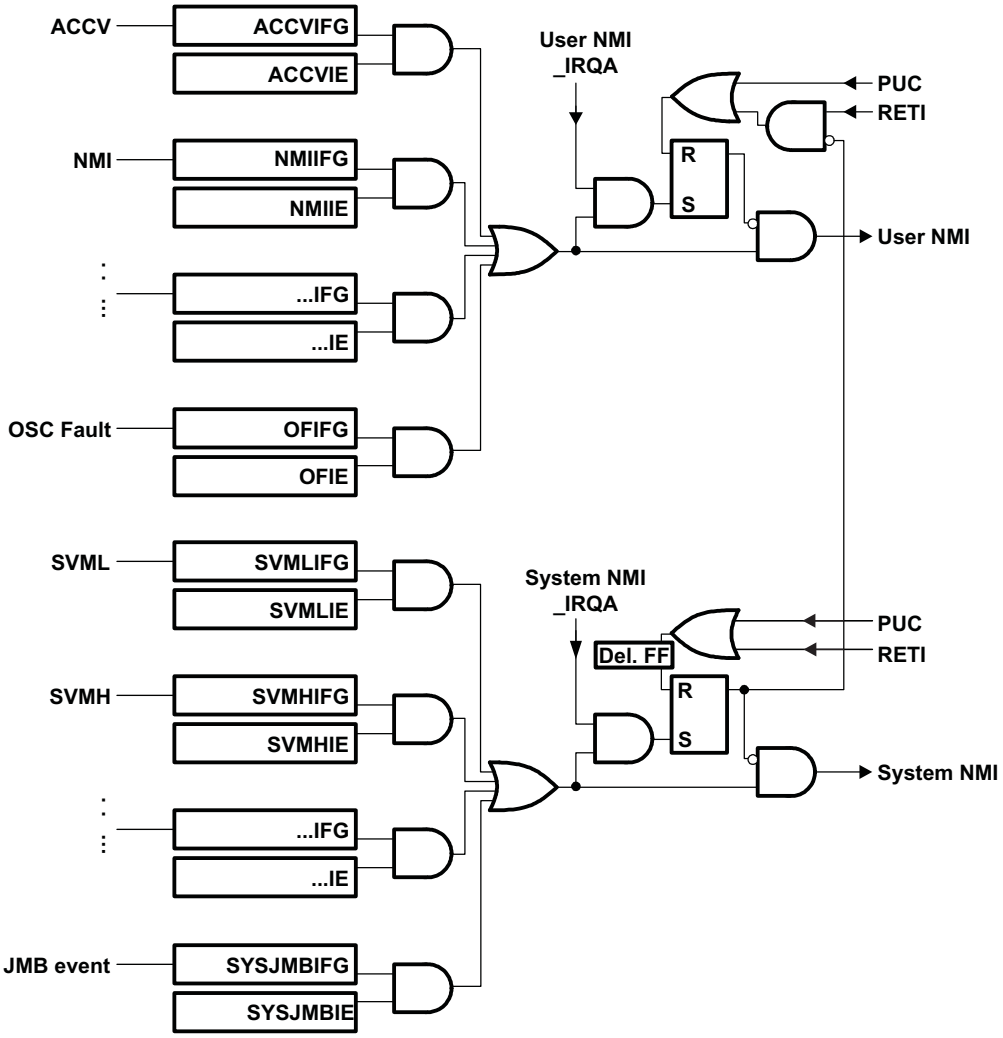


Figure 1-3. NMIs With Reentrance Protection

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.3.3 Maskable Interrupts

Maskable interrupts are caused by peripherals with interrupt capability. Each maskable interrupt source can be disabled individually by an interrupt enable bit, or all maskable interrupts can be disabled by the general interrupt enable (GIE) bit in the status register (SR).

Each individual peripheral interrupt is discussed in its respective module chapter in this manual.

1.3.4 Interrupt Processing

When an interrupt is requested from a peripheral and the peripheral interrupt enable bit and GIE bit are set, the interrupt service routine is requested. Only the individual enable bit must be set for (non)-maskable interrupts (NMI) to be requested.

1.3.4.1 Interrupt Acceptance

The interrupt latency is six cycles, starting with the acceptance of an interrupt request, and lasting until the start of execution of the first instruction of the interrupt service routine, as shown in Figure 1-4. The interrupt logic executes the following:

1. Any currently executing instruction is completed.
2. The PC, which points to the next instruction, is pushed onto the stack.
3. The SR is pushed onto the stack.
4. The interrupt with the highest priority is selected if multiple interrupts occurred during the last instruction and are pending for service.
5. The interrupt request flag resets automatically on single-source flags. Multiple source flags remain set for servicing by software.
6. All bits of SR are cleared except SCG0, thereby terminating any low-power mode. Because the GIE bit is cleared, further interrupts are disabled.
7. The content of the interrupt vector is loaded into the PC; the program continues with the interrupt service routine at that address.

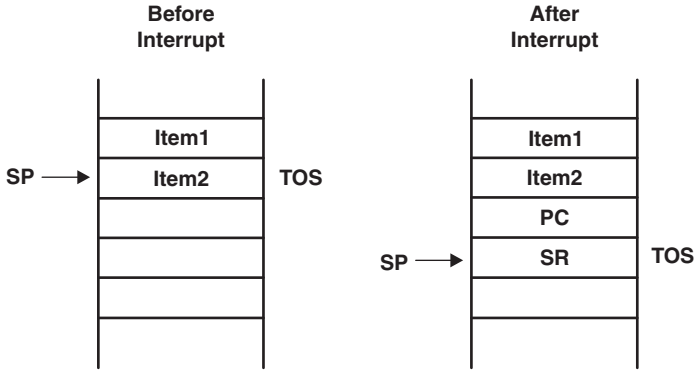


Figure 1-4. Interrupt Processing

NOTE: Enable and Disable Interrupt

Due to the pipelined CPU architecture, the instruction following the enable interrupt instruction (EINT) is always executed, even if an interrupt service request is pending when the interrupts are enabled.

If the enable interrupt instruction (EINT) is immediately followed by a disable interrupt instruction (DINT), a pending interrupt might not be serviced. Further instructions after DINT might execute incorrectly and result in unexpected CPU execution. It is recommended to always insert at least one instruction between EINT and DINT. Note that any alternative instruction use that sets and immediately clears the CPU status register GIE bit must be considered in the same fashion.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.3.4.2 Return From Interrupt

The interrupt handling routine terminates with the instruction:

```
RETI //return from an interrupt service routine
```

The return from the interrupt takes five cycles to execute the following actions and is illustrated in Figure 1-5.

1. The SR with all previous settings pops from the stack. All previous settings of GIE, CPUOFF, etc. are now in effect, regardless of the settings used during the interrupt service routine.
2. The PC pops from the stack and begins execution at the point where it was interrupted.

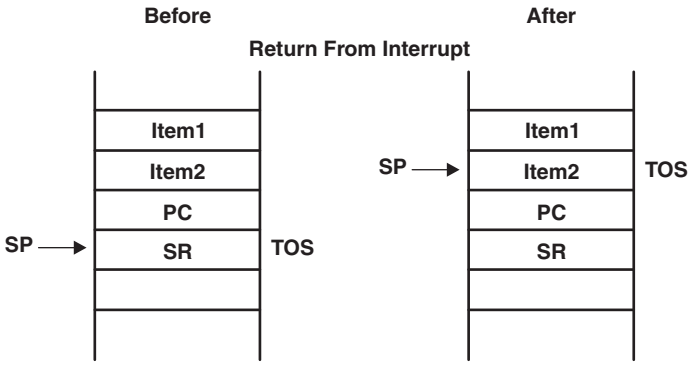


Figure 1-5. Return From Interrupt

1.3.5 Interrupt Nesting

Interrupt nesting is enabled if the GIE bit is set inside an interrupt service routine. When interrupt nesting is enabled, any interrupt occurring during an interrupt service routine interrupts the routine, regardless of the interrupt priorities.

1.3.6 Interrupt Vectors

The interrupt vectors are located in the address range 0FFFFh to 0FF80h, for a maximum of 64 interrupt sources. A vector is programmed by the user and points to the start location of the corresponding interrupt service routine. Table 1-1 is an example of the interrupt vectors available. See the device-specific data sheet for the complete interrupt vector list.

Table 1-1. Interrupt Sources, Flags, and Vectors

Interrupt Source	Interrupt Flag	System Interrupt	Word Address	Priority
Reset: power up, external reset watchdog, flash password	... WDTIFG KEYV	... Reset	... 0FFFEh	... Highest
System NMI: PMM		(Non)maskable	0FFFCh	…
User NMI: NMI, oscillator fault, flash memory access violation	... NMIIFG OFIFG ACCVIFG	... (Non)maskable (Non)maskable (Non)maskable	... 0FFFAh	... …
Device specific			0FFF8h	…
...			...	...
Watchdog timer	WDTIFG	Maskable	...	...
...			...	...
Device specific			…	…
Reserved		Maskable	…	Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

Some interrupt enable bits, and interrupt flags, as well as, control bits for the $\overline{\text{RST}}$/NMI pin are located in the special function registers (SFR). The SFR are located in the peripheral address range and are byte and word accessible. See the device-specific data sheet for the SFR configuration.

1.3.6.1 Alternate Interrupt Vectors

It is possible to use the RAM as an alternate location for the interrupt vector locations. Setting the SYSRIVECT bit in SYSCTL causes the interrupt vectors to be remapped to the top of RAM. Once set, any interrupt vectors to the alternate locations now residing in RAM. Because SYSRIVECT is automatically cleared on a BOR, it is critical that the reset vector at location 0FFFEh still be available and handled properly in firmware.

1.3.7 SYS Interrupt Vector Generators

SYS collects all system NMI (SNMI) sources, user NMI (UNMI) sources, and BOR/POR/PUC (reset) sources of all the other modules. They are combined into three interrupt vectors. The interrupt vector registers SYSRSTIV, SYSSNIV, SYSUNIV are used to determine which flags requested an interrupt or a reset. The interrupt with the highest priority of a group, when enabled, generates a number in the corresponding SYSRSTIV, SYSSNIV, SYSUNIV register. This number can be directly added to the program counter, causing a branch to the appropriate portion of the interrupt service routine. Disabled interrupts do not affect the SYSRSTIV, SYSSNIV, SYSUNIV values. Reading SYSRSTIV, SYSSNIV, SYSUNIV register automatically resets the highest pending interrupt flag of that register. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. Writing to the SYSRSTIV, SYSSNIV, SYSUNIV register automatically resets all pending interrupt flags of the group.

1.3.7.1 SYSSNIV Software Example

The following software example shows the recommended use of SYSSNIV. The SYSSNIV value is added to the PC to automatically jump to the appropriate routine. For SYSRSTIV and SYSUNIV, a similar software approach can be used. The following is an example for a generic device. Vectors can change in priority for a given device. The device specific data sheet should be referenced for the vector locations. All vectors should be coded symbolically to allow for easy portability of code.

```
SNI_ISR:    ADD      &SYSSNIV,PC ; Add offset to jump table
         RETI              ; Vector 0: No interrupt
         JMP      SVML_ISR       ; Vector 2: SVMLIFG
         JMP      SVMH_ISR       ; Vector 4: SVMHIFG
         JMP      DLYL_ISR       ; Vector 6: SVSMLDLYIFG
         JMP      DLYH_ISR       ; Vector 8: SVSMHDLYIFG
         JMP      VMA_ISR        ; Vector 10: VMAIFG
         JMP      JMBI_ISR       ; Vector 12: JMBINIFG
JMBO_ISR:                        ; Vector 14: JMBOUTIFG
        ...                      ; Task_E starts here
        RETI                     ; Return
SVML_ISR:                        ; Vector 2
        ...                      ; Task_2 starts here
        RETI                     ; Return
SVMH_ISR:                        ; Vector 4
        ...                  ; Task_4 starts here
        RETI                     ; Return
DLYL_ISR:                        ; Vector 6
        ...                      ; Task_6 starts here
        RETI                     ; Return
DLYH_ISR:                        ; Vector 8
        ...                      ; Task_8 starts here
        RETI                     ; Return
VMA_ISR:                            ; Vector A
       ...                       ; Task_A starts here
         RETI                    ; Return
JMBI_ISR:                        ; Vector C
       ...                       ; Task_C starts here
       RETI                      ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.3.7.2 SYSBERRIV Bus Error Interrupt Vector Generator

Some devices, for example those that contain the USB module, include an additional system interrupt vector generator, SYSBERRIV. In general, any type of system related bus error or timeout error is associated with a user NMI event. Upon this event, the SYSUNIV contains an offset value corresponding to a bus error event (BUSIFG). This offset can be added to the PC to automatically jump to the appropriate NMI routine. Similarly, SYSBERRIV also contains an offset value corresponding to which specific event caused the bus error event. The offset value in SYSBERRIV can be added inside the NMI routine to automatically jump to the appropriate routine. In this way, the SYSBERRIV can be thought of as an extension to the user NMI vectors.

1.4 Operating Modes

The MSP430 family is designed for ultralow-power applications and uses different operating modes shown in Figure 1-6.

The operating modes take into account three different needs:

- Ultralow power
- Speed and data throughput
- Minimization of individual peripheral current consumption

The low-power modes LPM0 through LPM4 are configured with the CPUOFF, OSCOFF, SCG0, and SCG1 bits in the SR. The advantage of including the CPUOFF, OSCOFF, SCG0, and SCG1 mode-control bits in the SR is that the present operating mode is saved onto the stack during an interrupt service routine. Program flow returns to the previous operating mode if the saved SR value is not altered during the interrupt service routine. Program flow can be returned to a different operating mode by manipulating the saved SR value on the stack inside of the interrupt service routine. When setting any of the mode-control bits, the selected operating mode takes effect immediately. Peripherals operating with any disabled clock are disabled until the clock becomes active. Peripherals may also be disabled with their individual control register settings. All I/O port pins and RAM/registers are unchanged. Wakeup from LPM0 through LPM4 is possible through all enabled interrupts.

When LPMx.5 (LPM3.5 or LPM4.5) is entered, the voltage regulator of the Power Management Module (PMM) is disabled. All RAM and register contents are lost. Although the I/O register contents are lost, the I/O pin states are locked upon LPMx.5 entry. See the Digital I/O chapter for further details. Wakeup from LPM4.5 is possible via a power sequence, a $\overline{\text{RST}}$ event, or from specific I/O. Wakeup from LPM3.5 is possible via a power sequence, a $\overline{\text{RST}}$ event, RTC event, or from specific I/O.

NOTE: LPM3.5 and LPM4.5 low power modes are not available on all devices. See the device specific data sheet to see which LPMx.5 power modes are available.

NOTE: The TEST/SBWTCK pin is used for interfacing to the development tools via Spy-Bi-Wire and JTAG. When the TEST/SBWTCK pin is high, wakeup times from LPM2, LPM3, and LPM4 may be different compared to when TEST/SBWTCK is low. Pay careful attention to the real-time behavior when exiting from LPM2, LPM3, and LPM4 with the device connected to a development tool (for example, MSP-FET430UIF). See the PMM chapter for details.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

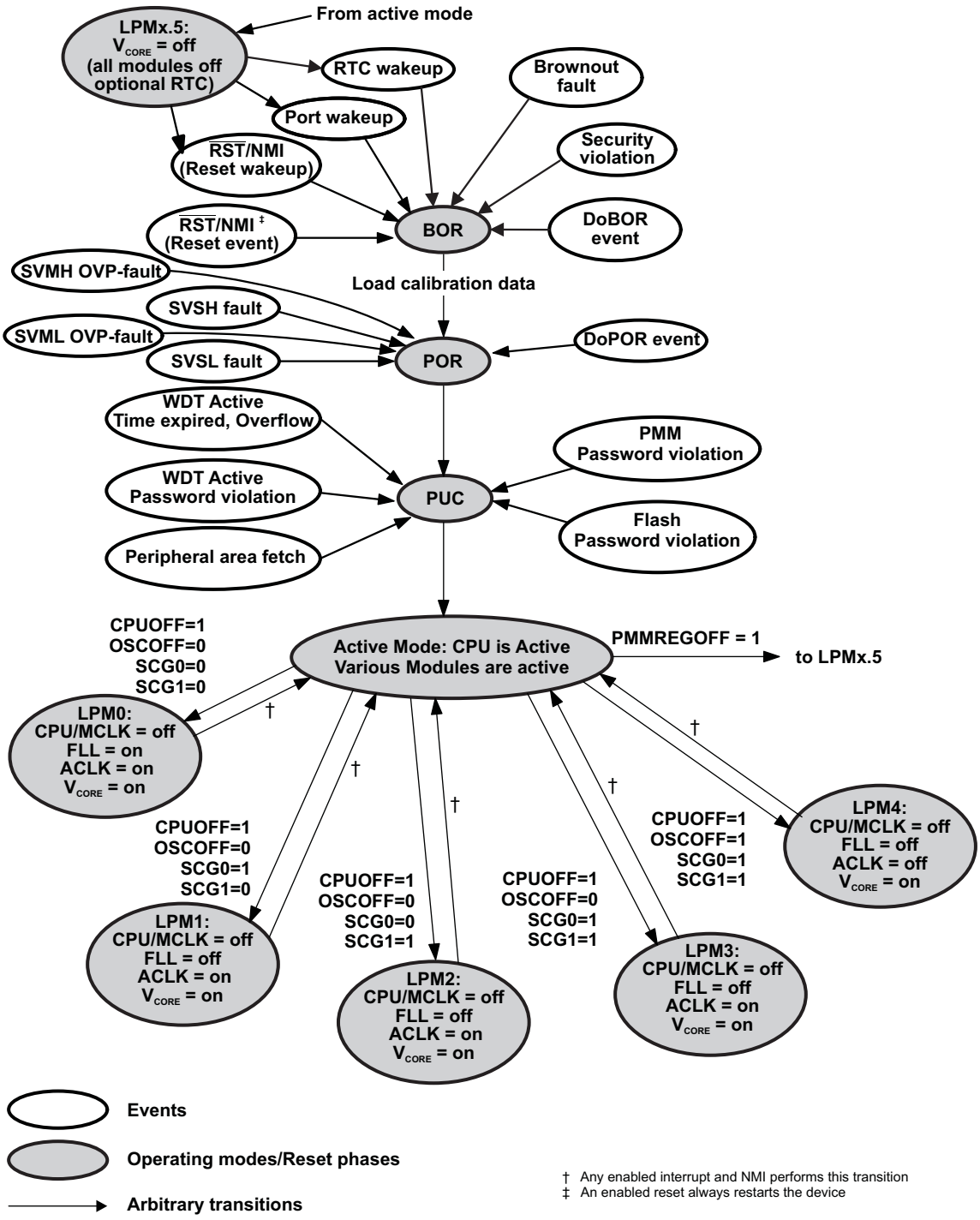


Figure 1-6. Operation Modes

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 1-2. Operation Modes

SCG1 (1)	SCG0	OSCOFF (1)	CPUOFF (1)	Mode	CPU and Clocks Status (2)
0	0	0	0	Active	CPU, MCLK are active. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK, MCLK, or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0). FLL is enabled if DCO is enabled.
0	0	0	1	LPM0	CPU, MCLK are disabled. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0). FLL is enabled if DCO is enabled.
0	1	0	1	LPM1	CPU, MCLK are disabled. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0). FLL is disabled.
1	0	0	1	LPM2	CPU, MCLK are disabled. ACLK is active. SMCLK is disabled. DCO is enabled if sources ACLK. FLL is disabled.
1	1	0	1	LPM3	CPU, MCLK are disabled. ACLK is active. SMCLK is disabled. DCO is enabled if sources ACLK. FLL is disabled.
1	1	1	1	LPM4	CPU and all clocks are disabled.
1	1	1	1	LPM3.5 (3)	When PMMREGOFF = 1, regulator is disabled. No memory retention. In this mode, RTC operation is possible when configured properly. See the *RTC* module for further details.
1	1	1	1	LPM4.5 (3)	When PMMREGOFF = 1, regulator is disabled. No memory retention. In this mode, all clock sources are disabled; that is, no RTC operation is possible.

(1) This bit is automatically reset when exiting low power modes. Please refer to Section 1.4.1 for details.

(2) The low-power modes and, hence, the system clocks can be affected by the clock request system. See the UCS chapter for details.

(3) LPM3.5 and LPM4.5 modes are not available on all devices. See the device-specific data sheet for availability.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.4.1 Entering and Exiting Low-Power Modes LPM0 Through LPM4

An enabled interrupt event wakes the device from low-power operating modes LPM0 through LPM4. The program flow for exiting LPM0 through LPM4 is:

- Enter interrupt service routine
  - The PC and SR are stored on the stack.
  - The CPUOFF, SCG1, and OSCOFF bits are automatically reset.
- Options for returning from the interrupt service routine
  - The original SR is popped from the stack, restoring the previous operating mode.
  - The SR bits stored on the stack can be modified within the interrupt service routine returning to a different operating mode when the RETI instruction is executed.

```
; Enter LPM0 Example
   BIS   #GIE+CPUOFF,SR                    ; Enter LPM0
;  ...                                     ; Program stops here
;
; Exit LPM0 Interrupt Service Routine
   BIC   #CPUOFF,0(SP)                     ; Exit LPM0 on RETI
   RETI


; Enter LPM3 Example
   BIS   #GIE+CPUOFF+SCG1+SCG0,SR          ; Enter LPM3
;  ...                                     ; Program stops here
;
; Exit LPM3 Interrupt Service Routine
   BIC   #CPUOFF+SCG1+SCG0,0(SP)           ; Exit LPM3 on RETI
   RETI


; Enter LPM4 Example
   BIS   #GIE+CPUOFF+OSCOFF+SCG1+SCG0,SR   ; Enter LPM4
;  ...                                     ; Program stops here
;
; Exit LPM4 Interrupt Service Routine
   BIC   #CPUOFF+OSCOFF+SCG1+SCG0,0(SP)    ; Exit LPM4 on RETI
   RETI
```

1.4.2 Entering and Exiting Low-Power Modes LPMx.5

LPMx.5 entry and exit is handled differently than the other low power modes. LPMx.5, when used properly, gives the lowest power consumption available on a device. To achieve this, entry to LPMx.5 disables the LDO of the PMM module, removing the supply voltage from the core of the device. Since the supply voltage is removed from the core, all register contents, as well as, SRAM contents are lost. Exit from LPMx.5 causes a BOR event, which forces a complete reset of the system. Therefore, it is the application's responsibility to properly reconfigure the device upon exit from LPMx.5.

The wakeup time from LPMx.5 is significantly longer than the wakeup time from the other power modes (please see the device specific data sheet). This is primarily due to the facts that after exit from LPMx.5, time is required for the core voltage supply to be regenerated, as well as, boot code execution to complete before the application code can begin. Therefore, the usage of LPMx.5 is restricted to very low duty cycle events.

There are two LPMx.5 power modes, LPM3.5 and LPM4.5. Not all of these are available on all devices. See the device specific data sheet to see which LPMx.5 power modes are available. LPM4.5 allows for the lowest power consumption available. No clock sources are active during LPM4.5. LPM3.5 is similar to LPM4.5, but has the additional capability of having a RTC mode available. In addition to the wakeup events possible in LPM4.5, RTC wakeup events are also possible in LPM3.5.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The program flow for entering LPMx.5 is:

1. Configure I/O appropriately. See the Digital I/O chapter for complete details on configuring I/O for LPMx.5.
   - Set all ports to general purpose I/O. Configure each port to ensure no floating inputs based on the application requirements.
   - If wakeup from I/O is desired, configure input ports with interrupt capability appropriately.
2. If LPM3.5 is available, and desired, enable RTC operation. In addition, configure any RTC interrupts, if desired for LPM3.5 wakeup event. See the RTC Overview chapter for complete details.
3. Ensure clock system settings allow LPMx.5 entry according to Table 3-1 in UCS chapter.
4. Enter LPMx.5 by setting PMMREGOFF=1 and LPM4 status register bits. The following code example shows how to enter LPMx.5 mode. See the PMM chapter for further details.

```
; Enter LPMx.5 Example
   MOV.B #PMMPW_H, &PMMCTL0_H               ; Open PMM registers for write
   BIS.B #PMMREGOFF, &PMMCTL0_L             ;
   BIS   #GIE+CPUOFF+OSCOFF+SCG1+SCG0,SR    ; Enter LPMx.5 when PMMREGOFF is set.
```

NOTE: It is not possible to wakeup from LPMx.5 if its respective interrupt flag is already asserted. It is recommended that the respective flag be cleared prior to entering LPMx.5. It is also recommended that GIE = 1 be set prior to entry into LPMx.5. Any pending flags in this case could then be serviced prior to LPMx.5 entry.

Although it is recommended to set GIE = 1 prior to entering LPMx.5, it is not required. Device wakeup from LPMx.5 with an enabled wakeup function will still cause the device to wake up from LPMx.5 even with GIE = 0. If GIE = 0 prior to LPMx.5, additional care may be required. Should the respective interrupt event should occur during LPMx.5 entry, the device may not recognize this or any future interrupt wakeup event on this function.

Exit from LPMx.5 is possible with a $\overline{\text{RST}}$ event, a power on cycle, or via specific I/O. Any exit from LPMx.5 causes a BOR. Program execution continues at the location stored in the system reset vector location 0FFFEh after execution of the boot code. The PMMLPM5IFG bit inside the PMM module is set indicating that the device was in LPMx.5 prior to the wakeup event. Additionally, SYSRSTIV = 08h which can be used to generate an efficient reset handler routine. During LPMx.5, all I/O pin conditions are automatically locked to the current state. Upon exit from LPMx.5, the I/O pin conditions remain locked until the application unlocks them. See the Digital I/O chapter for complete details. If LPM3.5 was in effect, RTC operation continues uninterrupted upon wake-up. The program flow for exiting LPMx.5 is:

- Enter system reset service routine
  - Reconfigure system as required for the application.
  - Reconfigure I/O as required for the application.

1.4.3 Extended Time in Low-Power Modes

The temperature coefficient of the DCO should be considered when the DCO is disabled for extended low-power mode periods. If the temperature changes significantly, the DCO frequency at wakeup may be significantly different from when the low-power mode was entered and may be out of the specified operating range. To avoid this, the DCO can be set to it lowest value before entering the low-power mode for extended periods of time where temperature can change.

```
; Enter LPM4 Example with lowest DCO Setting
   BIC   #SCG0, SR                          ; Disable FLL
   MOV   #0100h, &UCSCTL0                   ; Set DCO tap to first tap, clear
modulation.
   BIC   #DCORSEL2+DCORSEL1+DCORSEL0,&UCSCTL1 ; Lowest DCORSEL
   BIS   #GIE+CPUOFF+OSCOFF+SCG1+SCG0,SR    ; Enter LPM4
;  ...                                      ; Program stops
;

; Interrupt Service Routine
   BIC   #CPUOFF+OSCOFF+SCG1+SCG0,0(SR)     ; Exit LPM4 on RETI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.


```
        RETI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.5 Principles for Low-Power Applications

Often, the most important factor for reducing power consumption is using the device clock system to maximize the time in LPM3 or LPM4 modes whenever possible.

- Use interrupts to wake the processor and control program flow.
- Peripherals should be switched on only when needed.
- Use low-power integrated peripheral modules in place of software driven functions. For example, Timer_A and Timer_B can automatically generate PWM and capture external timing with no CPU resources.
- Calculated branching and fast table look-ups should be used in place of flag polling and long software calculations.
- Avoid frequent subroutine and function calls due to overhead.
- For longer software routines, single-cycle CPU registers should be used.
- Overwrite RAM control register RCCTL0 with all not available and unused segments set to powered down (= 1). For information about used RAM segments see the device-specific data sheet.

If the application has low duty cycle, slow response time events, maximizing time in LPMx.5 can further reduce power consumption significantly.

1.6 Connection of Unused Pins

The correct termination of all unused pins is listed in Table 1-3.

Table 1-3. Connection of Unused Pins[(1)]

Pin	Potential	Comment
AVCC	DV_{CC}	
AVSS	DV_{SS}	
Px.0 to Px.7	Open	Switched to port function, output direction (PxDIR.n = 1)
XIN	DV_{SS}	For dedicated XIN pins only. XIN pins with shared GPIO functions should be programmed to GPIO and follow Px.0 to Px.7 recommendations.
XOUT	Open	For dedicated XOUT pins only. XOUT pins with shared GPIO functions should be programmed to GPIO and follow Px.0 to Px.7 recommendations.
XT2IN	DV_{SS}	For dedicated XT2IN pins only. XT2IN pins with shared GPIO functions should be programmed to GPIO and follow Px.0 to Px.7 recommendations.
XT2OUT	Open	For dedicated XT2OUT pins only. XT2OUT pins with shared GPIO functions should be programmed to GPIO and follow Px.0 to Px.7 recommendations.
LCDCAP	DV_{SS}	
$\overline{RST}$/NMI	DV_{CC} or V_{CC}	47-kΩ pullup or internal pullup selected with 10-nF (2.2 nF) pulldown[(2)]
PJ.0/TDO PJ.1/TDI PJ.2/TMS PJ.3/TCK	Open	The JTAG pins are shared with general purpose I/O function (PJ.x). If not being used, these should be switched to port function, output direction (PJDIR.n = 1). When used as JTAG pins, these pins should remain open.
TEST	Open	This pin always has an internal pulldown enabled.
GUARD	DV_{CC}	If radio module is not used in the application
R_BIAS	DV_{SS}	If radio module is not used in the application
AVCC_RF	DV_{CC}	If radio module is not used in the application
RF_N	Open	If radio module is not used in the application
RF_P	Open	If radio module is not used in the application
RF_XIN	DV_{SS}	If radio module is not used in the application
RF_XOUT	Open	If radio module is not used in the application
V18	Open	For USB devices only when USB module is not being used in the application

(1) Any unused pin with a secondary function that is shared with general purpose I/O should follow the Px.0 to Px.7 unused pin connection guidelines.

(2) The pulldown capacitor should not exceed 2.2 nF when using devices with Spy-Bi-Wire interface in Spy-Bi-Wire mode or in 4-wire JTAG mode with TI tools such as FET interfaces or GANG programmers.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 1-3. Connection of Unused Pins[1] (continued)

Pin	Potential	Comment
PUR[3]	DV_{SS}	For USB devices only when USB module is not being used in the application
VUSB	Open	For USB devices only when USB module is not being used in the application
PU.0/DP PU.1/DM	Open	For USB devices only when USB module is not being used in the application
VBUS, VSSU	DV_{SS}	For USB devices only when USB module is not being used in the application
LDOI	DV_{SS}	For devices with LDO-PWR module when not being used in the application.
LDOO	Open	For devices with LDO-PWR module when not being used in the application.

(3) The default USB BSL evaluates the state of the PUR pin after a BOR reset. If it is pulled high externally, then the BSL is invoked. Therefore, unless invoking the BSL, it is important to keep PUR pulled low after a BOR reset, even if BSL or USB is never used. A 1-MΩ resistor to ground is recommended.

1.7 Reset Pin (RST/NMI) Configuration

The reset pin can be configured as a reset function (default) or as an NMI function via the Special Function Register (SFR), SFRRPCR. Setting SYSNMI causes the RST/NMI pin to be configured as an external NMI source. The external NMI is edge sensitive, and its edge is selectable by SYSNMIIES. Setting the NMIIE enables the interrupt of the external NMI. Upon an external NMI event, the NMIIFG is set.

The RST/NMI pin can have either a pullup or pulldown present or not. SYSRSTUP selects either pullup or pulldown and SYSRSTRE causes the pullup or pulldown to be enabled or not. If the RST/NMI pin is unused, it is required to have either the internal pullup selected and enabled or an external resistor connected to the RST/NMI pin as shown in Table 1-3.

NOTE: All devices except the MSP430F543x (non-A devices) have the internal pullup enabled. In this case, no external pullup resistor is required.

1.8 Configuring JTAG pins

The JTAG pins are shared with general purpose I/O pins. There are several ways that the JTAG pins can be selected for four wire JTAG mode via software. Normally, upon a BOR, SYSJTAGPIN is cleared. With SYSJTAGPIN cleared, the JTAG are configured as general-purpose I/O. See the Digital I/O chapter for details on controlling the JTAG pins as general purpose I/O. If SYSJTAG = 1, the JTAG pins are configured to four wire JTAG mode and remain in this mode until another BOR condition occurs. Therefore, SYSJTAGPIN is a write only once function. Clearing it by software is not possible, and the device does not change from four wire JTAG mode to general purpose I/O.

1.9 Boot Code

The boot code is always executed after a BOR. The boot code loads factory stored calibration values of the oscillator and reference voltages. In addition, it checks for the presence of a user-defined boot strap loader (BSL).

1.10 Bootstrap Loader (BSL)

The BSL is software that is executed after start-up when a certain BSL entry condition is applied. The BSL enables the user to communicate with the embedded memory in the microcontroller during the prototyping phase, final production, and in service. All memory mapped resources, the programmable memory (flash memory), the data memory (RAM), and the peripherals, can be modified by the BSL as required. The user can define custom BSL code for flash-based devices and protect it against erasure and unintentional or unauthorized access.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

On devices without USB, a basic BSL program is provided by TI. This supports the commonly used UART protocol with RS232 interfacing, allowing flexible use of both hardware and software. To use the BSL, a specific BSL entry sequence must be applied to specific device pins. The correct entry sequence causes SYSBSLIND to be set. An added sequence of commands initiates the desired function. A boot-loading session can be exited by continuing operation at a defined user program address or by applying the standard reset sequence. Access to the device memory via the BSL is protected against misuse by a user-defined password. Devices with USB have a USB based BSL program provided by TI. For more details, see the *MSP430 Programming Via the Bootstrap Loader User's Guide* (SLAU319) at www.ti.com/msp430.

The amount of BSL memory that is available is device specific. The BSL memory size is organized into segments and can be set using the SYSBSLSIZE bits. See the device specific data sheet for the number and size of the segments available. It is possible to assign a small amount of RAM to the allocated BSL memory. Setting SYSBSLR allocates the lowest 16 bytes of RAM for the BSL. When the BSL memory is protected, access to these RAM locations is only possible from within the protected BSL memory segments.

It may be desirable in some BSL applications to only allow changing of the Power Management Module settings from the protected BSL segments. This is possible with the SYSPMMPE bit. Normally, this bit is cleared and allows access of the PMM control registers from any memory location. Setting SYSPMMPE, allows access to the PMM control registers only from the protected BSL memory. Once set, SYSPMMPE can only be cleared by a BOR event.

Copyright © 2009–2013, Texas Instruments Incorporated

1.11 Memory Map – Uses and Abilities

This memory map represents the MSP430F5438 device. Though the address ranges differs from device to device, overall behavior remains the same.

Address Range	Name and Usage	Properties						
		Can generate NMI on read/write/fetch						
		Generates PUC on fetch access						
		Protectable for read/write accesses						
		Always able to access PMM registers from[1]; Mass erase by user possible						
		Mass erase by user possible						
		Bank erase by user possible						
		Segment erase by user possible						
00000h-00FFFh	Peripherals with gaps							
00000h-000FFh	Reserved for system extension							
00100h-00FEFh	Peripherals						x	
00FF0h-00FF3h	Descriptor type[2]						x	
00FF4h-00FF7h	Start address of descriptor structure						x	
01000h-011FFh	BSL 0	x				x		
01200h-013FFh	BSL 1	x				x		
01400h-015FFh	BSL 2	x				x		
01600h-017FFh	BSL 3	x			x	x		
017FCh-017FFh	BSL Signature Location							
01800h-0187Fh	Info D	x						
01880h-018FFh	Info C	x						
01900h-0197Fh	Info B	x						
01980h-019FFh	Info A	x						
01A00h-01A7Fh	Device Descriptor Table						x	
01C00h-05BFFh	RAM 16 KB							
05B80-05BFFh	Alternate Interrupt Vectors							
05C00h-0FFFFh	Program	x	x[1]	x				
0FF80h-0FFFFh	Interrupt Vectors							
10000h-45BFFh	Program	x	x	x				
45C00h-FFFFFh	Vacant							x[3]

(1) Access rights are separately programmable for SYS and PMM.
(2) Fixed ID for all MSP430 devices. See Section 1.13.1 for further details.
(3) On vacant memory space, the value 03FFFh is driven on the data bus.

1.11.1 Vacant Memory Space

Vacant memory is non-existent memory space. Accesses to vacant memory space generate a system (non)maskable interrupt (SNMI) when enabled (VMAIE = 1). Reads from vacant memory results in the value 3FFFh. In the case of a fetch, this is taken as JMP $. Fetch accesses from vacant peripheral space result in a PUC. After the boot code is executed, it behaves like vacant memory space and also causes an NMI on access.

1.11.2 JTAG Lock Mechanism via the Electronic Fuse

A device can be protected from unauthorized access by disabling the JTAG and SBW interface. This is achieved by programming the electronic fuse. Programming the electronic fuse, completely disables the debug and access capabilities associated with the JTAG and Spy-Bi-Wire interface. The JTAG is locked by programming a certain signature into the device's' flash memory at dedicated addresses. The JTAG security lock key resides at the end of the bootstrap loader (BSL) memory at addresses 17FCh through 17FFh. Anything other than 0h or FFFFFFFFh programmed to these addresses locks the JTAG interface.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

All of the 5xx MSP430 devices come with a preprogrammed BSL (TI-BSL) code that, by default, protects itself from unintended erase and write access. This is done by setting SYSBSLPE in the SYSBSLC register. Since the JTAG security lock key resides in the BSL memory address range, appropriate action must be taken to unprotect the BSL memory area before programming the protection key. For more details on the electronic fuse see the *MSP430 Programming Via the JTAG Interface User's Guide* (SLAU320) at www.ti.com/msp430.

Some JTAG commands are still possible after the device is secured, including the BYPASS command (see IEEE1149-2001 Standard) and the JMB_EXCHANGE command which allows access to the JTAG Mailbox System (see Section 1.12 for details).

NOTE: If a device has been protected, Texas Instruments cannot access the device for a customer return. Access is only possible if a BSL is provided with its corresponding key or an unlock mechanism is provided by the customer.

1.12 JTAG Mailbox (JMB) System

The SYS module provides the capability to exchange user data via the regular JTAG test/debug interface. The idea behind the JMB is to have a direct interface to the CPU during debugging, programming, and test that is identical for all '430 devices of this family and uses only few or no user application resources. The JTAG interface was chosen because it is available on all '430 devices and is a dedicated resource for debugging, programming and test.

Applications of the JMB are:

- Providing entry password for device lock/unlock protection
- Run-time data exchange (RTDX)

1.12.1 JMB Configuration

The JMB supports two transfer modes, 16-bit and 32-bit. Setting JMBMODE enables 32-bit transfer mode. Clearing JMBMODE enables 16-bit transfer mode.

1.12.2 JMBOUT0 and JMBOUT1 Outgoing Mailbox

Two 16-bit registers are available for outgoing messages to the JTAG port. JMBOUT0 is only used when using 16-bit transfer mode (JMBMODE = 0). JMBOUT1 is used in addition to JMBOUT0 when using 32-bit transfer mode (JMBMODE = 1). When the application wishes to send a message to the JTAG port, it writes data to JMBOUT0 for 16-bit mode, or JMBOUT0 and JMBOUT1 for 32-bit mode.

JMBOUT0FG and JMBOUT1FG are read only flags that indicate the status of JMBOUT0 and JMBOUT1, respectively. When JMBOUT0FG is set, JMBOUT0 has been read by the JTAG port and is ready to receive new data. When JMBOUT0FG is reset, the JMBOUT0 is not ready to receive new data. JMBOUT1FG behaves similarly.

1.12.3 JMBIN0 and JMBIN1 Incoming Mailbox

Two 16-bit registers are available for incoming messages from the JTAG port. Only JMBIN0 is used when in 16-bit transfer mode (JMBMODE = 0). JMBIN1 is used in addition to JMBIN0 when using 32-bit transfer mode (JMBMODE = 1). When the JTAG port wishes to send a message to the application, it writes data to JMBIN0 for 16-bit mode, or JMBIN0 and JMBIN1 for 32-bit mode.

JMBIN0FG and JMBIN1FG are flags that indicate the status of JMBIN0 and JMBIN1, respectively. When JMBIN0FG is set, JMBIN0 has data that is available for reading. When JMBIN0FG is reset, no new data is available in JMBIN0. JMBIN1FG behaves similarly.

JMBIN0FG and JMBIN1FG can be configured to clear automatically by clearing JMBCLR0OFF and JMBCLR1OFF, respectively. Otherwise, these flags must be cleared by software.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.12.4 JMB NMI Usage

The JMB handshake mechanism can be configured to use interrupts to avoid unnecessary polling if desired. In 16-bit mode, JMBOUTIFG is set when JMBOUT0 has been read by the JTAG port and is ready to receive data. In 32-bit mode, JMBOUTIFG is set when both JMBOUT0 and JMBOUT1 has been read by the JTAG port and are ready to receive data. If JMBOUTIE is set, these events cause a system NMI. In 16-bit mode, JMBOUTIFG is cleared automatically when data is written to JMBOUT0. In 32-bit mode, JMBOUTIFG Is cleared automatically when data is written to both JMBOUT0 and JMBOUT1. In addition, the JMBOUTIFG can be cleared when reading SYSSNIV. Clearing JMBOUTIE disables the NMI interrupt.

In 16-bit mode, JMBINIFG is set when JMBIN0 is available for reading. In 32-bit mode, JMBINIFG is set when both JMBIN0 and JMBIN1 are available for reading. If JMBOUTIE is set, these events cause a system NMI. In 16-bit mode, JMBINIFG is cleared automatically when JMBIN0 is read. In 32-bit mode, JMBINIFG Is cleared automatically when both JMBIN0 and JMBIN1 are read. In addition, the JMBINIFG can be cleared when reading SYSSNIV. Clearing JMBINIE disables the NMI interrupt.

1.13 Device Descriptor Table

Each device provides a data structure in memory that allows an unambiguous identification of the device, as well as, a more detailed description of the available modules on a given device. SYS provides this information and can be used by device-adaptive SW tools and libraries to clearly identify a particular device and all modules and capabilities contained within it. The validity of the device descriptor can be verified by cyclic redundancy check (CRC).Figure 1-7 shows the logical order and structure of the device descriptor table. The complete device descriptor table and its contents can be found in the device specific data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

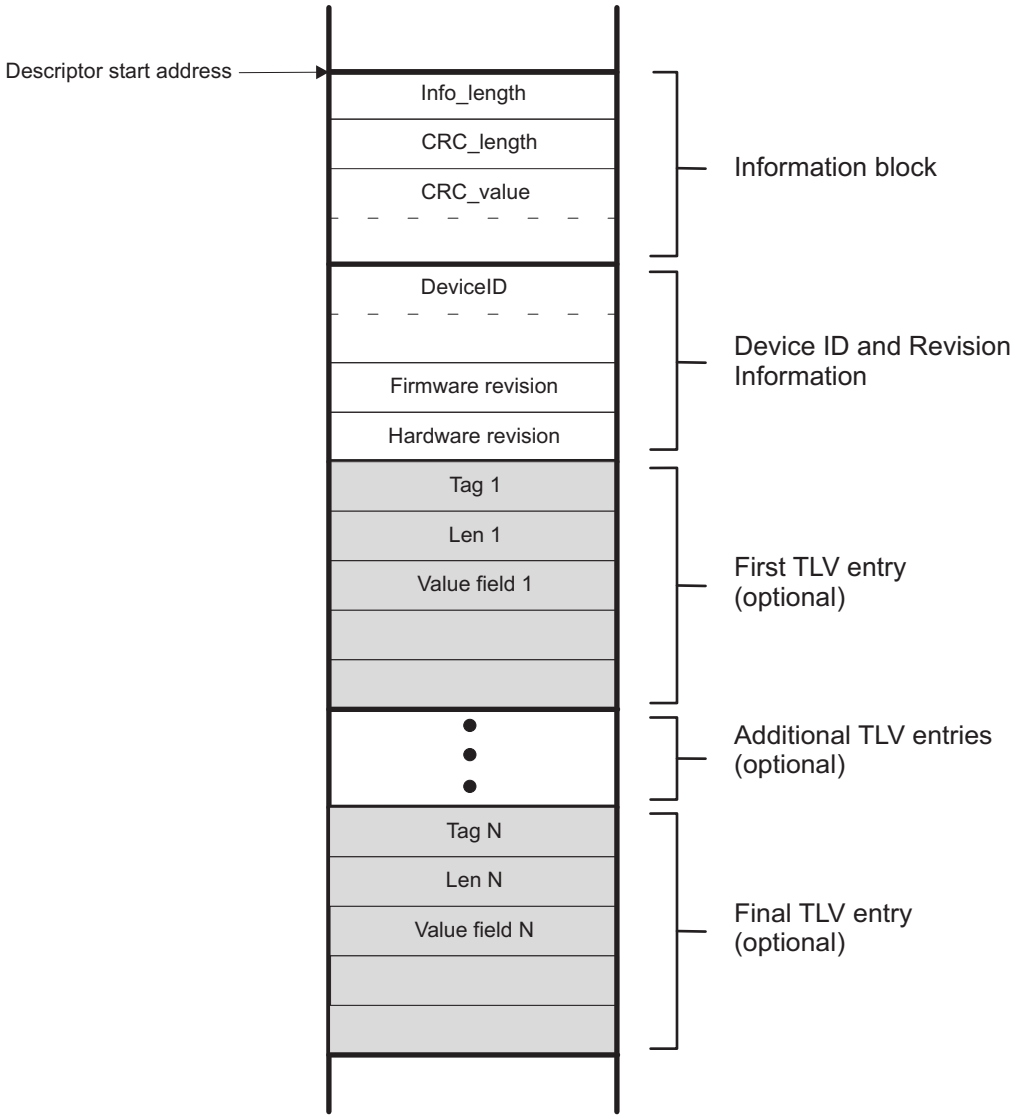


Figure 1-7. Devices Descriptor Table

1.13.1 Identifying Device Type

The value read at address location 00FF0h identifies the family branch of the device. All values starting with 80h indicate a hierarchical structure consisting of the information block and a TLV tag-length-value (TLV) structure containing the various descriptors. Any other value than 80h read at address location 00FF0h indicates the device is of an older family and contains a flat descriptor beginning at location 0FF0h. The information block, shown in Figure 1-7 contains the device ID, die revisions, firmware revisions, and other manufacturer and tool related information. The descriptors contains information about the available peripherals, their subtypes and addresses and provides the information required to build adaptive hardware drivers for operating systems.

The length of the descriptors represented by Info_length is computed as follows:

Length = 2^{Info_length} in 32-bit words (1)

For example, if Info_length = 5, then the length of the descriptors equals 128 bytes.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.13.2 TLV Descriptors

The TLV descriptors follow the information block. Because the information block is always a fixed length, the start location of the TLV descriptors is fixed for a given device family. For the MSP430x5xx family, this location is 01A08h. See the device-specific data sheet for the complete TLV structure and what descriptors are available.

The TLV descriptors are unique to their respective TLV block and are always followed by the descriptor block length.

Each TLV descriptor contains a tag field which identifies the descriptor type. Table 1-4 shows the currently supported tags.

Table 1-4. Tag Values

Short Name	Value	Description
LDTAG	01h	Legacy descriptor (1xx, 2xx, 4xx families)
PDTAG	02h	Peripheral discovery descriptor
Reserved	03h	Future usage
Reserved	04h	Future usage
BLANK	05h	Blank descriptor
Reserved	06h	Future usage
ADCCAL	11h	ADC calibration
REFCAL	12h	REF calibration
Reserved	13h - FDh	Future usage
TAGEXT	FEh	Tag extender

Each tag field is unique to its respective descriptor and is always followed by a length field. The length field is one byte if the tag value is 01h through 0FDh and represents the length of the descriptor in bytes. If the tag value equals 0FEh (TAGEXT), the next byte extends the tag values, and the following two bytes represent the length of the descriptor in bytes. In this way, a user can search through the TLV descriptor table for a particular tag value, using a routine similar to below written in pseudo code:

```
// Identify the descriptor ID (d_ID_value) for the TLV descriptor of interest:
descriptor_address = TLV_START address;

while ( value at descriptor_address != d_ID_value && descriptor_address != TLV_TAGEND &&
descriptor_address < TLV_END)
{
  // Point to next descriptor
  descriptor_address = descriptor_address + (length of the current TLV block) + 2;
}

if (value at descriptor_address == d_ID_value) {
  // Appropriate TLV decriptor has been found!
  Return length of descriptor & descriptor_address as the location of the TLV descriptor
} else {
  // No TLV descriptor found with a matching d_ID_value
  Return a failing condition
}
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.13.3 Peripheral Discovery Descriptor

This descriptor type can describe concatenated or distributed memory or peripheral mappings, as well as, the number of interrupt vectors and their order. The peripheral discovery descriptor has tag value 02h (PDTAG). Table 1-5 shows the structure of the peripheral discovery descriptor.

NOTE: Peripheral Discovery Descriptor is not available in every device. See the Device Descriptors section in the device-specific data sheet for the availability and details on Peripheral Discovery Descriptor.

Table 1-5. Peripheral Discovery Descriptor

Element	Size (bytes)	Comments
memory entry 1	2	Optional
memory entry 2	2	Optional
...	2	Optional
delimiter (00h)	1	Mandatory
peripheral count	1	Mandatory
peripheral entry 1	2	Optional
peripheral entry 2	2	Optional
...	2	Optional
Interrupt priority N-3	1	Optional
Interrupt priority N-4	1	Optional
...	1	Optional
delimiter (00h)	1	Mandatory

The structures for a memory entry and peripheral entry are shown below. A memory entry consists of two bytes (one word). Table 1-6 shows the individual bit fields of a memory entry word and their respective meanings. Similarly, a peripheral entry consists of two bytes (one word). Table 1-7 shows the individual bit fields of a peripheral entry word and their respective meanings.

Table 1-6. Values for Memory Entry

Bit fields				
[15:13]	[12:9]	[8]	[7]	[6:0]
Memory type	**Size**	**More**	**Unit Size**	**Address value**
000: None	0000: 0 B	0: End Entry	0: 0200h	0000000
001: RAM	0001: 128 B	1: More Entries	1: 010000h	0000001

Copyright © 2009–2013, Texas Instruments Incorporated

Table 1-6. Values for Memory Entry (continued)

Bit fields				
[15:13]	[12:9]	[8]	[7]	[6:0]
Memory type	Size	More	Unit Size	Address value
010: EEPROM	0010: 256 B			0000010
011: Reserved	0011: 512 B			0000011
100: FLASH	0100: 1 KB			0000100
101: ROM	0101: 2KB			0000101
110: MemType appended	0110: 4 KB			0000110
111: Undefined	0111: 8 KB			0000111
	1000: 16 KB			0001000
	1001: 32 KB			0001001
	1010: 64 KB			0001010
	1011: 128 KB			0001011
	1100: 256 KB			0001100
	1101: 512 KB			...
	1110: Size appended			...
	1111: Undefined			1111111

Table 1-7. Values for Peripheral Entry

Bit fields		
[15:8]	[7]	[6:0]
Peripheral ID (PID) (1)	UnitSize	AdrVal
Any PID	0: 010h	0000000
Any PID	1: 0800h	0000001
Any PID		0000010
Any PID		0000011
Any PID		0000100
Any PID		0000101
Any PID		...
Any PID		...
Any PID		1111111

(1) The Peripheral IDs are listed in Table 1-8. This is not a complete list, but shown as an example.

Table 1-8. Peripheral IDs(1)

Peripheral or Module	PID
No Module	00h
WDT	01h
SFR	02h
UCS	03h
SYS	04h
PMM	05h
Flash Controller	08h
CRC16	09h
Port 1, 2	51h

(1) This table is not a complete list of all peripheral IDs available on a device, but is shown here for illustrative purposes only.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 1-8. Peripheral IDs[(1)] (continued)

Peripheral or Module	PID
Port 3, 4	52h
Port 5, 6	53h
Port 7, 8	54h
Port 9, 10	55h
Port J	5Fh
Timer A0	81h
Timer A1	82h
Special info appended	FEh
Undefined module	FFh

Copyright © 2009–2013, Texas Instruments Incorporated

Table 1-9 shows a simple example for a peripheral discovery descriptor of a hypothetical device:

Table 1-9. Sample Peripheral Discovery Descriptor

Hex	Binary	Entry type	Description
030h, 0Eh	001_1000_ 0_0_0001110	memory	RAM 16 KB; Start address = 01C00h (0Eh * 0200h)(1)
09Bh, 02Eh	100_1011_0_0_0101110	memory	FLASH 128 KB Start address = 05C00h (2Eh * 0200h)
00h	0000_0000_0000_0000	delimiter	No more memory entries
0Fh	0000_1111	peripheral count	Peripheral count = 15
02h, 10h	00000010_0_0010000	peripheral	SFR at address = 0100h (10h * 10h)
01h, 01h	00000001_0_0000001	peripheral	WDT at address = 0110h (0100h + 10h)
05h, 01h	00000101_0_0000001	peripheral	PMM at address = 0120h (0110h + 10h)
03h, 01h	00000011_0_0000001	peripheral	UCS at address = 0130h (0120h + 10h)
08h, 01h	00001000_0_0000001	peripheral	FLCTL at address = 0140h (0130h + 10h)
09h, 01h	00001001_0_0000001	peripheral	CRC16 at address = 0150h (0140h + 10h)
04h, 01h	00000100_0_0000001	peripheral	SYS at address = 0160h (0150h + 10h)
51h, 0Ah	01010001_0_0001010	peripheral	Port 1, 2 at address = 0200h (0160h + 10h * 10h)
52h, 02h	01010010_0_0000010	peripheral	Port 3, 4 at address = 0220h (0200h + 02h * 10h)
53h, 02h	01010011_0_0000010	peripheral	Port 5, 6 at address = 0240h (0220h + 02h * 10h)
54h, 02h	01010100_0_0000010	peripheral	Port 7, 8 at address = 0260h (0240h + 02h * 10h)
55h, 02h	01010101_0_0000010	peripheral	Port 9, 10 at address = 0280h (0260h + 02h * 10h)
5Fh, 0Ah	01011111_0_0001010	peripheral	Port J at address = 0320h (0280h + 0Ah * 10h)
81h, 02h	10000001_0_0000010	peripheral	Timer A0 at address = 0340h (0320h + 02h * 10h)
82h, 04h	10000010_0_0000100	peripheral	Timer A1 at address = 0380h (0340h + 04h * 10h)
–			No appended entries
			SYSRSTIV @0FFFEh (implied)
			SYSSNIV @0FFFC (implied)
			SYSUNIV @ 0FFFA (implied)
81h	1000_0001	interrupt	TA0 CCR0 @ 0FFF8
81h	1000_0001	interrupt	TA0 CCR1, CCR1, TA0IFG@ 0FFF6
51h	0101_0001	interrupt	Port 1 @ 0FFF4
82h	1000_0010	interrupt	TA1CCR0 @ 0FFF2
51h	0101_0001	interrupt	Port 2 @ 0FFF0
81h	1000_0010	interrupt	TA1 CCR1, CCR1, TA1IFG@ 0FFEE
00h	0000_0000	delimiter	No more interrupt entries

(1) In this example, the memory type is RAM (bits[15:13] = 001), the size is 16KB (bits[12:9] = 1000), and the starting address is 01C00h. The starting address is computed by taking the size field indicated by bit[7] (in this case 0200h) and multiplying it by the address value (bits[6:0] = 0001110. In this case, we have 0200h * 00Eh = 01C00h.

NOTE: The interrupt ordering has some implied rules:

- For timers, CCR0 interrupt has higher priority over all other CCRn interrupts.
- For communication ports, RX has higher priority over TX
- For port pairs, Port 1 has higher priority over Port 2, Port 3 has higher priority over Port 4, etc.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.13.4 CRC Computation

The CRC checksum for the TLV structure is stored at memory locations 0x1A02 and 0x1A03. The least significant byte (LSB) and most significant byte (MSB) reside at memory locations 0x1A02 and 0x1A03, respectively. The checksum is computed using data stored at memory locations 0x1A04 through 0x1AFF. The CRC checksum can be easily computed using the CRC16 module. The following simplified C code utilizes the CRC16 module to compute the checksum. Please see the CRC16 chapter for further details on the CRC algorithm implementation.

NOTE: The CRC module on the CC430F613x, CC430F612x, and CC430F513x devices does not support the bit-wise reverse feature used in this code example. Registers CRCDIRB and CRCRESR, along with their respective functionality, are not available.

```
unsigned int i;
unsigned char CRCRESULT_LSB, CRCRESULT_MSB;

WDTCTL = WDTPW + WDTHOLD;
CRCINIRES = 0xFFFF;
for (i = 0x01A04; i <= 0x01AFF; i++){
  CRCDIRB_L = *(unsigned char*)(i);
}

CRCRESULT_LSB = CRCINIRES_L;  // value stored @ 0x1A02
CRCRESULT_MSB = CRCINIRES_H;  // value stored @ 0x1A03
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.13.5 Calibration Values

The TLV structure contains calibration values that can be used to improve the measurement capability of various functions. The calibration values available on a given device are shown in the TLV structure of the device-specific data sheet.

1.13.5.1 REF Calibration

The calibration data for the REF module consists of three words, one word for each reference voltage available (1.5, 2.0, and 2.5 V). The reference voltages are measured at room temperature. The measured values are normalized by 1.5/2.0/2.5 V before being stored into the TLV structure:

$$CAL_ADC_15VREF_FACTOR = \frac{V_{REF+}}{1.5V} \times 2^{15}$$

$$CAL_ADC_20VREF_FACTOR = \frac{V_{REF+}}{2.0V} \times 2^{15}$$

$$CAL_ADC_25VREF_FACTOR = \frac{V_{REF+}}{2.5V} \times 2^{15} \quad (2)$$

In this way, a conversion result is corrected by multiplying it with the CAL_15VREF_FACTOR (or CAL_20VREF_FACTOR, CAL_25VREF_FACTOR) and dividing the result by 2^{15} as shown for each of the respective reference voltages:

$$ADC(corrected) = ADC(raw) \times CAL_ADC15VREF_FACTOR \times \frac{1}{2^{15}}$$

$$ADC(corrected) = ADC(raw) \times CAL_ADC20VREF_FACTOR \times \frac{1}{2^{15}}$$

$$ADC(corrected) = ADC(raw) \times CAL_ADC25VREF_FACTOR \times \frac{1}{2^{15}} \quad (3)$$

In the following example, the integrated 1.5-V reference voltage is used during a conversion.

- Conversion result: 0x0100 = 256 decimal
- Reference voltage calibration factor (CAL_15VREF_FACTOR) : 0x7BBB

The following steps show how the ADC conversion result can be corrected:

- Multiply the conversion result by 2 (this step simplifies the final division): 0x0100 x 0x0002 = 0x0200
- Multiply the result by CAL_15VREF_FACTOR: 0x200 x 0x7FEE = 0x00F7_7600
- Divide the result by 2^{16}: 0x00F7_7600 / 0x0001_0000 = 0x0000_00F7 = 247 decimal

1.13.5.2 ADC Offset and Gain Calibration

The offset of the ADC is determined and stored as a twos-complement number in the TLV structure. The offset error correction is done by adding the CAL_ADC_OFFSET to the conversion result.

$$ADC(offset_corrected) = ADC(raw) + CAL_ADC_OFFSET \quad (4)$$

The gain of the ADC12 is calculated by Equation 5:

$$CAL_ADC_GAIN_FACTOR = \frac{1}{GAIN} \times 2^{15} \quad (5)$$

The conversion result is gain corrected by multiplying it with the CAL_ADC_GAIN_FACTOR and dividing the result by 2^{15}:

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

$$ADC(gain_corrected) = ADC(raw) \times CAL_ADC_GAIN_FACTOR \times \frac{1}{2^{15}} \quad (6)$$

If both gain and offset are corrected, the gain correction is done first:

$$ADC(gain_corrected) = ADC(raw) \times CAL_ADC_GAIN_FACTOR \times \frac{1}{2^{15}}$$

$$ADC(final) = ADC(gain_corrected) + CAL_ADC_OFFSET \quad (7)$$

1.13.5.3 Temperature Sensor Calibration

The temperature sensor is calibrated using the internal voltage references. Each reference voltage (1.5/2.0/2.5 V) contains a measured value for two temperatures, 30°C±3°C and 85°C ±3°C and are stored in the TLV structure. The characteristic equation of the temperature sensor voltage, in mV is:

$$V_{SENSE} = TC_{SENSOR} \times Temp + V_{SENSOR} \quad (8)$$

The temperature coefficient, TC_{SENSOR} in mV/°C, represents the slope of the equation. V_{SENSOR}, in mV, represents the y-intercept of the equation. Temp, in °C, is the temperature of interest.

The temperature (Temp, °C) can be computed as follows for each of the reference voltages used in the ADC measurement:

$$Temp = \left(ADC(raw) - CAL_ADC_15T30\right) \times \left(\frac{85-30}{CAL_ADC_15T85 - CAL_ADC_15T30}\right) + 30$$

$$Temp = \left(ADC(raw) - CAL_ADC_20T30\right) \times \left(\frac{85-30}{CAL_ADC_20T85 - CAL_ADC_20T30}\right) + 30$$

$$Temp = \left(ADC(raw) - CAL_ADC_25T30\right) \times \left(\frac{85-30}{CAL_ADC_25T85 - CAL_ADC_25T30}\right) + 30 \quad (9)$$

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.14 SFR Registers

The SFRs are listed in Table 1-11. The base address for the SFRs is listed in Table 1-10. Many of the bits inside the SFRs are described in other chapters throughout this user's guide. These bits are marked with a note and a reference. See the specific chapter of the respective module for details.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 1-10. SFR Base Address

Module	Base Address
SFR	00100h

Table 1-11. SFR Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	SFRIE1	Interrupt Enable	Read/write	Word	0000h	Section 5.4.4
00h	SFRIE1_L (IE1)		Read/write	Byte	00h	
01h	SFRIE1_H (IE2)		Read/write	Byte	00h	
02h	SFRIFG1	Interrupt Flag	Read/write	Word	0082h	Section 1.14.2
02h	SFRIFG1_L (IFG1)		Read/write	Byte	82h	
03h	SFRIFG1_H (IFG2)		Read/write	Byte	00h	
04h	SFRRPCR	Reset Pin Control	Read/write	Word	0000h	Section 1.14.3
04h	SFRRPCR_L		Read/write	Byte	00h	
05h	SFRRPCR_H		Read/write	Byte	00h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.14.1 SFRIE1 Register

Interrupt Enable Register

Figure 1-8. SFRIE1 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
JMBOUTIE	JMBINIE	ACCVIE(1)	NMIIE	VMAIE	Reserved	OFIE(2)	WDTIE(3)
rw-0	rw-0	rw-0	rw-0	rw-0	r0	rw-0	rw-0

(1) See the Flash Controller chapter for details.
(2) See the UCS chapter for details.
(3) See the WDT_A chapter for details.

Table 1-12. SFRIE1 Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved. Always reads as 0.
7	JMBOUTIE	RW	0h	JTAG mailbox output interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
6	JMBINIE	RW	0h	JTAG mailbox input interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
5	ACCVIE	RW	0h	Flash controller access violation interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
4	NMIIE	RW	0h	NMI pin interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
3	VMAIE	RW	0h	Vacant memory access interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
2	Reserved	R	0h	Reserved. Always reads as 0.
1	OFIE	RW	0h	Oscillator fault interrupt enable flag 0b = Interrupts disabled 1b = Interrupts enabled
0	WDTIE	RW	0h	Watchdog timer interrupt enable. This bit enables the WDTIFG interrupt for interval timer mode. It is not necessary to set this bit for watchdog mode. Because other bits in ~IE1 may be used for other modules, it is recommended to set or clear this bit using BIS.B or BIC.B instructions, rather than MOV.B or CLR.B instruction 0b = Interrupts disabled 1b = Interrupts enabled

Copyright © 2009–2013, Texas Instruments Incorporated

1.14.2 SFRIFG1 Register

Interrupt Flag Register

Figure 1-9. SFRIFG1 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
JMBOUTIFG	JMBINIFG	Reserved	NMIIFG	VMAIFG	Reserved	OFIFG(1)	WDTIFG(2)
rw-(1)	rw-(0)	r0	rw-0	rw-0	r0	rw-(1)	rw-0

(1) See the UCS chapter for details.
(2) See the WDT_A chapter for details.

Table 1-13. SFRIFG1 Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved. Always reads as 0.
7	JMBOUTIFG	RW	1h	JTAG mailbox output interrupt flag 0b = No interrupt pending. When in 16-bit mode (JMBMODE = 0), this bit is cleared automatically when JMBO0 has been written with a new message to the JTAG module by the CPU. When in 32-bit mode (JMBMODE = 1), this bit is cleared automatically when both JMBO0 and JMBO1 have been written with new messages to the JTAG module by the CPU. This bit is also cleared when the associated vector in SYSUNIV has been read. 1b = Interrupt pending, JMBO registers are ready for new messages. In 16-bit mode (JMBMODE = 0), JMBO0 has been received by the JTAG module and is ready for a new message from the CPU. In 32-bit mode (JMBMODE = 1) , JMBO0 and JMBO1 have been received by the JTAG module and are ready for new messages from the CPU.
6	JMBINIFG	RW	0h	JTAG mailbox input interrupt flag 0b = No interrupt pending. When in 16-bit mode (JMBMODE = 0), this bit is cleared automatically when JMBI0 is read by the CPU. When in 32-bit mode (JMBMODE = 1), this bit is cleared automatically when both JMBI0 and JMBI1 have been read by the CPU. This bit is also cleared when the associated vector in SYSUNIV has been read 1b = Interrupt pending, a message is waiting in the JMBIN registers. In 16-bit mode (JMBMODE = 0) when JMBI0 has been written by the JTAG module. In 32-bit mode (JMBMODE = 1) when JMBI0 and JMBI1 have been written by the JTAG module.
5	Reserved	R	0h	Reserved. Always reads as 0.
4	NMIIFG	RW	0h	NMI pin interrupt flag 0b = No interrupt pending 1b = Interrupt pending
3	VMAIFG	RW	0h	Vacant memory access interrupt flag 0b = No interrupt pending 1b = Interrupt pending
2	Reserved	R	0h	Reserved. Always reads as 0.
1	OFIFG	RW	1h	Oscillator fault interrupt flag 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 1-13. SFRIFG1 Register Description (continued)

Bit	Field	Type	Reset	Description
0	WDTIFG	RW	0h	Watchdog timer interrupt flag. In watchdog mode, WDTIFG will self clear upon a watchdog timeout event. The SYSRSTIV can be read to determine if the reset was caused by a watchdog timeout event. In interval mode, WDTIFG is reset automatically by servicing the interrupt, or can be reset by software. Because other bits in ~IFG1 may be used for other modules, it is recommended to set or clear WDTIFG by using BIS.B or BIC.B instructions, rather than MOV.B or CLR.B instructions. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

1.14.3 SFRRPCR Register

Reset Pin Control Register

Figure 1-10. SFRRPCR Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved				SYSRSTRE[1]	SYSRSTUP[1]	SYSNMIIES	SYSNMI
r0	r0	r0	r0	rw-1	rw-1	rw-0	rw-0

(1) All devices except the MSP430F5438 (non-A) default to pullup enabled on the reset pin.

Table 1-14. SFRRPCR Register Description

Bit	Field	Type	Reset	Description
15-4	Reserved	R	0h	Reserved. Always reads as 0.
3	SYSRSTRE	RW	1h	Reset pin resistor enable 0b = Pullup/pulldown resistor at the $\overline{\text{RST}}$/NMI pin is disabled 1b = Pullup/pulldown resistor at the $\overline{\text{RST}}$/NMI pin is enabled
2	SYSRSTUP	RW	1h	Reset resistor pin pullup/pulldown 0b = Pulldown is selected 1b = Pullup is selected
1	SYSNMIIES	RW	0h	NMI edge select. This bit selects the interrupt edge for the NMI when SYSNMI = 1. Modifying this bit can trigger an NMI. Modify this bit when SYSNMI = 0 to avoid triggering an accidental NMI. 0b = NMI on rising edge 1b = NMI on falling edge
0	SYSNMI	RW	0h	NMI select. This bit selects the function for the RST/NMI pin. 0b = Reset function 1b = NMI function

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15 SYS Registers

The SYS configuration registers are listed in Table 1-16 and the base address is listed in Table 1-15. A detailed description of each register and its bits is also provided. Each register starts at a word boundary. Either word or byte data can be written to the SYS configuration registers.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 1-15. SYS Base Address

Module	Base Address
SYS	00180h

Table 1-16. SYS Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	SYSCTL	System Control	Read/write	Word	0000h	Section 1.15.1
00h	SYSCTL_L		Read/write	Byte	00h	
01h	SYSCTL_H		Read/write	Byte	00h	
02h	SYSBSLC	Bootstrap Loader Configuration	Read/write	Word	0003h	Section 1.15.2
02h	SYSBSLC_L		Read/write	Byte	03h	
03h	SYSBSLC_H		Read/write	Byte	00h	
06h	SYSJMBC	JTAG Mailbox Control	Read/write	Word	0000h	Section 1.15.3
06h	SYSJMBC_L		Read/write	Byte	00h	
07h	SYSJMBC_H		Read/write	Byte	00h	
08h	SYSJMBI0	JTAG Mailbox Input 0	Read/write	Word	0000h	Section 1.15.4
08h	SYSJMBI0_L		Read/write	Byte	00h	
09h	SYSJMBI0_H		Read/write	Byte	00h	
0Ah	SYSJMBI1	JTAG Mailbox Input 1	Read/write	Word	0000h	Section 1.15.5
0Ah	SYSJMBI1_L		Read/write	Byte	00h	
0Bh	SYSJMBI1_H		Read/write	Byte	00h	
0Ch	SYSJMBO0	JTAG Mailbox Output 0	Read/write	Word	0000h	Section 1.15.6
0Ch	SYSJMBO0_L		Read/write	Byte	00h	
0Dh	SYSJMBO0_H		Read/write	Byte	00h	
0Eh	SYSJMBO1	JTAG Mailbox Output 1	Read/write	Word	0000h	Section 1.15.7
0Eh	SYSJMBO1_L		Read/write	Byte	00h	
0Fh	SYSJMBO1_H		Read/write	Byte	00h	
18h	SYSBERRIV	Bus Error Vector Generator	Read	Word	0000h	Section 1.15.11
1Ah	SYSUNIV	User NMI Vector Generator	Read	Word	0000h	Section 1.15.8
1Ch	SYSSNIV	System NMI Vector Generator	Read	Word	0000h	Section 1.15.9
1Eh	SYSRSTIV	Reset Vector Generator	Read	Word	0002h	Section 1.15.10

Copyright © 2009–2013, Texas Instruments Incorporated

1.15.1 SYSCTL Register

SYS Control Register

Figure 1-11. SYSCTL Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved		SYSJTAGPIN	SYSBSLIND	Reserved	SYSPMMPE	Reserved	SYSRIVECT
r0	r0	rw-[0]	r-0	r0	rw-[0]	r0	rw-[0]

Table 1-17. SYSCTL Register Description

Bit	Field	Type	Reset	Description
15-6	Reserved	R	0h	Reserved. Always reads as 0.
5	SYSJTAGPIN	RW	0h	Dedicated JTAG pins enable. Setting this bit disables the shared functionality of the JTAG pins and permanently enables the JTAG function. This bit can only be set once. Once it is set it remains set until a BOR occurs. 0b = Shared JTAG pins (JTAG mode selectable via SBW sequence) 1b = Dedicated JTAG pins (explicit 4-wire JTAG mode selection)
4	SYSBSLIND	RW	0h	BSL entry indication. This bit indicates a BSL entry sequence detected on the Spy-Bi-Wire pins. 0b = No BSL entry sequence detected 1b = BSL entry sequence detected
3	Reserved	R	0h	Reserved. Always reads as 0.
2	SYSPMMPE	RW	0h	PMM access protect. This controls the accessibility of the PMM control registers. Once set to 1, it only can be cleared by a BOR. 0b = Access from anywhere in memory 1b = Access only from the protected BSL segments
1	Reserved	R	0h	Reserved. Always reads as 0.
0	SYSRIVECT	RW	0h	RAM-based interrupt vectors 0b = Interrupt vectors generated with end address TOP of lower 64k flash FFFFh 1b = Interrupt vectors generated with end address TOP of RAM

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.2 SYSBSLC Register

Bootstrap Loader Configuration Register

Figure 1-12. SYSBSLC Register

15	14	13	12	11	10	9	8
SYSBSLPE	SYSBSLOFF	Reserved					
rw-[0]	rw-[0]	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved					SYSBSLR	SYSBSLSIZE	
r0	r0	r0	r0	r0	rw-[0]	rw-[1]	rw-[1]

Table 1-18. SYSBSLC Register Description

Bit	Field	Type	Reset	Description
15	SYSBSLPE	RW	0h	Bootstrap loader memory protection enable for the size covered in SYSBSLSIZE. By default, this bit is cleared by hardware with a BOR event (as indicated above), however the boot code that checks for an available BSL may set this bit via software in order to protect the BSL. Since devices normally come with a TI BSL preprogrammed and protected, the boot code sets this bit. 0b = Area not protected. Read, program, and erase of BSL memory is possible. 1b = Area protected
14	SYSBSLOFF	RW	0h	Bootstrap loader memory disable for the size covered in SYSBSLSIZE 0b = BSL memory is addressed when this area is read. 1b = BSL memory behaves like vacant memory. Reads cause 3FFFh to be read. Fetches cause JMP $ to be executed.
13-3	Reserved	R	0h	Reserved. Always reads as 0.
2	SYSBSLR	RW	0h	RAM assigned to BSL 0b = No RAM assigned to BSL area 1b = Lowest 16 bytes of RAM assigned to BSL
1-0	SYSBSLSIZE	RW	03h	Bootstrap loader size. Defines the space and size of flash memory that is reserved for the BSL. 00b = Size: BSL segment 3 01b = Size: BSL segments 2 and 3 10b = Size: BSL segments 1, 2, and 3 11b = Size: BSL segments 1, 2, 3, and 4

Copyright © 2009–2013, Texas Instruments Incorporated

1.15.3 SYSJMBC Register

JTAG Mailbox Control Register

Figure 1-13. SYSJMBC Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
JMBCLR1OFF	JMBCLR0OFF	Reserved	JMBM0DE	JMBOUT1FG	JMBOUT0FG	JMBIN1FG	JMBIN0FG
rw-(0)	rw-(0)	r0	rw-0	r-(1)	r-(1)	rw-(0)	rw-(0)

Table 1-19. SYSJMBC Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved. Always reads as 0.
7	JMBCLR1OFF	RW	0h	Incoming JTAG Mailbox 1 flag auto-clear disable 0b = JMBIN1FG cleared on read of JMB1IN register 1b = JMBIN1FG cleared by software
6	JMBCLR0OFF	RW	0h	Incoming JTAG Mailbox 0 flag auto-clear disable 0b = JMBIN0FG cleared on read of JMB0IN register 1b = JMBIN0FG cleared by software
5	Reserved	R	0h	Reserved. Always reads as 0.
4	JMBMODE	RW	0h	This bit defines the operation mode of JMB for JMBI0/1 and JMBO0/1. Before switching this bit, pad and flush out any partial content to avoid data drops. 0b = 16-bit transfers using JMBO0 and JMBI0 only 1b = 32-bit transfers using JMBO0/1 and JMBI0/1
3	JMBOUT1FG	RW	1h	Outgoing JTAG Mailbox 1 flag. This bit is cleared automatically when a message is written to the upper byte of JMBO1 or as word access (by the CPU, DMA,…) and is set after the message was read via JTAG. 0b = JMBO1 is not ready to receive new data. 1b = JMBO1 is ready to receive new data.
2	JMBOUT0FG	RW	1h	Outgoing JTAG Mailbox 0 flag. This bit is cleared automatically when a message is written to the upper byte of JMBO0 or as word access (by the CPU, DMA,…) and is set after the message was read via JTAG. 0b = JMBO0 is not ready to receive new data. 1b = JMBO0 is ready to receive new data.
1	JMBIN1FG	RW	0h	Incoming JTAG Mailbox 1 flag. This bit is set when a new message (provided via JTAG) is available in JMBI1. This flag is cleared automatically on read of JMBI1 when JMBCLR1OFF = 0 (auto clear mode). On JMBCLR1OFF = 1, JMBIN1FG needs to be cleared by SW. 0b = JMBI1 has no new data. 1b = JMBI1 has new data available.
0	JMBIN0FG	RW	0h	Incoming JTAG Mailbox 0 flag. This bit is set when a new message (provided via JTAG) is available in JMBI0. This flag is cleared automatically on read of JMBI0 when JMBCLR0OFF = 0 (auto clear mode). On JMBCLR0OFF = 1, JMBIN0FG needs to be cleared by SW. 0b = JMBI1 has no new data. 1b = JMBI1 has new data available.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.4 SYSJMBI0 Register

JTAG Mailbox Input 0 Register

Figure 1-14. SYSJMBI0 Register

15	14	13	12	11	10	9	8
MSGHI							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
7	6	5	4	3	2	1	0
MSGLO							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0

Table 1-20. SYSJMBI0 Register Description

Bit	Field	Type	Reset	Description
15-8	MSGHI	R	0h	JTAG mailbox incoming message high byte
7-0	MSGLO	R	0h	JTAG mailbox incoming message low byte

1.15.5 SYSJMBI1 Register

JTAG Mailbox Input 0 Register

Figure 1-15. SYSJMBI1 Register

15	14	13	12	11	10	9	8
MSGHI							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
7	6	5	4	3	2	1	0
MSGLO							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0

Table 1-21. SYSJMBI1 Register Description

Bit	Field	Type	Reset	Description
15-8	MSGHI	R	0h	JTAG mailbox incoming message high byte
7-0	MSGLO	R	0h	JTAG mailbox incoming message low byte

Copyright © 2009–2013, Texas Instruments Incorporated

1.15.6 SYSJMBO0 Register

JTAG Mailbox Output 0 Register

Figure 1-16. SYSJMBO0 Register

15	14	13	12	11	10	9	8
MSGHI							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
MSGL0							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 1-22. SYSJMBO0 Register Description

Bit	Field	Type	Reset	Description
15-8	MSGHI	RW	0h	JTAG mailbox outgoing message high byte
7-0	MSGLO	RW	0h	JTAG mailbox outgoing message low byte

1.15.7 SYSJMBO1 Register

JTAG Mailbox Output 1 Register

Figure 1-17. SYSJMBO1 Register

15	14	13	12	11	10	9	8
MSGHI							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
MSGL0							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 1-23. SYSJMBO1 Register Description

Bit	Field	Type	Reset	Description
15-8	MSGHI	RW	0h	JTAG mailbox outgoing message high byte
7-0	MSGLO	RW	0h	JTAG mailbox outgoing message low byte

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.8 SYSUNIV Register

User NMI Vector Register

NOTE: Additional events for more complex devices are appended to this table; sources that are removed reduce the length of this table. The vectors are expected to be accessed symbolic only with the corresponding include file of the device in use.

Figure 1-18. SYSUNIV Register

15	14	13	12	11	10	9	8
SYSUNVEC							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
SYSUNVEC							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 1-24. SYSUNIV Register Description

Bit	Field	Type	Reset	Description
15-0	SYSUNIV	R	0h	User NMI vector. Generates a value that can be used as address offset for fast interrupt service routine handling. Writing to this register clears all pending user NMI flags. 00h = No interrupt pending 02h = NMIIFG interrupt pending (highest priority) 04h = OFIFG interrupt pending 06h = ACCVIFG interrupt pending 08h = BUSIFG interrupt pending (Not present on all devices. See device-specific datasheet)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.9 SYSSNIV Register

System NMI Vector Register

NOTE: Additional events for more complex devices are appended to this table; sources that are removed reduce the length of this table. The vectors are expected to be accessed symbolic only with the corresponding include file of the used device.

Figure 1-19. SYSSNIV Register

15	14	13	12	11	10	9	8
SYSSNVEC							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
SYSSNVEC							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 1-25. SYSSNIV Register Description

Bit	Field	Type	Reset	Description
15-0	SYSSNIV	R	0h	System NMI vector. Generates a value that can be used as address offset for fast interrupt service routine handling. Writing to this register clears all pending system NMI flags. 00h = No interrupt pending 02h = SVMLIFG interrupt pending (highest priority) 04h = SVMHIFG interrupt pending 06h = SVSMLDLYIFG interrupt pending 08h = SVSMHDLYIFG interrupt pending 0Ah = VMAIFG interrupt pending 0Ch = JMBINIFG interrupt pending 0Eh = JMBOUTIFG interrupt pending 10h = SVMLVLRIFG interrupt pending 12h = SVMHVLRIFG interrupt pending 14h = Reserved

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.10 SYSRSTIV Register

Reset Interrupt Vector Register

NOTE: Additional events for more complex devices are appended to this table; sources that are removed reduce the length of this table. The vectors are expected to be accessed symbolic only with the corresponding include file of the used device.

Figure 1-20. SYSRSTIV Register

15	14	13	12	11	10	9	8
SYSRSTVEC							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
SYSRSTVEC							
r0	r0	r-0	r-0	r-0	r-0	r-1	r0

Table 1-26. SYSRSTIV Register Description

Bit	Field	Type	Reset	Description
15-0	SYSRSTIV	R	0h	Reset interrupt vector. Generates a value that can be used as address offset for fast interrupt service routine handling to identify the last cause of a reset (BOR, POR, PUC) . Writing to this register clears all pending reset source flags. 00h = No interrupt pending 02h = Brownout (BOR) (highest priority) 04h = $\overline{\text{RST}}$/NMI (BOR) 06h = PMMSWBOR (BOR) 08h = Wakeup from LPMx.5 (BOR) 0Ah = Security violation (BOR) 0Ch = SVSL (POR) 0Eh = SVSH (POR) 10h = SVML_OVP (POR) 12h = SVMH_OVP (POR) 14h = PMMSWPOR (POR) 16h = WDT time out (PUC) 18h = WDT password violation (PUC) 1Ah = Flash password violation (PUC) 1Ch = Reserved 1Eh = PERF peripheral/configuration area fetch (PUC) 20h = PMM password violation (PUC) 22h to 3Eh = Reserved

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1.15.11 SYSBERRIV Register

System Bus Error Interrupt Vector Register

NOTE: Additional events for more complex devices are appended to this table; sources that are removed reduce the length of this table. The vectors are expected to be accessed symbolic only with the corresponding include file of the used device.

Figure 1-21. SYSBERRIV Register

15	14	13	12	11	10	9	8
SYSBERRIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
SYSBERRIV							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 1-27. SYSBERRIV Register Description

Bit	Field	Type	Reset	Description
15-0	SYSBERRIV	R	0h	System bus error interrupt vector. Generates a value that can be used as an address offset for fast interrupt service routine handling. Writing to this register clears all pending flags. 00h = No interrupt pending 02h = USB module timed out. Wait state time out of 8 clock cycles. 16 clock cycles only on the F552x and F551x devices. 04h = Reserved for future extensions 06h = Reserved for future extensions 08h = Reserved for future extensions

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Power Management Module and Supply Voltage Supervisor

This chapter describes the operation of the Power Management Module (PMM) and Supply Voltage Supervisor (SVS).

Topic **Page**

2.1 Power Management Module (PMM) Introduction 81
2.2 PMM Operation 83
2.3 PMM Registers 96

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.1 Power Management Module (PMM) Introduction

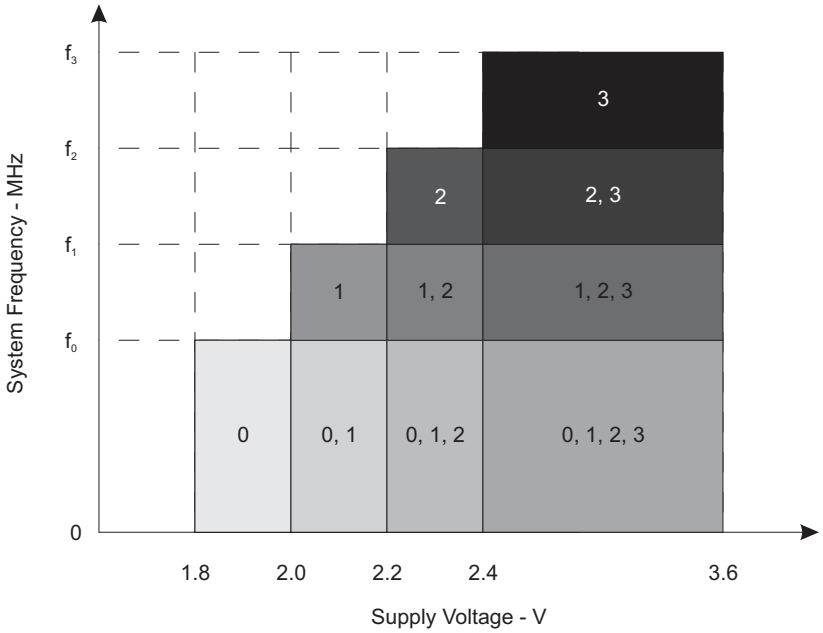
PMM features include:

- Wide supply voltage (DV_{CC}) range: 1.8 V to 3.6 V
- Generation of voltage for the device core (V_{CORE}) with up to four programmable levels
- Supply voltage supervisor (SVS) for DV_{CC} and V_{CORE} with programmable threshold levels
- Supply voltage monitor (SVM) for DV_{CC} and V_{CORE} with programmable threshold levels
- Brownout reset (BOR)
- Software accessible power-fail indicators
- I/O protection during power-fail condition
- Software selectable supervisor or monitor state output (optional)

The PMM manages all functions related to the power supply and its supervision for the device. Its primary functions are first to generate a supply voltage for the core logic, and second, provide several mechanisms for the supervision and monitoring of both the voltage applied to the device (DV_{CC}) and the voltage generated for the core (V_{CORE}).

The PMM uses an integrated low-dropout voltage regulator (LDO) to produce a secondary core voltage (V_{CORE}) from the primary one applied to the device (DV_{CC}). In general, V_{CORE} supplies the CPU, memories (flash and RAM), and the digital modules, while DV_{CC} supplies the I/Os and all analog modules (including the oscillators). The V_{CORE} output is maintained using a dedicated voltage reference. V_{CORE} is programmable up to four steps, to provide only as much power as is needed for the speed that has been selected for the CPU. This enhances power efficiency of the system. The input or primary side of the regulator is referred to in this chapter as its high side. The output or secondary side is referred to in this chapter as its low side.

The required minimum voltage for the core depends on the selected MCLK rate. Figure 2-1 shows the relationship between the system frequency for a given core voltage setting, as well as the minimum required voltage applied to the device. Figure 2-1 is only an example—see the device-specific data sheet to determine which core voltage levels are supported and what level of system frequency performance is possible for a given device.



The numbers within the fields denote the supported PMMCOREVx settings.

Figure 2-1. System Frequency, Supply Voltage, and Core Voltage – See Device-Specific Data Sheet

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The PMM module provides means for DV_{CC} and V_{CORE} to be supervised and monitored. Both of these functions detect when a voltage falls under a specific threshold. In general, the difference is that supervision results in a power-on reset (POR) event, while monitoring results in the generation of an interrupt flag that software may then handle. As such, DV_{CC} is supervised and monitored by the high-side supervisor (SVS_H) and high-side monitor (SVM_H), respectively. V_{CORE} is supervised and monitored by the low-side supervisor (SVS_L) and low-side monitor (SVM_L), respectively. Thus, there are four separate supervision and monitoring modules that can be active at any given time. The thresholds enforced by these modules are derived from the same voltage reference used by the regulator to generate V_{CORE}.

In addition to the SVS_H, SVM_H, SVS_L, and SVM_L modules, V_{CORE} is further monitored by the brownout reset (BOR) circuit. As DV_{CC} ramps up from 0 V at power up, the BOR keeps the device in reset until V_{CORE} is at a sufficient level for operation at the default MCLK rate and for the SVS_H and SVS_L mechanisms to be activated. During operation, the BOR also generates a reset if V_{CORE} falls below a preset threshold. BOR can be used to provide an even lower-power means of monitoring the supply rail if the flexibility of the SVS_L is not required.

The block diagram of the PMM is shown in Figure 2-2.

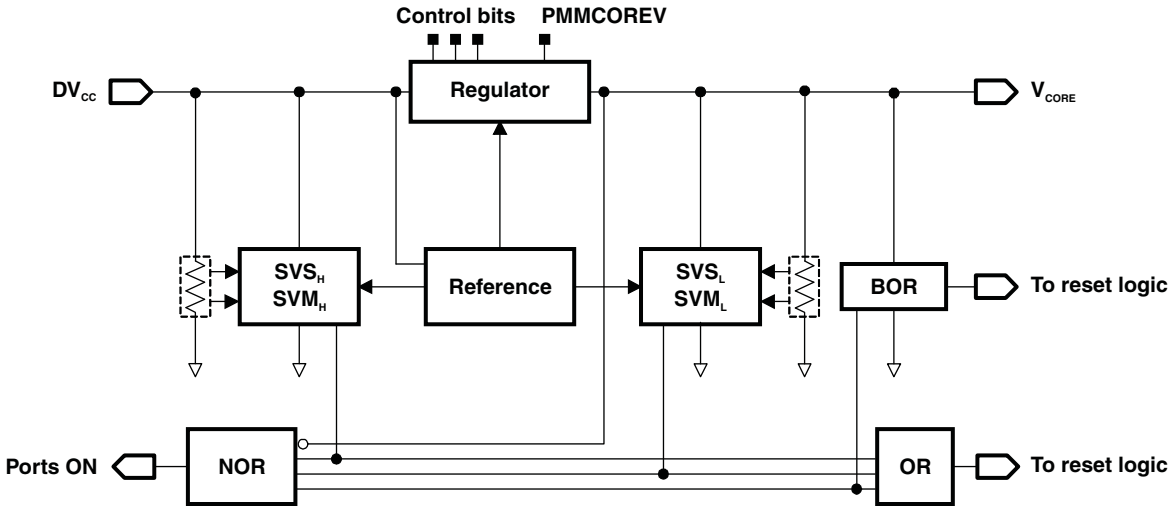


Figure 2-2. PMM Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.2 PMM Operation

2.2.1 V_{CORE} and the Regulator

DV_{CC} can be powered from a wide input voltage range, but the core logic of the device must be kept at a voltage lower than what this range allows. For this reason, a regulator has been integrated into the PMM. The regulator derives the necessary core voltage (V_{CORE}) from DV_{CC}.

Higher MCLK speeds require higher levels of V_{CORE}. Higher levels of V_{CORE} consume more power, and so the core voltage has been made programmable in up to four steps to allow it to provide only as much power as is required for a given MCLK setting. The level is controlled by the PMMCOREV bits. Note that the default setting, the lowest value of PMMCOREV, enables operation of MCLK over a very wide frequency range. As such, no PMM changes are required for many applications. See the device-specific data sheet for performance characteristics and core step levels supported.

Before increasing MCLK to a higher speed, it is necessary for software to ensure that the V_{CORE} level is sufficiently high for the chosen frequency. Failure to do so may force the CPU to attempt operation without sufficient power, which can cause unpredictable results. See Section 2.2.4 for more information on the appropriate procedure to raise V_{CORE} for higher MCLK frequencies.

The regulator supports two different load settings to optimize power. The high-current mode is required when:

- The CPU is in active, LPM0, or LPM1 modes
- A clock source greater than 32 kHz is used to drive any module
- An interrupt is executed

Otherwise, the low-current mode is used. The hardware controls the load settings automatically, according to the criteria above.

2.2.2 Supply Voltage Supervisor and Monitor

The high-side supervisor and monitor (SVS_H and SVM_H) oversee DV_{CC}, and the low-side supervisor and monitor (SVS_L and SVM_L) oversee V_{CORE}. By default, all of these modules are active, but each can be disabled using the corresponding enable bit (SVSHE, SVMHE, SVSLE, SVMLE), resulting in some power savings.

Typical application scenarios for supply voltage supervisors and monitors are:

- High-Side Supervisor, SVSH
  - Supervision of external power supply (DVCC)
  - Device reset because of low battery or supply voltage
- High-Side Monitor, SVMH
  - Monitoring of external power supply (DVCC)
  - Detection of low battery voltage (Pre-warning)
- Low-Side Supervisor, SVSL
  - Supervision of internal core voltage used to supply digital core
  - Device reset because of disruptive conditions at external VCORE pin (for example a short). The internal core voltage never drops below a critical level if parasitic events at the external VCORE pin are avoided.
- Low-Side Monitor, SVML
  - Monitoring of internal core voltage used to supply digital core
  - Detection of correct internal voltage levels when changing (especially increasing) the core voltage level before changing, for example, to higher system frequencies (also see Section 2.2.4).

2.2.2.1 SVS and SVM Thresholds

The voltage thresholds enforced by the SVS and SVM modules are selectable. Table 2-1 shows the SVS and SVM threshold registers, the voltage threshold they control, and the number of threshold options.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 2-1. SVS and SVM Thresholds

Register	Description	Threshold	Available Steps
SVSHRVL	SVS_H reset voltage level	SVS_{H_IT-}	4
SVSMHRRL	SVS_H, SVM_H reset release voltage level	SVS_{H_IT+}, SVM_H	8
SVSLRVL	SVS_L reset voltage level	SVS_{L_IT-}	4
SVSMLRRL	SVS_L, SVM_L reset release voltage level	SVS_{L_IT+}, SVM_L	4(1)

(1) The register settings support up to eight levels (0 through 7); however, levels 3 through 7 are identical.

2.2.2.1.1 Recommended SVS_L Settings

For each of the core voltages, there are two supply voltage supervisor levels available. The SVSLRVL bits define the voltage level of VCORE below which the reset is activated. The SVSMLRRL bits define the voltage level of VCORE at which the reset is released. Although various settings can be chosen, there is one set of SVSLRVL and SVSMLRRL settings that is well suited for each core voltage selected by PMMCOREV. By default, an SVS_L event always generates a POR (SVSLPE = 1), and it is recommended to always configure SVSLPE = 1 for reliable device startup. The most commonly used and recommended settings are shown in Table 2-2.

Table 2-2. Recommended SVS_L Settings

PMMCOREV[1:0]	DVCC (V)	SVSLRVL[1:0] Sets SVS_{L_IT-} Level	SVSMLRRL[2:0] Sets SVS_{L_IT+} and SVM_L levels
00	≥ 1.8	00	000
01	≥ 2.0	01	001
10	≥ 2.2	10	010
11	≥ 2.4	11	011

2.2.2.1.1.1 Recommended SVS_H Settings

For the high-side supply, there are two supply voltage supervisor levels available. The SVSMHRRL bits define the voltage level of DVCC at which the reset is released. The SVSHRVL register defines the voltage level of DVCC below which the reset is turned on. These settings should be selected according to the minimum voltages required for device operation in a given application, as well as system power supply characteristics. See the device-specific data sheet for threshold values corresponding to the settings shown here. Although various settings are available, the most common are based on the maximum frequency required which, in turn, determines the minimum DVCC level supervised. By default, an SVS_H event always generates a POR (SVSHPE = 1), and it is recommended to always configure SVSHPE = 1 for reliable device startup. The most commonly used and recommended settings are shown in Table 2-3 .

Table 2-3. Recommended SVS_H Settings

f_{SYS} Max (MHz)	DVCC (V)	SVSHRVL[1:0] Sets SVS_{H_IT-} Level	SVSMHRRL[2:0] Sets SVS_{H_IT+} and SVM_H Levels	PMMCOREV[1:0]
8	>1.8	00	000	00
12	>2.0	01	001	01
20	>2.2	10	010	10
25	>2.4	11	011	11

The available voltage threshold settings of SVS_H and SVM_H are dependent on the voltage level setting of VCORE. Table 2-4 summarizes all the possible settings available. All other settings not listed are invalid and should not be used. Figure 2-3 shows the available settings for the SVM_H.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 2-4. Available SVS_H and SVM_H Settings Versus V_{CORE} Settings

PMMCOREV[1:0]	SVSHRVL[1:0] Sets SVS_{H_IT-} Level	SVSMHRRL[2:0] Sets SVS_{H_IT+} and SVM_H Levels
00	00 through 11	000 through 011
01	00 through 11	001 through 100
10	00 through 11	010 through 101
11	00 through 11	011 through 111

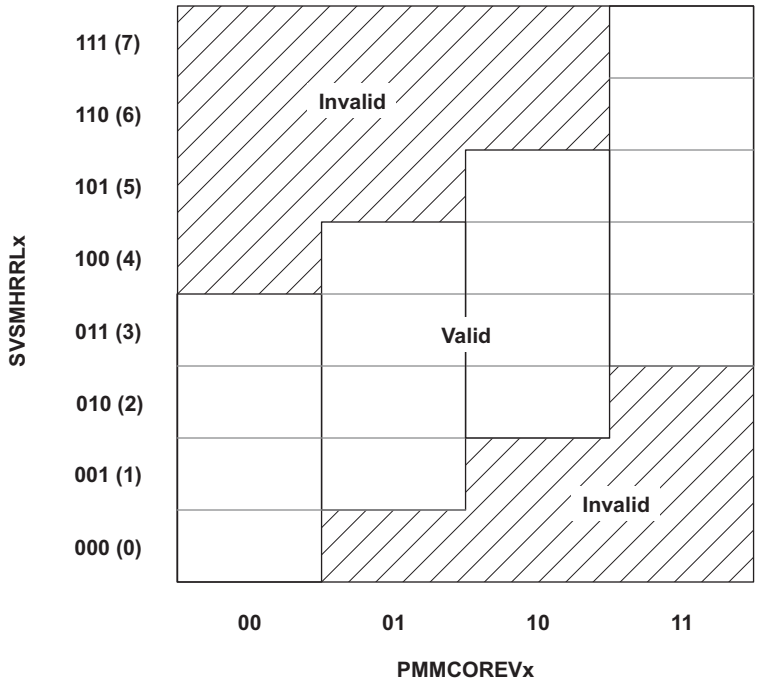


Figure 2-3. Available SVM_H Settings Versus VCORE Settings

The behavior of the SVS and SVM according to these thresholds is best portrayed graphically. Figure 2-4 shows how the supervisors and monitors respond to various supply failure conditions.

As Figure 2-4 shows, there is hysteresis built into the supervision thresholds, such that the thresholds in force depend on whether the voltage rail is going up or down. There is no hysteresis in the monitoring thresholds.

Copyright © 2009–2013, Texas Instruments Incorporated

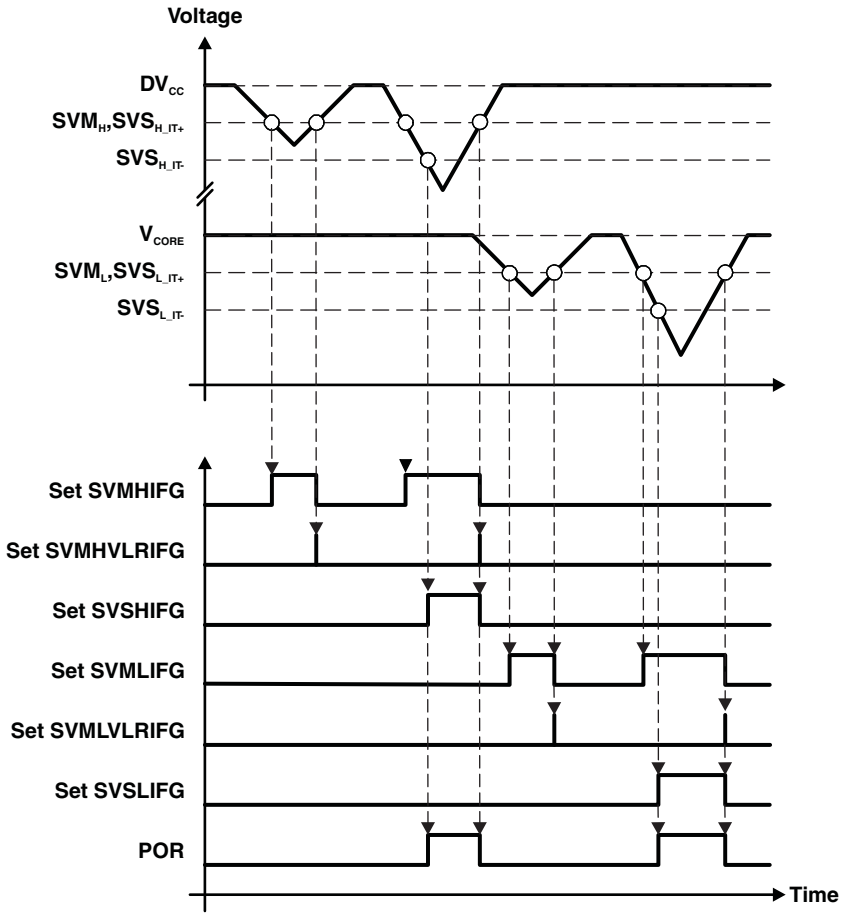


Figure 2-4. High-Side and Low-Side Voltage Failure and Resulting PMM Actions

2.2.2.2 High-Side Supervisor (SVS_H) and High-Side Monitor (SVM_H)

The SVS_H and SVM_H modules are enabled by default. They can be disabled by clearing the SVSHE and SVMHE bits, respectively. Their block diagrams are shown in Figure 2-5.

Copyright © 2009–2013, Texas Instruments Incorporated

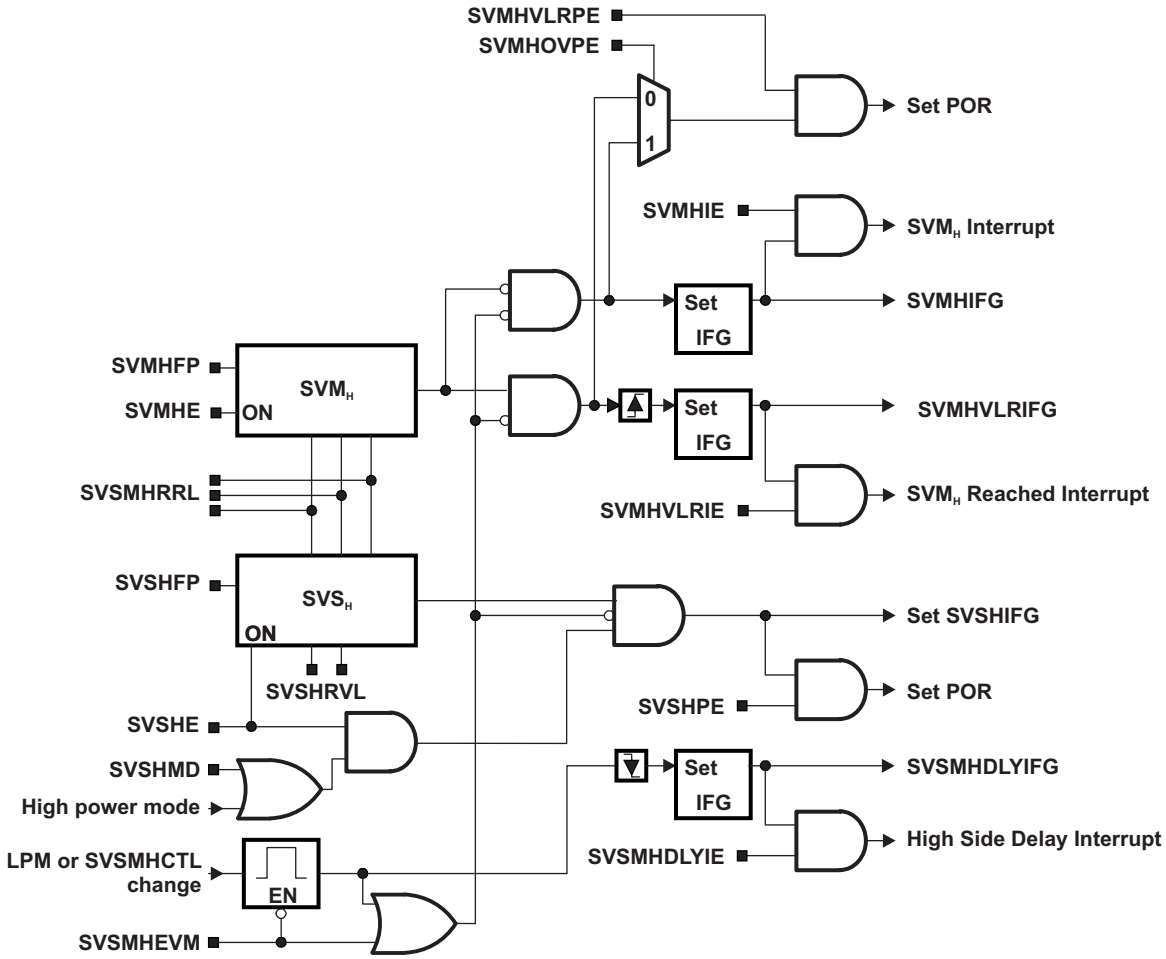


Figure 2-5. High-Side SVS and SVM

If DV_{CC} falls below the SVS_H level, SVSHIFG (SVS_H interrupt flag) is set. If DV_{CC} remains below the SVS_H level and software attempts to clear SVSHIFG, it is immediately set again by hardware. If the SVSHPE (SVS_H POR enable) bit is set when SVSHIFG gets set, a POR is generated.

If DV_{CC} falls below the SVM_H level, SVMHIFG (SVM_H interrupt flag) is set. If DV_{CC} remains below the SVM_H level and software attempts to clear SVMHIFG, it is immediately set again by hardware. If the SVMHIE (SVM_H interrupt enable) bit is set when SVMHIFG gets set, an interrupt is generated. If a POR is desired when SVMHIFG is set, the SVM_H can be configured to do so by setting the SVMHVLRPE (SVM_H voltage level reached POR enable) bit while SVMHOVPE bit is cleared.

If DV_{CC} rises above the SVM_H level, the SVMHVLRIFG (SVM_H voltage level reached) interrupt flag is set. If SVMHVLRIE (SVM_H voltage level reached interrupt enable) is set when this occurs, an interrupt is also generated.

Alternatively the SVM_H module can be used for overvoltage detection, but only with the highest core voltage setting (PMMCOREV = 11b), . This is accomplished by setting the SVMHOVPE (SVM_H overvoltage POR enable) bit in addition to setting SVMHVLRPE. Under these conditions, if a rising DVCC exceeds safe device operation, a POR is generated.

The SVS_H and SVM_H modules have configurable performance modes for power-saving operation. (See Section 2.2.9 for more information.) If these SVS_H and SVM_H power modes are modified, or if a voltage level is modified, a delay element masks the interrupts and POR sources until the SVS_H and SVM_H circuits have settled. When SVSMHDLYST (delay status) reads zero, the delay has expired. In addition, the SVSMHDLYIFG (SVS_H and SVM_H delay expired) interrupt flag is set. If the SVSMHDLYIE (SVS_H and SVM_H delay expired interrupt enable) is set when this occurs, an interrupt is also generated.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

In case of power-fail conditions, setting SVSHMD causes the SVS_H interrupt flag to be set in LPM2, LPM3, and LPM4. If SVSHMD is not set, the SVS_H interrupt flag is not set in LPM2, LPM3, and LPM4. In addition, all SVS_H and SVM_H events can be masked by setting SVSMHEVM. For most applications, SVSMHEVM should be cleared.

All the interrupt flags of SVS_H and SVM_H remain set until cleared by a BOR or by software.

2.2.2.3 Low-Side Supervisor (SVS_L) and Low-Side Monitor (SVM_L)

The SVS_L and SVM_L modules are enabled by default. They can be disabled by clearing SVSLE and SVMLE bits, respectively. Their block diagrams are shown in Figure 2-6.

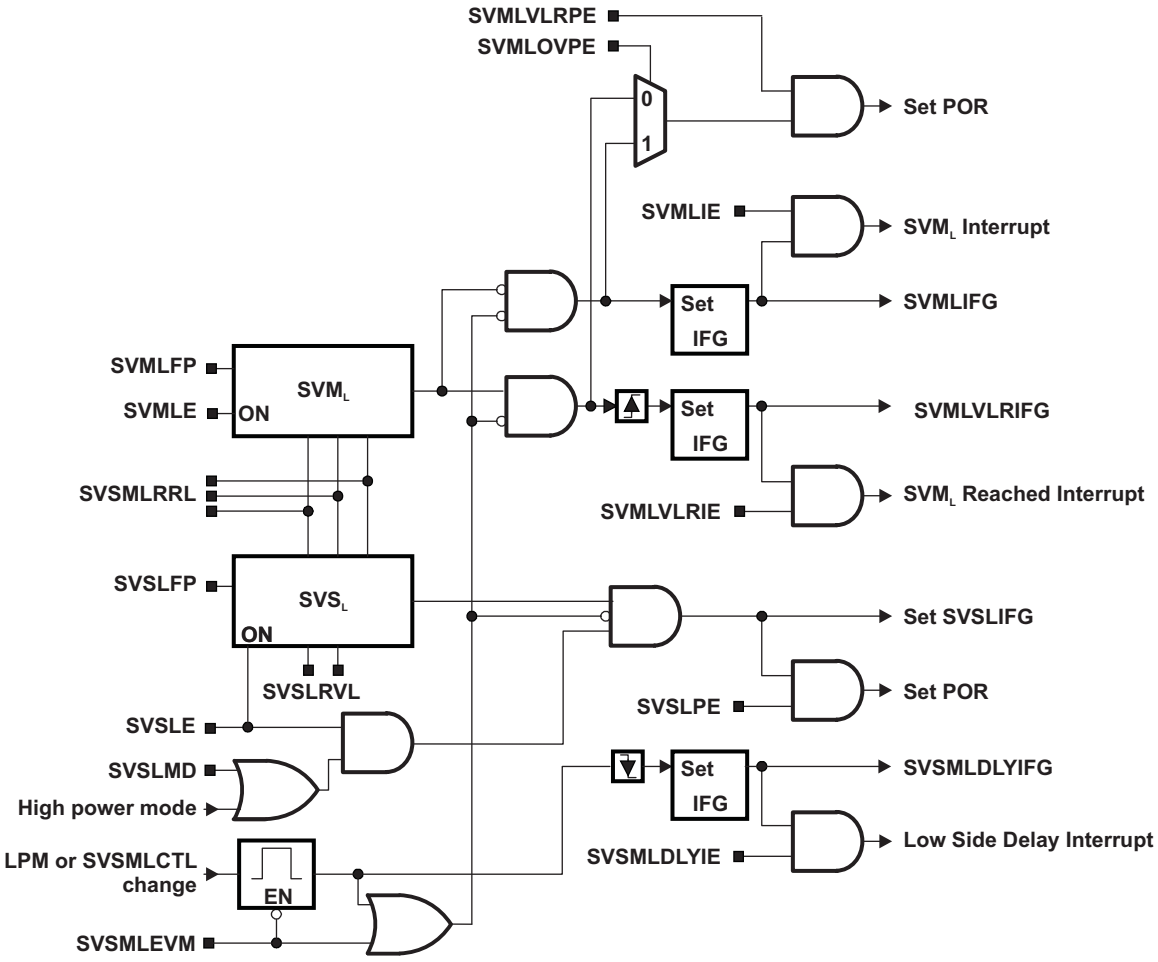


Figure 2-6. Low-Side SVS and SVM

If V_{CORE} falls below the SVS_L level, SVSLIFG (SVS_L interrupt flag) is set. If V_{CORE} remains below the SVS_L level and software attempts to clear SVSLIFG, it is immediately set again by hardware. If the SVSLPE (SVS_L POR enable) bit is set when SVSLIFG gets set, a POR is generated.

If V_{CORE} falls below the SVM_L level, SVMLIFG (SVM_L interrupt flag) is set. If V_{CORE} remains below the SVM_L level and software attempts to clear SVMLIFG, it is immediately set again by hardware. If the SVMLIE (SVM_L interrupt enable) bit is set when SVMLIFG gets set, an interrupt is generated. If a POR is desired when SVMLIFG is set, the SVM_L can be configured to do so by setting the SVMLVLRPE (SVM_L voltage level reached POR enable) bit while SVMLOVPE bit is cleared.

If V_{CORE} rises above the SVM_L level, the SVMLVLRIFG (SVM_L voltage level reached) interrupt flag is set. If SVMLVLRIE (SVM_L voltage level reached interrupt enable) is set when this occurs, an interrupt is also generated.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The SVM_L module can also be used for overvoltage detection. This is accomplished by setting the SVMLOVPE (SVM_L overvoltage POR enable) bit, in addition to setting SVMLVLRPE. Under these conditions, if V_{CORE} exceeds safe device operation, a POR is generated.

The SVS_L and SVM_L modules have configurable performance modes for power-saving operation. (See Section 2.2.9 for more information.) If these SVS_L and SVM_L power modes are modified, or if a voltage level is modified, a delay element masks the interrupts and POR sources until the SVS_L and SVM_L circuits have settled. When SVSMLDLYST (delay status) reads zero, the delay has expired. In addition, the SVSMLDLYIFG (SVS_L/SVM_L delay expired) interrupt flag is set. If the SVSMLDLYIE (SVS_L /SVM_L delay expired interrupt enable) is set when this occurs, an interrupt is also generated.

In case of power-fail conditions, setting SVSLMD causes the SVS_L interrupt flag to be set in LPM2, LPM3, and LPM4. If SVSLMD is not set, the SVS_L interrupt flag is not set in LPM2, LPM3, and LPM4. In addition, all SVS_L and SVM_L events can be masked by setting SVSMLEVM. For most applications, SVSMLEVM should be cleared.

All the interrupt flags of SVS_L and SVM_L remain set until cleared by a BOR or by software.

2.2.3 Supply Voltage Supervisor and Monitor - Power-Up

When the device is powering up, the SVS_H and SVS_L functions are enabled by default. Initially, DV_{CC} is low, and therefore the PMM holds the device in POR reset. When both the SVS_H and SVS_L levels are met, the reset is released. Figure 2-7 shows this process.

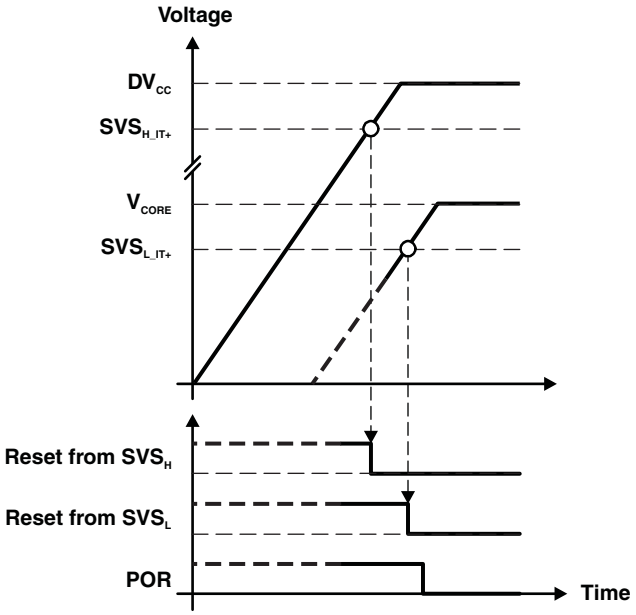


Figure 2-7. PMM Action at Device Power-Up

After this point, both voltage domains are supervised and monitored while the respective modules are enabled.

2.2.4 Increasing V_{CORE} to Support Higher MCLK Frequencies

With a reset, V_{CORE} and all the PMM thresholds, default to their lowest possible levels. These default settings allow a wide range of MCLK operation, and in many applications no change to these levels is required. However, if the application requires the performance provided by higher MCLK frequencies, software should ensure that V_{CORE} has been raised to a sufficient voltage level before changing MCLK, since failing to supply sufficient voltage to the CPU could produce unpredictable results. For a given device, minimum V_{CORE} levels required for maximum MCLK frequencies have been established (See the device data sheet for specific values).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

After setting PMMCOREV to increase V_{CORE}, there is a time delay until the new voltage has been established. Software must not raise MCLK until the necessary core voltage has settled. SVM_L can be used to verify that V_{CORE} has met the required minimum value, prior to increasing MCLK. Figure 2-8 shows this procedure.

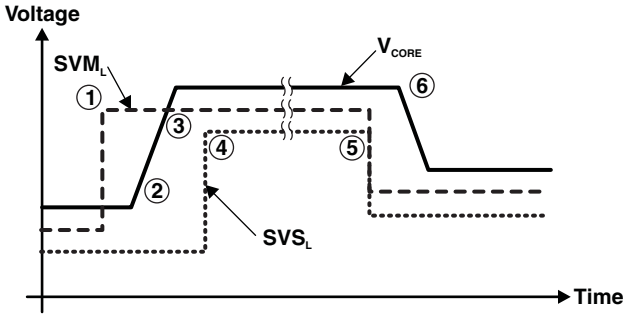


Figure 2-8. Changing V_{CORE} and SVM_L and SVS_L Levels

It is critical that the V_{CORE} level be increased by only one level at a time. The following steps 1 through 4 show the procedure to increase V_{CORE} by one level. This sequence is repeated to change the V_{CORE} level until the targeted level is obtained:

- Step 1: Program the SVM_H and SVS_H to the next level to ensure DV_{CC} is high enough for the next V_{CORE} level. Program the SVM_L to the next level and wait for (SVSMLDLYIFG) to be set.
- Step 2: Program PMMCOREV to the next V_{CORE} level.
- Step 3: Wait for the voltage level reached (SVMLVLRIFG) flag.
- Step 4: Program the SVS_L to the next level.

As a reference, the following is a C code example for increasing V_{CORE}. The sample libraries provide routines for increasing and decreasing the V_{CORE} and should be used whenever possible.

```
; C Code example for increasing core voltage.
; Note: Change core voltage one level at a time.


void SetVCoreUp (unsigned int level)
{
 // Open PMM registers for write access
 PMMCTL0_H = 0xA5;
 // Make sure no flags are set for iterative sequences
 while ((PMMIFG & SVSMHDLYIFG) == 0);
 while ((PMMIFG & SVSMLDLYIFG) == 0);
 // Set SVS/SVM high side new level
 SVSMHCTL = SVSHE + SVSHRVL0 * level + SVMHE + SVSMHRRL0 * level;
 // Set SVM low side to new level
 SVSMLCTL = SVSLE + SVMLE + SVSMLRRL0 * level;
 // Wait till SVM is settled
 while ((PMMIFG & SVSMLDLYIFG) == 0);
 // Clear already set flags
 PMMIFG &= ~(SVMLVLRIFG + SVMLIFG);
 // Set VCore to new level
 PMMCTL0_L = PMMCOREV0 * level;
 // Wait till new level reached
 if ((PMMIFG & SVMLIFG))
   while ((PMMIFG & SVMLVLRIFG) == 0);
 // Set SVS/SVM low side to new level
 SVSMLCTL = SVSLE + SVSLRVL0 * level + SVMLE + SVSMLRRL0 * level;
 // Lock PMM registers for write access
 PMMCTL0_H = 0x00;
}
```


Copyright © 2009–2013, Texas Instruments Incorporated

NOTE: See the *MSP430x5xx and MSP430x6xx Core Libraries* (SLAA448). These libraries contain useful and ready-to-use functions for easily configuring and using the PMM module.

2.2.5 Decreasing V_{CORE} for Power Optimization

The risk posed by increasing MCLK frequency does not exist when decreasing MCLK from the current V_{CORE} or higher settings, because higher V_{CORE} levels can still support MCLK frequencies below the ones for which they were intended. However, significant power efficiency gains can be made by operating V_{CORE} at the lowest value required for a given MCLK frequency. It is critical that the V_{CORE} level be decreased by only one level at a time. The following steps show the procedure to decrease V_{CORE} by one level. This sequence is repeated to change the V_{CORE} level until the targeted level is obtained:

Steps 5 through 6 show the procedure to decrease V_{CORE}:

- Step 5: Program the SVM_L and SVS_L to the new level and wait for (SVSMLDLYIFG) to be set.
- Step 6: Program PMMCOREV to the new V_{CORE} level.

It is critical when lowering the V_{CORE} setting that the maximum MCLK frequency for the new V_{CORE} setting is not violated (see the device-specific data sheet).

2.2.6 Transition From LPM3 and LPM4 Modes to AM

The LDO requires time to settle when the application transitions from low-power modes to active modes. If a transition from LPM3 or LPM4 occurs and the devices does not stay in active mode long enough, the LDO does not have time to settle sufficiently. Circuitry inside the LDO ensures that the LDO has its minimum required time to settle to its proper operating voltage. The circuitry ensures that every eighth transition from LPM3 or LPM4 causes the LDO to remain on long enough to properly settle. This is handled automatically and requires no setting by the application.

2.2.7 LPM3.5 and LPM4.5

LPM3.5 and LMP4.5 are additional low-power modes in which the regulator of the PMM is completely disabled, providing additional power savings. Not all devices support all LPMx.5 modes, so see the device-specific data sheet. Because there is no power supplied to VCORE during LPMx.5, the CPU and all digital modules including RAM are unpowered. This disables the entire device and, as a result, the contents of the registers and RAM are lost. Any essential values should be stored to flash prior to entering LPMx.5. PMMREGOFF bit is used to disable the regulator. See the SYS module for complete descriptions and proper uses of LMPx.5.

Because the regulator of the PMM is disabled upon entering LPMx.5, all I/O register configurations are lost. Therefore, the configuration of I/O pins must be handled differently to ensure that all pins in the application behave in a controlled manner upon entering and exiting LPMx.5. Properly setting the I/O pins is critical to achieving the lowest possible power consumption in LPMx.5, as well as preventing any possible uncontrolled input or output I/O state in the application. The application has complete control of the I/O pin conditions preventing the possibility of unwanted spurious activity upon entry and exit from LPMx.5. The I/O pin state is held and locked based on the settings prior to LPMx.5 entry. Upon entry into LPMx.5, the LOCKLPM5 bit in PM5CTL0 of the PMM module is set automatically. Note that only the pin condition is retained. All other port configuration register settings are lost. See the Digital I/O chapter for further details.

2.2.8 Brownout Reset (BOR), Software BOR, Software POR

The primary function of the brownout reset (BOR) circuit occurs when the device is powering up. It is functional very early in the power-up ramp, generating a POR that initializes the system. It also functions when no SVS is enabled and a brownout condition occurs. It sustains this reset until the input power is sufficient for the logic, for proper reset of the system.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

In an application, it may be desired to cause a BOR via software. Setting PMMSWBOR causes a software-driven BOR. PMMBORIFG is set accordingly. Note that a BOR also initiates a POR and PUC. PMMBORIFG can be cleared by software or by reading SYSRSTIV. Similarly, it is possible to cause a POR via software by setting PMMSWPOR. PMMPORIFG is set accordingly. A POR also initiates a PUC. PMMPORIFG can be cleared by software or by reading SYSRSTIV. Both PMMSWBOR and PMMSWPOR are self clearing. See the SYS module for complete descriptions of BOR, POR, and PUC resets.

2.2.9 SVS and SVM Performance Modes and Wakeup Times

The supervisors/monitors can function in one of two modes: normal and full performance. The difference is a tradeoff in response time versus the power consumed; full-performance mode has a faster response time but consumes considerably more power than normal mode. Full-performance mode might be considered in applications in which the decoupling of the external power supply cannot adequately prevent fast spikes on DV_{CC} from occurring, or when the application has a particular intolerance to failure. In such cases, full-performance mode provides an additional layer of protection.

There are two ways to control the performance mode: manual and automatic. In manual mode, the normal/full-performance selection is the same for every operational mode except LPMx.5 (the SVS and SVM are always disabled in LPMx.5). In this case, the normal or full-performance selection is made with the SVSHFP, SVMHFP, SVSLFP, or SVMLFP bit, for their respective modules.

In automatic mode, hardware changes the normal or full-performance selection depending on the operational mode in effect.

The wakeup time of the device from low-power modes is affected by the settings of the SVS_L and SVM_L performance modes as listed in Table 2-6, Table 2-7, Table 2-8, and Table 2-9. The wakeup time from low-power modes is not affected by the settings of the SVS_H and SVM_H. All wake-ups from LPMx.5 (LPM3.5 or LPM4.5), are defined by the data sheet parametric, $t_{WAKE-UP-LPM5}$, regardless of the performance modes for SVS_L or SVM_L, because these are disabled in LPMx.5.

The tables in Section 2.2.9.1 and Section 2.2.9.2 show the required settings to select the control and performance modes for SVS_L, SVM_L, SVS_H, and SVM_H.

NOTE: Low-Power Modes

Even if the CPU requests a specific low-power mode, the device might not go into that state because of modules requesting clocks that should be switched off or have higher frequencies or because of modules requesting a higher drive capability of the LDO. The low-power modes mentioned in the tables assume that the device is actually in the requested state; that is, no module is requesting a deviating clock setting or drive capability.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.2.9.1 Low-Side SVS and SVM Control and Performance Mode Selection

Table 2-5. SVS_L and SVM_L Control Mode Selection

SVSMLACE	SVSLMD	SVS_L Control Mode	SVM_L Control Mode
0	0	Automatic (see Table 2-6)	Manual (see Table 2-8)
0	1	Manual (see Table 2-7)	Manual (see Table 2-8)
1	0	Automatic (see Table 2-6)	Automatic (see Table 2-9)
1	1	Automatic (see Table 2-6)	Automatic (see Table 2-9)

Table 2-6. SVS_L Automatic Performance Control

SVSLE	SVSLMD	SVSLFP	AM, LPM0, LPM1 SVS_L State	LPM2, LPM3, LPM4 SVS_L State	Wakeup Time LPM2, LPM3, LPM4
0	x	x	Off	Off	$t_{WAKE-UP-FAST}$
1	0	0	Normal	Off	$t_{WAKE-UP-SLOW}$
1	0	1	Full performance	Off	$t_{WAKE-UP-FAST}$
1	1	0	Normal	Off	$t_{WAKE-UP-SLOW}$
1	1	1	Full performance	Normal	$t_{WAKE-UP-FAST}$

Table 2-7. SVS_L Manual Performance Modes

SVSLE		SVSLFP	AM, LPM0, LPM1 SVS_L State	LPM2, LPM3, LPM4 SVS_L State	Wakeup Time LPM2, LPM3, LPM4
0		x	Off	Off	$t_{WAKE-UP-FAST}$
1		0	Normal	Normal	$t_{WAKE-UP-SLOW}$
1		1	Full performance	Full performance	$t_{WAKE-UP-FAST}$

Table 2-8. SVM_L Automatic Performance Control

SVMLE		SVMLFP	AM, LPM0, LPM1 SVM_L State	LPM2, LPM3, LPM4 SVM_L State	Wakeup Time LPM2, LPM3, LPM4
0		x	Off	Off	$t_{WAKE-UP-FAST}$
1		0	Normal	Off	$t_{WAKE-UP-SLOW}$
1		1	Full performance	Normal	$t_{WAKE-UP-FAST}$

Table 2-9. SVM_L Manual Performance Modes

SVMLE		SVMLFP	AM, LPM0, LPM1 SVM_L State	LPM2, LPM3, LPM4 SVM_L State	Wakeup Time LPM2, LPM3, LPM4
0		x	Off	Off	$t_{WAKE-UP-FAST}$
1		0	Normal	Normal	$t_{WAKE-UP-SLOW}$
1		1	Full performance	Full performance	$t_{WAKE-UP-FAST}$

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.2.9.2 High-Side SVS and SVM Control and Performance Mode Selection

Table 2-10. SVS_H and SVM_H Control Mode Selection

SVSMHACE	SVSHMD	SVS_H Control Mode	SVM_H Control Mode
0	0	Automatic (see Table 2-11)	Manual (see Table 2-13)
0	1	Manual (see Table 2-12)	Manual (see Table 2-13)
1	0	Automatic (see Table 2-11)	Automatic (see Table 2-14)
1	1	Automatic (see Table 2-11)	Automatic (see Table 2-14)

Table 2-11. SVS_H Automatic Performance Control

SVSHE	SVSHMD	SVSHFP	AM, LPM0, LPM1 SVS_H State	LPM2, LPM3, LPM4 SVS_H State
0	x	x	Off	Off
1	0	0	Normal	Off
1	0	1	Full performance	Off
1	1	0	Normal	Off
1	1	1	Full performance	Normal

Table 2-12. SVS_H Manual Performance Modes

SVSHE		SVSHFP	AM, LPM0, LPM1 SVS_H State	LPM2, LPM3, LPM4 SVS_H State
0		x	Off	Off
1		0	Normal	Normal
1		1	Full performance	Full performance

Table 2-13. SVM_H Automatic Performance Control

SVMHE		SVMHFP	AM, LPM0, LPM1 SVMH State	LPM2, LPM3, LPM4 SVM_H State
0		x	Off	Off
1		0	Normal	Off
1		1	Full performance	Normal

Table 2-14. SVM_H Manual Performance Modes

SVMHE		SVMHFP	AM, LPM0, LPM1 SVM_H State	LPM2, LPM3, LPM4 SVM_H State
0		x	Off	Off
1		0	Normal	Normal
1		1	Full performance	Full performance

2.2.9.3 Wakeup Times in Debug Mode

The TEST/SBWTCK pin is used for interfacing to the development tools via Spy-Bi-Wire and JTAG. When the TEST/SBWTCK pin is high, wakeup times from LPM2, LPM3, and LPM4 may be different compared to when TEST/SBWTCK is low. When the TEST/SBWTCK pin is high, all delays associated with the SVS_L and SVM_L settings have no effect and the device wakes within $t_{WAKE-UP-FAST}$. Pay careful attention to the real-time behavior when exiting from LPM2, LPM3, and LPM4 with the device connected to a development tool (for example, MSP-FET430UIF).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.2.10 PMM Interrupts

Interrupt flags generated by the PMM are routed to the system NMI interrupt vector generator register, SYSSNIV. When the PMM causes a reset, a value is generated in the system reset interrupt vector generator register, SYSRSTIV, corresponding to the source of the reset. These registers are defined within the SYS module. More information on the relationship between the PMM and SYS modules is available in the SYS chapter.

2.2.11 Port I/O Control

The PMM provides a means of ensuring that I/O pins cannot behave in uncontrolled fashion during an undervoltage event. During these times, outputs are disabled, both normal drive and the weak pullup/pulldown function. If the CPU is functioning normally, and then an undervoltage event occurs, any pin configured as an input has its PxIN register value locked in at the point the event occurs, until voltage is restored. During the undervoltage event, external voltage changes on the pin are not registered internally. This helps prevent erratic behavior from occurring.

2.2.12 Supply Voltage Monitor Output (SVMOUT, Optional)

The state of SVMLIFG, SVMLVLRIFG, SVMHIFG, and SVMLVLRIFG can be monitored on the external SVMOUT pin. Each of these interrupt flags can be enabled (SVMLOE, SVMLVLROE, SVMHOE, SVMLVLROE) to generate an output signal. The polarity of the output is selected by the SVMOUTPOL bit. If SVMOUTPOL is set, the output is set to 1 if an enabled interrupt flag is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3 PMM Registers

The PMM registers are listed in Table 2-15. The base address of the PMM module can be found in the device-specific data sheet. The address offset of each PMM register is given in Table 2-15. The password, PMMPW, defined in the PMMCTL0 register controls access to all PMM, SVS, and SVM registers. Once the correct password is written, the write access is enabled. The write access is disabled by writing a wrong password in byte mode to the PMMCTL0 upper byte. Word accesses to PMMCTL0 with a wrong password triggers a PUC. A write access to a register other than PMMCTL0 while write access is not enabled causes a PUC.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 2-15. PMM Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	PMMCTL0	PMM control register 0	Read/write	Word	9600h	Section 2.3.1
00h	PMMCTL0_L		Read/write	Byte	00h	
01h	PMMCTL0_H		Read/write	Byte	96h	
02h	PMMCTL1	PMM control register 1	Read/write	Word	0000h	Section 2.3.2
02h	PMMCTL1_L		Read/write	Byte	00h	
03h	PMMCTL1_H		Read/write	Byte	00h	
04h	SVSMHCTL	SVS and SVM high side control register	Read/write	Word	4400h	Section 2.3.3
04h	SVSMHCTL_L		Read/write	Byte	00h	
05h	SVSMHCTL_H		Read/write	Byte	44h	
06h	SVSMLCTL	SVS and SVM low side control register	Read/write	Word	4400h	Section 2.3.4
06h	SVSMLCTL_L		Read/write	Byte	00h	
07h	SVSMLCTL_H		Read/write	Byte	44h	
08h	SVSMIO	SVSIN and SVMOUT control register (optional)	Read/write	Word	0020h	Section 2.3.5
08h	SVSMIO_L		Read/write	Byte	20h	
09h	SVSMIO_H		Read/write	Byte	00h	
0Ch	PMMIFG	PMM interrupt flag register	Read/write	Word	0000h	Section 2.3.6
0Ch	PMMIFG_L		Read/write	Byte	00h	
0Dh	PMMIFG_H		Read/write	Byte	00h	
0Eh	PMMRIE	PMM interrupt enable register	Read/write	Word	1100h	Section 2.3.7
0Eh	PMMRIE_L		Read/write	Byte	00h	
0Fh	PMMRIE_H		Read/write	Byte	11h	
10h	PM5CTL0	Power mode 5 control register 0	Read/write	Word	0000h	Section 2.3.8
10h	PM5CTL0_L		Read/write	Byte	00h	
11h	PM5CTL0_H		Read/write	Byte	00h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.1 PMMCTL0 Register

Power Management Module Control Register 0

Figure 2-9. PMMCTL0 Register

15	14	13	12	11	10	9	8
PMMPW							
rw-1	rw-0	rw-0	rw-1	rw-0	rw-1	rw-1	rw-0
7	6	5	4	3	2	1	0
PMMHPMRE	Reserved		PMMREGOFF	PMMSWPOR	PMMSWBOR	PMMCOREV	
rw-0	r-0	r-0	rw-0	rw-0	rw-0	rw-[0]	rw-[0]

Table 2-16. PMMCTL0 Register Description

Bit	Field	Type	Reset	Description
15-8	PMMPW	RW	96h	PMM password. Always read as 096h. When using word operations, must be written with 0A5h or a PUC is generated. When using byte operation, writing 0A5h unlocks all PMM registers. When using byte operation, writing anything different than 0A5h locks all PMM registers.
7	PMMHPMRE	RW	0h	Global high power module request enable. If the PMMHPMRE bit is set, any module is able to request the PMM high-power mode.
6-5	Reserved	R	0h	Reserved. Always reads as 0.
4	PMMREGOFF	RW	0h	Regulator off (see the SYS chapter for details)
3	PMMSWPOR	RW	0h	Software power-on reset. Setting this bit to 1 triggers a POR. This bit is self clearing.
2	PMMSWBOR	RW	0h	Software brownout reset. Setting this bit to 1 triggers a BOR. This bit is self clearing.
1-0	PMMCOREV	RW	0h	Core voltage (see the device-specific data sheet for supported levels and corresponding voltages) 00b = V(CORE) level 0 01b = V(CORE) level 1 10b = V(CORE) level 2 11b = V(CORE) level 3

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.2 PMMCTL1 Register

Power Management Module Control Register 1

Figure 2-10. PMMCTL1 Register

15	14	13	12	11	10	9	8
Reserved							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
7	6	5	4	3	2	1	0
Reserved		Reserved		Reserved		Reserved	
r-0	r-0	rw-[0]	rw-[0]	r-0	r-0	rw-0	rw-0

Table 2-17. PMMCTL1 Register Description

Bit	Field	Type	Reset	Description
15-6	Reserved	R	0h	Reserved. Always reads as 0.
5-4	Reserved	RW	0h	Reserved. Must always be written with 0.
3-2	Reserved	R	0h	Reserved. Always reads as 0.
1-0	Reserved	RW	0h	Reserved. Must always be written with 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.3 SVSMHCTL Register

Supply Voltage Supervisor and Monitor High-Side Control Register

Figure 2-11. SVSMHCTL Register

15	14	13	12	11	10	9	8
SVMHFP	SVMHE	Reserved	SVMHOVPE	SVSHFP	SVSHE	SVSHRVL	
rw-[0]	rw-1	r-0	rw-[0]	rw-[0]	rw-1	rw-[0]	rw-[0]
7	6	5	4	3	2	1	0
SVSMHACE	SVSMHEVM	Reserved	SVSHMD	SVSMHDLYST	SVSMHRRL		
rw-[0]	rw-0	r-0	rw-0	r-0	rw-[0]	rw-[0]	rw-[0]

Table 2-18. SVSMHCTL Register Description

Bit	Field	Type	Reset	Description
15	SVMHFP	RW	0h	SVM high-side full-performance mode. If this bit is set, the SVMH operates in full-performance mode. 0b = Normal mode. See the device-specific data sheet for response times. 1b = Full-performance mode. See the device-specific data sheet for response times.
14	SVMHE	RW	1h	SVM high-side enable. If this bit is set, the SVMH is enabled.
13	Reserved	R	0h	Reserved. Always reads as 0.
12	SVMHOVPE	RW	0h	SVM high-side overvoltage enable. If this bit is set, the SVMH overvoltage detection is enabled. If SVMHVLRPE is also set, a POR occurs on an overvoltage condition.
11	SVSHFP	RW	0h	SVS high-side full-performance mode. If this bit is set, the SVSH operates in full-performance mode. 0b = Normal mode. See the device-specific data sheet for response times. 1b = Full-performance mode. See the device-specific data sheet for response times.
10	SVSHE	RW	1h	SVS high-side enable. If this bit is set, the SVSH is enabled.
9-8	SVSHRVL	RW	0h	SVS high-side reset voltage level. If DVCC falls short of the SVSH voltage level selected by SVSHRVL, a reset is triggered (if SVSHPE = 1). The voltage levels are defined in the device-specific data sheet.
7	SVSMHACE	RW	0h	SVS and SVM high-side automatic control enable. If this bit is set, the low-power mode of the SVSH and SVMH circuits is under hardware control.
6	SVSMHEVM	RW	0h	SVS and SVM high-side event mask. If this bit is set, the SVSH and SVMH events are masked. 0b = No events are masked. 1b = All events are masked.
5	Reserved	R	0h	Reserved. Always reads as 0.
4	SVSHMD	RW	0h	SVS high-side mode. If this bit is set, the SVSH interrupt flag is set in LPM2, LPM3, and LPM4 in case of power-fail conditions. If this bit is not set, the SVSH interrupt is not set in LPM2, LPM3, and LPM4.
3	SVSMHDLYST	RW	0h	SVS and SVM high-side delay status. If this bit is set, the SVSH and SVMH events are masked for some delay time. The delay time depends on the power mode of the SVSH and SVMH. If SVMHFP = 1 and SVSHFP = 1 (that is, full-performance mode), the delay is shorter. See the device-specific data sheet for details. The bit is cleared by hardware if the delay has expired.
2-0	SVSMHRRL	RW	0h	SVS and SVM high-side reset release voltage level. These bits define the reset release voltage level of the SVSH. It is also used for the SVMH to define the voltage reached level. The voltage levels are defined in the device-specific data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.4 SVSMLCTL Register

Supply Voltage Supervisor and Monitor Low-Side Control Register

Figure 2-12. SVSMLCTL Register

15	14	13	12	11	10	9	8
SVMLFP	SVMLE	Reserved	SVMLOVPE	SVSLFP	SVSLE	SVSLRVL	
rw-[0]	rw-1	r-0	rw-[0]	rw-[0]	rw-1	rw-[0]	rw-[0]
7	6	5	4	3	2	1	0
SVSMLACE	SVSMLEVM	Reserved	SVSLMD	SVSMLDLYST	SVSMLRRL		
rw-[0]	rw-0	r-0	rw-0	r-0	rw-[0]	rw-[0]	rw-[0]

Table 2-19. SVSMLCTL Register Description

Bit	Field	Type	Reset	Description
15	SVMLFP	RW	0h	SVM low-side full-performance mode. If this bit is set, the SVML operates in full-performance mode. 0b = Normal mode. See the device-specific data sheet for response times. 1b = Full-performance mode. See the device-specific data sheet for response times.
14	SVMLE	RW	1h	SVM low-side enable. If this bit is set, the SVML is enabled.
13	Reserved	R	0h	Reserved. Always reads as 0.
12	SVMLOVPE	RW	0h	SVM low-side overvoltage enable. If this bit is set, the SVML overvoltage detection is enabled.
11	SVSLFP	RW	0h	SVS low-side full-performance mode. If this bit is set, the SVSL operates in full-performance mode. 0b = Normal mode. See the device-specific data sheet for response times. 1b = Full-performance mode. See the device-specific data sheet for response times.
10	SVSLE	RW	1h	SVS low-side enable. If this bit is set, the SVSL is enabled.
9-8	SVSLRVL	RW	0h	SVS low-side reset voltage level. If V(CORE) falls short of the SVSL voltage level selected by SVSLRVL, a reset is triggered (if SVSLPE = 1).
7	SVSMLACE	RW	0h	SVS and SVM low-side automatic control enable. If this bit is set, the low-power mode of the SVSL and SVML circuits is under hardware control.
6	SVSMLEVM	RW	0h	SVS and SVM low-side event mask. If this bit is set, the SVSL and SVML events are masked. 0b = No events are masked. 1b = All events are masked.
5	Reserved	R	0h	Reserved. Always reads as 0.
4	SVSLMD	RW	0h	SVS low-side mode. If this bit is set, the SVSL interrupt flag is set in LPM2, LPM3 and LPM4 in case of power-fail conditions. If this bit is not set, the SVSL interrupt is not set in LPM2, LPM3, and LPM4.
3	SVSMLDLYST	RW	0h	SVS and SVM low-side delay status. If this bit is set, the SVSL and SVML events are masked for a delay time. The delay time depends on the power mode of the SVSL and SVML. If SVMLFP = 1 and SVSLFP = 1 (that is, full-performance mode), the delay is shorter. The bit is cleared by hardware if the delay has expired.
2-0	SVSMLRRL	RW	0h	SVS and SVM low-side reset release voltage level. These bits define the reset release voltage level of the SVSL. It is also used for the SVML to define the voltage reached level.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.5 SVSMIO Register

SVSIN and SVMOUT Control Register

Figure 2-13. SVSMIO Register

15	14	13	12	11	10	9	8
Reserved			SVMHVLROE	SVMHOE	Reserved		
r-0	r-0	r-0	rw-[0]	rw-[0]	r-0	r-0	r-0
7	6	5	4	3	2	1	0
Reserved		SVMOUTPOL	SVMLVLROE	SVMLOE	Reserved		
r-0	r-0	rw-[1]	rw-[0]	rw-[0]	r-0	r-0	r-0

Table 2-20. SVSMIO Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12	SVMHVLROE	RW	0h	SVM high-side voltage level reached output enable. If this bit is set, the SVMHVLRIFG bit is output to the device SVMOUT pin. The device-specific port logic has to be configured accordingly.
11	SVMHOE	RW	0h	SVM high-side output enable. If this bit is set, the SVMHIFG bit is output to the device SVMOUT pin. The device-specific port logic has to be configured accordingly.
10-6	Reserved	R	0h	Reserved. Always reads as 0.
5	SVMOUTPOL	RW	1h	SVMOUT pin polarity. If this bit is set, SVMOUT is active high. An error condition is signaled by a 1 at SVMOUT. If SVMOUTPOL is cleared, the error condition is signaled by a 0 at the SVMOUT pin.
4	SVMLVLROE	RW	0h	SVM low-side voltage level reached output enable. If this bit is set, the SVMLVLRIFG bit is output to the device SVMOUT pin. The device-specific port logic has to be configured accordingly.
3	SVMLOE	RW	0h	SVM low-side output enable. If this bit is set, the SVMLIFG bit is output to the device SVMOUT pin. The device-specific port logic has to be configured accordingly.
2-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.6 PMMIFG Register

Power Management Module Interrupt Flag Register

Figure 2-14. PMMIFG Register

15	14	13	12	11	10	9	8
PMMLPM5IFG	Reserved	SVSLIFG(1)	SVSHIFG(1)	Reserved	PMMPORIFG	PMMRSTIFG	PMMBORIFG
rw-[0]	r-0	rw-[0]	rw-[0]	r-0	rw-[0]	rw-[0]	rw-[0]
7	6	5	4	3	2	1	0
Reserved	SVMHVLRIFG(1)	SVMHIFG	SVSMHDLYIFG	Reserved	SVMLVLRIFG(1)	SVMLIFG	SVSMLDLYIFG
r-0	rw-[0]	rw-[0]	rw-0	r-0	rw-[0]	rw-[0]	rw-0

(1) After power up, the reset value depends on the power sequence.

(1) After power up, the reset value depends on the power sequence.

Table 2-21. PMMIFG Register Description

Bit	Field	Type	Reset	Description
15	PMMLPM5IFG	RW	0h	LPMx.5 flag. This bit is set if the system was in LPMx.5 before. The bit is cleared by software or by reading the reset vector word. A power failure on the DVCC domain clears the bit. 0b = No interrupt pending 1b = Interrupt pending
14	Reserved	R	0h	Reserved. Always reads as 0.
13	SVSLIFG	RW	0h	SVS low-side interrupt flag. The bit is cleared by software or by reading the reset vector word. 0b = No interrupt pending 1b = Interrupt pending
12	SVSHIFG	RW	0h	SVS high-side interrupt flag. The bit is cleared by software or by reading the reset vector word. 0b = No interrupt pending 1b = Interrupt pending
11	Reserved	R	0h	Reserved. Always reads as 0.
10	PMMPORIFG	RW	0h	PMM software power-on reset interrupt flag. This interrupt flag is set if a software POR is triggered. The bit is cleared by software or by reading the reset vector word, SYSRSTIV. 0b = No interrupt pending 1b = Interrupt pending
9	PMMRSTIFG	RW	0h	PMM reset pin interrupt flag. This interrupt flag is set if the RST/NMI pin is the reset source. The bit is cleared by software or by reading the reset vector word. 0b = No interrupt pending 1b = Interrupt pending
8	PMMBORIFG	RW	0h	PMM software brownout reset interrupt flag. This interrupt flag is set if a software BOR (PMMSWBOR) is triggered. The bit is cleared by software or by reading the reset vector word, SYSRSTIV. 0b = No interrupt pending 1b = Interrupt pending
7	Reserved	R	0h	Reserved. Always reads as 0.
6	SVMHVLRIFG	RW	0h	SVM high-side voltage level reached interrupt flag. The bit is cleared by software or by reading the reset vector (SVSHPE = 1) word or by reading the interrupt vector (SVSHPE = 0) word. 0b = No interrupt pending 1b = Interrupt pending
5	SVMHIFG	RW	0h	SVM high-side interrupt flag. The bit is cleared by software. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 2-21. PMMIFG Register Description (continued)

Bit	Field	Type	Reset	Description
4	SVSMHDLYIFG	RW	0h	SVS and SVM high-side delay expired interrupt flag. This interrupt flag is set if the delay element expired. The bit is cleared by software or by reading the interrupt vector word. 0b = No interrupt pending 1b = Interrupt pending
3	Reserved	R	0h	Reserved. Always reads as 0.
2	SVMLVLRIFG	RW	0h	SVM low-side voltage level reached interrupt flag. The bit is cleared by software or by reading the reset vector (SVSLPE = 1) word or by reading the interrupt vector (SVSLPE = 0) word. 0b = No interrupt pending 1b = Interrupt pending
1	SVMLIFG	RW	0h	SVM low-side interrupt flag. The bit is cleared by software. 0b = No interrupt pending 1b = Interrupt pending
0	SVSMLDLYIFG	RW	0h	SVS and SVM low-side delay expired interrupt flag. This interrupt flag is set if the delay element expired. The bit is cleared by software or by reading the interrupt vector word. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.7 PMMRIE Register

Power Management Module Reset and Interrupt Enable Register

Figure 2-15. PMMRIE Register

15	14	13	12	11	10	9	8
Reserved		SVMHVLRPE	SVSHPE	Reserved		SVMLVLRPE	SVSLPE
r-0	r-0	rw-[0]	rw-[1]	r-0	r-0	rw-[0]	rw-[1]
7	6	5	4	3	2	1	0
Reserved	SVMHVLRIE	SVMHIE	SVSMHDLYIE	Reserved	SVMLVLRIE	SVMLIE	SVSMLDLYIE
r-0	rw-0	rw-0	rw-0	r-0	rw-0	rw-0	rw-0

Table 2-22. PMMRIE Register Description

Bit	Field	Type	Reset	Description
15-14	Reserved	R	0h	Reserved. Always reads as 0.
13	SVMHVLRPE	RW	0h	SVM high-side voltage level reached power-on reset enable. If this bit is set, exceeding the SVMH voltage level triggers a POR.
12	SVSHPE	RW	1h	SVS high-side power-on reset enable. If this bit is set, falling below the SVSH voltage level triggers a POR.
11-10	Reserved	R	0h	Reserved. Always reads as 0.
9	SVMLVLRPE	RW	0h	SVM low-side voltage level reached power-on reset enable. If this bit is set, exceeding the SVML voltage level triggers a POR.
8	SVSLPE	RW	1h	SVS low-side power-on reset enable. If this bit is set, falling below the SVSL voltage level triggers a POR.
7	Reserved	R	0h	Reserved. Always reads as 0.
6	SVMHVLRIE	RW	0h	SVM high-side reset voltage level interrupt enable
5	SVMHIE	RW	0h	SVM high-side interrupt enable. This bit is cleared by software or if the interrupt vector word is read.
4	SVSMHDLYIE	RW	0h	SVS and SVM high-side delay expired interrupt enable
3	Reserved	R	0h	Reserved. Always reads as 0.
2	SVMLVLRIE	RW	0h	SVM low-side reset voltage level interrupt enable
1	SVMLIE	RW	0h	SVM low-side interrupt enable. This bit is cleared by software or if the interrupt vector word is read.
0	SVSMLDLYIE	RW	0h	SVS and SVM low-side delay expired interrupt enable

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

2.3.8 PM5CTL0 Register

Power Mode 5 Control Register 0

Figure 2-16. PM5CTL0 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved							LOCKLPM5
r0	r0	r0	r0	r0	r0	r0	rw-[0]

Table 2-23. PM5CTL0 Register Description

Bit	Field	Type	Reset	Description
15-1	Reserved	R	0h	Reserved. Always reads as 0.
0	LOCKLPM5	RW	0h	Lock I/O pin configuration upon entry to or exit from LPMx.5. When power is applied to the device, this bit, once set, can only be cleared by the user or via another power cycle. Note: This bit was formerly named LOCKIO, and some application reports and code examples may continue to use this terminology. 0b = I/O pin configuration is not locked and defaults to its reset condition. 1b = I/O pin configuration remains locked. Pin state is held during LPMx.5 entry and exit.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Unified Clock System (UCS)

The Unified Clock System (UCS) module provides the various clocks for a device. This chapter describes the operation of the UCS module, which is implemented in all devices.

Topic **Page**

3.1	**Unified Clock System (UCS) Introduction**	**107**
3.2	**UCS Operation**	**109**
3.3	**Module Oscillator (MODOSC)**	**119**
3.4	**UCS Module Registers**	**120**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.1 Unified Clock System (UCS) Introduction

The UCS module supports low system cost and ultralow power consumption. Using three internal clock signals, the user can select the best balance of performance and low power consumption. The UCS module can be configured to operate without any external components, with one or two external crystals, or with resonators, under full software control.

The UCS module includes up to five clock sources:

- XT1CLK: Low-frequency oscillator that can be used with low-frequency 32768-Hz watch crystals
- VLOCLK: Internal very low power, low frequency oscillator with 10 kHz typical frequency
- REFOCLK: Internal, trimmed, low-frequency oscillator with 32768 Hz typical frequency, with the ability to be used as a clock reference into the FLL
- DCOCLK: Internal digitally-controlled oscillator (DCO) that can be stabilized by the FLL
- XT2CLK: RF XT2 oscillator required for radio functionality

Three clock signals are available from the UCS module:

- ACLK: Auxiliary clock. The ACLK is software selectable as XT1CLK, REFOCLK, VLOCLK, DCOCLK, DCOCLKDIV, and when available, XT2CLK. DCOCLKDIV is the DCOCLK frequency divided by 1, 2, 4, 8, 16, or 32 within the FLL block. ACLK can be divided by 1, 2, 4, 8, 16, or 32. ACLK/n is ACLK divided by 1, 2, 4, 8, 16, or 32 and is available externally at a pin. ACLK is software selectable by individual peripheral modules.
- MCLK: Master clock. MCLK is software selectable as XT1CLK, REFOCLK, VLOCLK, DCOCLK, DCOCLKDIV, and XT2CLK. DCOCLKDIV is the DCOCLK frequency divided by 1, 2, 4, 8, 16, or 32 within the FLL block. MCLK can be divided by 1, 2, 4, 8, 16, or 32. MCLK is used by the CPU and system.
- SMCLK: Subsystem master clock. SMCLK is software selectable as XT1CLK, REFOCLK, VLOCLK, DCOCLK, DCOCLKDIV, and XT2CLK. DCOCLKDIV is the DCOCLK frequency divided by 1, 2, 4, 8, 16, or 32 within the FLL block. SMCLK can be divided by 1, 2, 4, 8, 16, or 32. SMCLK is software selectable by individual peripheral modules.

The block diagram of the UCS module is shown in Figure 3-1.

Copyright © 2009–2013, Texas Instruments Incorporated

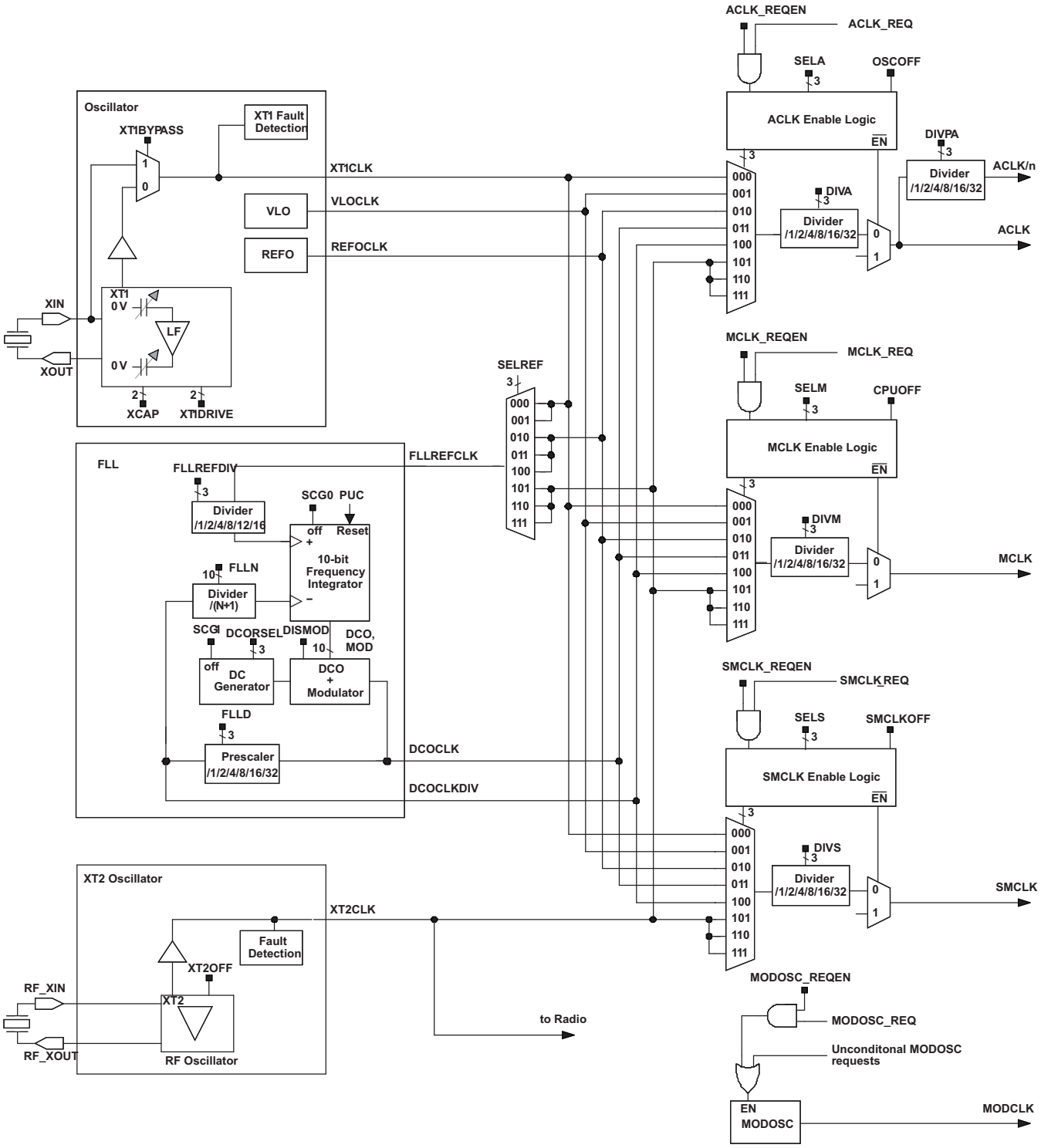


Figure 3-1. UCS Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.2 UCS Operation

After a PUC, the UCS module default configuration is:

- XT1 is selected as the oscillator source for XT1CLK. XT1CLK is selected for ACLK.
- DCOCLKDIV is selected for MCLK.
- DCOCLKDIV is selected for SMCLK.
- FLL operation is enabled and XT1CLK is selected as the FLL reference clock, FLLREFCLK.
- XIN and XOUT pins are set to general-purpose I/Os and XT1 remains disabled until the I/O ports are configured for XT1 operation.
- RF oscillator sourcing XT2CLK disabled.

As previously stated, FLL operation with XT1 is selected by default, but XT1 is disabled. The crystal pins (XIN, XOUT) are shared with general-purpose I/Os. To enable XT1, the PSEL bits associated with the crystal pins must be set. When a 32,768 Hz crystal is used for XT1CLK, the fault control logic immediately causes ACLK to be sourced by the REFOCLK, because XT1 is not stable immediately (see Section 3.2.12). Once crystal startup is obtained and settled, the FLL stabilizes MCLK and SMCLK to 1.048576 MHz and f_{DCO} = 2.097152 MHz.

Status register control bits (SCG0, SCG1, OSCOFF, and CPUOFF) configure the MSP430 operating modes and enable or disable portions of the UCS module (see the SYS chapter). Registers UCSCTL0 through UCSCTL8, configure the UCS module.

The UCS module can be configured or reconfigured by software at any time during program execution.

NOTE: For devices using RTC_B, RTC_C, or RTC_D (RTC modules supporting LPM3.5) setting bit RTCHOLD = 0 in register RTCCTL1 also enables XT1, independent from UCS configuration.

3.2.1 UCS Module Features for Low-Power Applications

Conflicting requirements typically exist in battery-powered applications:

- Low clock frequency for energy conservation and time keeping
- High clock frequency for fast response times and fast burst processing capabilities
- Clock stability over operating temperature and supply voltage
- Low-cost applications with less-constrained clock accuracy requirements

The UCS module addresses these conflicting requirements by allowing the user to select from the three available clock signals: ACLK, MCLK, and SMCLK.

All three available clock signals can be sourced via any of the available clock sources (XT1CLK, VLOCLK, REFOCLK, DCOCLK, DCOCLKDIV, or XT2CLK), giving complete flexibility in the system clock configuration. A flexible clock distribution and divider system is provided to fine tune the individual clock requirements.

3.2.2 Internal Very-Low-Power Low-Frequency Oscillator (VLO)

The internal VLO provides a typical frequency of 10 kHz (see device-specific data sheet for parameters) without requiring a crystal. The VLO provides for a low-cost ultralow-power clock source for applications that do not require an accurate time base.

The VLO is enabled when it is used to source ACLK, MCLK, or SMCLK (SELA = {1} or SELM = {1} or SELS = {1}).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.2.3 Internal Trimmed Low-Frequency Reference Oscillator (REFO)

The internal trimmed low-frequency REFO can be used for cost-sensitive applications where a crystal is not required or desired. REFO is internally trimmed to 32.768 kHz typical and provides for a stable reference frequency that can be used as FLLREFCLK. REFO, combined with the FLL, provides for a flexible range of system clock settings without the need for a crystal. REFO consumes no power when not being used.

REFO is enabled under any of the following conditions:

- REFO is a source for ACLK (SELA = {2}) and in active mode (AM) through LPM3 (OSCOFF = 0)
- REFO is a source for MCLK (SELM = {2}) and in active mode (AM) (CPUOFF = 0)
- REFO is a source for SMCLK (SELS = {2}) and in active mode (AM) through LPM1 (SMCLKOFF = 0)
- REFO is a source for FLLREFCLK (SELREF = {2}) and the DCO is a source for ACLK (SELA = {3,4}) and in active mode (AM) through LPM3 (OSCOFF = 0)
- REFO is a source for FLLREFCLK (SELREF = {2}) and the DCO is a source for MCLK (SELM = {3,4}) and in active mode (AM) (CPUOFF = 0)
- REFO is a source for FLLREFCLK (SELREF = {2}) and the DCO is a source for SMCLK (SELS = {3,4}) and in active mode (AM) through LPM1 (SMCLKOFF = 0)

3.2.4 XT1 Oscillator

The XT1 oscillator supports ultralow-current consumption using a 32,768 Hz watch crystal. A watch crystal connects to XIN and XOUT without any other external components. The software-selectable XCAP bits configure the internally provided load capacitance for the XT1 crystal in LF mode. This capacitance can be selected as 2 pF, 6 pF, 9 pF, or 12 pF (typical). Additional external capacitors can be added if necessary.

The drive settings of XT1 can be increased with the XT1DRIVE bits. At power up, the XT1 starts with the highest drive settings for fast, reliable startup. If needed, user software can reduce the drive strength to further reduce power. .

XT1 may be used with an external clock signal on the XIN pin in either LF or HF mode by setting XT1BYPASS. When used with an external signal, the external frequency must meet the data sheet parameters for the chosen mode. XT1 is powered down when used in bypass mode.

Some devices support XT1 bypass operation with external clock inputs that reside on a different external supply domain, called DV_{IO}. Please refer to the device specific datasheet. On these devices, DV_{IO} has a voltage range of 1.8V ± 10 %. When using the XT1 bypass operation with external clock inputs that reside on DV_{IO}, it is required that XT1BYPASSLV = 1. For example, when XT1BYPASSLV = 1, it is assumed the external clock signal swings from 0V to DV_{IO}. With XT1BYPASS = 0, it is assumed the external clock signal swings from 0V to DV_{CC}. The usage of XT1BYPASSLV allows for interfacing to external clock sources that reside on either the DV_{CC} or DV_{IO} supply domains. When used with an external signal, the external frequency must meet the data sheet parameters for the chosen mode. XT1 is powered down when used in bypass mode.

The XT1 pins are shared with general-purpose I/O ports. At power up, the default operation is XT1. However, XT1 remains disabled until the ports shared with XT1 are configured for XT1 operation. The configuration of the shared I/O is determined by the PSEL bit associated with XIN and the XT1BYPASS bit. Setting the PSEL bit causes the XIN and XOUT ports to be configured for XT1 operation. If XT1BYPASS is also set, XT1 is configured for bypass mode of operation, and the oscillator associated with XT1 is powered down. In bypass mode of operation, XIN can accept an external clock input signal and XOUT is configured as a general-purpose I/O. The PSEL bit associated with XOUT is a don't care.

If the PSEL bit associated with XIN is cleared, both XIN and XOUT ports are configured as general-purpose I/Os, and XT1 is disabled.

XT1 is enabled under any of the following conditions:

- XT1 is a source for ACLK (SELA = {0}) and in active mode (AM) through LPM3 (OSCOFF = 0)
- XT1 is a source for MCLK (SELM = {0}) and in active mode (AM) (CPUOFF = 0)
- XT1 is a source for SMCLK (SELS = {0}) and in active mode (AM) through LPM1 (SMCLKOFF = 0)
- XT1 is a source for FLLREFCLK (SELREF = {0}) and the DCO is a source for ACLK (SELA = {3,4}) and in active mode (AM) through LPM3 (OSCOFF = 0)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

- XT1 is a source for FLLREFCLK (SELREF = {0}) and the DCO is a source for MCLK (SELM = {3,4}) and in active mode (AM) (CPUOFF = 0)
- XT1 is a source for FLLREFCLK (SELREF = {0}) and the DCO is a source for SMCLK (SELS = {3,4}) and in active mode (AM) through LPM1 (SMCLKOFF = 0)
- XT1OFF = 0. XT1 enabled in active mode (AM) through LPM4. For devices that support LPMx.5, XT1 also remains enabled.

3.2.5 RF XT2 Oscillator

The RF XT2 oscillator's main purpose is to provide a reference clock to the on-chip radio module. But it also sources XT2CLK and XT2CLK can be used to source ACLK, MCLK, SMCLK or FLLREFCLK.

The RF oscillator is enabled if it is used by the radio; that is, if the radio module is not in its sleep state. With XT2OFF = 0 the RF oscillator is permanently enabled even when the radio is in sleep mode. With XT2OFF = 1 the RF oscillator is disabled when the radio enters sleep mode. When the RF oscillator is disabled the corresponding fault flag XT2OFFG is set and if the RF oscillator is selected to source ACLK, MCLK, SMCLK or FLLREFCLK the corresponding fail-safe mechanism takes over.

3.2.6 Digitally-Controlled Oscillator (DCO)

The DCO is an integrated digitally controlled oscillator. The DCO frequency can be adjusted by software using the DCORSEL, DCO, and MOD bits. The DCO frequency can be optionally stabilized by the FLL to a multiple frequency of FLLREFCLK/n. The FLL can accept different reference sources selectable via the SELREF bits. Reference sources include XT1CLK, REFOCLK, or XT2CLK. The value of n is defined by the FLLREFDIV bits (n = 1, 2, 4, 8, 12, or 16). The default is n = 1. There may be scenarios in which FLL operation is not required or desired; in these cases, no FLLREFCLK is necessary. This can be accomplished by setting SELREF = {7}.

The FLLD bits configure the FLL prescaler divider value D to 1, 2, 4, 8, 16, or 32. By default, D = 2, and MCLK and SMCLK are sourced from DCOCLKDIV, providing a clock frequency DCOCLK/2.

The divider (N + 1) and the divider value D define the DCOCLK and DCOCLKDIV frequencies, where N > 0. Writing N = 0 causes the divider to be set to 2.

$f_{DCOCLK} = D \times (N + 1) \times (f_{FLLREFCLK} \div n)$

$f_{DCOCLKDIV} = (N + 1) \times (f_{FLLREFCLK} \div n)$

Adjusting DCO Frequency

By default, FLL operation is enabled. FLL operation can be disabled by setting SCG0 or SCG1. Once disabled, the DCO continues to operate at the current settings defined in UCSCTL0 and UCSCTL1. The DCO frequency can be adjusted manually if desired. Otherwise, the DCO frequency is stabilized by the FLL operation.

After a PUC, DCORSEL = {2} and DCO = {0}. MCLK and SMCLK are sourced from DCOCLKDIV. Because the CPU executes code from MCLK, which is sourced from the fast-starting DCO, code execution begins from PUC in less than 5 µs.

The frequency of DCOCLK is set by the following functions:

- The three DCORSEL bits select one of eight nominal frequency ranges for the DCO. These ranges are defined for an individual device in the device-specific data sheet.
- The five DCO bits divide the DCO range selected by the DCORSEL bits into 32 frequency steps, separated by approximately 8%.
- The five MOD bits switch between the frequency selected by the DCO bits and the next-higher frequency set by {DCO + 1}. When DCO = {31}, the MOD bits have no effect, because the DCO is already at the highest setting for the selected DCORSEL range.

3.2.7 Frequency Locked Loop (FLL)

The FLL continuously counts up or down a frequency integrator. The output of the frequency integrator that drives the DCO can be read in UCSCTL0, UCSCTL1 (bits MOD and DCO). The count is adjusted +1 with the frequency $f_{FLLREFCLK}/n$ (n = 1, 2, 4, 8, 12, or 16) or –1 with the frequency $f_{DCOCLK}/[D \times (N+1)]$.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: Reading MOD and DCO bits

The integrator is updated via the DCOCLK, which may differ in frequency of operation of MCLK. It is possible that immediate reads of a previously written value are not visible to the user since the update to the integrator has not occurred. This is normal. Once the integrator is updated at the next successive DCOCLK, the correct value can be read.

In addition, since the MCLK can be asynchronous to the integrator updates, reading the values may be cause a corrupted value to be read under this condition. In this case, a majority vote method should be performed.

Five of the integrator bits (UCSCTL0 bits 12 to 8) set the DCO frequency tap. Thirty-two taps are implemented for the DCO, and each is approximately 8% higher than the previous. The modulator mixes two adjacent DCO frequencies to produce fractional taps.

For a given DCO bias range setting, time must be allowed for the DCO to settle on the proper tap for normal operation. (n × 32) $f_{FLLREFCLK}$ cycles are required between taps requiring a worst case of (n × 32 × 32) $f_{FLLREFCLK}$ cycles for the DCO to settle. The value n is defined by the FLLREFDIV bits (n = 1, 2, 4, 8, 12, or 16).

3.2.8 DCO Modulator

The modulator mixes two DCO frequencies, f_{DCO} and f_{DCO}+1 to produce an intermediate effective frequency between f_{DCO} and f_{DCO}+1 and spread the clock energy, reducing electromagnetic interference (EMI). The modulator mixes f_{DCO} and f_{DCO}+1 for 32 DCOCLK clock cycles and is configured with the MOD bits. When MOD = {0}, the modulator is off.

The modulator mixing formula is:

$t = (32 – MOD) \times t_{DCO} + MOD \times t_{DCO+1}$

Figure 3-2 shows the modulator operation.

When FLL operation is enabled, the modulator settings and DCO are controlled by the FLL hardware. If FLL operation is not desired, the modulator settings and DCO control can be configured with software.

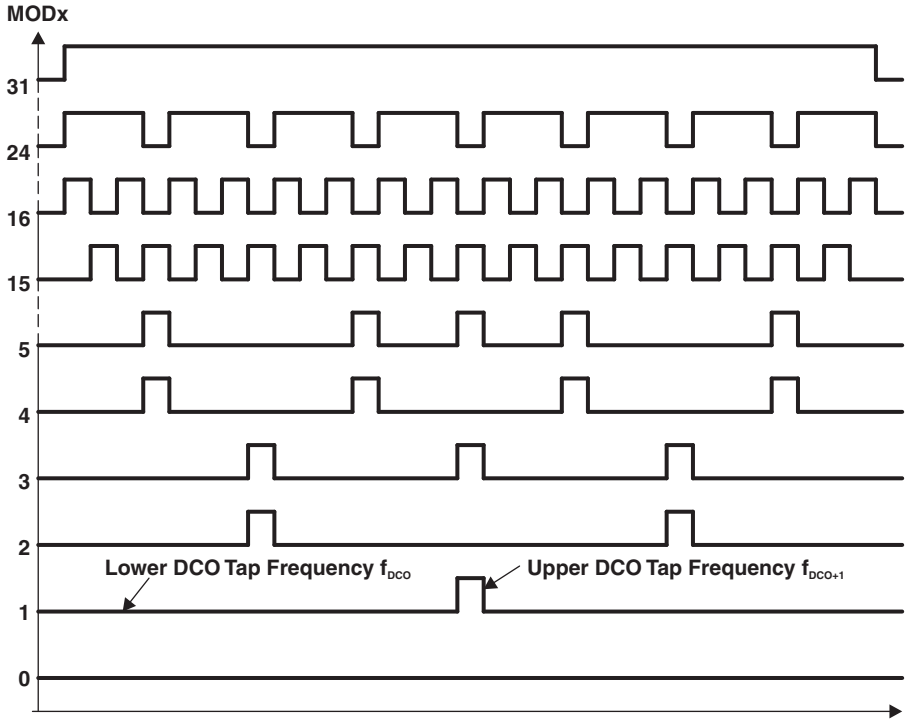


Figure 3-2. Modulator Patterns

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.2.9 Disabling FLL Hardware and Modulator

The FLL is disabled when the status register bits SCG0 or SCG1 are set. When the FLL is disabled, the DCO runs at the previously selected tap and DCOCLK is not automatically stabilized.

The DCO modulator is disabled when DISMOD is set. When the DCO modulator is disabled, the DCOCLK is adjusted to the DCO tap selected by the DCO bits.

NOTE: DCO operation without FLL

When the FLL operation is disabled, the DCO continues to operate at the current settings. Because it is not stabilized by the FLL, temperature and voltage variations influence the frequency of operation. See the device-specific data sheet for voltage and temperature coefficients to ensure reliable operation.

3.2.10 FLL Operation From Low-Power Modes

An interrupt service request clears SCG1, CPUOFF, and OSCOFF if set, but does not clear SCG0. This means that for FLL operation from within an interrupt service routine entered from LPM1, 3, or 4, the FLL remains disabled and the DCO operates at the previous setting as defined in UCSCTL0 and UCSCTL1. SCG0 can be cleared by user software if FLL operation is required.

3.2.11 Operation From Low-Power Modes, Requested by Peripheral Modules

A peripheral module requests its clock sources automatically from the UCS module if required for its proper operation, regardless of the current mode of operation, as shown in Figure 3-3.

A peripheral module asserts one of three possible clock request signals based on its control bits: ACLK_REQ, MCLK_REQ, or SMCLK_REQ. These request signals are based on the configuration and clock selection of the respective module. For example, if a timer selects ACLK as its clock source and the timer is enabled, the timer generates an ACLK_REQ signal to the UCS system. The UCS, in turn, enables ACLK regardless of the LPM settings.

Any clock request from a peripheral module causes its respective clock off signal to be overridden, but does not change the setting of clock off control bit. For example, a peripheral module may require ACLK that is currently disabled by the OSCOFF bit (OSCOFF = 1). The module can request ACLK by generating an ACLK_REQ. This causes the OSCOFF bit to have no effect, thereby allowing ACLK to be available to the requesting peripheral module. The OSCOFF bit remains at its current setting (OSCOFF = 1).

If the requested source is not active, the software NMI handler must take care of the required actions. For the previous example, if ACLK was sourced by XT1 and XT1 was not enabled, an oscillator fault condition occurs and the software must handle the event. The watchdog, due to its security requirement, actively selects the VLOCLK source if the originally selected clock source is not available.

Due to the clock request feature, care must be taken in the application when entering low-power modes to save power. Although the device enters the selected low-power mode, a clock request may exhibit more current consumption than the specified values in the data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

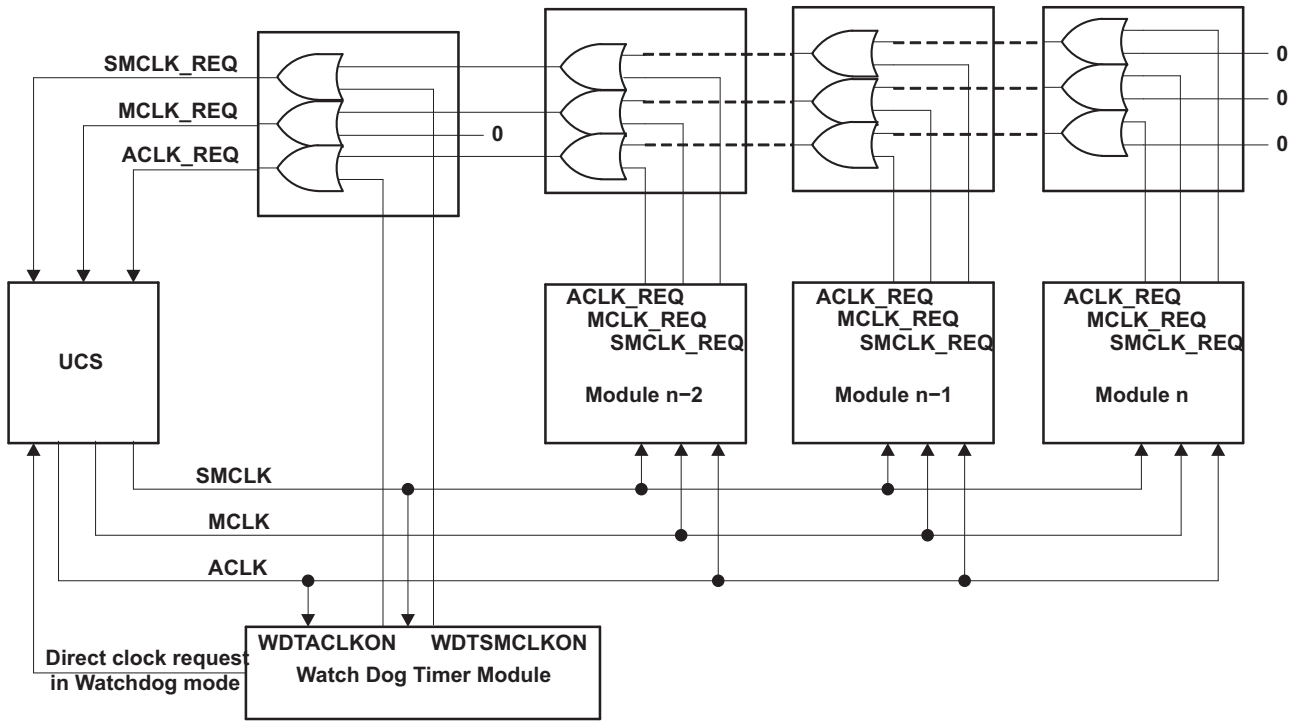


Figure 3-3. Module Request Clock System

By default, the clock request logic is enabled. The clock request logic can be disabled by clearing ACLKREQEN, MCLKREQEN, or SMCLKREQEN, for each respective system clock. When ACLKREQEN or MCLKREQEN bits are set, or active, the clock is available to the system and prevents entry into a low-power mode until all modules requesting the clock are disabled. When ACLKREQEN or MCLKREQEN bits are cleared, or disabled, the clock is always halted as defined by the low-power modes. The SMCLKREQEN logic behaves similarly, but is also influenced by the SMCLKOFF bit in the UCSCTL6 register.Table 3-1 shows the relationship between the system clocks and the low-power modes in conjunction with the clock request logic.

Table 3-1. Clock Request System and Power Modes

Mode	ACLK		MCLK		SMCLK			
					SMCLKOFF = 0		SMCLKOFF = 1	
	ACLKREQEN = 0	ACLKREQEN = 1	MCLKREQEN = 0	MCLKREQEN = 1	SMCLKREQEN = 0	SMCLKREQEN = 1	SMCLKREQEN = 0	SMCLKREQEN = 1
AM	Active	Active	Active	Active	Active	Active	Disabled	Active
LPM0	Active	Active	Disabled	Active	Active	Active	Disabled	Active
LPM1	Active	Active	Disabled	Active	Active	Active	Disabled	Active
LPM2	Active	Active	Disabled	Active	Disabled	Active	Disabled	Active
LPM3	Active	Active	Disabled	Active	Disabled	Active	Disabled	Active
LPM4	Disabled	Active	Disabled	Active	Disabled	Active	Disabled	Active
LPM3.5 (1)	Disabled (2)	Disabled (3)	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
LPM4.5 (1)	Disabled (2)	Disabled (3)	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled

(1) Any clock request prior to entry into LPM3.5 or LPM4.5 is ignored and LPM3.5 or LPM4.5 entry occurs. For the special case when XT1OFF = 0 or XT2OFF = 0, the LPMx.5 request is ignored and the device does not enter LPMx.5.

(2) Watchdog mode requesting ACLK prevents LPMx.5 entry.

(3) Any module requesting ACLK prevents LPMx.5 entry.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.2.11.1 LPM3.5 and LPM4.5 Clock Request Handling

While ACLK is requested by at least one module (ACLKREQEN = 1), LPMx.5 cannot be entered. After clearing the ACLK request enable signal (ACLKREQEN = 0), the device is able to enter LPMx.5 (except for watchdog mode–see following note). See Entering and Exiting Low-Power Modes LPMx.5 for details on how to enter LPMx.5.

NOTE: Watchdog Mode Using ACLK

When a device is configured to use watchdog mode (WDTTMSEL = 0) and is sourced by ACLK (WDTSSEL = 01b), the device cannot enter LPMx.5 regardless of whether or not the ACLK request enable signal (ACLKREQEN) is cleared.

3.2.12 UCS Module Fail-Safe Operation

The UCS module incorporates an oscillator-fault fail-safe feature. This feature detects an oscillator fault for XT1, DCO, and XT2 as shown in Figure 3-4. The available fault conditions are:

- Low-frequency oscillator fault (XT1LFOFFG) for XT1 in LF mode
- High-frequency oscillator fault (XT1HFOFFG) for XT1 in HF mode
- High-frequency oscillator fault (XT2OFFG) for XT2
- DCO fault flag (DCOFFG) for the DCO

The crystal oscillator fault bits XT1LFOFFG, XT1HFOFFG, and XT2OFFG are set if the corresponding crystal oscillator is turned on and not operating properly. Once set, the fault bits remain set until reset in software, regardless if the fault condition no longer exists. If the user clears the fault bits and the fault condition still exists, the fault bits are automatically set, otherwise they remain cleared.

When using XT1 operation in LF mode as the reference source into the FLL (SELREF = {0}), a crystal fault automatically causes the FLL reference source, FLLREFCLK, to be sourced by the REFO. XT1LFOFFG is set. When using XT1 operation in HF mode as the reference source into the FLL, a crystal fault causes no FLLREFCLK signal to be generated and the FLL continues to count down to zero in an attempt to lock FLLREFCLK and DCOCLK/[D × (N + 1)]. The DCO tap moves to the lowest position (DCO are cleared) and the DCOFFG is set. DCOFFG is also set if the N-multiplier value is set too high for the selected DCO frequency range, resulting in the DCO tap moving to the highest position (UCSCTL0.12 to UCSCTL0.8 are set). The DCOFFG remains set until cleared by the user. If the user clears the DCOFFG and the fault condition remains, it is automatically set, otherwise it remains cleared. XT1HFOFFG is set.

When using XT2 as the reference source into the FLL, a crystal fault causes no FLLREFCLK signal to be generated, and the FLL continues to count down to zero in an attempt to lock FLLREFCLK and DCOCLK/[D × (N + 1)]. The DCO tap moves to the lowest position (DCO are cleared) and the DCOFFG is set. DCOFFG is also set if the N-multiplier value is set too high for the selected DCO frequency range, resulting in the DCO tap moving to the highest position (UCSCTL0.12 to UCSCTL0.8 are set). The DCOFFG remains set until cleared by the user. If the user clears the DCOFFG and the fault condition remains, it is automatically set, otherwise it remains cleared. XT2OFFG is set.

The OFIFG oscillator-fault interrupt flag is set and latched at POR or when any oscillator fault (XT1LFOFFG, XT1HFOFFG, XT2OFFG, or DCOFFG) is detected. When OFIFG is set and OFIE is set, the OFIFG requests an NMI. When the interrupt is granted, the OFIE is not reset automatically as it is in previous MSP430 families. It is no longer required to reset the OFIE. NMI entry and exit circuitry removes this requirement. The OFIFG flag must be cleared by software. The source of the fault can be identified by checking the individual fault bits.

If a fault is detected for the oscillator sourcing MCLK, MCLK is automatically switched to the DCO for its clock source (DCOCLKDIV) for all clock sources except XT1 LF mode. If MCLK is sourced from XT1 in LF mode, an oscillator fault causes MCLK to be automatically switched to the REFO for its clock source (REFOCLK). This does not change the SELM bit settings. This condition must be handled by user software.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

If a fault is detected for the oscillator sourcing SMCLK, SMCLK is automatically switched to the DCO for its clock source (DCOCLKDIV) for all clock sources except XT1 LF mode. If SMCLK is sourced from XT1 in LF mode, an oscillator fault causes SMCLK to be automatically switched to the REFO for its clock source (REFOCLK). This does not change the SELS bit settings. This condition must be handled by user software.

If a fault is detected for the oscillator sourcing ACLK, ACLK is automatically switched to the DCO for its clock source (DCOCLKDIV) for all clock sources except XT1 LF mode. If ACLK is sourced from XT1 in LF mode, an oscillator fault causes ACLK to be automatically switched to the REFO for its clock source (REFOCLK). This does not change the SELA bit settings. This condition must be handled by user software.

NOTE: DCO active during oscillator fault

DCOCLKDIV is active even at the lowest DCO tap. The clock signal is available for the CPU to execute code and service an NMI during an oscillator fault.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

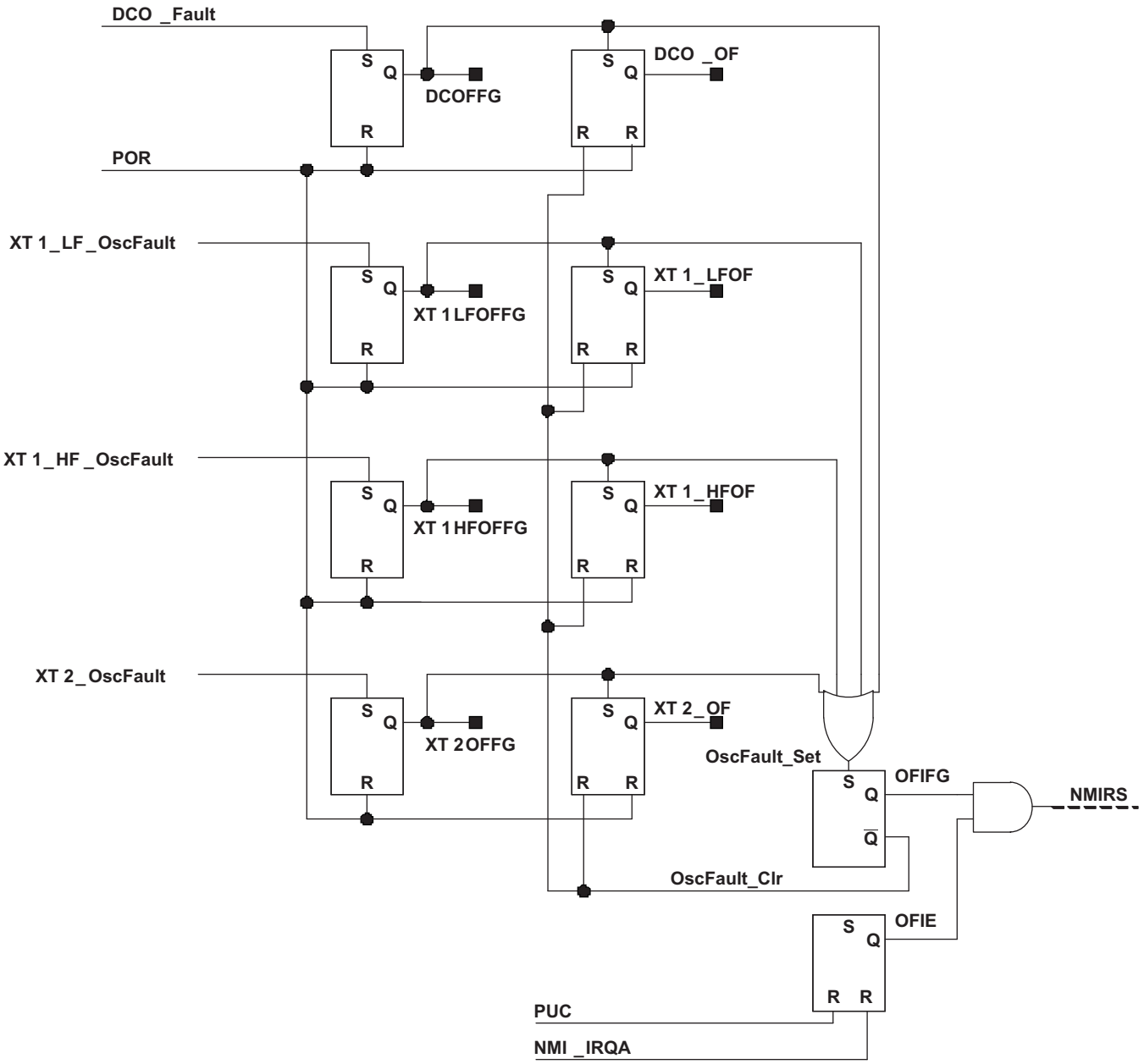


Figure 3-4. Oscillator Fault Logic

Copyright © 2009–2013, Texas Instruments Incorporated

NOTE: Fault conditions

DCO_Fault: DCOFFG is set if DCO bits in UCSCTL0 register value equals {0} or {31}.

XT1_LF_OscFault: This signal is set after the XT1 (LF mode) oscillator has stopped operation and cleared after operation resumes. The fault condition causes XT1LFOFFG to be set and remain set. If the user clears XT1LFOFFG and the fault condition still exists, XT1LFOFFG remains set.

XT1_HF_OscFault: This signal is set after the XT1 (HF mode) oscillator has stopped operation and cleared after operation resumes. The fault condition causes XT1HFOFFG to be set and remain set. If the user clears XT1HFOFFG and the fault condition still exists, XT1HFOFFG remains set.

XT2_OscFault: This signal is set after the XT2 oscillator has stopped operation and cleared after operation resumes. The fault condition causes XT2OFFG to be set and remain set. If the user clears XT2OFFG and the fault condition still exists, XT2OFFG remains set.

NOTE: Fault logic

Please note that as long as a fault condition still exists, the OFIFG remains set. The application must take special care when clearing the OFIFG signal. If no fault condition remains when the OFIFG signal is cleared, the clock logic switches back to the original user settings prior to the fault condition.

NOTE: Fault logic counters

Each crystal oscillator circuit has hardware counters. These counters are reset each time a fault condition occurs on its respective oscillator, causing the fault flag to be set. The counters begin to count after the fault condition is removed. Once the maximum count is reached, the fault flag is removed.

In XT1 LF mode, the maximum count is 8192. In XT1 HF mode (and XT2 when available), the maximum count is 1024. In bypass modes, regardless of LF or HF settings, the maximum count is 8192.

3.2.13 Synchronization of Clock Signals

When switching MCLK or SMCLK from one clock source to the another, the switch is synchronized to avoid critical race conditions as shown in Figure 3-5:

- The current clock cycle continues until the next rising edge.
- The clock remains high until the next rising edge of the new clock.
- The new clock source is selected and continues with a full high period.

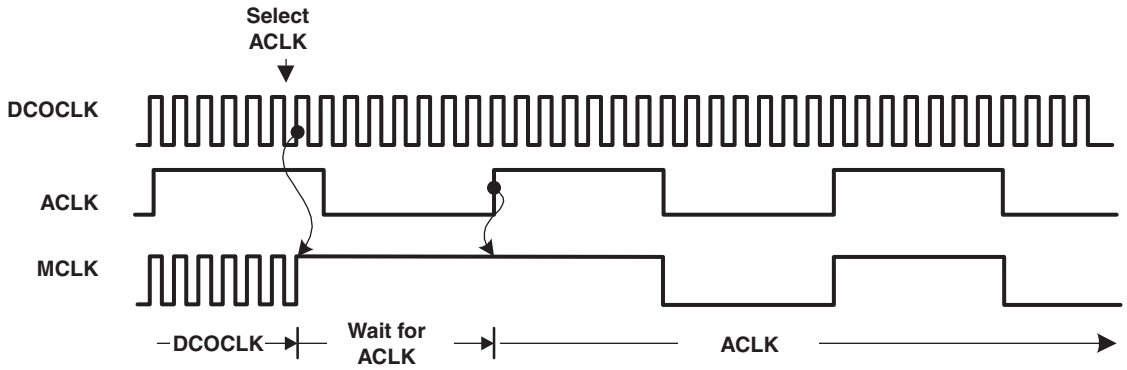


Figure 3-5. Switch MCLK from DCOCLK to XT1CLK

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.3 Module Oscillator (MODOSC)

The UCS module also supports an internal oscillator, MODOSC, that is used by the flash memory controller module and, optionally, by other modules in the system. The MODOSC sources MODCLK.

3.3.1 MODOSC Operation

To conserve power, MODOSC is powered down when not needed and enabled only when required. When the MODOSC source is required, the respective module requests it. MODOSC is enabled based on unconditional and conditional requests. Setting MODOSCREQEN enables conditional requests. Unconditional requests are always enabled. It is not necessary to set MODOSCREQEN for modules that use unconditional requests; for example, flash controller or ADC12_A.

The flash memory controller only requires MODCLK when performing write or erase operations. When performing such operations, the flash memory controller issues an unconditional request for the MODOSC source. Upon doing so, the MODOSC source is enabled, if not already enabled from other modules' previous requests.

The ADC12_A may optionally use MODOSC as a clock source for its conversion clock. The user chooses the ADC12OSC as the conversion clock source. During a conversion, the ADC12_A module issues an unconditional request for the ADC12OSC clock source. Upon doing so, the MODOSC source is enabled, if not already enabled from other modules' previous requests.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4 UCS Module Registers

The UCS module registers are listed in Table 3-2. The base address can be found in the device-specific data sheet. The address offset is listed in Table 3-2.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 3-2. UCS Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCSCTL0	Unified Clock System Control 0	Read/write	Word	0000h	Section 3.4.1
00h	UCSCTL0_L		Read/write	Byte	00h	
01h	UCSCTL0_H		Read/write	Byte	00h	
02h	UCSCTL1	Unified Clock System Control 1	Read/write	Word	0020h	Section 3.4.2
02h	UCSCTL1_L		Read/write	Byte	20h	
03h	UCSCTL1_H		Read/write	Byte	00h	
04h	UCSCTL2	Unified Clock System Control 2	Read/write	Word	101Fh	Section 3.4.3
04h	UCSCTL2_L		Read/write	Byte	1Fh	
05h	UCSCTL2_H		Read/write	Byte	10h	
06h	UCSCTL3	Unified Clock System Control 3	Read/write	Word	0000h	Section 3.4.4
06h	UCSCTL3_L		Read/write	Byte	00h	
07h	UCSCTL3_H		Read/write	Byte	00h	
08h	UCSCTL4	Unified Clock System Control 4	Read/write	Word	0044h	Section 3.4.5
08h	UCSCTL4_L		Read/write	Byte	44h	
09h	UCSCTL4_H		Read/write	Byte	00h	
0Ah	UCSCTL5	Unified Clock System Control 5	Read/write	Word	0000h	Section 3.4.6
0Ah	UCSCTL5_L		Read/write	Byte	00h	
0Bh	UCSCTL5_H		Read/write	Byte	00h	
0Ch	UCSCTL6	Unified Clock System Control 6	Read/write	Word	C1CDh	Section 3.4.7
0Ch	UCSCTL6_L		Read/write	Byte	CDh	
0Dh	UCSCTL6_H		Read/write	Byte	C1h	
0Eh	UCSCTL7	Unified Clock System Control 7	Read/write	Word	0703h	Section 3.4.8
0Eh	UCSCTL7_L		Read/write	Byte	03h	
0Fh	UCSCTL7_H		Read/write	Byte	07h	
10h	UCSCTL8	Unified Clock System Control 8	Read/write	Word	0707h	Section 3.4.9
10h	UCSCTL8_L		Read/write	Byte	07h	
11h	UCSCTL8_H		Read/write	Byte	07h	
12h	UCSCTL9	Unified Clock System Control 9[(1)]	Read/write	Word	0000h	Section 3.4.10
12h	UCSCTL9_L		Read/write	Byte	00h	
13h	UCSCTL9_H		Read/write	Byte	00h	

(1) This register is not available on all devices. See the device-specific data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.1 UCSCTL0 Register

Unified Clock System Control 0 Register

Figure 3-6. UCSCTL0 Register

15	14	13	12	11	10	9	8
Reserved			DCO				
r0	r0	r0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
MOD					Reserved		
rw-0	rw-0	rw-0	rw-0	rw-0	r0	r0	r0

Table 3-3. UCSCTL0 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-8	DCO	RW	0h	DCO tap selection. These bits select the DCO tap and are modified automatically during FLL operation.
7-3	MOD	RW	0h	Modulation bit counter. These bits select the modulation pattern. All MOD bits are modified automatically during FLL operation. The DCO register value is incremented when the modulation bit counter rolls over from 31 to 0. If the modulation bit counter decrements from 0 to the maximum count, the DCO register value is also decremented.
2-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.2 UCSCTL1 Register

Unified Clock System Control 1 Register

Figure 3-7. UCSCTL1 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved	DCORSEL			Reserved		Reserved	DISMOD
r0	rw-0	rw-1	rw-0	r0	r0	rw-0	rw-0

Table 3-4. UCSCTL1 Register Description

Bit	Field	Type	Reset	Description
15-7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	DCORSEL	RW	2h	DCO frequency range select. These bits select the DCO frequency range of operation defined in the device-specific datasheet.
3-2	Reserved	R	0h	Reserved. Always reads as 0.
1	Reserved	RW	0h	Reserved. Always reads as 0.
0	DISMOD	RW	0h	Modulation. This bit enables or disables the modulation. 0b = Modulation enabled 1b = Modulation disabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.3 UCSCTL2 Register

Unified Clock System Control 2 Register

Figure 3-8. UCSCTL2 Register

15	14	13	12	11	10	9	8
Reserved	FLLD			Reserved		FLLN	
r0	rw-0	rw-0	rw-1	r0	r0	rw-0	rw-0
7	6	5	4	3	2	1	0
FLLN							
rw-0	rw-0	rw-0	rw-1	rw-1	rw-1	rw-1	rw-1

Table 3-5. UCSCTL2 Register Description

Bit	Field	Type	Reset	Description
15	Reserved	R	0h	Reserved. Always reads as 0.
14-12	FLLD	RW	1h	FLL loop divider. These bits divide f(DCOCLK) in the FLL feedback loop. This results in an additional multiplier for the multiplier bits. See also multiplier bits. 000b = f(DCOCLK)/1 001b = f(DCOCLK)/2 010b = f(DCOCLK)/4 011b = f(DCOCLK)/8 100b = f(DCOCLK)/16 101b = f(DCOCLK)/32 110b = Reserved for future use. Defaults to f(DCOCLK)/32. 111b = Reserved for future use. Defaults to f(DCOCLK)/32.
11-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	FLLN	RW	1Fh	Multiplier bits. These bits set the multiplier value N of the DCO. N must be greater than 0. Writing zero to FLLN causes N to be set to 1.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.4 UCSCTL3 Register

Unified Clock System Control 3 Register

Figure 3-9. UCSCTL3 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved	SELREF			Reserved	FLLREFDIV		
r0	rw-0	rw-0	rw-0	r0	rw-0	rw-0	rw-0

Table 3-6. UCSCTL3 Register Description

Bit	Field	Type	Reset	Description
15-7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	SELREF	RW	0h	FLL reference select. These bits select the FLL reference clock source. 000b = XT1CLK 001b = Reserved for future use. Defaults to XT1CLK. 010b = REFOCLK 011b = Reserved for future use. Defaults to REFOCLK. 100b = Reserved for future use. Defaults to REFOCLK. 101b = XT2CLK when available, otherwise REFOCLK. 110b = Reserved for future use. XT2CLK when available, otherwise REFOCLK. 111b =
3	Reserved	R	0h	Reserved. Always reads as 0.
2-0	FLLREFDIV	RW	0h	FLL reference divider. These bits define the divide factor for f(FLLREFCLK). The divided frequency is used as the FLL reference frequency. 000b = f(FLLREFCLK)/1 001b = f(FLLREFCLK)/2 010b = f(FLLREFCLK)/4 011b = f(FLLREFCLK)/8 100b = f(FLLREFCLK)/12 101b = f(FLLREFCLK)/16 110b = Reserved for future use. Defaults to f(FLLREFCLK)/16. 111b = Reserved for future use. Defaults to f(FLLREFCLK)/16.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.5 UCSCTL4 Register

Unified Clock System Control 4 Register

Figure 3-10. UCSCTL4 Register

15	14	13	12	11	10	9	8
Reserved					SELA		
r0	r0	r0	r0	r0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
Reserved	SELS			Reserved	SELM		
r0	rw-1	rw-0	rw-0	r0	rw-1	rw-0	rw-0

Table 3-7. UCSCTL4 Register Description

Bit	Field	Type	Reset	Description
15-11	Reserved	R	0h	Reserved. Always reads as 0.
10-8	SELA	RW	0h	Selects the ACLK source 000b = XT1CLK 001b = VLOCLK 010b = REFOCLK 011b = DCOCLK 100b = DCOCLKDIV 101b = XT2CLK when available, otherwise DCOCLKDIV 110b = Reserved for future use. Defaults to XT2CLK. 111b = Reserved for future use. Defaults to XT2CLK.
7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	SELS	RW	4h	Selects the SMCLK source 000b = XT1CLK 001b = VLOCLK 010b = REFOCLK 011b = DCOCLK 100b = DCOCLKDIV 101b = XT2CLK when available, otherwise DCOCLKDIV 110b = Reserved for future use. Defaults to XT2CLK. 111b = Reserved for future use. Defaults to XT2CLK.
3	Reserved	R	0h	Reserved. Always reads as 0.
2-0	SELM	RW	4h	Selects the MCLK source 000b = XT1CLK 001b = VLOCLK 010b = REFOCLK 011b = DCOCLK 100b = DCOCLKDIV 101b = XT2CLK when available, otherwise DCOCLKDIV 110b = Reserved for future use. Defaults to XT2CLK. 111b = Reserved for future use. Defaults to XT2CLK.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.6 UCSCTL5 Register

Unified Clock System Control 5 Register

Figure 3-11. UCSCTL5 Register

15	14	13	12	11	10	9	8
Reserved	DIVPA			Reserved	DIVA		
r0	rw-0	rw-0	rw-0	r0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
Reserved	DIVS			Reserved	DIVM		
r0	rw-0	rw-0	rw-0	r0	rw-0	rw-0	rw-0

Table 3-8. UCSCTL5 Register Description

Bit	Field	Type	Reset	Description
15	Reserved	R	0h	Reserved. Always reads as 0.
14-12	DIVPA	RW	0h	ACLK source divider available at external pin. Divides the frequency of ACLK and presents it to an external pin. 000b = f(ACLK)/1 001b = f(ACLK)/2 010b = f(ACLK)/4 011b = f(ACLK)/8 100b = f(ACLK)/16 101b = f(ACLK)/32 110b = Reserved for future use. Defaults to f(ACLK)/32. 111b = Reserved for future use. Defaults to f(ACLK)/32.
11	Reserved	R	0h	Reserved. Always reads as 0.
10-8	DIVA	RW	0h	ACLK source divider. Divides the frequency of the ACLK clock source. 000b = f(ACLK)/1 001b = f(ACLK)/2 010b = f(ACLK)/4 011b = f(ACLK)/8 100b = f(ACLK)/16 101b = f(ACLK)/32 110b = Reserved for future use. Defaults to f(ACLK)/32. 111b = Reserved for future use. Defaults to f(ACLK)/32.
7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	DIVS	RW	0h	SMCLK source divider 000b = f(SMCLK)/1 001b = f(SMCLK)/2 010b = f(SMCLK)/4 011b = f(SMCLK)/8 100b = f(SMCLK)/16 101b = f(SMCLK)/32 110b = Reserved for future use. Defaults to f(SMCLK)/32. 111b = Reserved for future use. Defaults to f(SMCLK)/32.
3	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 3-8. UCSCTL5 Register Description (continued)

Bit	Field	Type	Reset	Description
2-0	DIVM	RW	0h	MCLK source divider 000b = f(MCLK)/1 001b = f(MCLK)/2 010b = f(MCLK)/4 011b = f(MCLK)/8 100b = f(MCLK)/16 101b = f(MCLK)/32 110b = Reserved for future use. Defaults to f(MCLK)/32. 111b = Reserved for future use. Defaults to f(MCLK)/32.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.7 UCSCTL6 Register

Unified Clock System Control 6 Register

Figure 3-12. UCSCTL6 Register

15	14	13	12	11	10	9	8
Reserved							XT2OFF
r0	r0	r0	r0	r0	r0	r0	rw-1
7	6	5	4	3	2	1	0
XT1DRIVE(1)		XTS	XT1BYPASS	XCAP(1)		SMCLKOFF	XT1OFF
rw-1	rw-1	rw-0	rw-0	rw-1	rw-1	rw-0	rw-1

(1) The configuration of these bits is retained during LPM3.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPM3.5 before clearing LOCKLPM5 is required.

Table 3-9. UCSCTL6 Register Description

Bit	Field	Type	Reset	Description
15-9	Reserved	R	0h	Reserved. Always reads as 0.
8	XT2OFF	RW	1h	Turns off the XT2 oscillator 0b = XT2 is on 1b = XT2 is off if it is not used by the radio; that is, if the radio is not in sleep state
7-6	XT1DRIVE	RW	3h	The XT1 oscillator current can be adjusted to its drive needs. Initially, it starts with the highest supply current for reliable and quick startup. If needed, user software can reduce the drive strength. 00b = Lowest current consumption for XT1 LF mode. XT1 oscillator operating range in HF mode is 4 MHz to 8 MHz. 01b = Increased drive strength for XT1 LF mode. XT1 oscillator operating range in HF mode is 8 MHz to 16 MHz. 10b = Increased drive capability for XT1 LF mode. XT1 oscillator operating range in HF mode is 16 MHz to 24 MHz. 11b = Maximum drive capability and maximum current consumption for XT1 LF mode. XT1 oscillator operating range in HF mode is 24 MHz to 32 MHz.
5	XTS	RW	0h	XT1 mode select 0b = Low-frequency mode. XCAP bits define the capacitance at the XIN and XOUT pins. 1b = High-frequency mode. XCAP bits are not used.
4	XT1BYPASS	RW	0h	XT1 bypass select 0b = XT1 sourced internally 1b = XT1 sourced externally from pin
3-2	XCAP	RW	3h	Oscillator capacitor selection. These bits select the capacitors applied to the LF crystal or resonator in the LF mode (XTS = 0). The effective capacitance (seen by the crystal) is C(eff) ≈ (C(XIN) + 2 pF) / 2. It is assumed that C(XIN) = C(XOUT) and that a parasitic capacitance of 2 pF is added by the package and the printed circuit board. For details about the typical internal and the effective capacitors, see the device-specific data sheet.
1	SMCLKOFF	RW	0h	SMCLK off. This bit turns off the SMCLK. 0b = SMCLK on 1b = SMCLK off
0	XT1OFF	RW	1h	XT1 off. This bit turns off the XT1. 0b = XT1 is on if XT1 is selected via the port selection and XT1 is not in bypass mode of operation. 1b = XT1 is off if it is not used as a source for ACLK, MCLK, or SMCLK or is not used as a reference source required for FLL operation.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.8 UCSCTL7 Register

Unified Clock System Control 7 Register

Figure 3-13. UCSCTL7 Register

15	14	13	12	11	10	9	8
Reserved		Reserved		Reserved		Reserved	
r0	r0	rw-0	rw-(0)	rw-(1)	rw-(1)	r-1	r-1
7	6	5	4	3	2	1	0
Reserved			Reserved	XT2OFFG(1)	XT1HFOFFG(1)	XT1LFOFFG	DCOFFG
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(1)	rw-(1)

(1) Not available on all devices. When not available, this bit is reserved.

Table 3-10. UCSCTL7 Register Description

Bit	Field	Type	Reset	Description
15-14	Reserved	R	0h	Reserved. Always reads as 0.
13-12	Reserved	RW	0h	Reserved. Must always be written with 0.
11-10	Reserved	RW	3h	Reserved. The states of these bits should be ignored.
9-8	Reserved	R	3h	Reserved. The states of these bits should be ignored.
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4	Reserved	RW	0h	Reserved. The state of this bit should be ignored.
3	XT2OFFG(1)	RW	0h	XT2 oscillator fault flag. If this bit is set, the OFIFG flag is also set. XT2OFFG is set if a XT2 fault condition exists. XT2OFFG can be cleared via software. If the XT2 fault condition still remains, XT2OFFG is set. 0b = No fault condition occurred after the last reset. 1b = XT2 fault. An XT2 fault occurred after the last reset.
2	XT1HFOFFG(1)	RW	0h	XT1 oscillator fault flag (HF mode). If this bit is set, the OFIFG flag is also set. XT1HFOFFG is set if a XT1 fault condition exists. XT1HFOFFG can be cleared via software. If the XT1 fault condition still remains, XT1HFOFFG is set. 0b = No fault condition occurred after the last reset. 1b = XT1 fault. An XT1 fault occurred after the last reset.
1	XT1LFOFFG	RW	1h	XT1 oscillator fault flag (LF mode). If this bit is set, the OFIFG flag is also set. XT1LFOFFG is set if a XT1 fault condition exists. XT1LFOFFG can be cleared via software. If the XT1 fault condition still remains, XT1LFOFFG is set. 0b = No fault condition occurred after the last reset. 1b = XT1 fault (LF mode). A XT1 fault occurred after the last reset.
0	DCOFFG	RW	1h	DCO fault flag. If this bit is set, the OFIFG flag is also set. The DCOFFG bit is set if DCO = {0} or DCO = {31}. DCOFFG can be cleared via software. If the DCO fault condition still remains, DCOFFG is set. 0b = No fault condition occurred after the last reset. 1b = DCO fault. A DCO fault occurred after the last reset.

(1) Not available on all devices. When not available, this bit is reserved.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

3.4.9 UCSCTL8 Register

Unified Clock System Control 8 Register

Figure 3-14. UCSCTL8 Register

15	14	13	12	11	10	9	8
Reserved					Reserved		
r0	r0	r0	r0	r0	rw-(1)	rw-(1)	rw-(1)
7	6	5	4	3	2	1	0
Reserved			Reserved	MODOSCREQEN	SMCLKREQEN	MCLKREQEN	ACLKREQEN
r0	r0	r0	rw-(0)	rw-(0)	rw-(1)	rw-(1)	rw-(1)

Table 3-11. UCSCTL8 Register Description

Bit	Field	Type	Reset	Description
15-11	Reserved	R	0h	Reserved. Always reads as 0.
10-8	Reserved	R	0h	Reserved. Must always be written as 1.
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4	Reserved	R	0h	Reserved. Must always be written as 0.
3	MODOSCREQEN	RW	0h	MODOSC clock request enable. Setting this enables conditional module requests for MODOSC. 0b = MODOSC conditional requests are disabled. 1b = MODOSC conditional requests are enabled.
2	SMCLKREQEN	RW	1h	SMCLK clock request enable. Setting this enables conditional module requests for SMCLK 0b = SMCLK conditional requests are disabled. 1b = SMCLK conditional requests are enabled.
1	MCLKREQEN	RW	1h	MCLK clock request enable. Setting this enables conditional module requests for MCLK 0b = MCLK conditional requests are disabled. 1b = MCLK conditional requests are enabled.
0	ACLKREQEN	RW	1h	ACLK clock request enable. Setting this enables conditional module requests for ACLK 0b = ACLK conditional requests are disabled. 1b = ACLK conditional requests are enabled.

Copyright © 2009–2013, Texas Instruments Incorporated

3.4.10 UCSCTL9 Register

Unified Clock System Control 9 Register

This register is not available on all devices. See the device-specific data sheet.

Figure 3-15. UCSCTL9 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved						XT2BYPASSLV	XT1BYPASSLV
r0	r0	r0	r0	r0	r0	rw-0	rw-0

Table 3-12. UCSCTL9 Register Description

Bit	Field	Type	Reset	Description
15-2	Reserved	R	0h	Reserved. Always reads as 0.
1	XT2BYPASSLV	RW	0h	Selects XT2 bypass input swing level. Must be set for reduced swing operation. 0b = Input range from 0 to DVCC 1b = Input range from 0 to DVIO
0	XT1BYPASSLV	RW	0h	Selects XT1 bypass input swing level. Must be set for reduced swing operation. 0b = Input range from 0 to DVCC 1b = Input range from 0 to DVIO

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

CPUX

This chapter describes the extended MSP430X 16-bit RISC CPU (CPUX) with 1MB memory access, its addressing modes, and instruction set.

NOTE: The MSP430X CPU implemented on these devices has, in some cases, slightly different cycle counts from the MSP430X CPU implemented on the 2xx and 4xx families.

Topic **Page**

4.1 MSP430X CPU (CPUX) Introduction 133
4.2 Interrupts 135
4.3 CPU Registers 136
4.4 Addressing Modes 142
4.5 MSP430 and MSP430X Instructions 159
4.6 Instruction Set Description 175

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.1 MSP430X CPU (CPUX) Introduction

The MSP430X CPU incorporates features specifically designed for modern programming techniques, such as calculated branching, table processing, and the use of high-level languages such as C. The MSP430X CPU can address a 1MB address range without paging. The MSP430X CPU is completely backward compatible with the MSP430 CPU.

The MSP430X CPU features include:

- RISC architecture
- Orthogonal architecture
- Full register access including program counter (PC), status register (SR), and stack pointer (SP)
- Single-cycle register operations
- Large register file reduces fetches to memory.
- 20-bit address bus allows direct access and branching throughout the entire memory range without paging.
- 16-bit data bus allows direct manipulation of word-wide arguments.
- Constant generator provides the six most often used immediate values and reduces code size.
- Direct memory-to-memory transfers without intermediate register holding
- Byte, word, and 20-bit address-word addressing

The block diagram of the MSP430X CPU is shown in Figure 4-1.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

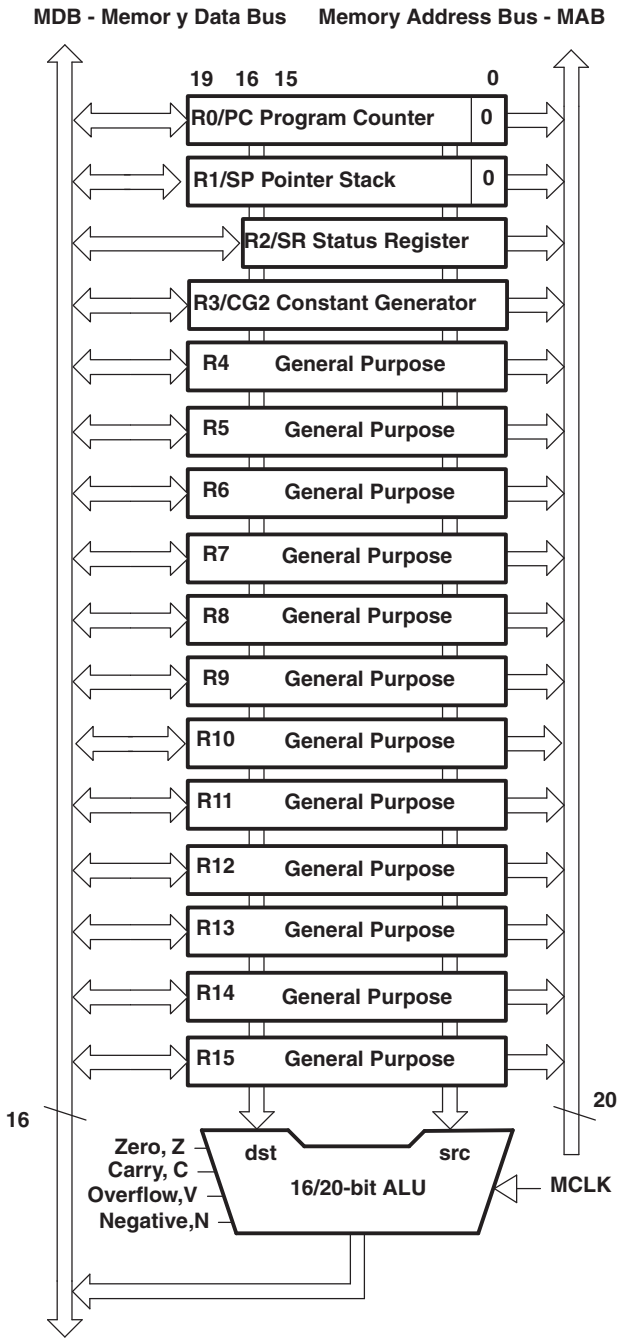


Figure 4-1. MSP430X CPU Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.2 Interrupts

The MSP430X has the following interrupt structure:

- Vectored interrupts with no polling necessary
- Interrupt vectors are located downward from address 0FFFEh.

The interrupt vectors contain 16-bit addresses that point into the lower 64-KB memory. This means all interrupt handlers must start in the lower 64-KB memory.

During an interrupt, the program counter (PC) and the status register (SR) are pushed onto the stack as shown in Figure 4-2. The MSP430X architecture stores the complete 20-bit PC value efficiently by appending the PC bits 19:16 to the stored SR value automatically on the stack. When the RETI instruction is executed, the full 20-bit PC is restored making return from interrupt to any address in the memory range possible.

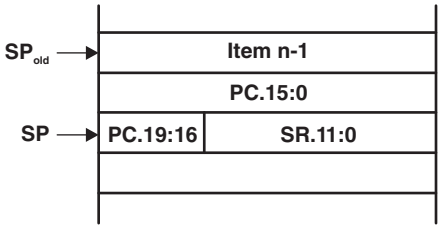


Figure 4-2. PC Storage on the Stack for Interrupts

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.3 CPU Registers

The CPU incorporates 16 registers (R0 through R15). Registers R0, R1, R2, and R3 have dedicated functions. Registers R4 through R15 are working registers for general use.

4.3.1 Program Counter (PC)

The 20-bit Program Counter (PC, also called R0) points to the next instruction to be executed. Each instruction uses an even number of bytes (2, 4, 6, or 8 bytes), and the PC is incremented accordingly. Instruction accesses are performed on word boundaries, and the PC is aligned to even addresses. Figure 4-3 shows the PC.

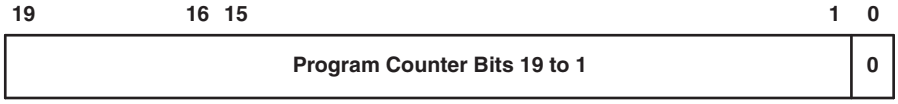


Figure 4-3. Program Counter

The PC can be addressed with all instructions and addressing modes. A few examples:

```
MOV.W  #LABEL,PC  ; Branch to address LABEL (lower 64 KB)

MOVA   #LABEL,PC  ; Branch to address LABEL (1MB memory)

MOV.W  LABEL,PC   ; Branch to address in word LABEL
                  ; (lower 64 KB)

MOV.W  @R14,PC    ; Branch indirect to address in
                  ; R14 (lower 64 KB)

ADDA   #4,PC      ; Skip two words (1 MB memory)
```

The BR and CALL instructions reset the upper four PC bits to 0. Only addresses in the lower 64-KB address range can be reached with the BR or CALL instruction. When branching or calling, addresses beyond the lower 64-KB range can only be reached using the BRA or CALLA instructions. Also, any instruction to directly modify the PC does so according to the used addressing mode. For example, `MOV.W #value,PC` clears the upper four bits of the PC, because it is a .W instruction.

The PC is automatically stored on the stack with CALL (or CALLA) instructions and during an interrupt service routine. Figure 4-4 shows the storage of the PC with the return address after a CALLA instruction. A CALL instruction stores only bits 15:0 of the PC.

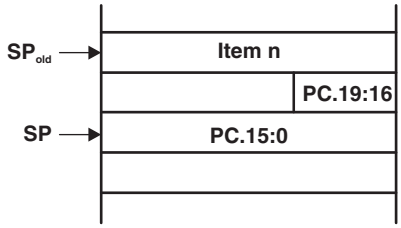


Figure 4-4. PC Storage on the Stack for CALLA

The RETA instruction restores bits 19:0 of the PC and adds 4 to the stack pointer (SP). The RET instruction restores bits 15:0 to the PC and adds 2 to the SP.

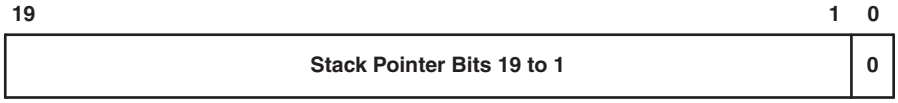
4.3.2 Stack Pointer (SP)

The 20-bit Stack Pointer (SP, also called R1) is used by the CPU to store the return addresses of subroutine calls and interrupts. It uses a predecrement, postincrement scheme. In addition, the SP can be used by software with all instructions and addressing modes. Figure 4-5 shows the SP. The SP is initialized into RAM by the user, and is always aligned to even addresses.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Figure 4-6 shows the stack usage. Figure 4-7 shows the stack usage when 20-bit address words are pushed.




```
MOV.W   2(SP),R6      ; Copy Item I2 to R6
MOV.W   R7,0(SP)      ; Overwrite TOS with R7
PUSH    #0123h        ; Put 0123h on stack
POP     R8            ; R8 = 0123h
```

Figure 4-5. Stack Pointer

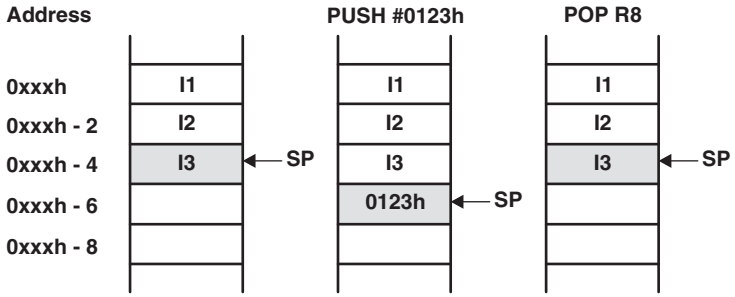


Figure 4-6. Stack Usage

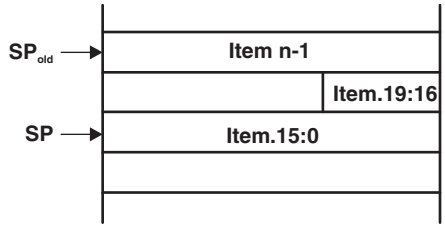


Figure 4-7. PUSHX.A Format on the Stack

The special cases of using the SP as an argument to the PUSH and POP instructions are described and shown in Figure 4-8.

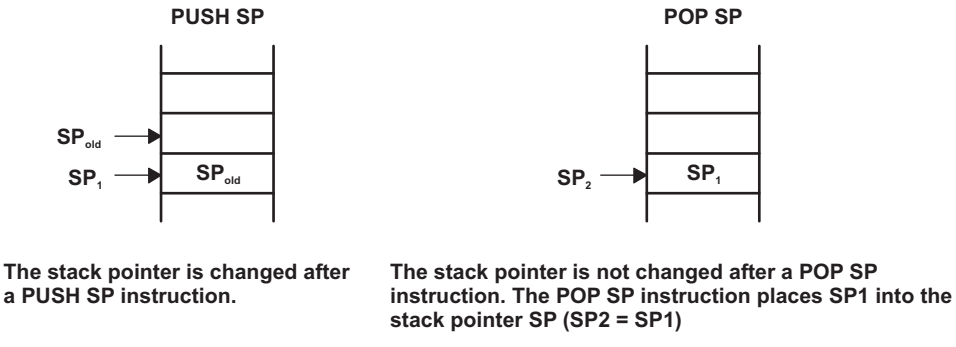


Figure 4-8. PUSH SP, POP SP Sequence

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.3.3 Status Register (SR)

The 16-bit Status Register (SR, also called R2), used as a source or destination register, can only be used in register mode addressed with word instructions. The remaining combinations of addressing modes are used to support the constant generator. Figure 4-9 shows the SR bits. Do not write 20-bit values to the SR. Unpredictable operation can result.

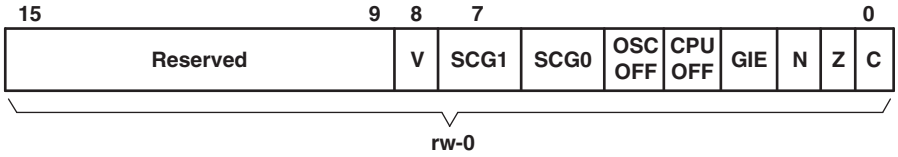


Figure 4-9. SR Bits

Table 4-1 describes the SR bits.

Table 4-1. SR Bit Description

Bit	Description	
Reserved	Reserved	
V	Overflow. This bit is set when the result of an arithmetic operation overflows the signed-variable range.	
	`ADD(.B), ADDX(.B,.A), ADDC(.B), ADDCX(.B.A), ADDA`	Set when: positive + positive = negative negative + negative = positive otherwise reset
	`SUB(.B), SUBX(.B,.A), SUBC(.B),SUBCX(.B,.A), SUBA, CMP(.B), CMPX(.B,.A), CMPA`	Set when: positive – negative = negative negative – positive = positive otherwise reset
SCG1	System clock generator 1. This bit may be used to enable or disable functions in the clock system depending on the device family; for example, DCO bias enable or disable.	
SCG0	System clock generator 0. This bit may be used to enable or disable functions in the clock system depending on the device family; for example, FLL enable or disable.	
OSCOFF	Oscillator off. This bit, when set, turns off the LFXT1 crystal oscillator when LFXT1CLK is not used for MCLK or SMCLK.	
CPUOFF	CPU off. This bit, when set, turns off the CPU.	
SCG1	The bits CPUOFF, OSCOFF, SCG0 and SCG1 request the system to enter a low-power mode	
SCG0		
OSCOFF		
CPUOFF		
GIE	General interrupt enable. This bit, when set, enables maskable interrupts. When reset, all maskable interrupts are disabled.	
N	Negative. This bit is set when the result of an operation is negative and cleared when the result is positive.	
Z	Zero. This bit is set when the result of an operation is 0 and cleared when the result is not 0.	
C	Carry. This bit is set when the result of an operation produced a carry and cleared when no carry occurred.	

NOTE: Bit manipulations of the SR should be done by the following instructions: MOV, BIS, and BIC.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.3.4 Constant Generator Registers (CG1 and CG2)

Six commonly-used constants are generated with the constant generator registers R2 (CG1) and R3 (CG2), without requiring an additional 16-bit word of program code. The constants are selected with the source register addressing modes (As), as described in Table 4-2.

Table 4-2. Values of Constant Generators CG1, CG2

Register	As	Constant	Remarks
R2	00	–	Register mode
R2	01	(0)	Absolute address mode
R2	10	00004h	+4, bit processing
R2	11	00008h	+8, bit processing
R3	00	00000h	0, word processing
R3	01	00001h	+1
R3	10	00002h	+2, bit processing
R3	11	FFh, FFFFh, FFFFFh	–1, word processing

The constant generator advantages are:

- No special instructions required
- No additional code word for the six constants
- No code memory access required to retrieve the constant

The assembler uses the constant generator automatically if one of the six constants is used as an immediate source operand. Registers R2 and R3, used in the constant mode, cannot be addressed explicitly; they act as source-only registers.

4.3.4.1 Constant Generator – Expanded Instruction Set

The RISC instruction set of the MSP430 has only 27 instructions. However, the constant generator allows the MSP430 assembler to support 24 additional emulated instructions. For example, the single-operand instruction:

```
CLR dst
```

is emulated by the double-operand instruction with the same length:

```
MOV R3,dst
```

where the #0 is replaced by the assembler, and R3 is used with As = 00.

```
INC dst
```

is replaced by:

```
ADD 0(R3),dst
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.3.5 General-Purpose Registers (R4 –R15)

The 12 CPU registers (R4 to R15) contain 8-bit, 16-bit, or 20-bit values. Any byte-write to a CPU register clears bits 19:8. Any word-write to a register clears bits 19:16. The only exception is the SXT instruction. The SXT instruction extends the sign through the complete 20-bit register.

Figure 4-10 through Figure 4-14 show the handling of byte, word, and address-word data. Note the reset of the leading most significant bits (MSBs) if a register is the destination of a byte or word instruction.

Figure 4-10 shows byte handling (8-bit data, .B suffix). The handling is shown for a source register and a destination memory byte and for a source memory byte and a destination register.

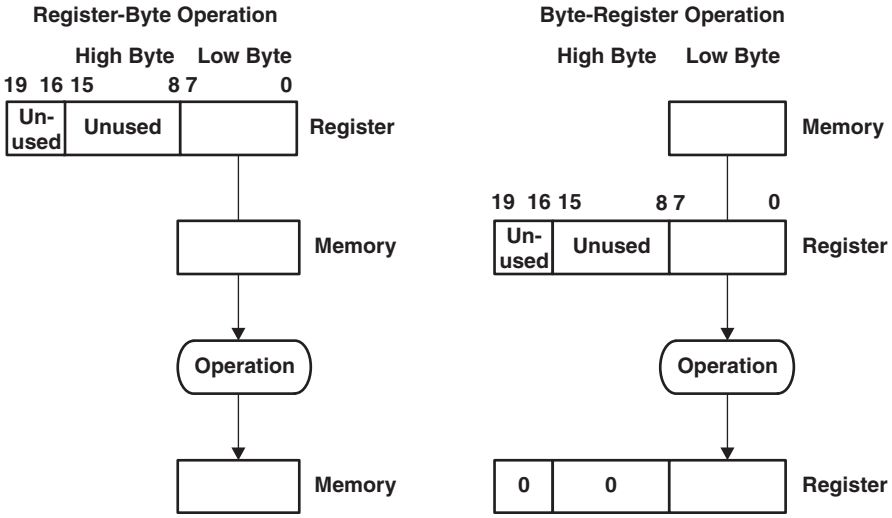


Figure 4-10. Register-Byte and Byte-Register Operation

Figure 4-11 and Figure 4-12 show 16-bit word handling (.W suffix). The handling is shown for a source register and a destination memory word and for a source memory word and a destination register.

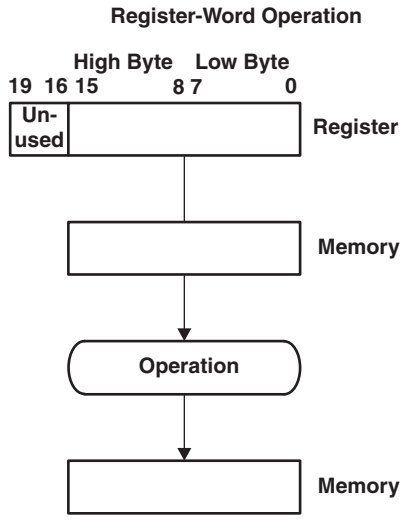


Figure 4-11. Register-Word Operation

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

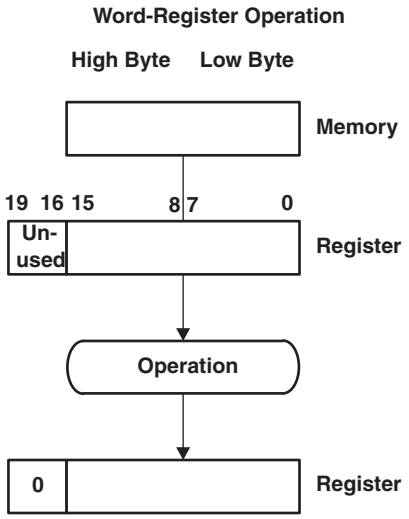


Figure 4-12. Word-Register Operation

Figure 4-13 and Figure 4-14 show 20-bit address-word handling (.A suffix). The handling is shown for a source register and a destination memory address-word and for a source memory address-word and a destination register.

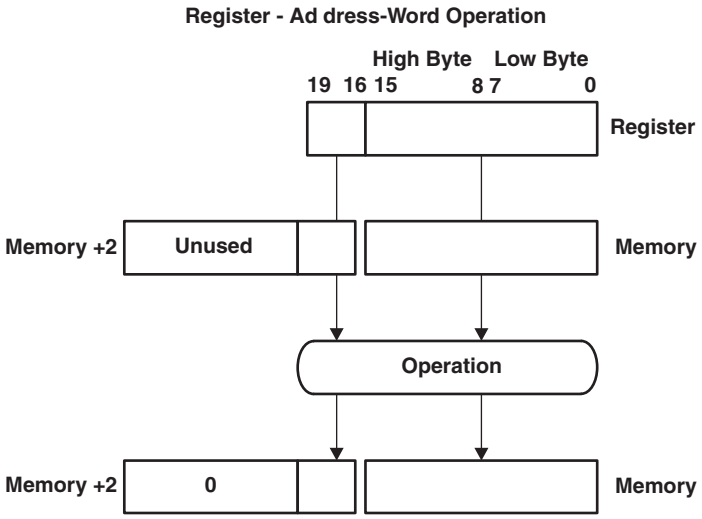


Figure 4-13. Register – Address-Word Operation

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

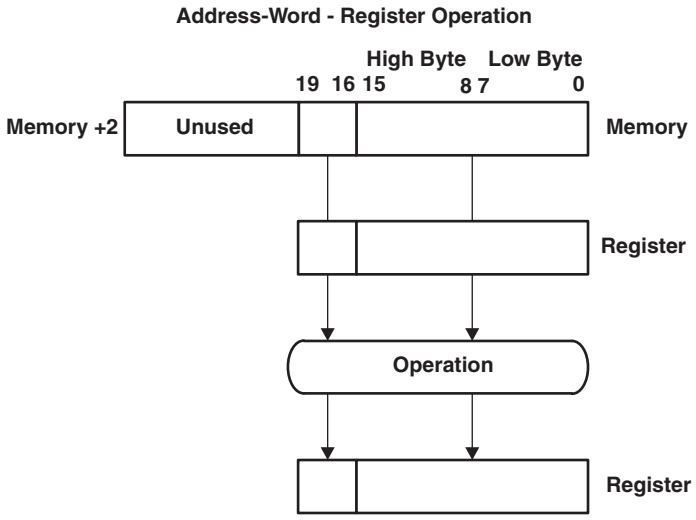


Figure 4-14. Address-Word – Register Operation

4.4 Addressing Modes

Seven addressing modes for the source operand and four addressing modes for the destination operand use 16-bit or 20-bit addresses (see Table 4-3). The MSP430 and MSP430X instructions are usable throughout the entire 1MB memory range.

Table 4-3. Source and Destination Addressing

As, Ad	Addressing Mode	Syntax	Description
00, 0	Register	Rn	Register contents are operand.
01, 1	Indexed	X(Rn)	(Rn + X) points to the operand. X is stored in the next word, or stored in combination of the preceding extension word and the next word.
01, 1	Symbolic	ADDR	(PC + X) points to the operand. X is stored in the next word, or stored in combination of the preceding extension word and the next word. Indexed mode X(PC) is used.
01, 1	Absolute	&ADDR	The word following the instruction contains the absolute address. X is stored in the next word, or stored in combination of the preceding extension word and the next word. Indexed mode X(SR) is used.
10, –	Indirect Register	@Rn	Rn is used as a pointer to the operand.
11, –	Indirect Autoincrement	@Rn+	Rn is used as a pointer to the operand. Rn is incremented afterwards by 1 for .B instructions, by 2 for .W instructions, and by 4 for .A instructions.
11, –	Immediate	#N	N is stored in the next word, or stored in combination of the preceding extension word and the next word. Indirect autoincrement mode @PC+ is used.

The seven addressing modes are explained in detail in the following sections. Most of the examples show the same addressing mode for the source and destination, but any valid combination of source and destination addressing modes is possible in an instruction.

NOTE: Use of Labels EDE, TONI, TOM, and LEO

Throughout MSP430 documentation, EDE, TONI, TOM, and LEO are used as generic labels. They are only labels and have no special meaning.

Copyright © 2009–2013, Texas Instruments Incorporated

4.4.1 Register Mode

Operation: The operand is the 8-, 16-, or 20-bit content of the used CPU register.

Length: One, two, or three words

Comment: Valid for source and destination

Byte operation: Byte operation reads only the eight least significant bits (LSBs) of the source register Rsrc and writes the result to the eight LSBs of the destination register Rdst. The bits Rdst.19:8 are cleared. The register Rsrc is not modified.

Word operation: Word operation reads the 16 LSBs of the source register Rsrc and writes the result to the 16 LSBs of the destination register Rdst. The bits Rdst.19:16 are cleared. The register Rsrc is not modified.

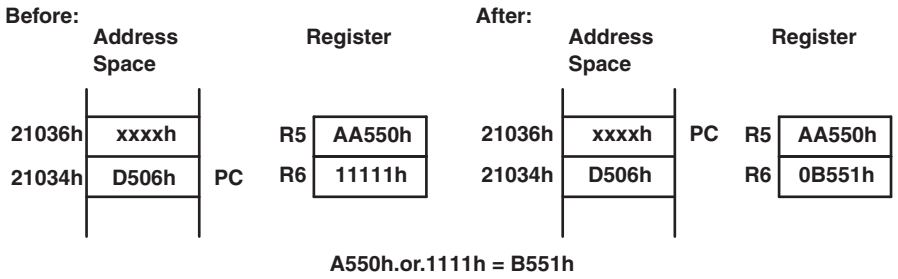
Address-word operation: Address-word operation reads the 20 bits of the source register Rsrc and writes the result to the 20 bits of the destination register Rdst. The register Rsrc is not modified

SXT exception: The SXT instruction is the only exception for register operation. The sign of the low byte in bit 7 is extended to the bits Rdst.19:8.

Example:

```
BIS.W R5,R6 ;
```

This instruction logically ORs the 16-bit data contained in R5 with the 16-bit contents of R6. R6.19:16 is cleared.



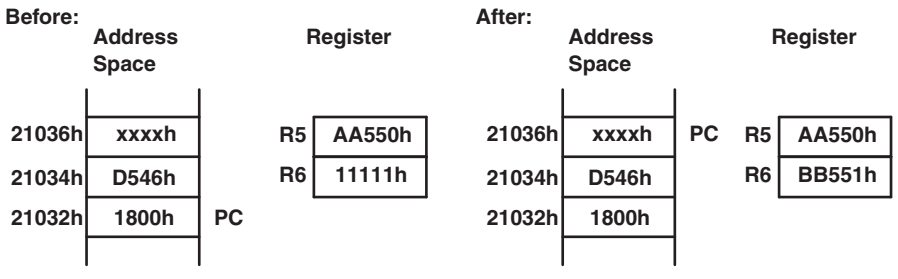
A550h.or.1111h = B551h

Example:

```
BISX.A R5,R6 ;
```

This instruction logically ORs the 20-bit data contained in R5 with the 20-bit contents of R6.

The extension word contains the A/L bit for 20-bit data. The instruction word uses byte mode with bits A/L:B/W = 01. The result of the instruction is:



AA550h.or.11111h = BB551h

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.4.2 Indexed Mode

The Indexed mode calculates the address of the operand by adding the signed index to a CPU register. The Indexed mode has three addressing possibilities:

- Indexed mode in lower 64-KB memory
- MSP430 instruction with Indexed mode addressing memory above the lower 64-KB memory
- MSP430X instruction with Indexed mode

4.4.2.1 Indexed Mode in Lower 64-KB Memory

If the CPU register Rn points to an address in the lower 64 KB of the memory range, the calculated memory address bits 19:16 are cleared after the addition of the CPU register Rn and the signed 16-bit index. This means the calculated memory address is always located in the lower 64 KB and does not overflow or underflow out of the lower 64-KB memory space. The RAM and the peripheral registers can be accessed this way and existing MSP430 software is usable without modifications as shown in Figure 4-15.

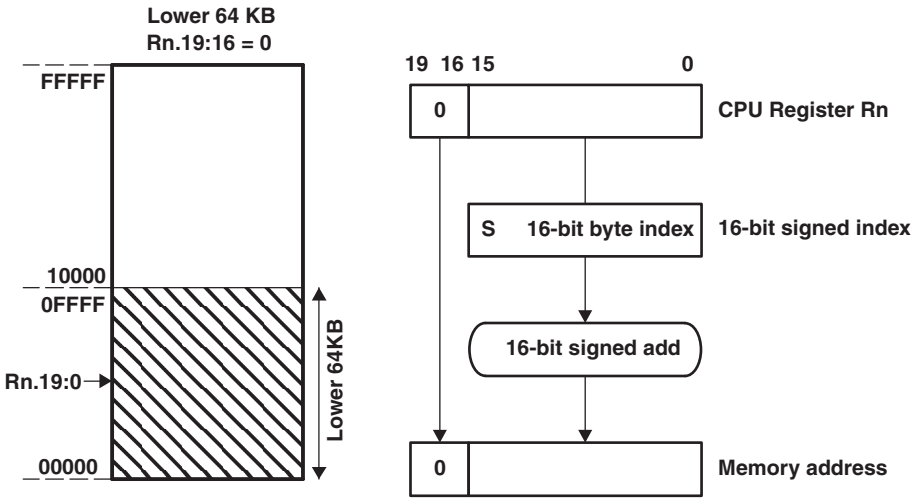
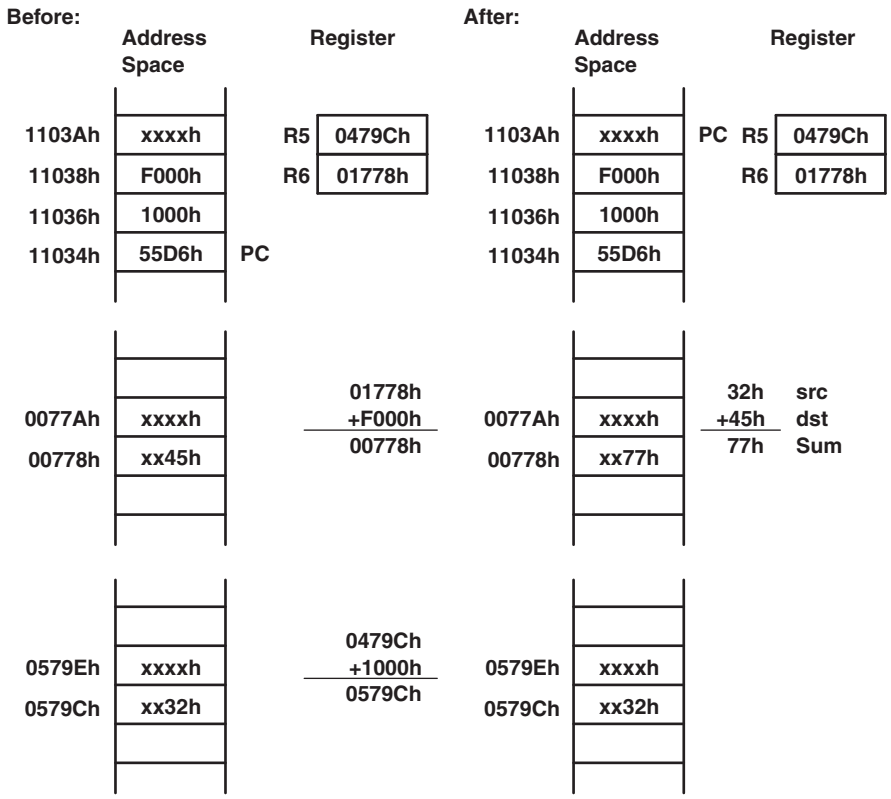


Figure 4-15. Indexed Mode in Lower 64 KB

Length:	Two or three words
Operation:	The signed 16-bit index is located in the next word after the instruction and is added to the CPU register Rn. The resulting bits 19:16 are cleared giving a truncated 16-bit memory address, which points to an operand address in the range 00000h to 0FFFFh. The operand is the content of the addressed memory location.
Comment:	Valid for source and destination. The assembler calculates the register index and inserts it.
Example:	`ADD.B 1000h(R5),0F000h(R6);`
	This instruction adds the 8-bit data contained in source byte 1000h(R5) and the destination byte 0F000h(R6) and places the result into the destination byte. Source and destination bytes are both located in the lower 64 KB due to the cleared bits 19:16 of registers R5 and R6.
Source:	The byte pointed to by R5 + 1000h results in address 0479Ch + 1000h = 0579Ch after truncation to a 16-bit address.
Destination:	The byte pointed to by R6 + F000h results in address 01778h + F000h = 00778h after truncation to a 16-bit address.

Copyright © 2009–2013, Texas Instruments Incorporated



4.4.2.2 MSP430 Instruction With Indexed Mode in Upper Memory

If the CPU register Rn points to an address above the lower 64-KB memory, the Rn bits 19:16 are used for the address calculation of the operand. The operand may be located in memory in the range Rn ±32 KB, because the index, X, is a signed 16-bit value. In this case, the address of the operand can overflow or underflow into the lower 64-KB memory space (see Figure 4-16 and Figure 4-17).

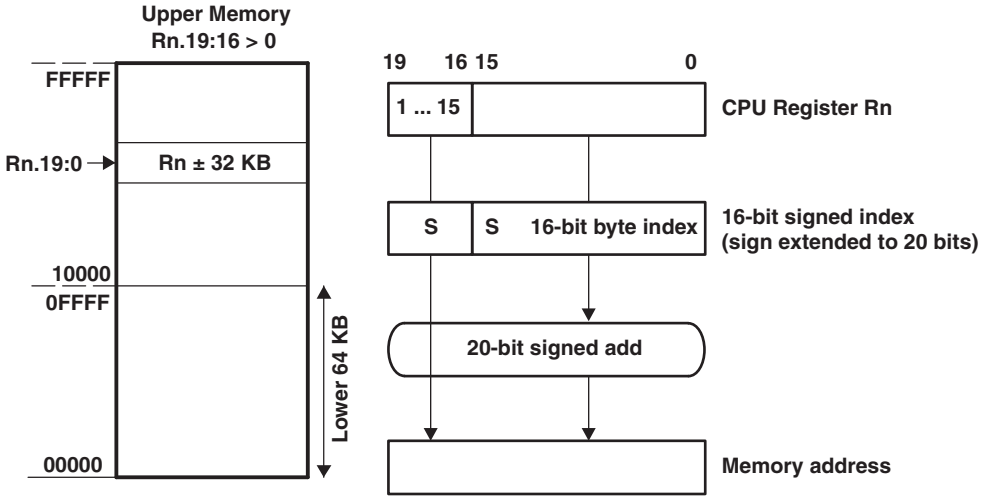


Figure 4-16. Indexed Mode in Upper Memory

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

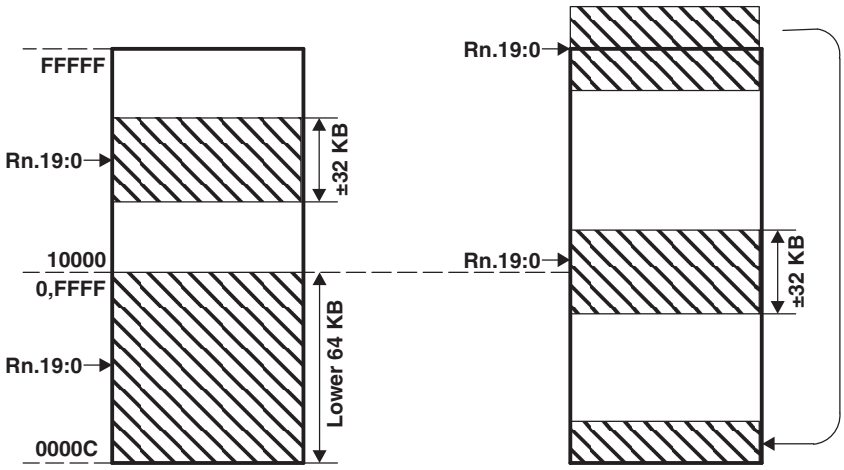


Figure 4-17. Overflow and Underflow for Indexed Mode

Length:	Two or three words
Operation:	The sign-extended 16-bit index in the next word after the instruction is added to the 20 bits of the CPU register Rn. This delivers a 20-bit address, which points to an address in the range 0 to FFFFFh. The operand is the content of the addressed memory location.
Comment:	Valid for source and destination. The assembler calculates the register index and inserts it.
Example:	`ADD.W 8346h(R5),2100h(R6) ;` This instruction adds the 16-bit data contained in the source and the destination addresses and places the 16-bit result into the destination. Source and destination operand can be located in the entire address range.
Source:	The word pointed to by R5 + 8346h. The negative index 8346h is sign extended, which results in address 23456h + F8346h = 1B79Ch.
Destination:	The word pointed to by R6 + 2100h results in address 15678h + 2100h = 17778h.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

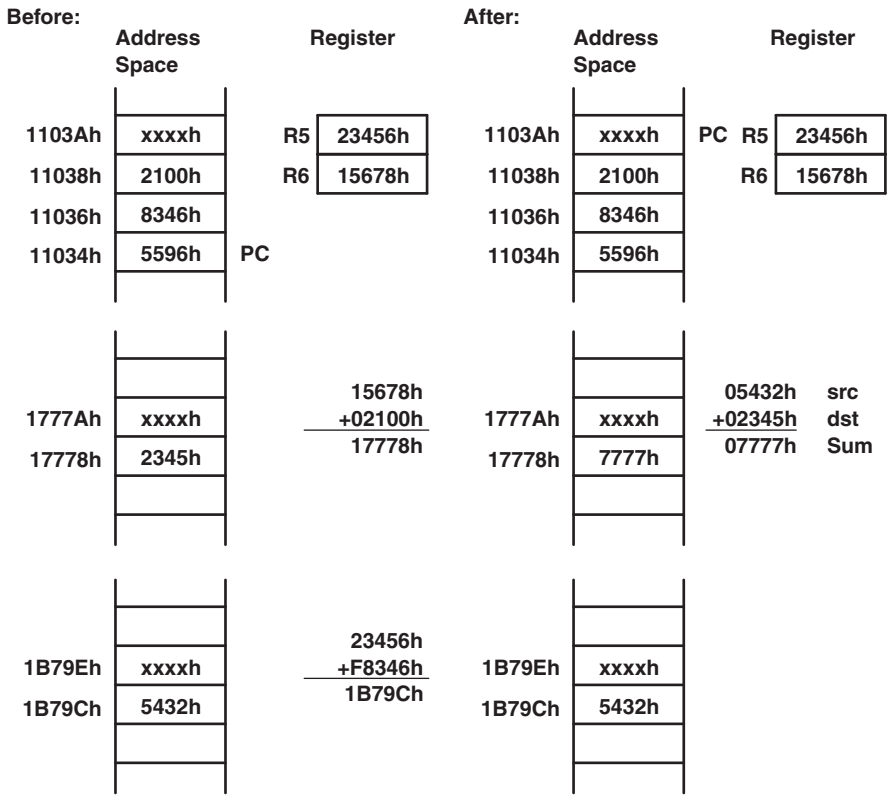


Figure 4-18. Example for Indexed Mode

4.4.2.3 MSP430X Instruction With Indexed Mode

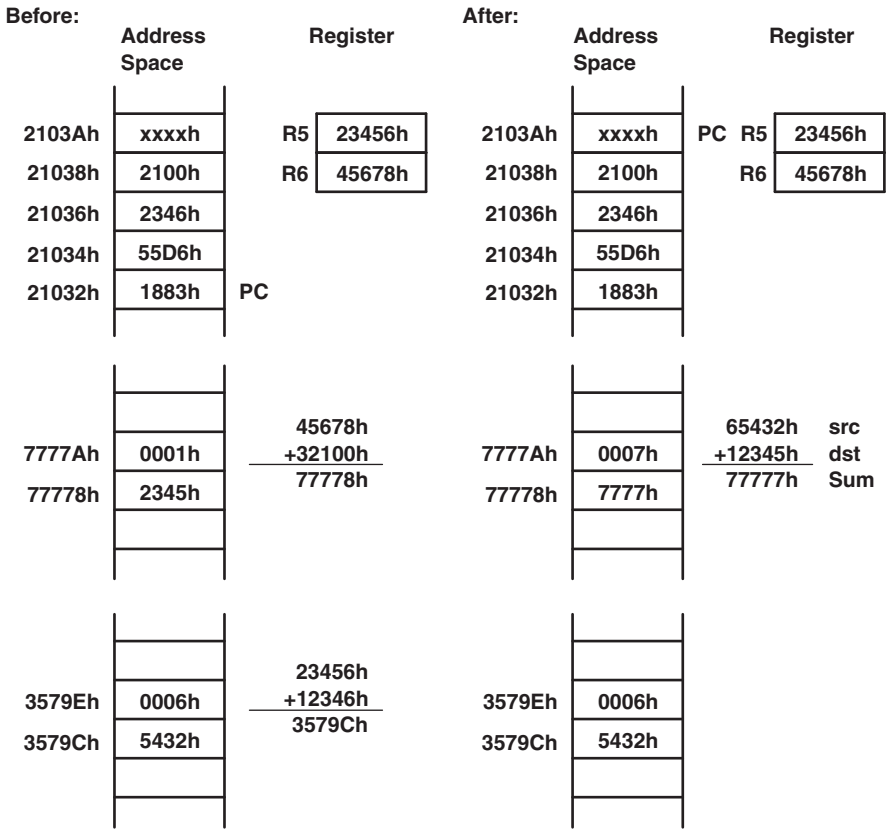
When using an MSP430X instruction with Indexed mode, the operand can be located anywhere in the range of Rn + 19 bits.

Length:	Three or four words
Operation:	The operand address is the sum of the 20-bit CPU register content and the 20-bit index. The 4 MSBs of the index are contained in the extension word; the 16 LSBs are contained in the word following the instruction. The CPU register is not modified
Comment:	Valid for source and destination. The assembler calculates the register index and inserts it.
Example:	`ADDX.A 12346h(R5),32100h(R6) ;` This instruction adds the 20-bit data contained in the source and the destination addresses and places the result into the destination.
Source:	Two words pointed to by R5 + 12346h which results in address 23456h + 12346h = 3579Ch.
Destination:	Two words pointed to by R6 + 32100h which results in address 45678h + 32100h = 77778h.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The extension word contains the MSBs of the source index and of the destination index and the A/L bit for 20-bit data. The instruction word uses byte mode due to the 20-bit data length with bits A/L:B/W = 01.



4.4.3 Symbolic Mode

The Symbolic mode calculates the address of the operand by adding the signed index to the PC. The Symbolic mode has three addressing possibilities:

- Symbolic mode in lower 64-KB memory
- MSP430 instruction with Symbolic mode addressing memory above the lower 64-KB memory.
- MSP430X instruction with Symbolic mode

4.4.3.1 Symbolic Mode in Lower 64 KB

If the PC points to an address in the lower 64 KB of the memory range, the calculated memory address bits 19:16 are cleared after the addition of the PC and the signed 16-bit index. This means the calculated memory address is always located in the lower 64 KB and does not overflow or underflow out of the lower 64-KB memory space. The RAM and the peripheral registers can be accessed this way and existing MSP430 software is usable without modifications as shown in Figure 4-19.

Copyright © 2009–2013, Texas Instruments Incorporated

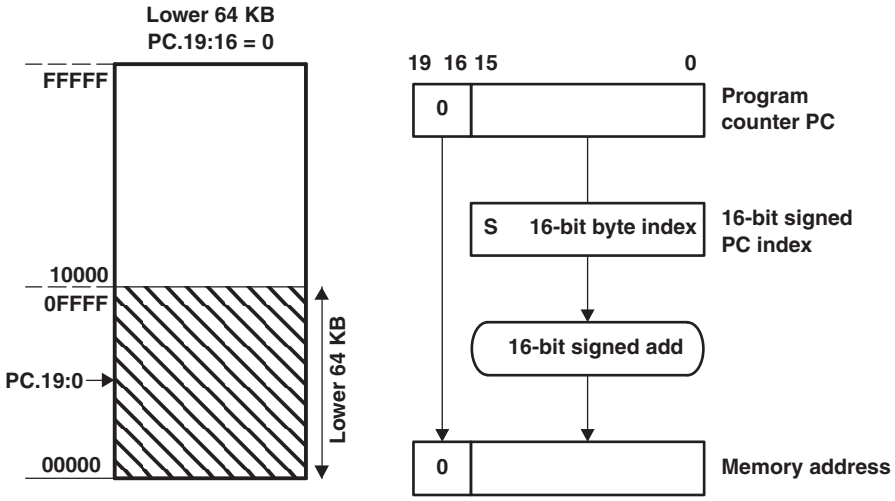
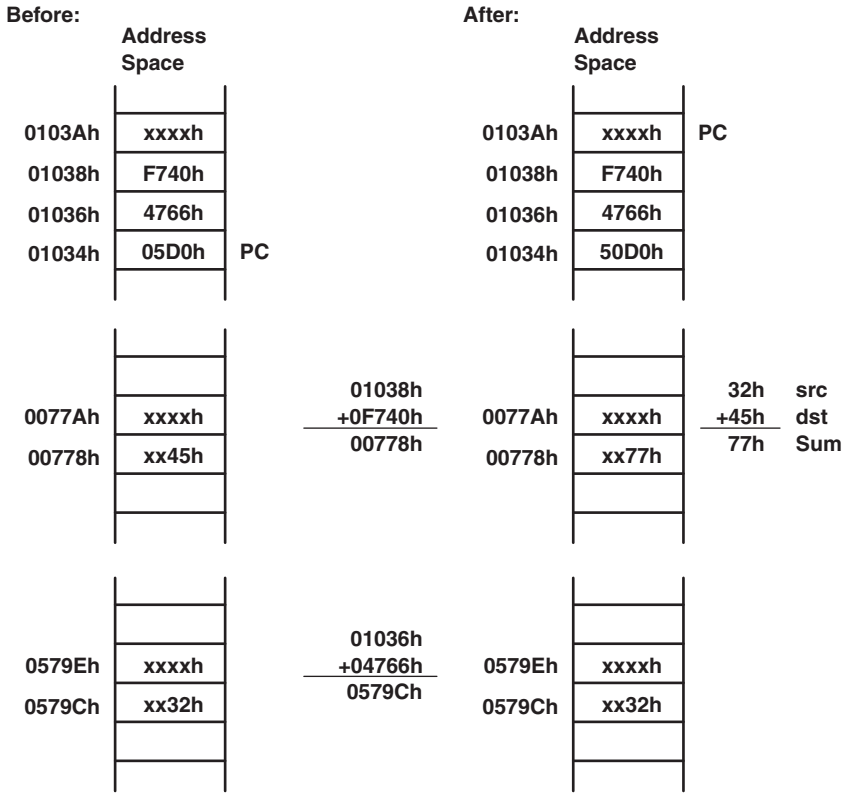


Figure 4-19. Symbolic Mode Running in Lower 64 KB

Operation:	The signed 16-bit index in the next word after the instruction is added temporarily to the PC. The resulting bits 19:16 are cleared giving a truncated 16-bit memory address, which points to an operand address in the range 00000h to 0FFFFh. The operand is the content of the addressed memory location.
Length:	Two or three words
Comment:	Valid for source and destination. The assembler calculates the PC index and inserts it.
Example:	`ADD.B EDE,TONI ;` This instruction adds the 8-bit data contained in source byte EDE and destination byte TONI and places the result into the destination byte TONI. Bytes EDE and TONI and the program are located in the lower 64 KB.
Source:	Byte EDE located at address 0579Ch, pointed to by PC + 4766h, where the PC index 4766h is the result of 0579Ch – 01036h = 04766h. Address 01036h is the location of the index for this example.
Destination:	Byte TONI located at address 00778h, pointed to by PC + F740h, is the truncated 16-bit result of 00778h – 1038h = FF740h. Address 01038h is the location of the index for this example.

Copyright © 2009–2013, Texas Instruments Incorporated



4.4.3.2 MSP430 Instruction With Symbolic Mode in Upper Memory

If the PC points to an address above the lower 64-KB memory, the PC bits 19:16 are used for the address calculation of the operand. The operand may be located in memory in the range PC ± 32 KB, because the index, X, is a signed 16-bit value. In this case, the address of the operand can overflow or underflow into the lower 64-KB memory space as shown in Figure 4-20 and Figure 4-21.

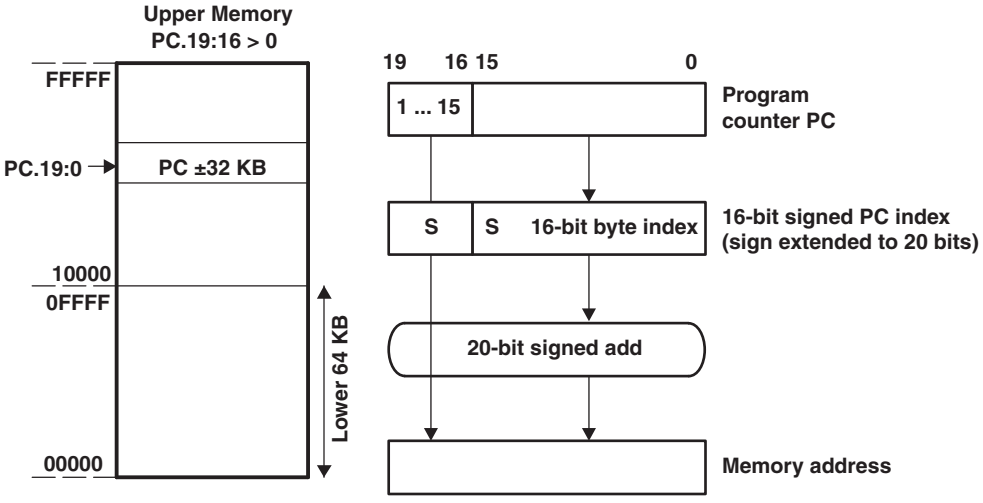


Figure 4-20. Symbolic Mode Running in Upper Memory

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

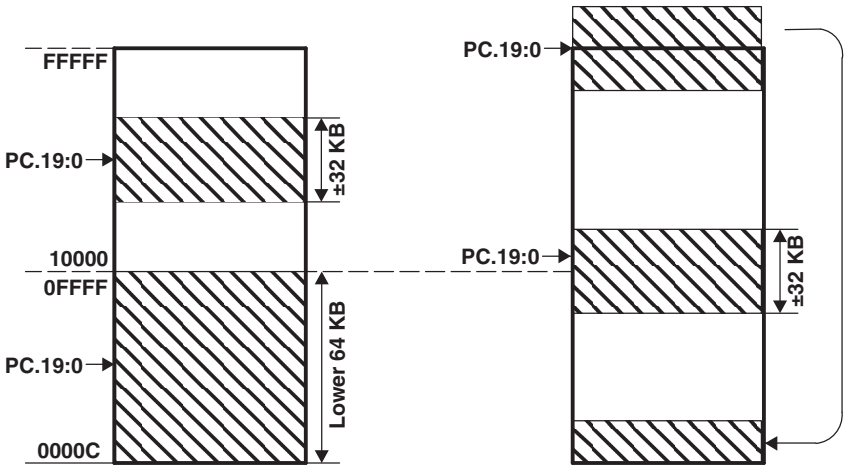
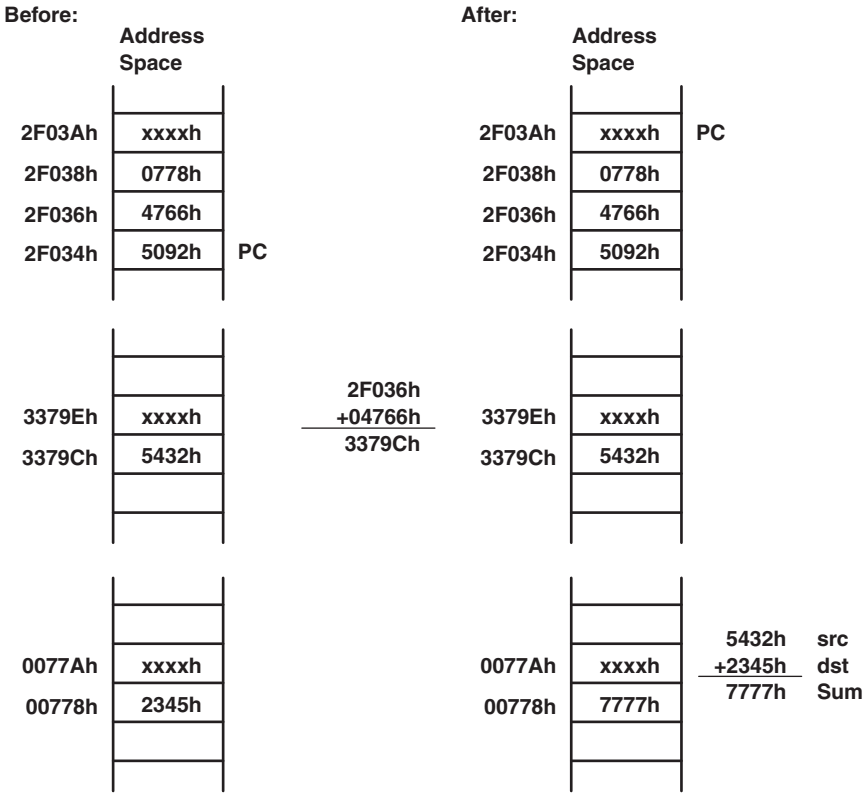


Figure 4-21. Overflow and Underflow for Symbolic Mode

Length:	Two or three words
Operation:	The sign-extended 16-bit index in the next word after the instruction is added to the 20 bits of the PC. This delivers a 20-bit address, which points to an address in the range 0 to FFFFFh. The operand is the content of the addressed memory location.
Comment:	Valid for source and destination. The assembler calculates the PC index and inserts it
Example:	`ADD.W EDE,&TONI ;` This instruction adds the 16-bit data contained in source word EDE and destination word TONI and places the 16-bit result into the destination word TONI. For this example, the instruction is located at address 2F034h.
Source:	Word EDE at address 3379Ch, pointed to by PC + 4766h, which is the 16-bit result of 3379Ch – 2F036h = 04766h. Address 2F036h is the location of the index for this example.
Destination:	Word TONI located at address 00778h pointed to by the absolute address 00778h

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.



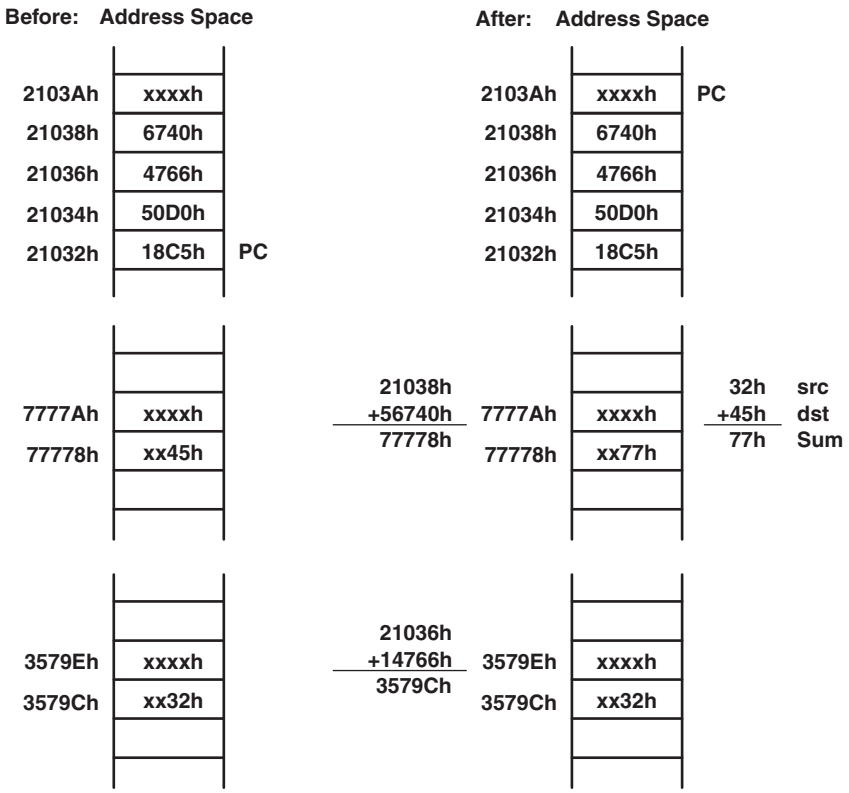
4.4.3.3 MSP430X Instruction With Symbolic Mode

When using an MSP430X instruction with Symbolic mode, the operand can be located anywhere in the range of PC + 19 bits.

Length:	Three or four words
Operation:	The operand address is the sum of the 20-bit PC and the 20-bit index. The 4 MSBs of the index are contained in the extension word; the 16 LSBs are contained in the word following the instruction.
Comment:	Valid for source and destination. The assembler calculates the register index and inserts it.
Example:	`ADDX.B EDE,TONI ;` This instruction adds the 8-bit data contained in source byte EDE and destination byte TONI and places the result into the destination byte TONI.
Source:	Byte EDE located at address 3579Ch, pointed to by PC + 14766h, is the 20-bit result of 3579Ch – 21036h = 14766h. Address 21036h is the address of the index in this example.
Destination:	Byte TONI located at address 77778h, pointed to by PC + 56740h, is the 20-bit result of 77778h – 21038h = 56740h. Address 21038h is the address of the index in this example.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.



4.4.4 Absolute Mode

The Absolute mode uses the contents of the word following the instruction as the address of the operand. The Absolute mode has two addressing possibilities:

- Absolute mode in lower 64-KB memory
- MSP430X instruction with Absolute mode

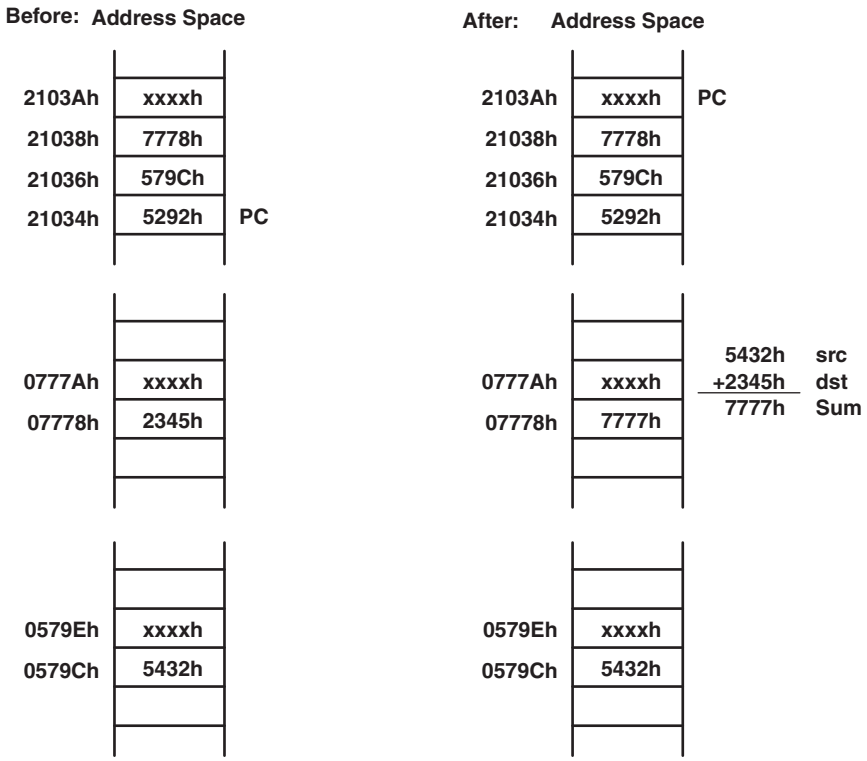
4.4.4.1 Absolute Mode in Lower 64 KB

If an MSP430 instruction is used with Absolute addressing mode, the absolute address is a 16-bit value and, therefore, points to an address in the lower 64 KB of the memory range. The address is calculated as an index from 0 and is stored in the word following the instruction The RAM and the peripheral registers can be accessed this way and existing MSP430 software is usable without modifications.

Length:	Two or three words
Operation:	The operand is the content of the addressed memory location.
Comment:	Valid for source and destination. The assembler calculates the index from 0 and inserts it.
Example:	`ADD.W &EDE,&TONI ;`
	This instruction adds the 16-bit data contained in the absolute source and destination addresses and places the result into the destination.
Source:	Word at address EDE
Destination:	Word at address TONI

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.



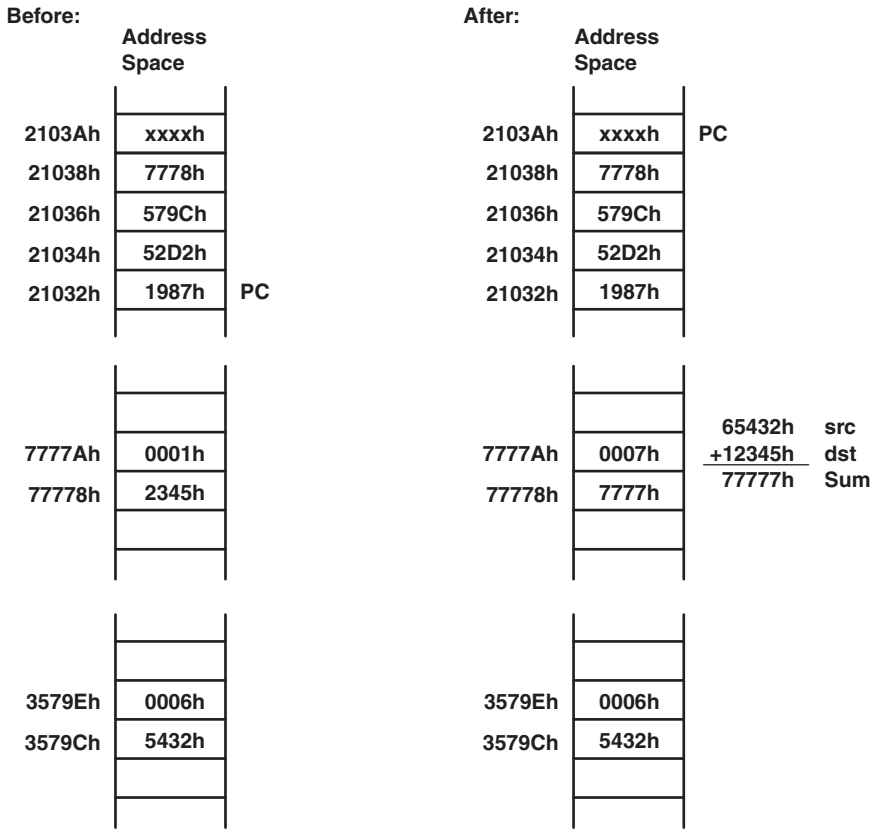
4.4.4.2 MSP430X Instruction With Absolute Mode

If an MSP430X instruction is used with Absolute addressing mode, the absolute address is a 20-bit value and, therefore, points to any address in the memory range. The address value is calculated as an index from 0. The 4 MSBs of the index are contained in the extension word, and the 16 LSBs are contained in the word following the instruction.

Length:	Three or four words
Operation:	The operand is the content of the addressed memory location.
Comment:	Valid for source and destination. The assembler calculates the index from 0 and inserts it.
Example:	`ADDX.A &EDE,&TONI ;`
	This instruction adds the 20-bit data contained in the absolute source and destination addresses and places the result into the destination.
Source:	Two words beginning with address EDE
Destination:	Two words beginning with address TONI

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

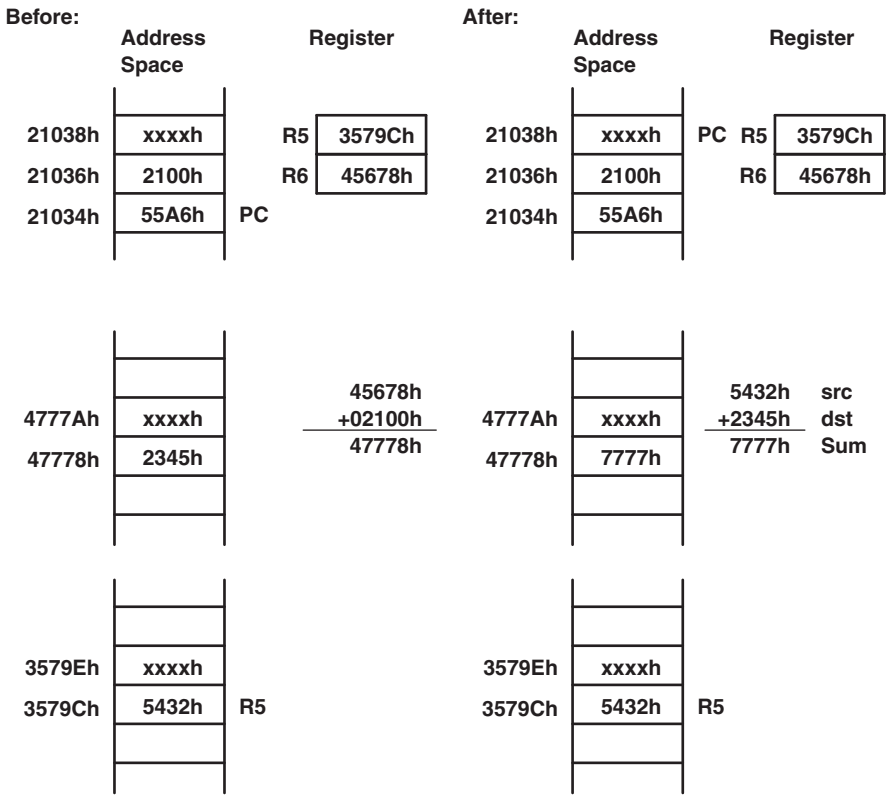


4.4.5 Indirect Register Mode

The Indirect Register mode uses the contents of the CPU register Rsrc as the source operand. The Indirect Register mode always uses a 20-bit address.

Length:	One, two, or three words
Operation:	The operand is the content the addressed memory location. The source register Rsrc is not modified.
Comment:	Valid only for the source operand. The substitute for the destination operand is 0(Rdst).
Example:	`ADDX.W @R5,2100h(R6)`
	This instruction adds the two 16-bit operands contained in the source and the destination addresses and places the result into the destination.
Source:	Word pointed to by R5. R5 contains address 3579Ch for this example.
Destination:	Word pointed to by R6 + 2100h, which results in address 45678h + 2100h = 7778h

Copyright © 2009–2013, Texas Instruments Incorporated



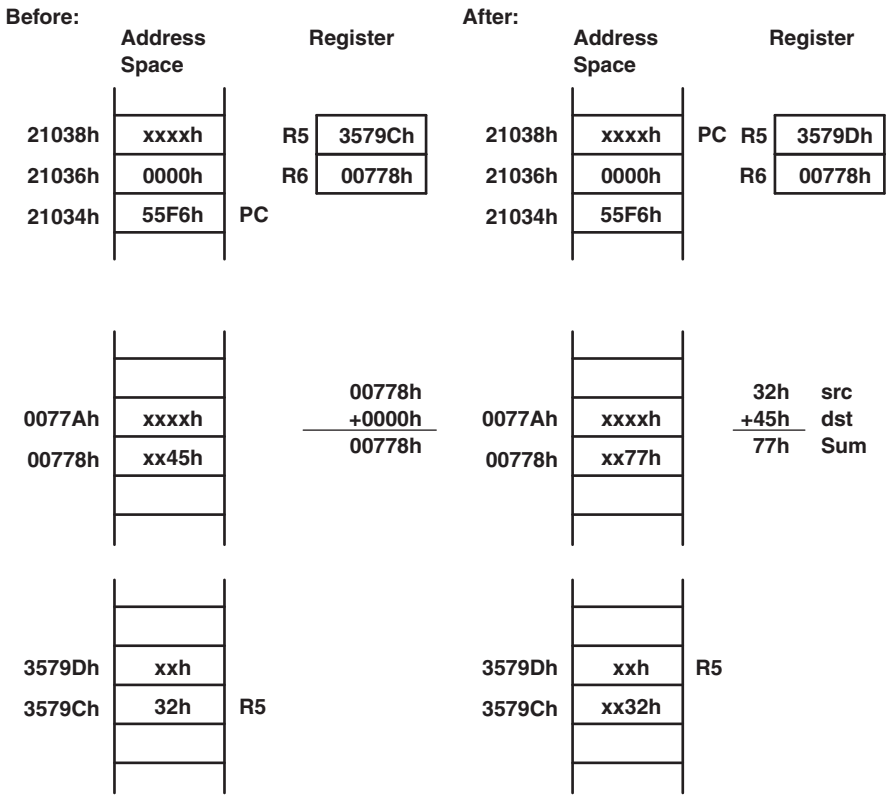
4.4.6 Indirect Autoincrement Mode

The Indirect Autoincrement mode uses the contents of the CPU register Rsrc as the source operand. Rsrc is then automatically incremented by 1 for byte instructions, by 2 for word instructions, and by 4 for address-word instructions immediately after accessing the source operand. If the same register is used for source and destination, it contains the incremented address for the destination access. Indirect Autoincrement mode always uses 20-bit addresses.

Length:	One, two, or three words
Operation:	The operand is the content of the addressed memory location.
Comment:	Valid only for the source operand
Example:	`ADD.B @R5+,0(R6)`
	This instruction adds the 8-bit data contained in the source and the destination addresses and places the result into the destination.
Source:	Byte pointed to by R5. R5 contains address 3579Ch for this example.
Destination:	Byte pointed to by R6 + 0h, which results in address 0778h for this example

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.



4.4.7 Immediate Mode

The Immediate mode allows accessing constants as operands by including the constant in the memory location following the instruction. The PC is used with the Indirect Autoincrement mode. The PC points to the immediate value contained in the next word. After the fetching of the immediate operand, the PC is incremented by 2 for byte, word, or address-word instructions. The Immediate mode has two addressing possibilities:

- 8-bit or 16-bit constants with MSP430 instructions
- 20-bit constants with MSP430X instruction

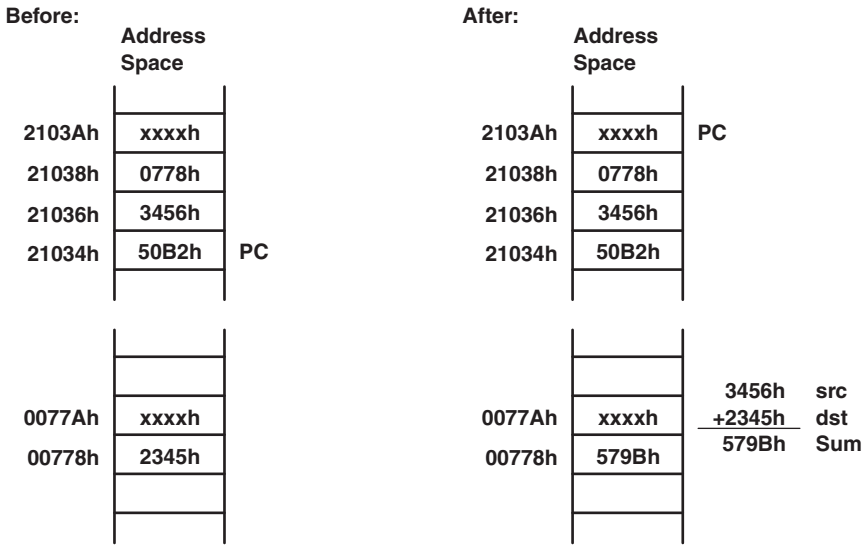
4.4.7.1 MSP430 Instructions With Immediate Mode

If an MSP430 instruction is used with Immediate addressing mode, the constant is an 8- or 16-bit value and is stored in the word following the instruction.

Length:	Two or three words. One word less if a constant of the constant generator can be used for the immediate operand.
Operation:	The 16-bit immediate source operand is used together with the 16-bit destination operand.
Comment:	Valid only for the source operand
Example:	`ADD #3456h,&TONI` This instruction adds the 16-bit immediate operand 3456h to the data in the destination address TONI.
Source:	16-bit immediate value 3456h
Destination:	Word at address TONI

Copyright © 2009–2013, Texas Instruments Incorporated

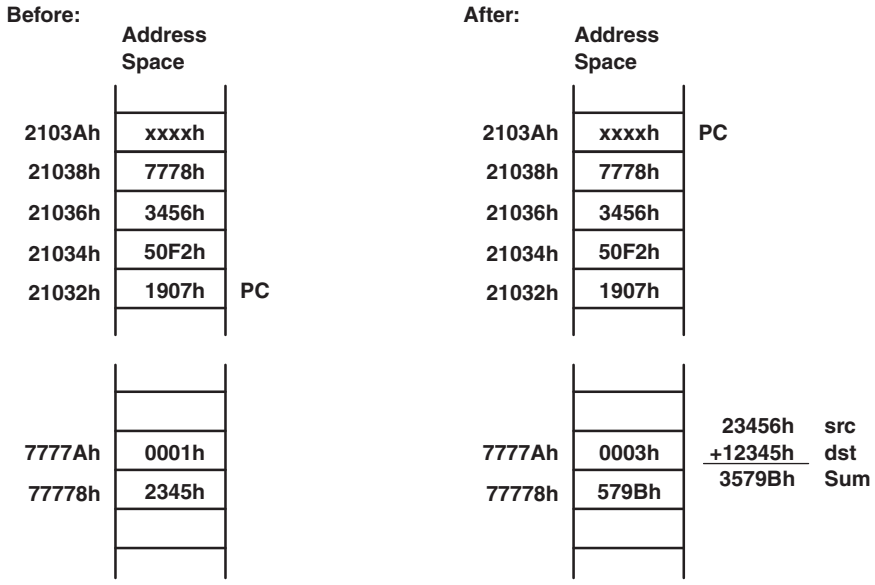
ECCN 5E002 TSPA - Technology / Software Publicly Available.



4.4.7.2 MSP430X Instructions With Immediate Mode

If an MSP430X instruction is used with Immediate addressing mode, the constant is a 20-bit value. The 4 MSBs of the constant are stored in the extension word, and the 16 LSBs of the constant are stored in the word following the instruction.

Length:	Three or four words. One word less if a constant of the constant generator can be used for the immediate operand.
Operation:	The 20-bit immediate source operand is used together with the 20-bit destination operand.
Comment:	Valid only for the source operand
Example:	`ADDX.A #23456h,&TONI ;`
	This instruction adds the 20-bit immediate operand 23456h to the data in the destination address TONI.
Source:	20-bit immediate value 23456h
Destination:	Two words beginning with address TONI



Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5 MSP430 and MSP430X Instructions

MSP430 instructions are the 27 implemented instructions of the MSP430 CPU. These instructions are used throughout the 1MB memory range unless their 16-bit capability is exceeded. The MSP430X instructions are used when the addressing of the operands or the data length exceeds the 16-bit capability of the MSP430 instructions.

There are three possibilities when choosing between an MSP430 and MSP430X instruction:

- To use only the MSP430 instructions – The only exceptions are the CALLA and the RETA instruction. This can be done if a few, simple rules are met:
  - Place all constants, variables, arrays, tables, and data in the lower 64 KB. This allows the use of MSP430 instructions with 16-bit addressing for all data accesses. No pointers with 20-bit addresses are needed.
  - Place subroutine constants immediately after the subroutine code. This allows the use of the symbolic addressing mode with its 16-bit index to reach addresses within the range of PC + 32 KB.
- To use only MSP430X instructions – The disadvantages of this method are the reduced speed due to the additional CPU cycles and the increased program space due to the necessary extension word for any double-operand instruction.
- Use the best fitting instruction where needed.

Section 4.5.1 lists and describes the MSP430 instructions, and Section 4.5.2 lists and describes the MSP430X instructions.

4.5.1 MSP430 Instructions

The MSP430 instructions can be used, regardless if the program resides in the lower 64 KB or beyond it. The only exceptions are the instructions CALL and RET, which are limited to the lower 64-KB address range. CALLA and RETA instructions have been added to the MSP430X CPU to handle subroutines in the entire address range with no code size overhead.

4.5.1.1 MSP430 Double-Operand (Format I) Instructions

Figure 4-22 shows the format of the MSP430 double-operand instructions. Source and destination words are appended for the Indexed, Symbolic, Absolute, and Immediate modes. Table 4-4 lists the 12 MSP430 double-operand instructions.

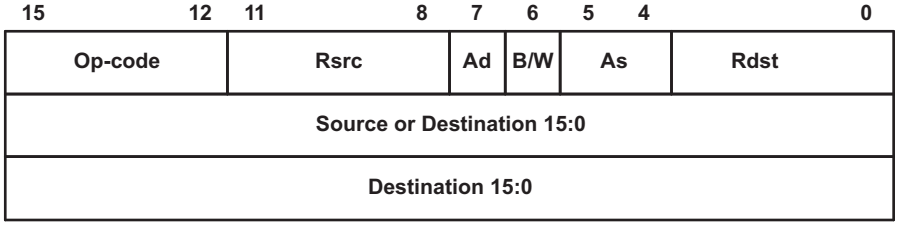


Figure 4-22. MSP430 Double-Operand Instruction Format

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 4-4. MSP430 Double-Operand Instructions

Mnemonic	S-Reg, D-Reg	Operation	Status Bits (1)			
			V	N	Z	C
MOV(.B)	src,dst	src → dst	–	–	–	–
ADD(.B)	src,dst	src + dst → dst	*	*	*	*
ADDC(.B)	src,dst	src + dst + C → dst	*	*	*	*
SUB(.B)	src,dst	dst + .not.src + 1 → dst	*	*	*	*
SUBC(.B)	src,dst	dst + .not.src + C → dst	*	*	*	*
CMP(.B)	src,dst	dst - src	*	*	*	*
DADD(.B)	src,dst	src + dst + C → dst (decimally)	*	*	*	*
BIT(.B)	src,dst	src .and. dst	0	*	*	$\overline{Z}$
BIC(.B)	src,dst	.not.src .and. dst → dst	–	–	–	–
BIS(.B)	src,dst	src .or. dst → dst	–	–	–	–
XOR(.B)	src,dst	src .xor. dst → dst	*	*	*	$\overline{Z}$
AND(.B)	src,dst	src .and. dst → dst	0	*	*	$\overline{Z}$

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

4.5.1.2 MSP430 Single-Operand (Format II) Instructions

Figure 4-23 shows the format for MSP430 single-operand instructions, except RETI. The destination word is appended for the Indexed, Symbolic, Absolute, and Immediate modes. Table 4-5 lists the seven single-operand instructions.

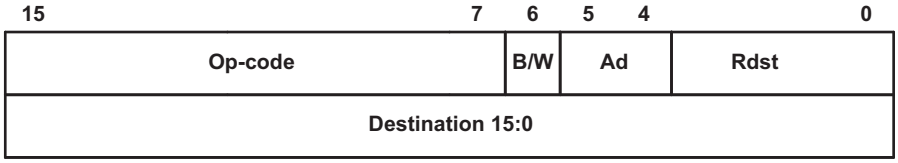


Figure 4-23. MSP430 Single-Operand Instructions

Table 4-5. MSP430 Single-Operand Instructions

Mnemonic	S-Reg, D-Reg	Operation	Status Bits (1)			
			V	N	Z	C
RRC(.B)	dst	C → MSB →.....LSB → C	0	*	*	*
RRA(.B)	dst	MSB → MSB →....LSB → C	0	*	*	*
PUSH(.B)	src	SP - 2 → SP, src → SP	–	–	–	–
SWPB	dst	bit 15...bit 8 ↔ bit 7...bit 0	–	–	–	–
CALL	dst	Call subroutine in lower 64 KB	–	–	–	–
RETI		TOS → SR, SP + 2 → SP	*	*	*	*
		TOS → PC,SP + 2 → SP				
SXT	dst	Register mode: bit 7 → bit 8...bit 19 Other modes: bit 7 → bit 8...bit 15	0	*	*	$\overline{Z}$

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

Copyright © 2009–2013, Texas Instruments Incorporated

4.5.1.3 Jump Instructions

Figure 4-24 shows the format for MSP430 and MSP430X jump instructions. The signed 10-bit word offset of the jump instruction is multiplied by two, sign-extended to a 20-bit address, and added to the 20-bit PC. This allows jumps in a range of –511 to +512 words relative to the PC in the full 20-bit address space. Jumps do not affect the status bits. Table 4-6 lists and describes the eight jump instructions.

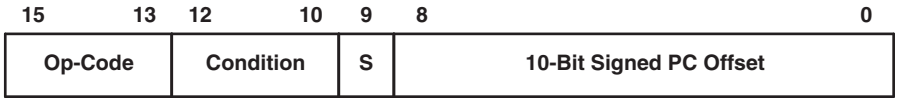


Figure 4-24. Format of Conditional Jump Instructions

Table 4-6. Conditional Jump Instructions

Mnemonic	S-Reg, D-Reg	Operation
JEQ, JZ	Label	Jump to label if zero bit is set
JNE, JNZ	Label	Jump to label if zero bit is reset
JC	Label	Jump to label if carry bit is set
JNC	Label	Jump to label if carry bit is reset
JN	Label	Jump to label if negative bit is set
JGE	Label	Jump to label if (N .XOR. V) = 0
JL	Label	Jump to label if (N .XOR. V) = 1
JMP	Label	Jump to label unconditionally

4.5.1.4 Emulated Instructions

In addition to the MSP430 and MSP430X instructions, emulated instructions are instructions that make code easier to write and read, but do not have op-codes themselves. Instead, they are replaced automatically by the assembler with a core instruction. There is no code or performance penalty for using emulated instructions. The emulated instructions are listed in Table 4-7.

Table 4-7. Emulated Instructions

Instruction	Explanation	Emulation	Status Bits (1)			
			V	N	Z	C
ADC(.B) dst	Add Carry to dst	ADDC(.B) #0,dst	*	*	*	*
BR dst	Branch indirectly dst	MOV dst,PC	–	–	–	–
CLR(.B) dst	Clear dst	MOV(.B) #0,dst	–	–	–	–
CLRC	Clear Carry bit	BIC #1,SR	–	–	–	0
CLRN	Clear Negative bit	BIC #4,SR	–	0	–	–
CLRZ	Clear Zero bit	BIC #2,SR	–	–	0	–
DADC(.B) dst	Add Carry to dst decimally	DADD(.B) #0,dst	*	*	*	*
DEC(.B) dst	Decrement dst by 1	SUB(.B) #1,dst	*	*	*	*
DECD(.B) dst	Decrement dst by 2	SUB(.B) #2,dst	*	*	*	*
DINT	Disable interrupt	BIC #8,SR	–	–	–	–
EINT	Enable interrupt	BIS #8,SR	–	–	–	–
INC(.B) dst	Increment dst by 1	ADD(.B) #1,dst	*	*	*	*
INCD(.B) dst	Increment dst by 2	ADD(.B) #2,dst	*	*	*	*
INV(.B) dst	Invert dst	XOR(.B) #–1,dst	*	*	*	*

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 4-7. Emulated Instructions (continued)

Instruction	Explanation	Emulation	Status Bits (1)			
			V	N	Z	C
NOP	No operation	MOV R3,R3	–	–	–	–
POP dst	Pop operand from stack	MOV @SP+,dst	–	–	–	–
RET	Return from subroutine	MOV @SP+,PC	–	–	–	–
RLA(.B) dst	Shift left dst arithmetically	ADD(.B) dst,dst	*	*	*	*
RLC(.B) dst	Shift left dst logically through Carry	ADDC(.B) dst,dst	*	*	*	*
SBC(.B) dst	Subtract Carry from dst	SUBC(.B) #0,dst	*	*	*	*
SETC	Set Carry bit	BIS #1,SR	–	–	–	1
SETN	Set Negative bit	BIS #4,SR	–	1	–	–
SETZ	Set Zero bit	BIS #2,SR	–	–	1	–
TST(.B) dst	Test dst (compare with 0)	CMP(.B) #0,dst	0	*	*	1

4.5.1.5 MSP430 Instruction Execution

The number of CPU clock cycles required for an instruction depends on the instruction format and the addressing modes used – not the instruction itself. The number of clock cycles refers to MCLK.

4.5.1.5.1 Instruction Cycles and Length for Interrupt, Reset, and Subroutines

Table 4-8 lists the length and the CPU cycles for reset, interrupts, and subroutines.

Table 4-8. Interrupt, Return, and Reset Cycles and Length

Action	Execution Time (MCLK Cycles)	Length of Instruction (Words)
Return from interrupt RETI	5	1
Return from subroutine RET	4	1
Interrupt request service (cycles needed before first instruction)	6	–
WDT reset	4	–
Reset ($\overline{\text{RST}}$/NMI)	4	–

4.5.1.5.2 Format II (Single-Operand) Instruction Cycles and Lengths

Table 4-9 lists the length and the CPU cycles for all addressing modes of the MSP430 single-operand instructions.

Table 4-9. MSP430 Format II Instruction Cycles and Length

Addressing Mode	No. of Cycles			Length of Instruction	Example
	RRA, RRC SWPB, SXT	PUSH	CALL		
Rn	1	3	4	1	SWPB R5
@Rn	3	3	4	1	RRC @R9
@Rn+	3	3	4	1	SWPB @R10+
#N	N/A	3	4	2	CALL #LABEL
X(Rn)	4	4	5	2	CALL 2(R7)
EDE	4	4	5	2	PUSH EDE
&EDE	4	4	6	2	SXT &EDE

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.1.5.3 Jump Instructions Cycles and Lengths

All jump instructions require one code word and take two CPU cycles to execute, regardless of whether the jump is taken or not.

4.5.1.5.4 Format I (Double-Operand) Instruction Cycles and Lengths

Table 4-10 lists the length and CPU cycles for all addressing modes of the MSP430 Format I instructions.

Table 4-10. MSP430 Format I Instructions Cycles and Length

Addressing Mode		No. of Cycles	Length of Instruction	Example
Source	Destination			
Rn	Rm	1	1	MOV R5,R8
	PC	3	1	BR R9
	x(Rm)	4(1)	2	ADD R5,4(R6)
	EDE	4(1)	2	XOR R8,EDE
	&EDE	4(1)	2	MOV R5,&EDE
@Rn	Rm	2	1	AND @R4,R5
	PC	4	1	BR @R8
	x(Rm)	5(1)	2	XOR @R5,8(R6)
	EDE	5(1)	2	MOV @R5,EDE
	&EDE	5(1)	2	XOR @R5,&EDE
@Rn+	Rm	2	1	ADD @R5+,R6
	PC	4	1	BR @R9+
	x(Rm)	5(1)	2	XOR @R5,8(R6)
	EDE	5(1)	2	MOV @R9+,EDE
	&EDE	5(1)	2	MOV @R9+,&EDE
#N	Rm	2	2	MOV #20,R9
	PC	3	2	BR #2AEh
	x(Rm)	5(1)	3	MOV #0300h,0(SP)
	EDE	5(1)	3	ADD #33,EDE
	&EDE	5(1)	3	ADD #33,&EDE
x(Rn)	Rm	3	2	MOV 2(R5),R7
	PC	5	2	BR 2(R6)
	TONI	6(1)	3	MOV 4(R7),TONI
	x(Rm)	6(1)	3	ADD 4(R4),6(R9)
	&TONI	6(1)	3	MOV 2(R4),&TONI
EDE	Rm	3	2	AND EDE,R6
	PC	5	2	BR EDE
	TONI	6(1)	3	CMP EDE,TONI
	x(Rm)	6(1)	3	MOV EDE,0(SP)
	&TONI	6(1)	3	MOV EDE,&TONI
&EDE	Rm	3	2	MOV &EDE,R8
	PC	5	2	BR &EDE
	TONI	6(1)	3	MOV &EDE,TONI
	x(Rm)	6(1)	3	MOV &EDE,0(SP)
	&TONI	6(1)	3	MOV &EDE,&TONI

(1) MOV, BIT, and CMP instructions execute in one fewer cycle.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2 MSP430X Extended Instructions

The extended MSP430X instructions give the MSP430X CPU full access to its 20-bit address space. Most MSP430X instructions require an additional word of op-code called the extension word. Some extended instructions do not require an additional word and are noted in the instruction description. All addresses, indexes, and immediate numbers have 20-bit values when preceded by the extension word.

There are two types of extension words:

- Register or register mode for Format I instructions and register mode for Format II instructions
- Extension word for all other address mode combinations

4.5.2.1 Register Mode Extension Word

The register mode extension word is shown in Figure 4-25 and described in Table 4-11. An example is shown in Figure 4-27.

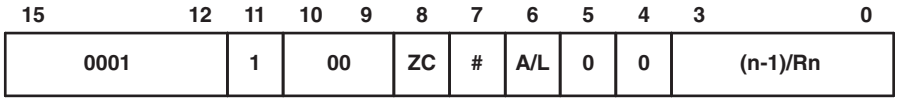


Figure 4-25. Extension Word for Register Modes

Table 4-11. Description of the Extension Word Bits for Register Mode

Bit	Description
15:11	Extension word op-code. Op-codes 1800h to 1FFFh are extension words.
10:9	Reserved
ZC	Zero carry
	0 The executed instruction uses the status of the carry bit C.
	1 The executed instruction uses the carry bit as 0. The carry bit is defined by the result of the final operation after instruction execution.
#	Repetition
	0 The number of instruction repetitions is set by extension word bits 3:0.
	1 The number of instruction repetitions is defined by the value of the four LSBs of Rn. See description for bits 3:0.
A/L	Data length extension. Together with the B/W bits of the following MSP430 instruction, the AL bit defines the used data length of the instruction.
	A/L B/W Comment
	0 0 Reserved
	0 1 20-bit address word
	1 0 16-bit word
	1 1 8-bit byte
5:4	Reserved
3:0	Repetition count
	# = 0 These four bits set the repetition count n. These bits contain n – 1.
	# = 1 These four bits define the CPU register whose bits 3:0 set the number of repetitions. Rn.3:0 contain n – 1.

4.5.2.2 Non-Register Mode Extension Word

The extension word for non-register modes is shown in Figure 4-26 and described in Table 4-12. An example is shown in Figure 4-28.

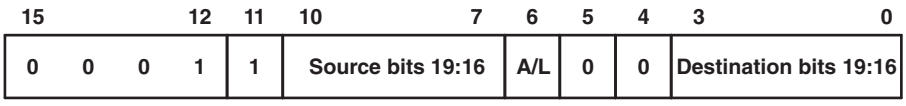


Figure 4-26. Extension Word for Non-Register Modes

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 4-12. Description of Extension Word Bits for Non-Register Modes

Bit	Description
15:11	Extension word op-code. Op-codes 1800h to 1FFFh are extension words.
Source Bits 19:16	The four MSBs of the 20-bit source. Depending on the source addressing mode, these four MSBs may belong to an immediate operand, an index, or to an absolute address.
A/L	Data length extension. Together with the B/W bits of the following MSP430 instruction, the AL bit defines the used data length of the instruction.
	A/L **B/W** **Comment** 0 0 Reserved 0 1 20-bit address word 1 0 16-bit word 1 1 8-bit byte
5:4	Reserved
Destination Bits 19:16	The four MSBs of the 20-bit destination. Depending on the destination addressing mode, these four MSBs may belong to an index or to an absolute address.

NOTE: B/W and A/L bit settings for SWPBX and SXTX

A/L	B/W	
0	0	SWPBX.A, SXTX.A
0	1	N/A
1	0	SWPB.W, SXTX.W
1	1	N/A

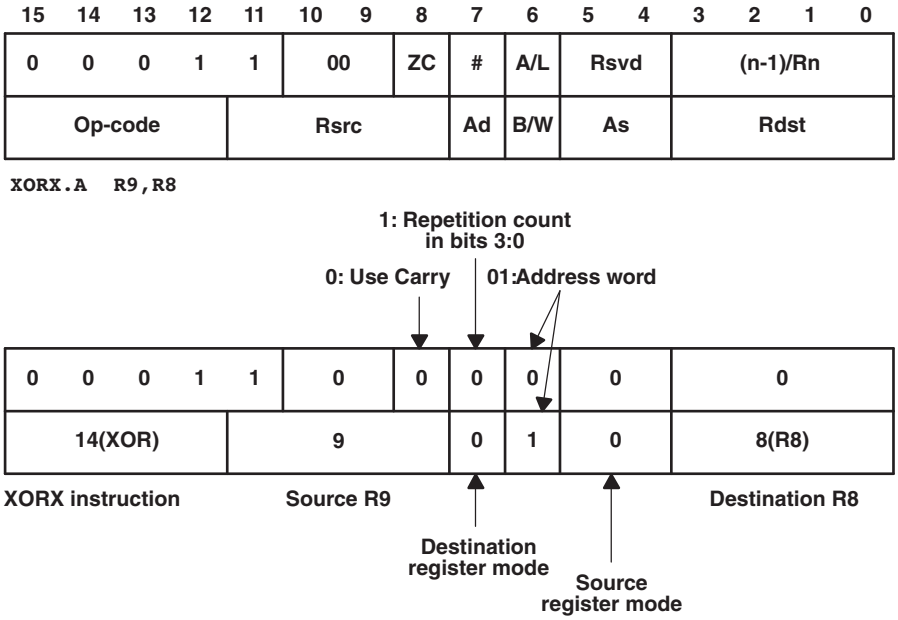


Figure 4-27. Example for Extended Register or Register Instruction

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

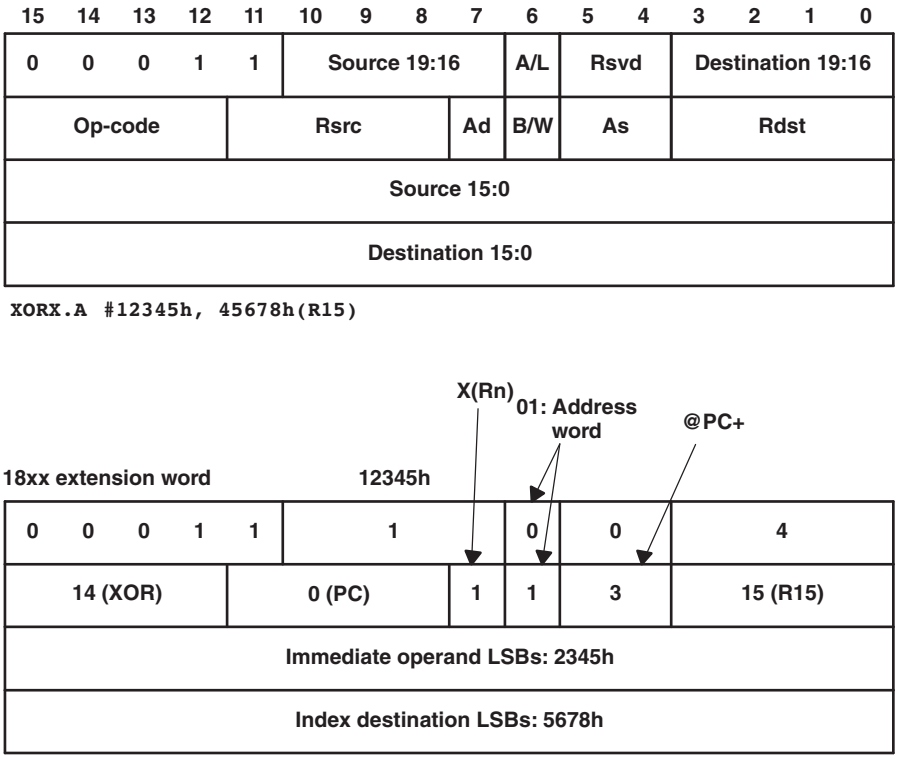


Figure 4-28. Example for Extended Immediate or Indexed Instruction

4.5.2.3 Extended Double-Operand (Format I) Instructions

All 12 double-operand instructions have extended versions as listed in Table 4-13.

Table 4-13. Extended Double-Operand Instructions

Mnemonic	Operands	Operation	Status Bits (1)			
			V	N	Z	C
MOVX(.B,.A)	src,dst	src → dst	–	–	–	–
ADDX(.B,.A)	src,dst	src + dst → dst	*	*	*	*
ADDCX(.B,.A)	src,dst	src + dst + C → dst	*	*	*	*
SUBX(.B,.A)	src,dst	dst + .not.src + 1 → dst	*	*	*	*
SUBCX(.B,.A)	src,dst	dst + .not.src + C → dst	*	*	*	*
CMPX(.B,.A)	src,dst	dst – src	*	*	*	*
DADDX(.B,.A)	src,dst	src + dst + C → dst (decimal)	*	*	*	*
BITX(.B,.A)	src,dst	src .and. dst	0	*	*	$\overline{Z}$
BICX(.B,.A)	src,dst	.not.src .and. dst → dst	–	–	–	–
BISX(.B,.A)	src,dst	src .or. dst → dst	–	–	–	–
XORX(.B,.A)	src,dst	src .xor. dst → dst	*	*	*	$\overline{Z}$
ANDX(.B,.A)	src,dst	src .and. dst → dst	0	*	*	$\overline{Z}$

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The four possible addressing combinations for the extension word for Format I instructions are shown in Figure 4-29.

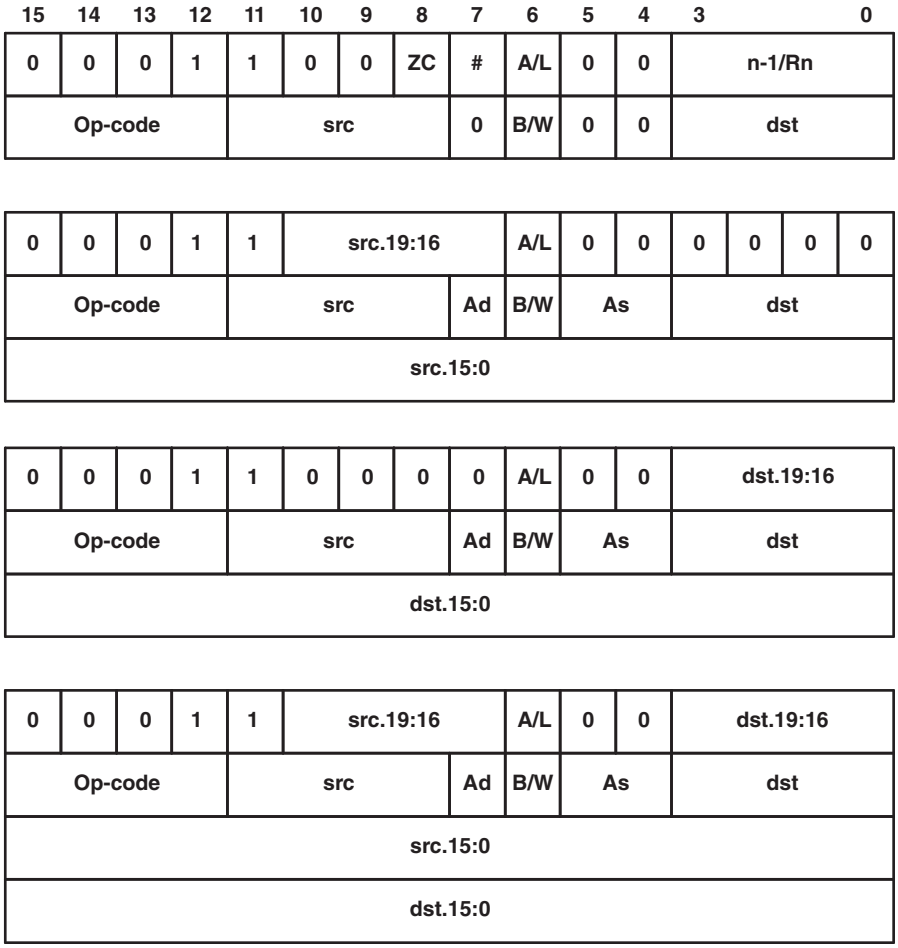


Figure 4-29. Extended Format I Instruction Formats

If the 20-bit address of a source or destination operand is located in memory, not in a CPU register, then two words are used for this operand as shown in Figure 4-30.

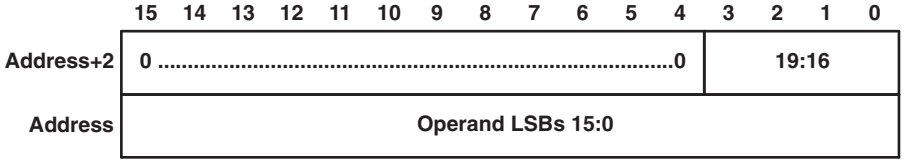


Figure 4-30. 20-Bit Addresses in Memory

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.4 Extended Single-Operand (Format II) Instructions

Extended MSP430X Format II instructions are listed in Table 4-14.

Table 4-14. Extended Single-Operand Instructions

Mnemonic	Operands	Operation	n	Status Bits (1)			
				V	N	Z	C
CALLA	dst	Call indirect to subroutine (20-bit address)		–	–	–	–
POPM.A	#n,Rdst	Pop n 20-bit registers from stack	1 to 16	–	–	–	–
POPM.W	#n,Rdst	Pop n 16-bit registers from stack	1 to 16	–	–	–	–
PUSHM.A	#n,Rsrc	Push n 20-bit registers to stack	1 to 16	–	–	–	–
PUSHM.W	#n,Rsrc	Push n 16-bit registers to stack	1 to 16	–	–	–	–
PUSHX(.B,.A)	src	Push 8-, 16-, or 20-bit source to stack		–	–	–	–
RRCM(.A)	#n,Rdst	Rotate right Rdst n bits through carry (16-, 20-bit register)	1 to 4	0	*	*	*
RRUM(.A)	#n,Rdst	Rotate right Rdst n bits unsigned (16-, 20-bit register)	1 to 4	0	*	*	*
RRAM(.A)	#n,Rdst	Rotate right Rdst n bits arithmetically (16-, 20-bit register)	1 to 4	0	*	*	*
RLAM(.A)	#n,Rdst	Rotate left Rdst n bits arithmetically (16-, 20-bit register)	1 to 4	*	*	*	*
RRCX(.B,.A)	dst	Rotate right dst through carry (8-, 16-, 20-bit data)	1	0	*	*	*
RRUX(.B,.A)	Rdst	Rotate right dst unsigned (8-, 16-, 20-bit)	1	0	*	*	*
RRAX(.B,.A)	dst	Rotate right dst arithmetically	1	0	*	*	*
SWPBX(.A)	dst	Exchange low byte with high byte	1	–	–	–	–
SXTX(.A)	Rdst	Bit7 → bit8 ... bit19	1	0	*	*	$\overline{Z}$
SXTX(.A)	dst	Bit7 → bit8 ... MSB	1	0	*	*	$\overline{Z}$

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

The three possible addressing mode combinations for Format II instructions are shown in Figure 4-31.

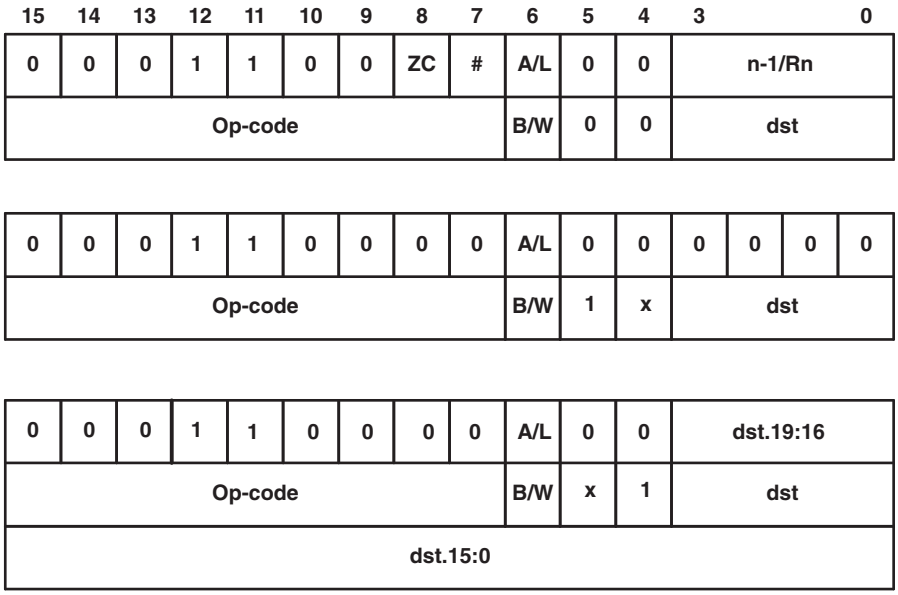


Figure 4-31. Extended Format II Instruction Format

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.4.1 Extended Format II Instruction Format Exceptions

Exceptions for the Format II instruction formats are shown in Figure 4-32 through Figure 4-35.

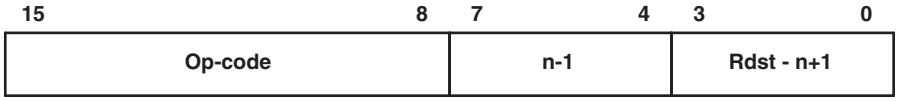


Figure 4-32. PUSHM and POPM Instruction Format

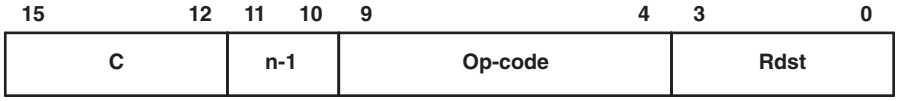


Figure 4-33. RRCM, RRAM, RRUM, and RLAM Instruction Format

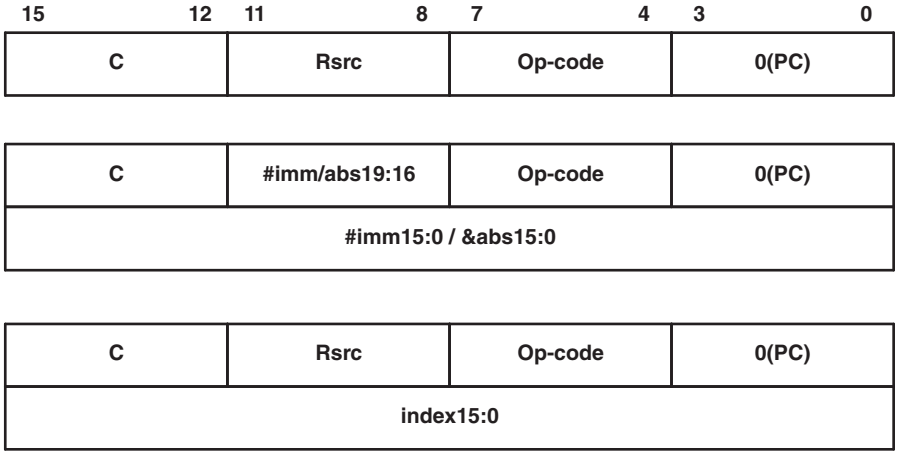


Figure 4-34. BRA Instruction Format

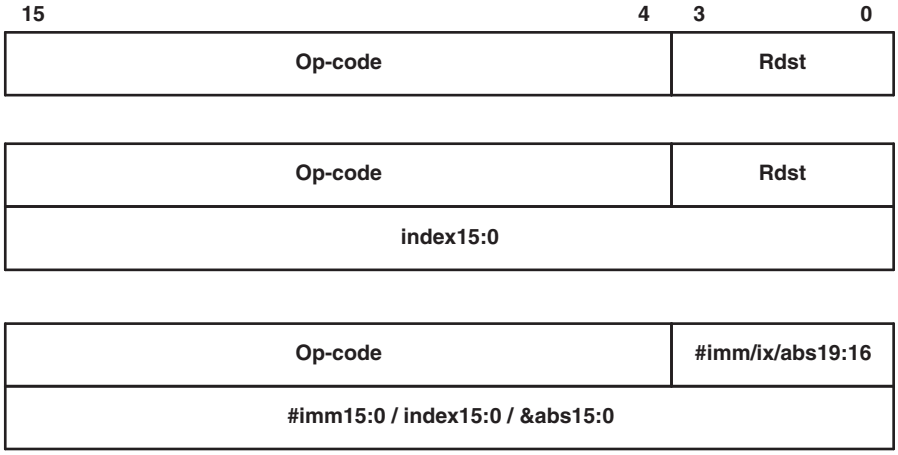


Figure 4-35. CALLA Instruction Format

Copyright © 2009–2013, Texas Instruments Incorporated

4.5.2.5 Extended Emulated Instructions

The extended instructions together with the constant generator form the extended emulated instructions. Table 4-15 lists the emulated instructions.

Table 4-15. Extended Emulated Instructions

Instruction	Explanation	Emulation
ADCX(.B,.A) dst	Add carry to dst	ADDCX(.B,.A) #0,dst
BRA dst	Branch indirect dst	MOVA dst,PC
RETA	Return from subroutine	MOVA @SP+,PC
CLRA Rdst	Clear Rdst	MOV #0,Rdst
CLRX(.B,.A) dst	Clear dst	MOVX(.B,.A) #0,dst
DADCX(.B,.A) dst	Add carry to dst decimally	DADDX(.B,.A) #0,dst
DECX(.B,.A) dst	Decrement dst by 1	SUBX(.B,.A) #1,dst
DECDA Rdst	Decrement Rdst by 2	SUBA #2,Rdst
DECDX(.B,.A) dst	Decrement dst by 2	SUBX(.B,.A) #2,dst
INCX(.B,.A) dst	Increment dst by 1	ADDX(.B,.A) #1,dst
INCDA Rdst	Increment Rdst by 2	ADDA #2,Rdst
INCDX(.B,.A) dst	Increment dst by 2	ADDX(.B,.A) #2,dst
INVX(.B,.A) dst	Invert dst	XORX(.B,.A) #-1,dst
RLAX(.B,.A) dst	Shift left dst arithmetically	ADDX(.B,.A) dst,dst
RLCX(.B,.A) dst	Shift left dst logically through carry	ADDCX(.B,.A) dst,dst
SBCX(.B,.A) dst	Subtract carry from dst	SUBCX(.B,.A) #0,dst
TSTA Rdst	Test Rdst (compare with 0)	CMPA #0,Rdst
TSTX(.B,.A) dst	Test dst (compare with 0)	CMPX(.B,.A) #0,dst
POPX dst	Pop to dst	MOVX(.B, .A) @SP+,dst

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.6 MSP430X Address Instructions

MSP430X address instructions are instructions that support 20-bit operands but have restricted addressing modes. The addressing modes are restricted to the Register mode and the Immediate mode, except for the MOVA instruction as listed in Table 4-16. Restricting the addressing modes removes the need for the additional extension-word op-code improving code density and execution time. Address instructions should be used any time an MSP430X instruction is needed with the corresponding restricted addressing mode.

Table 4-16. Address Instructions, Operate on 20-Bit Register Data

Mnemonic	Operands	Operation	Status Bits (1)			
			V	N	Z	C
ADDA	Rsrc,Rdst	Add source to destination register	*	*	*	*
	#imm20,Rdst					
MOVA	Rsrc,Rdst	Move source to destination	–	–	–	–
	#imm20,Rdst					
	z16(Rsrc),Rdst					
	EDE,Rdst					
	&abs20,Rdst					
	@Rsrc,Rdst					
	@Rsrc+,Rdst					
	Rsrc,z16(Rdst)					
	Rsrc,&abs20					
CMPA	Rsrc,Rdst	Compare source to destination register	*	*	*	*
	#imm20,Rdst					
SUBA	Rsrc,Rdst	Subtract source from destination register	*	*	*	*
	#imm20,Rdst					

(1) * = Status bit is affected.
– = Status bit is not affected.
0 = Status bit is cleared.
1 = Status bit is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.7 MSP430X Instruction Execution

The number of CPU clock cycles required for an MSP430X instruction depends on the instruction format and the addressing modes used, not the instruction itself. The number of clock cycles refers to MCLK.

4.5.2.7.1 MSP430X Format II (Single-Operand) Instruction Cycles and Lengths

Table 4-17 lists the length and the CPU cycles for all addressing modes of the MSP430X extended single-operand instructions.

Table 4-17. MSP430X Format II Instruction Cycles and Length

Instruction	Execution Cycles, Length of Instruction (Words)						
	Rn	@Rn	@Rn+	#N	X(Rn)	EDE	&EDE
RRAM	n, 1	–	–	–	–	–	–
RRCM	n, 1	–	–	–	–	–	–
RRUM	n, 1	–	–	–	–	–	–
RLAM	n, 1	–	–	–	–	–	–
PUSHM	2+n, 1	–	–	–	–	–	–
PUSHM.A	2+2n, 1	–	–	–	–	–	–
POPM	2+n, 1	–	–	–	–	–	–
POPM.A	2+2n, 1	–	–	–	–	–	–
CALLA	5, 1	6, 1	6, 1	5, 2	5(1), 2	7, 2	7, 2
RRAX(.B)	1+n, 2	4, 2	4, 2	–	5, 3	5, 3	5, 3
RRAX.A	1+n, 2	6, 2	6, 2	–	7, 3	7, 3	7, 3
RRCX(.B)	1+n, 2	4, 2	4, 2	–	5, 3	5, 3	5, 3
RRCX.A	1+n, 2	6, 2	6, 2	–	7, 3	7, 3	7, 3
PUSHX(.B)	4, 2	4, 2	4, 2	4, 3	5(1), 3	5, 3	5, 3
PUSHX.A	5, 2	6, 2	6, 2	5, 3	7(1), 3	7, 3	7, 3
POPX(.B)	3, 2	–	–	–	5, 3	5, 3	5, 3
POPX.A	4, 2	–	–	–	7, 3	7, 3	7, 3

(1) Add one cycle when Rn = SP

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.7.2 MSP430X Format I (Double-Operand) Instruction Cycles and Lengths

Table 4-18 lists the length and CPU cycles for all addressing modes of the MSP430X extended Format I instructions.

Table 4-18. MSP430X Format I Instruction Cycles and Length

Addressing Mode		No. of Cycles		Length of Instruction	Examples
Source	Destination	.B/.W	.A	.B/.W/.A	
Rn	Rm[1]	2	2	2	`BITX.B R5,R8`
	PC	4	4	2	`ADDX R9,PC`
	x(Rm)	5[2]	7[3]	3	`ANDX.A R5,4(R6)`
	EDE	5[2]	7[3]	3	`XORX R8,EDE`
	&EDE	5[2]	7[3]	3	`BITX.W R5,&EDE`
@Rn	Rm	3	4	2	`BITX @R5,R8`
	PC	5	6	2	`ADDX @R9,PC`
	x(Rm)	6[2]	9[3]	3	`ANDX.A @R5,4(R6)`
	EDE	6[2]	9[3]	3	`XORX @R8,EDE`
	&EDE	6[2]	9[3]	3	`BITX.B @R5,&EDE`
@Rn+	Rm	3	4	2	`BITX @R5+,R8`
	PC	5	6	2	`ADDX.A @R9+,PC`
	x(Rm)	6[2]	9[3]	3	`ANDX @R5+,4(R6)`
	EDE	6[2]	9[3]	3	`XORX.B @R8+,EDE`
	&EDE	6[2]	9[3]	3	`BITX @R5+,&EDE`
#N	Rm	3	3	3	`BITX #20,R8`
	PC[4]	4	4	3	`ADDX.A #FE000h,PC`
	x(Rm)	6[2]	8[3]	4	`ANDX #1234,4(R6)`
	EDE	6[2]	8[3]	4	`XORX #A5A5h,EDE`
	&EDE	6[2]	8[3]	4	`BITX.B #12,&EDE`
x(Rn)	Rm	4	5	3	`BITX 2(R5),R8`
	PC[4]	6	7	3	`SUBX.A 2(R6),PC`
	TONI	7[2]	10[3]	4	`ANDX 4(R7),4(R6)`
	x(Rm)	7[2]	10[3]	4	`XORX.B 2(R6),EDE`
	&TONI	7[2]	10[3]	4	`BITX 8(SP),&EDE`
EDE	Rm	4	5	3	`BITX.B EDE,R8`
	PC[4]	6	7	3	`ADDX.A EDE,PC`
	TONI	7[2]	10[3]	4	`ANDX EDE,4(R6)`
	x(Rm)	7[2]	10[3]	4	`ANDX EDE,TONI`
	&TONI	7[2]	10[3]	4	`BITX EDE,&TONI`
&EDE	Rm	4	5	3	`BITX &EDE,R8`
	PC[4]	6	7	3	`ADDX.A &EDE,PC`
	TONI	7[2]	10[3]	4	`ANDX.B &EDE,4(R6)`
	x(Rm)	7[2]	10[3]	4	`XORX &EDE,TONI`
	&TONI	7[2]	10[3]	4	`BITX &EDE,&TONI`

(1) Repeat instructions require n + 1 cycles, where n is the number of times the instruction is executed.
(2) Reduce the cycle count by one for MOV, BIT, and CMP instructions.
(3) Reduce the cycle count by two for MOV, BIT, and CMP instructions.
(4) Reduce the cycle count by one for MOV, ADD, and SUB instructions.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.5.2.7.3 MSP430X Address Instruction Cycles and Lengths

Table 4-19 lists the length and the CPU cycles for all addressing modes of the MSP430X address instructions.

Table 4-19. Address Instruction Cycles and Length

Addressing Mode		Execution Time (MCLK Cycles)		Length of Instruction (Words)		Example
Source	Destination	MOVA BRA	CMPA ADDA SUBA	MOVA	CMPA ADDA SUBA	
Rn	Rn	1	1	1	1	`CMPA R5,R8`
	PC	3	3	1	1	`SUBA R9,PC`
	x(Rm)	4	–	2	–	`MOVA R5,4(R6)`
	EDE	4	–	2	–	`MOVA R8,EDE`
	&EDE	4	–	2	–	`MOVA R5,&EDE`
@Rn	Rm	3	–	1	–	`MOVA @R5,R8`
	PC	5	–	1	–	`MOVA @R9,PC`
@Rn+	Rm	3	–	1	–	`MOVA @R5+,R8`
	PC	5	–	1	–	`MOVA @R9+,PC`
#N	Rm	2	3	2	2	`CMPA #20,R8`
	PC	3	3	2	2	`SUBA #FE000h,PC`
x(Rn)	Rm	4	–	2	–	`MOVA 2(R5),R8`
	PC	6	–	2	–	`MOVA 2(R6),PC`
EDE	Rm	4	–	2	–	`MOVA EDE,R8`
	PC	6	–	2	–	`MOVA EDE,PC`
&EDE	Rm	4	–	2	–	`MOVA &EDE,R8`
	PC	6	–	2	–	`MOVA &EDE,PC`

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6 Instruction Set Description

Table 4-20 shows all available instructions:

Table 4-20. Instruction Map of MSP430X

	000	040	080	0C0	100	140	180	1C0	200	240	280	2C0	300	340	380	3C0
0xxx	MOVA, CMPA, ADDA, SUBA, RRCM, RRAM, RLAM, RRUM															
10xx	RRC	RRC.B	SWPB		RRA	RRA.B	SXT		PUSH	PUSH.B	CALL		RETI	CALLA		
14xx	PUSHM.A, POPM.A, PUSHM.W, POPM.W															
18xx 1Cxx	Extension word for Format I and Format II instructions															
20xx	JNE, JNZ															
24xx	JEQ, JZ															
28xx	JNC															
2Cxx	JC															
30xx	JN															
34xx	JGE															
38xx	JL															
3Cxx	JMP															
4xxx	MOV, MOV.B															
5xxx	ADD, ADD.B															
6xxx	ADDC, ADDC.B															
7xxx	SUBC, SUBC.B															
8xxx	SUB, SUB.B															
9xxx	CMP, CMP.B															
Axxx	DADD, DADD.B															
Bxxx	BIT, BIT.B															
Cxxx	BIC, BIC.B															
Dxxx	BIS, BIS.B															
Exxx	XOR, XOR.B															
Fxxx	AND, AND.B															

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.1 Extended Instruction Binary Descriptions

Detailed MSP430X instruction binary descriptions are shown in the following tables.

Instruction	Instruction Group				src or data.19:16	Instruction Identifier				dst	
	15			12	11 8	7			4	3 0	
MOVA	0	0	0	0	src	0	0	0	0	dst	MOVA @Rsrc,Rdst
	0	0	0	0	src	0	0	0	1	dst	MOVA @Rsrc+,Rdst
	0	0	0	0	&abs.19:16	0	0	1	0	dst	MOVA &abs20,Rdst
	&abs.15:0										
	0	0	0	0	src	0	0	1	1	dst	MOVA x(Rsrc),Rdst
	x.15:0										±15-bit index x
	0	0	0	0	src	0	1	1	0	&abs.19:16	MOVA Rsrc,&abs20
	&abs.15:0										
	0	0	0	0	src	0	1	1	1	dst	MOVA Rsrc,X(Rdst)
	x.15:0										±15-bit index x
	0	0	0	0	imm.19:16	1	0	0	0	dst	MOVA #imm20,Rdst
	imm.15:0										
CMPA	0	0	0	0	imm.19:16	1	0	0	1	dst	CMPA #imm20,Rdst
	imm.15:0										
ADDA	0	0	0	0	imm.19:16	1	0	1	0	dst	ADDA #imm20,Rdst
	imm.15:0										
SUBA	0	0	0	0	imm.19:16	1	0	1	1	dst	SUBA #imm20,Rdst
	imm.15:0										
MOVA	0	0	0	0	src	1	1	0	0	dst	MOVA Rsrc,Rdst
CMPA	0	0	0	0	src	1	1	0	1	dst	CMPA Rsrc,Rdst
ADDA	0	0	0	0	src	1	1	1	0	dst	ADDA Rsrc,Rdst
SUBA	0	0	0	0	src	1	1	1	1	dst	SUBA Rsrc,Rdst

Instruction	Instruction Group				Bit Loc.	Inst. ID		Instruction Identifier				dst	
	15			12	11 10	9	8	7			4	3 0	
RRCM.A	0	0	0	0	n – 1	0	0	0	1	0	0	dst	RRCM.A #n,Rdst
RRAM.A	0	0	0	0	n – 1	0	1	0	1	0	0	dst	RRAM.A #n,Rdst
RLAM.A	0	0	0	0	n – 1	1	0	0	1	0	0	dst	RLAM.A #n,Rdst
RRUM.A	0	0	0	0	n – 1	1	1	0	1	0	0	dst	RRUM.A #n,Rdst
RRCM.W	0	0	0	0	n – 1	0	0	0	1	0	1	dst	RRCM.W #n,Rdst
RRAM.W	0	0	0	0	n – 1	0	1	0	1	0	1	dst	RRAM.W #n,Rdst
RLAM.W	0	0	0	0	n – 1	1	0	0	1	0	1	dst	RLAM.W #n,Rdst
RRUM.W	0	0	0	0	n – 1	1	1	0	1	0	1	dst	RRUM.W #n,Rdst

Copyright © 2009–2013, Texas Instruments Incorporated

Instruction	Instruction Identifier												dst				
	15			12	11			8	7	6	5	4	3			0	
RETI	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	
CALLA	0	0	0	1	0	0	1	1	0	1	0	0	dst				CALLA Rdst
	0	0	0	1	0	0	1	1	0	1	0	1	dst				CALLA x(Rdst)
	x.15:0																
	0	0	0	1	0	0	1	1	0	1	1	0	dst				CALLA @Rdst
	0	0	0	1	0	0	1	1	0	1	1	1	dst				CALLA @Rdst+
	0	0	0	1	0	0	1	1	1	0	0	0	&abs.19:16				CALLA &abs20
	&abs.15:0																
	0	0	0	1	0	0	1	1	1	0	0	1	x.19:16				CALLA EDE
	x.15:0																CALLA x(PC)
	0	0	0	1	0	0	1	1	1	0	1	1	imm.19:16				CALLA #imm20
	imm.15:0																
Reserved	0	0	0	1	0	0	1	1	1	0	1	0	x	x	x	x	
Reserved	0	0	0	1	0	0	1	1	1	1	x	x	x	x	x	x	
PUSHM.A	0	0	0	1	0	1	0	0	n – 1				dst				PUSHM.A #n,Rdst
PUSHM.W	0	0	0	1	0	1	0	1	n – 1				dst				PUSHM.W #n,Rdst
POPM.A	0	0	0	1	0	1	1	0	n – 1				dst – n + 1				POPM.A #n,Rdst
POPM.W	0	0	0	1	0	1	1	1	n – 1				dst – n + 1				POPM.W #n,Rdst

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2 MSP430 Instructions

The MSP430 instructions are listed and described on the following pages.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.1 ADC

*** ADC[.W]** Add carry to destination

*** ADC.B** Add carry to destination

Syntax `ADC dst` or `ADC.W dst`

`ADC.B dst`

Operation dst + C → dst

Emulation `ADDC #0,dst`

`ADDC.B #0,dst`

Description The carry bit (C) is added to the destination operand. The previous contents of the destination are lost.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if result is zero, reset otherwise
- C: Set if dst was incremented from 0FFFFh to 0000, reset otherwise
  Set if dst was incremented from 0FFh to 00, reset otherwise
- V: Set if an arithmetic overflow occurs, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 16-bit counter pointed to by R13 is added to a 32-bit counter pointed to by R12.

```
ADD     @R13,0(R12)      ; Add LSDs
ADC     2(R12)           ; Add carry to MSD
```

Example The 8-bit counter pointed to by R13 is added to a 16-bit counter pointed to by R12.

```
ADD.B   @R13,0(R12)     ; Add LSDs
ADC.B   1(R12)          ; Add carry to MSD
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.2 ADD

ADD[.W] Add source word to destination word

ADD.B Add source byte to destination byte

Syntax `ADD src,dst` or `ADD.W src,dst`

`ADD.B src,dst`

Operation src + dst → dst

Description The source operand is added to the destination operand. The previous content of the destination is lost.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if there is a carry from the MSB of the result, reset otherwise

V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Ten is added to the 16-bit counter CNTR located in lower 64 K.

```
ADD.W   #10,&CNTR       ; Add 10 to 16-bit counter
```

Example A table word pointed to by R5 (20-bit address in R5) is added to R6. The jump to label TONI is performed on a carry.

```
ADD.W    @R5,R6         ; Add table word to R6. R6.19:16 = 0
JC       TONI           ; Jump if carry
...                     ; No carry
```

Example A table byte pointed to by R5 (20-bit address) is added to R6. The jump to label TONI is performed if no carry occurs. The table pointer is auto-incremented by 1. R6.19:8 = 0

```
ADD.B    @R5+,R6        ; Add byte to R6. R5 + 1. R6: 000xxh
JNC      TONI           ; Jump if no carry
...                     ; Carry occurred
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.3 ADDC

ADDC[.W] Add source word and carry to destination word

ADDC.B Add source byte and carry to destination byte

Syntax `ADDC src,dst` or `ADDC.W src,dst`

`ADDC.B src,dst`

Operation src + dst + C → dst

Description The source operand and the carry bit C are added to the destination operand. The previous content of the destination is lost.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if there is a carry from the MSB of the result, reset otherwise
- V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Constant value 15 and the carry of the previous instruction are added to the 16-bit counter CNTR located in lower 64 K.

```
ADDC.W    #15,&CNTR     ; Add 15 + C to 16-bit CNTR
```

Example A table word pointed to by R5 (20-bit address) and the carry C are added to R6. The jump to label TONI is performed on a carry. R6.19:16 = 0

```
ADDC.W    @R5,R6        ; Add table word + C to R6
JC        TONI          ; Jump if carry
...                     ; No carry
```

Example A table byte pointed to by R5 (20-bit address) and the carry bit C are added to R6. The jump to label TONI is performed if no carry occurs. The table pointer is auto-incremented by 1. R6.19:8 = 0

```
ADDC.B    @R5+,R6       ; Add table byte + C to R6. R5 + 1
JNC       TONI          ; Jump if no carry
...                     ; Carry occurred
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.4 AND

AND[.W] Logical AND of source word with destination word

AND.B Logical AND of source byte with destination byte

Syntax `AND src,dst` or `AND.W src,dst`

`AND.B src,dst`

Operation src .and. dst → dst

Description The source operand and the destination operand are logically ANDed. The result is placed into the destination. The source operand is not affected.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if the result is not zero, reset otherwise. C = (.not. Z)
- V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The bits set in R5 (16-bit data) are used as a mask (AA55h) for the word TOM located in the lower 64 K. If the result is zero, a branch is taken to label TONI. R5.19:16 = 0

```
MOV     #AA55h,R5     ; Load 16-bit mask to R5
AND     R5,&TOM       ; TOM .and. R5 -> TOM
JZ      TONI          ; Jump if result 0
...                   ; Result > 0
```

or shorter:

```
AND     #AA55h,&TOM   ; TOM .and. AA55h -> TOM
JZ      TONI          ; Jump if result 0
```

Example A table byte pointed to by R5 (20-bit address) is logically ANDed with R6. R5 is incremented by 1 after the fetching of the byte. R6.19:8 = 0

```
AND.B   @R5+,R6       ; AND table byte with R6. R5 + 1
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.5 BIC

BIC[.W] Clear bits set in source word in destination word

BIC.B Clear bits set in source byte in destination byte

Syntax `BIC src,dst` or `BIC.W src,dst`

`BIC.B src,dst`

Operation (.not. src) .and. dst → dst

Description The inverted source operand and the destination operand are logically ANDed. The result is placed into the destination. The source operand is not affected.

Status Bits
- N: Not affected
- Z: Not affected
- C: Not affected
- V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The bits 15:14 of R5 (16-bit data) are cleared. R5.19:16 = 0

```
BIC     #0C000h,R5      ; Clear R5.19:14 bits
```

Example A table word pointed to by R5 (20-bit address) is used to clear bits in R7. R7.19:16 = 0

```
BIC.W   @R5,R7          ; Clear bits in R7 set in @R5
```

Example A table byte pointed to by R5 (20-bit address) is used to clear bits in Port1.

```
BIC.B   @R5,&P1OUT      ; Clear I/O port P1 bits set in @R5
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.6 BIS

BIS[.W] Set bits set in source word in destination word

BIS.B Set bits set in source byte in destination byte

Syntax `BIS src,dst` or `BIS.W src,dst`

`BIS.B src,dst`

Operation src .or. dst → dst

Description The source operand and the destination operand are logically ORed. The result is placed into the destination. The source operand is not affected.

Status Bits
N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Bits 15 and 13 of R5 (16-bit data) are set to one. R5.19:16 = 0

```
BIS     #A000h,R5       ; Set R5 bits
```

Example A table word pointed to by R5 (20-bit address) is used to set bits in R7. R7.19:16 = 0

```
BIS.W   @R5,R7          ; Set bits in R7
```

Example A table byte pointed to by R5 (20-bit address) is used to set bits in Port1. R5 is incremented by 1 afterwards.

```
BIS.B   @R5+,&P1OUT     ; Set I/O port P1 bits. R5 + 1
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.7 BIT

BIT[.W] Test bits set in source word in destination word

BIT.B Test bits set in source byte in destination byte

Syntax `BIT src,dst` or `BIT.W src,dst`

`BIT.B src,dst`

Operation src .and. dst

Description The source operand and the destination operand are logically ANDed. The result affects only the status bits in SR.

Register mode: the register bits Rdst.19:16 (.W) resp. Rdst. 19:8 (.B) are not cleared!

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if the result is not zero, reset otherwise. C = (.not. Z)

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Test if one (or both) of bits 15 and 14 of R5 (16-bit data) is set. Jump to label TONI if this is the case. R5.19:16 are not affected.

```
BIT     #C000h,R5       ; Test R5.15:14 bits
JNZ     TONI            ; At least one bit is set in R5
...                     ; Both bits are reset
```

Example A table word pointed to by R5 (20-bit address) is used to test bits in R7. Jump to label TONI if at least one bit is set. R7.19:16 are not affected.

```
BIT.W   @R5,R7          ; Test bits in R7
JC      TONI            ; At least one bit is set
...                     ; Both are reset
```

Example A table byte pointed to by R5 (20-bit address) is used to test bits in output Port1. Jump to label TONI if no bit is set. The next table byte is addressed.

```
BIT.B   @R5+,&P1OUT     ; Test I/O port P1 bits. R5 + 1
JNC     TONI            ; No corresponding bit is set
...                     ; At least one bit is set
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.8 BR, BRANCH

*** BR, BRANCH**	Branch to destination in lower 64K address space
Syntax	`BR dst`
Operation	dst → PC
Emulation	`MOV dst,PC`
Description	An unconditional branch is taken to an address anywhere in the lower 64K address space. All source addressing modes can be used. The branch instruction is a word instruction.
Status Bits	Status bits are not affected.
Example	Examples for all addressing modes are given.

```
BR      #EXEC    ; Branch to label EXEC or direct branch (for example #0A4h)
                 ; Core instruction MOV @PC+,PC

BR      EXEC     ; Branch to the address contained in EXEC
                 ; Core instruction MOV X(PC),PC
                 ; Indirect address

BR      &EXEC    ; Branch to the address contained in absolute
                 ; address EXEC
                 ; Core instruction MOV X(0),PC
                 ; Indirect address

BR      R5       ; Branch to the address contained in R5
                 ; Core instruction MOV R5,PC
                 ; Indirect R5

BR      @R5      ; Branch to the address contained in the word
                 ; pointed to by R5.
                 ; Core instruction MOV @R5,PC
                 ; Indirect, indirect R5

BR      @R5+     ; Branch to the address contained in the word pointed
                 ; to by R5 and increment pointer in R5 afterwards.
                 ; The next time-S/W flow uses R5 pointer-it can
                 ; alter program execution due to access to
                 ; next address in a table pointed to by R5
                 ; Core instruction MOV @R5,PC
                 ; Indirect, indirect R5 with autoincrement

BR      X(R5)    ; Branch to the address contained in the address
                 ; pointed to by R5 + X (for example table with address
                 ; starting at X). X can be an address or a label
                 ; Core instruction MOV X(R5),PC
                 ; Indirect, indirect R5 + X
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.9 CALL

CALL Call a subroutine in lower 64 K

Syntax `CALL dst`

Operation

dst → tmp 16-bit dst is evaluated and stored

SP – 2 → SP

PC → @SP updated PC with return address to TOS

tmp → PC saved 16-bit dst to PC

Description A subroutine call is made from an address in the lower 64 K to a subroutine address in the lower 64 K. All seven source addressing modes can be used. The call instruction is a word instruction. The return is made with the RET instruction.

Status Bits Status bits are not affected.
PC.19:16 cleared (address in lower 64 K)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Examples Examples for all addressing modes are given.

Immediate Mode: Call a subroutine at label EXEC (lower 64 K) or call directly to address.

```
CALL   #EXEC              ; Start address EXEC
CALL   #0AA04h            ; Start address 0AA04h
```

Symbolic Mode: Call a subroutine at the 16-bit address contained in address EXEC. EXEC is located at the address (PC + X) where X is within PC ± 32 K.

```
CALL   EXEC               ; Start address at @EXEC. z16(PC)
```

Absolute Mode: Call a subroutine at the 16-bit address contained in absolute address EXEC in the lower 64 K.

```
CALL   &EXEC              ; Start address at @EXEC
```

Register mode: Call a subroutine at the 16-bit address contained in register R5.15:0.

```
CALL   R5                 ; Start address at R5
```

Indirect Mode: Call a subroutine at the 16-bit address contained in the word pointed to by register R5 (20-bit address).

```
CALL   @R5                ; Start address at @R5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.10 CLR

*** CLR[.W]** Clear destination

*** CLR.B** Clear destination

Syntax `CLR dst` or `CLR.W dst`
`CLR.B dst`

Operation 0 → dst

Emulation
```
MOV #0,dst
MOV.B #0,dst
```

Description The destination operand is cleared.

Status Bits Status bits are not affected.

Example RAM word TONI is cleared.

```
CLR     TONI      ; 0 -> TONI
```

Example Register R5 is cleared.

```
CLR     R5
```

Example RAM byte TONI is cleared.

```
CLR.B   TONI      ; 0 -> TONI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.11 CLRC

*** CLRC**	Clear carry bit
Syntax	`CLRC`
Operation	0 → C
Emulation	`BIC #1,SR`
Description	The carry bit (C) is cleared. The clear carry instruction is a word instruction.
Status Bits	N: Not affected Z: Not affected C: Cleared V: Not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The 16-bit decimal counter pointed to by R13 is added to a 32-bit counter pointed to by R12.

```
CLRC                    ; C=0: defines start
DADD   @R13,0(R12)      ; add 16-bit counter to low word of 32-bit counter
DADC   2(R12)           ; add carry to high word of 32-bit counter
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.12 CLRN

*** CLRN** Clear negative bit

Syntax `CLRN`

Operation 0 → N

or

(.NOT.src .AND. dst → dst)

Emulation `BIC #4,SR`

Description The constant 04h is inverted (0FFFBh) and is logically ANDed with the destination operand. The result is placed into the destination. The clear negative bit instruction is a word instruction.

Status Bits
- N: Reset to 0
- Z: Not affected
- C: Not affected
- V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The negative bit in the SR is cleared. This avoids special treatment with negative numbers of the subroutine called.

```
            CLRN
            CALL   SUBR
            ......
            ......
SUBR        JN     SUBRET     ; If input is negative: do nothing and return
            ......
            ......
            ......
SUBRET      RET
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.13 CLRZ

*** CLRZ**	Clear zero bit
Syntax	`CLRZ`
Operation	0 → Z or (.NOT.src .AND. dst → dst)
Emulation	`BIC #2,SR`
Description	The constant 02h is inverted (0FFFDh) and logically ANDed with the destination operand. The result is placed into the destination. The clear zero bit instruction is a word instruction.
Status Bits	N: Not affected Z: Reset to 0 C: Not affected V: Not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The zero bit in the SR is cleared.

```
CLRZ
```

Indirect, Auto-Increment mode: Call a subroutine at the 16-bit address contained in the word pointed to by register R5 (20-bit address) and increment the 16-bit address in R5 afterwards by 2. The next time the software uses R5 as a pointer, it can alter the program execution due to access to the next word address in the table pointed to by R5.

```
CALL   @R5+          ; Start address at @R5. R5 + 2
```

Indexed mode: Call a subroutine at the 16-bit address contained in the 20-bit address pointed to by register (R5 + X); for example, a table with addresses starting at X. The address is within the lower 64 KB. X is within ±32 KB.

```
CALL   X(R5)         ; Start address at @(R5+X). z16(R5)
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.14 CMP

CMP[.W] Compare source word and destination word

CMP.B Compare source byte and destination byte

Syntax `CMP src,dst` or `CMP.W src,dst`

`CMP.B src,dst`

Operation (.not.src) + 1 + dst

or

dst – src

Emulation `BIC #2,SR`

Description The source operand is subtracted from the destination operand. This is made by adding the 1s complement of the source + 1 to the destination. The result affects only the status bits in SR.

Register mode: the register bits Rdst.19:16 (.W) resp. Rdst. 19:8 (.B) are not cleared.

Status Bits

N: Set if result is negative (src > dst), reset if positive (src = dst)

Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow).

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Compare word EDE with a 16-bit constant 1800h. Jump to label TONI if EDE equals the constant. The address of EDE is within PC + 32 K.

```
CMP     #01800h,EDE     ; Compare word EDE with 1800h
JEQ     TONI            ; EDE contains 1800h
...                     ; Not equal
```

Example A table word pointed to by (R5 + 10) is compared with R7. Jump to label TONI if R7 contains a lower, signed 16-bit number. R7.19:16 is not cleared. The address of the source operand is a 20-bit address in full memory range.

```
CMP.W   10(R5),R7       ; Compare two signed numbers
JL      TONI            ; R7 < 10(R5)
...                     ; R7 >= 10(R5)
```

Example A table byte pointed to by R5 (20-bit address) is compared to the value in output Port1. Jump to label TONI if values are equal. The next table byte is addressed.

```
CMP.B   @R5+,&P1OUT     ; Compare P1 bits with table. R5 + 1
JEQ     TONI            ; Equal contents
...                     ; Not equal
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.15 DADC

*** DADC[.W]** Add carry decimally to destination

*** DADC.B** Add carry decimally to destination

Syntax `DADC dst` or `DADC.W dst`

`DADC.B dst`

Operation dst + C → dst (decimally)

Emulation `DADD #0,dst`

`DADD.B #0,dst`

Description The carry bit (C) is added decimally to the destination.

Status Bits
- N: Set if MSB is 1
- Z: Set if dst is 0, reset otherwise
- C: Set if destination increments from 9999 to 0000, reset otherwise
  Set if destination increments from 99 to 00, reset otherwise
- V: Undefined

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The four-digit decimal number contained in R5 is added to an eight-digit decimal number pointed to by R8.

```
CLRC                ; Reset carry
                    ; next instruction's start condition is defined
DADD   R5,0(R8)     ; Add LSDs + C
DADC   2(R8)        ; Add carry to MSD
```

Example The two-digit decimal number contained in R5 is added to a four-digit decimal number pointed to by R8.

```
CLRC                ; Reset carry
                    ; next instruction's start condition is defined
DADD.B   R5,0(R8)   ; Add LSDs + C
DADC     1(R8)      ; Add carry to MSDs
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.16 DADD

*** DADD[.W]** Add source word and carry decimally to destination word

*** DADD.B** Add source byte and carry decimally to destination byte

Syntax `DADD src,dst` or `DADD.W src,dst`

`DADD.B src,dst`

Operation src + dst + C → dst (decimally)

Description The source operand and the destination operand are treated as two (.B) or four (.W) binary coded decimals (BCD) with positive signs. The source operand and the carry bit C are added decimally to the destination operand. The source operand is not affected. The previous content of the destination is lost. The result is not defined for non-BCD numbers.

Status Bits

N: Set if MSB of result is 1 (word > 7999h, byte > 79h), reset if MSB is 0

Z: Set if result is zero, reset otherwise

C: Set if the BCD result is too large (word > 9999h, byte > 99h), reset otherwise

V: Undefined

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Decimal 10 is added to the 16-bit BCD counter DECCNTR.

```
DADD   #10h,&DECCNTR   ; Add 10 to 4-digit BCD counter
```

Example The eight-digit BCD number contained in 16-bit RAM addresses BCD and BCD+2 is added decimally to an eight-digit BCD number contained in R4 and R5 (BCD+2 and R5 contain the MSDs). The carry C is added, and cleared.

```
CLRC                   ; Clear carry
DADD.W   &BCD,R4       ; Add LSDs. R4.19:16 = 0
DADD.W   &BCD+2,R5     ; Add MSDs with carry. R5.19:16 = 0
JC       OVERFLOW      ; Result >9999,9999: go to error routine
...                    ; Result ok
```

Example The two-digit BCD number contained in word BCD (16-bit address) is added decimally to a two-digit BCD number contained in R4. The carry C is added, also. R4.19:8 = 0

```
CLRC                   ; Clear carry
DADD.B   &BCD,R4       ; Add BCD to R4 decimally.
                         R4: 0,00ddh
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.17 DEC

*** DEC[.W]**	Decrement destination
*** DEC.B**	Decrement destination
Syntax	`DEC dst` or `DEC.W dst` `DEC.B dst`
Operation	dst – 1 → dst
Emulation	`SUB #1,dst` `SUB.B #1,dst`
Description	The destination operand is decremented by one. The original contents are lost.
Status Bits	N: Set if result is negative, reset if positive Z: Set if dst contained 1, reset otherwise C: Reset if dst contained 0, set otherwise V: Set if an arithmetic overflow occurs, otherwise reset. Set if initial value of destination was 08000h, otherwise reset. Set if initial value of destination was 080h, otherwise reset.
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	R10 is decremented by 1.

```
        DEC     R10                 ; Decrement R10

; Move a block of 255 bytes from memory location starting with EDE to
; memory location starting with TONI. Tables should not overlap: start of
; destination address TONI must not be within the range EDE to EDE+0FEh

        MOV     #EDE,R6
        MOV     #255,R10
L$1     MOV.B   @R6+,TONI-EDE-1(R6)
        DEC     R10
        JNZ     L$1
```

Do not transfer tables using the routine above with the overlap shown in Figure 4-36.

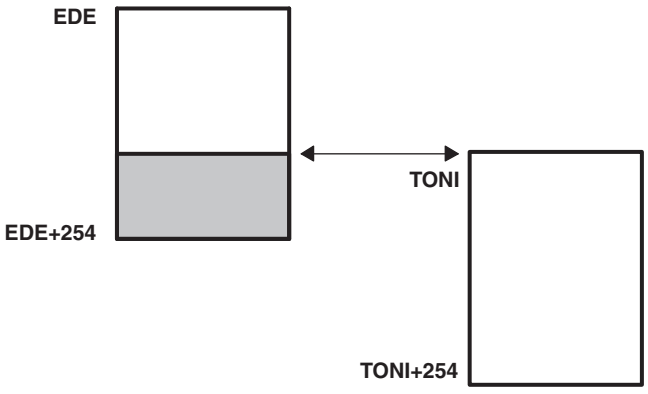


Figure 4-36. Decrement Overlap

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.18 DECD

*** DECD[.W]** Double-decrement destination

*** DECD.B** Double-decrement destination

Syntax `DECD dst` or `DECD.W dst`
`DECD.B dst`

Operation dst – 2 → dst

Emulation
```
SUB #2,dst
SUB.B #2,dst
```

Description The destination operand is decremented by two. The original contents are lost.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if dst contained 2, reset otherwise
- C: Reset if dst contained 0 or 1, set otherwise
- V: Set if an arithmetic overflow occurs, otherwise reset
  Set if initial value of destination was 08001 or 08000h, otherwise reset
  Set if initial value of destination was 081 or 080h, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example R10 is decremented by 2.

```
        DECD    R10             ; Decrement R10 by two
```

```
; Move a block of 255 bytes from memory location starting with EDE to
; memory location starting with TONI.
; Tables should not overlap: start of destination address TONI must not
; be within the range EDE to EDE+0FEh

        MOV     #EDE,R6
        MOV     #255,R10
L$1     MOV.B   @R6+,TONI-EDE-2(R6)
        DECD    R10
        JNZ     L$1
```

Example Memory at location LEO is decremented by two.

```
        DECD.B  LEO             ; Decrement MEM(LEO)
```

Decrement status byte STATUS by two

```
        DECD.B  STATUS
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.19 DINT

*** DINT**	Disable (general) interrupts
Syntax	`DINT`
Operation	0 → GIE or (0FFF7h .AND. SR → SR / .NOT.src .AND. dst → dst)
Emulation	`BIC #8,SR`
Description	All interrupts are disabled. The constant 08h is inverted and logically ANDed with the SR. The result is placed into the SR.
Status Bits	Status bits are not affected.
Mode Bits	GIE is reset. OSCOFF and CPUOFF are not affected.
Example	The general interrupt enable (GIE) bit in the SR is cleared to allow a nondisrupted move of a 32-bit counter. This ensures that the counter is not modified during the move by any interrupt.

```
DINT                ; All interrupt events using the GIE bit are disabled
NOP
MOV    COUNTHI,R5   ; Copy counter
MOV    COUNTLO,R6
EINT                ; All interrupt events using the GIE bit are enabled
```

NOTE: Disable interrupt

If any code sequence needs to be protected from interruption, DINT should be executed at least one instruction before the beginning of the uninterruptible sequence, or it should be followed by a NOP instruction.

NOTE: Enable and Disable Interrupt

Due to the pipelined CPU architecture, the instruction following the enable interrupt instruction (EINT) is always executed, even if an interrupt service request is pending when the interrupts are enabled.

If the enable interrupt instruction (EINT) is immediately followed by a disable interrupt instruction (DINT), a pending interrupt might not be serviced. Further instructions after DINT might execute incorrectly and result in unexpected CPU execution. It is recommended to always insert at least one instruction between EINT and DINT. Note that any alternative instruction use that sets and immediately clears the CPU status register GIE bit must be considered in the same fashion.

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.20 EINT

*** EINT**	Enable (general) interrupts
Syntax	`EINT`
Operation	1 → GIE or (0008h .OR. SR → SR / .src .OR. dst → dst)
Emulation	`BIS #8,SR`
Description	All interrupts are enabled. The constant #08h and the SR are logically ORed. The result is placed into the SR.
Status Bits	Status bits are not affected.
Mode Bits	GIE is set. OSCOFF and CPUOFF are not affected.
Example	The general interrupt enable (GIE) bit in the SR is set.

```
            PUSH.B   &P1IN
            BIC.B    @SP,&P1IFG  ; Reset only accepted flags
            EINT                 ; Preset port 1 interrupt flags stored on stack
                                 ; other interrupts are allowed
            BIT   #Mask,@SP
            JEQ   MaskOK         ; Flags are present identically to mask: jump
            ......
MaskOK      BIC   #Mask,@SP
            ......
            INCD   SP            ; Housekeeping: inverse to PUSH instruction
                                 ; at the start of interrupt subroutine. Corrects
                                 ; the stack pointer.
            RETI
```

NOTE: Enable and Disable Interrupt

Due to the pipelined CPU architecture, the instruction following the enable interrupt instruction (EINT) is always executed, even if an interrupt service request is pending when the interrupts are enabled.

If the enable interrupt instruction (EINT) is immediately followed by a disable interrupt instruction (DINT), a pending interrupt might not be serviced. Further instructions after DINT might execute incorrectly and result in unexpected CPU execution. It is recommended to always insert at least one instruction between EINT and DINT. Note that any alternative instruction use that sets and immediately clears the CPU status register GIE bit must be considered in the same fashion.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.21 INC

*** INC[.W]**	Increment destination
*** INC.B**	Increment destination
Syntax	`INC dst` or `INC.W dst` `INC.B dst`
Operation	dst + 1 → dst
Emulation	`ADD #1,dst`
Description	The destination operand is incremented by one. The original contents are lost.
Status Bits	N: Set if result is negative, reset if positive Z: Set if dst contained 0FFFFh, reset otherwise Set if dst contained 0FFh, reset otherwise C: Set if dst contained 0FFFFh, reset otherwise Set if dst contained 0FFh, reset otherwise V: Set if dst contained 07FFFh, reset otherwise Set if dst contained 07Fh, reset otherwise
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The status byte, STATUS, of a process is incremented. When it is equal to 11, a branch to OVFL is taken.

```
INC.B   STATUS
CMP.B   #11,STATUS
JEQ     OVFL
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.22 INCD

*** INCD[.W]** Double-increment destination

*** INCD.B** Double-increment destination

Syntax `INCD dst` or `INCD.W dst`

`INCD.B dst`

Operation dst + 2 → dst

Emulation `ADD #2,dst`

Description The destination operand is incremented by two. The original contents are lost.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if dst contained 0FFFEh, reset otherwise
  Set if dst contained 0FEh, reset otherwise
- C: Set if dst contained 0FFFEh or 0FFFFh, reset otherwise
  Set if dst contained 0FEh or 0FFh, reset otherwise
- V: Set if dst contained 07FFEh or 07FFFh, reset otherwise
  Set if dst contained 07Eh or 07Fh, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The item on the top of the stack (TOS) is removed without using a register.

```
.......
PUSH    R5        ; R5 is the result of a calculation, which is stored
                  ; in the system stack
INCD    SP        ; Remove TOS by double-increment from stack
                  ; Do not use INCD.B, SP is a word-aligned register
RET
```

Example The byte on the top of the stack is incremented by two.

```
INCD.B   0(SP)   ; Byte on TOS is increment by two
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.23 INV

*** INV[.W]** Invert destination

*** INV.B** Invert destination

Syntax `INV dst` or `INV.W dst`

`INV.B dst`

Operation .not.dst → dst

Emulation `XOR #0FFFFh,dst`

`XOR.B #0FFh,dst`

Description The destination operand is inverted. The original contents are lost.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if dst contained 0FFFFh, reset otherwise
  Set if dst contained 0FFh, reset otherwise
- C: Set if result is not zero, reset otherwise (= .NOT. Zero)
- V: Set if initial destination operand was negative, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Content of R5 is negated (2s complement).

```
MOV     #00AEh,R5    ;                      R5 = 000AEh
INV     R5           ; Invert R5,           R5 = 0FF51h
INC     R5           ; R5 is now negated,   R5 = 0FF52h
```

Example Content of memory byte LEO is negated.

```
MOV.B   #0AEh,LEO    ;                      MEM(LEO) = 0AEh
INV.B   LEO          ; Invert LEO,          MEM(LEO) = 051h
INC.B   LEO          ; MEM(LEO) is negated, MEM(LEO) = 052h
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.24 JC, JHS

JC Jump if carry

JHS Jump if higher or same (unsigned)

Syntax

```
JC label
JHS label
```

Operation

If C = 1: PC + (2 × Offset) → PC
If C = 0: execute the following instruction

Description The carry bit C in the SR is tested. If it is set, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range –511 to +512 words relative to the PC in the full memory range. If C is reset, the instruction after the jump is executed.

JC is used for the test of the carry bit C.

JHS is used for the comparison of unsigned numbers.

Status Bits Status bits are not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The state of the port 1 pin P1IN.1 bit defines the program flow.

```
BIT.B   #2,&P1IN      ; Port 1, bit 1 set? Bit -> C
JC      Label1        ; Yes, proceed at Label1
...                   ; No, continue
```

Example If R5 ≥ R6 (unsigned), the program continues at Label2.

```
CMP     R6,R 5        ; Is R5 >= R6? Info to C
JHS     Label2        ; Yes, C = 1
...                   ; No, R5 < R6. Continue
```

Example If R5 ≥ 12345h (unsigned operands), the program continues at Label2.

```
CMPA    #12345h,R5    ; Is R5 >= 12345h? Info to C
JHS     Label2        ; Yes, 12344h < R5 <= F,FFFFh. C = 1
...                   ; No, R5 < 12345h. Continue
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.25 JEQ, JZ

JEQ Jump if equal

JZ Jump if zero

Syntax
```
JEQ label
JZ label
```

Operation If Z = 1: PC + (2 × Offset) → PC
If Z = 0: execute following instruction

Description The zero bit Z in the SR is tested. If it is set, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range –511 to +512 words relative to the PC in the full memory range. If Z is reset, the instruction after the jump is executed.

JZ is used for the test of the zero bit Z.

JEQ is used for the comparison of operands.

Status Bits Status bits are not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The state of the P2IN.0 bit defines the program flow.

```
BIT.B   #1,&P2IN     ; Port 2, bit 0 reset?
JZ      Label1       ; Yes, proceed at Label1
...                  ; No, set, continue
```

Example If R5 = 15000h (20-bit data), the program continues at Label2.

```
CMPA    #15000h,R5   ; Is R5 = 15000h? Info to SR
JEQ     Label2       ; Yes, R5 = 15000h. Z = 1
...                  ; No, R5 not equal 15000h. Continue
```

Example R7 (20-bit counter) is incremented. If its content is zero, the program continues at Label4.

```
ADDA    #1,R7        ; Increment R7
JZ      Label4       ; Zero reached: Go to Label4
...                  ; R7 not equal 0. Continue here.
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.26 JGE

JGE Jump if greater or equal (signed)

Syntax `JGE label`

Operation If (N .xor. V) = 0: PC + (2 × Offset) → PC
If (N .xor. V) = 1: execute following instruction

Description The negative bit N and the overflow bit V in the SR are tested. If both bits are set or both are reset, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range -511 to +512 words relative to the PC in full Memory range. If only one bit is set, the instruction after the jump is executed.

JGE is used for the comparison of signed operands: also for incorrect results due to overflow, the decision made by the JGE instruction is correct.

Note that JGE emulates the nonimplemented JP (jump if positive) instruction if used after the instructions AND, BIT, RRA, SXTX, and TST. These instructions clear the V bit.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example If byte EDE (lower 64 K) contains positive data, go to Label1. Software can run in the full memory range.

```
TST.B   &EDE            ; Is EDE positive? V <- 0
JGE     Label1          ; Yes, JGE emulates JP
...                     ; No, 80h <= EDE <= FFh
```

Example If the content of R6 is greater than or equal to the memory pointed to by R7, the program continues a Label5. Signed data. Data and program in full memory range.

```
CMP     @R7,R6          ; Is R6 >= @R7?
JGE     Label5          ; Yes, go to Label5
...                     ; No, continue here
```

Example If R5 ≥ 12345h (signed operands), the program continues at Label2. Program in full memory range.

```
CMPA    #12345h,R5      ; Is R5 >= 12345h?
JGE     Label2          ; Yes, 12344h < R5 <= 7FFFFh
...                     ; No, 80000h <= R5 < 12345h
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.27 JL

JL Jump if less (signed)

Syntax `JL label`

Operation If (N .xor. V) = 1: PC + (2 × Offset) → PC
If (N .xor. V) = 0: execute following instruction

Description The negative bit N and the overflow bit V in the SR are tested. If only one is set, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range –511 to +512 words relative to the PC in full memory range. If both bits N and V are set or both are reset, the instruction after the jump is executed.

JL is used for the comparison of signed operands: also for incorrect results due to overflow, the decision made by the JL instruction is correct.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example If byte EDE contains a smaller, signed operand than byte TONI, continue at Label1. The address EDE is within PC ± 32 K.

```
CMP.B   &TONI,EDE     ; Is EDE < TONI
JL      Label1        ; Yes
...                   ; No, TONI <= EDE
```

Example If the signed content of R6 is less than the memory pointed to by R7 (20-bit address), the program continues at Label5. Data and program in full memory range.

```
CMP     @R7,R6        ; Is R6 < @R7?
JL      Label5        ; Yes, go to Label5
...                   ; No, continue here
```

Example If R5 < 12345h (signed operands), the program continues at Label2. Data and program in full memory range.

```
CMPA    #12345h,R5    ; Is R5 < 12345h?
JL      Label2        ; Yes, 80000h =< R5 < 12345h
...                   ; No, 12344h < R5 <= 7FFFFh
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.28 JMP

JMP	Jump unconditionally
Syntax	`JMP label`
Operation	PC + (2 × Offset) → PC
Description	The signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means an unconditional jump in the range –511 to +512 words relative to the PC in the full memory. The JMP instruction may be used as a BR or BRA instruction within its limited range relative to the PC.
Status Bits	Status bits are not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The byte STATUS is set to 10. Then a jump to label MAINLOOP is made. Data in lower 64 K, program in full memory range.

```
MOV.B   #10,&STATUS     ; Set STATUS to 10
JMP     MAINLOOP        ; Go to main loop
```

Example The interrupt vector TAIV of Timer_A3 is read and used for the program flow. Program in full memory range, but interrupt handlers always starts in lower 64 K.

```
ADD     &TAIV,PC        ; Add Timer_A interrupt vector to PC
RETI                    ; No Timer_A interrupt pending
JMP     IHCCR1          ; Timer block 1 caused interrupt
JMP     IHCCR2          ; Timer block 2 caused interrupt
RETI                    ; No legal interrupt, return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.29 JN

JN Jump if negative

Syntax `JN label`

Operation If N = 1: PC + (2 × Offset) → PC
If N = 0: execute following instruction

Description The negative bit N in the SR is tested. If it is set, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit program PC. This means a jump in the range -511 to +512 words relative to the PC in the full memory range. If N is reset, the instruction after the jump is executed.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The byte COUNT is tested. If it is negative, program execution continues at Label0. Data in lower 64 K, program in full memory range.

```
TST.B   &COUNT      ; Is byte COUNT negative?
JN      Label0      ; Yes, proceed at Label0
...                 ; COUNT >= 0
```

Example R6 is subtracted from R5. If the result is negative, program continues at Label2. Program in full memory range.

```
SUB     R6,R5       ; R5 - R6 -> R5
JN      Label2      ; R5 is negative: R6 > R5 (N = 1)
...                 ; R5 >= 0. Continue here.
```

Example R7 (20-bit counter) is decremented. If its content is below zero, the program continues at Label4. Program in full memory range.

```
SUBA    #1,R7       ; Decrement R7
JN      Label4      ; R7 < 0: Go to Label4
...                 ; R7 >= 0. Continue here.
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.30 JNC, JLO

JNC Jump if no carry

JLO Jump if lower (unsigned)

Syntax

```
JNC label
JLO label
```

Operation If C = 0: PC + (2 × Offset) → PC
If C = 1: execute following instruction

Description The carry bit C in the SR is tested. If it is reset, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range –511 to +512 words relative to the PC in the full memory range. If C is set, the instruction after the jump is executed.

JNC is used for the test of the carry bit C.

JLO is used for the comparison of unsigned numbers.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example If byte EDE < 15, the program continues at Label2. Unsigned data. Data in lower 64 K, program in full memory range.

```
CMP.B   #15,&EDE     ; Is EDE < 15? Info to C
JLO     Label2       ; Yes, EDE < 15. C = 0
...                  ; No, EDE >= 15. Continue
```

Example The word TONI is added to R5. If no carry occurs, continue at Label0. The address of TONI is within PC ± 32 K.

```
ADD     TONI,R5      ; TONI + R5 -> R5. Carry -> C
JNC     Label0       ; No carry
...                  ; Carry = 1: continue here
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.31 JNZ, JNE

JNZ Jump if not zero

JNE Jump if not equal

Syntax

```
JNZ label
JNE label
```

Operation If Z = 0: PC + (2 × Offset) → PC
If Z = 1: execute following instruction

Description The zero bit Z in the SR is tested. If it is reset, the signed 10-bit word offset contained in the instruction is multiplied by two, sign extended, and added to the 20-bit PC. This means a jump in the range –511 to +512 words relative to the PC in the full memory range. If Z is set, the instruction after the jump is executed.

JNZ is used for the test of the zero bit Z.

JNE is used for the comparison of operands.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The byte STATUS is tested. If it is not zero, the program continues at Label3. The address of STATUS is within PC ± 32 K.

```
TST.B   STATUS          ; Is STATUS = 0?
JNZ     Label3          ; No, proceed at Label3
...                     ; Yes, continue here
```

Example If word EDE ≠ 1500, the program continues at Label2. Data in lower 64 K, program in full memory range.

```
CMP     #1500,&EDE      ; Is EDE = 1500? Info to SR
JNE     Label2          ; No, EDE not equal 1500.
...                     ; Yes, R5 = 1500. Continue
```

Example R7 (20-bit counter) is decremented. If its content is not zero, the program continues at Label4. Program in full memory range.

```
SUBA    #1,R7           ; Decrement R7
JNZ     Label4          ; Zero not reached: Go to Label4
...                     ; Yes, R7 = 0. Continue here.
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.32 MOV

MOV[.W] Move source word to destination word

MOV.B Move source byte to destination byte

Syntax `MOV src,dst` or `MOV.W src,dst`

`MOV.B src,dst`

Operation src → dst

Description The source operand is copied to the destination. The source operand is not affected.

Status Bits
- N: Not affected
- Z: Not affected
- C: Not affected
- V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Move a 16-bit constant 1800h to absolute address-word EDE (lower 64 K)

```
MOV     #01800h,&EDE            ; Move 1800h to EDE
```

Example The contents of table EDE (word data, 16-bit addresses) are copied to table TOM. The length of the tables is 030h words. Both tables reside in the lower 64 K.

```
        MOV     #EDE,R10                ; Prepare pointer (16-bit address)
Loop    MOV     @R10+,TOM-EDE-2(R10)    ; R10 points to both tables.
                                        ; R10+2
        CMP     #EDE+60h,R10            ; End of table reached?
        JLO     Loop                    ; Not yet
        ...                             ; Copy completed
```

Example The contents of table EDE (byte data, 16-bit addresses) are copied to table TOM. The length of the tables is 020h bytes. Both tables may reside in full memory range, but must be within R10 ± 32 K.

```
        MOVA    #EDE,R10                ; Prepare pointer (20-bit)
        MOV     #20h,R9                 ; Prepare counter
Loop    MOV.B   @R10+,TOM-EDE-1(R10)    ; R10 points to both tables.
                                        ; R10+1
        DEC     R9                      ; Decrement counter
        JNZ     Loop                    ; Not yet done
        ...                             ; Copy completed
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.33 NOP

*** NOP**	No operation
Syntax	`NOP`
Operation	None
Emulation	`MOV #0, R3`
Description	No operation is performed. The instruction may be used for the elimination of instructions during the software check or for defined waiting times.
Status Bits	Status bits are not affected.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.34 POP

*** POP[.W]** Pop word from stack to destination

*** POP.B** Pop byte from stack to destination

Syntax
`POP dst`
`POP.B dst`

Operation
@SP → temp
SP + 2 → SP
temp → dst

Emulation `MOV @SP+,dst` or `MOV.W @SP+,dst`
`MOV.B @SP+,dst`

Description The stack location pointed to by the SP (TOS) is moved to the destination. The SP is incremented by two afterwards.

Status Bits Status bits are not affected.

Example The contents of R7 and the SR are restored from the stack.

```
POP     R7      ; Restore R7
POP     SR      ; Restore status register
```

Example The contents of RAM byte LEO is restored from the stack.

```
POP.B   LEO     ; The low byte of the stack is moved to LEO.
```

Example The contents of R7 is restored from the stack.

```
POP.B   R7      ; The low byte of the stack is moved to R7,
                ; the high byte of R7 is 00h
```

Example The contents of the memory pointed to by R7 and the SR are restored from the stack.

```
POP.B   0(R7)   ; The low byte of the stack is moved to the
                ; the byte which is pointed to by R7
                : Example:   R7 = 203h
                ;            Mem(R7) = low byte of system stack
                : Example:   R7 = 20Ah
                ;            Mem(R7) = low byte of system stack
POP     SR      ; Last word on stack moved to the SR
```

NOTE: System stack pointer

The system SP is always incremented by two, independent of the byte suffix.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.35 PUSH

PUSH[.W] Save a word on the stack

PUSH.B Save a byte on the stack

Syntax `PUSH dst` or `PUSH.W dst`

`PUSH.B dst`

Operation SP – 2 → SP
dst → @SP

Description The 20-bit SP SP is decremented by two. The operand is then copied to the RAM word addressed by the SP. A pushed byte is stored in the low byte; the high byte is not affected.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Save the two 16-bit registers R9 and R10 on the stack

```
PUSH    R9      ; Save R9 and R10 XXXXh
PUSH    R10     ; YYYYh
```

Example Save the two bytes EDE and TONI on the stack. The addresses EDE and TONI are within PC ± 32 K.

```
PUSH.B  EDE     ; Save EDE   xxXXh
PUSH.B  TONI    ; Save TONI   xxYYh
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.36 RET

*** RET**	Return from subroutine
Syntax	RET
Operation	@SP →PC.15:0 Saved PC to PC.15:0. PC.19:16 ← 0 SP + 2 → SP
Description	The 16-bit return address (lower 64 K), pushed onto the stack by a CALL instruction is restored to the PC. The program continues at the address following the subroutine call. The four MSBs of the PC.19:16 are cleared.
Status Bits	Status bits are not affected. PC.19:16: Cleared
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	Call a subroutine SUBR in the lower 64 K and return to the address in the lower 64 K after the CALL.

```
        CALL    #SUBR     ; Call subroutine starting at SUBR
        ...               ; Return by RET to here
SUBR    PUSH    R14       ; Save R14 (16 bit data)
        ...               ; Subroutine code
        POP     R14       ; Restore R14
        RET               ; Return to lower 64 K
```

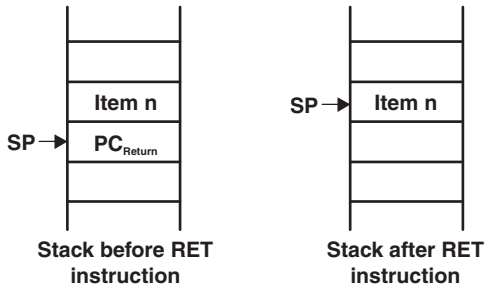


Figure 4-37. Stack After a RET Instruction

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.37 RETI

RETI	Return from interrupt
Syntax	`RETI`
Operation	@SP → SR.15:0 Restore saved SR with PC.19:16 SP + 2 → SP @SP → PC.15:0 Restore saved PC.15:0 SP + 2 → SP Housekeeping
Description	The SR is restored to the value at the beginning of the interrupt service routine. This includes the four MSBs of the PC.19:16. The SP is incremented by two afterward. The 20-bit PC is restored from PC.19:16 (from same stack location as the status bits) and PC.15:0. The 20-bit PC is restored to the value at the beginning of the interrupt service routine. The program continues at the address following the last executed instruction when the interrupt was granted. The SP is incremented by two afterward.
Status Bits	N: Restored from stack C: Restored from stack Z: Restored from stack V: Restored from stack
Mode Bits	OSCOFF, CPUOFF, and GIE are restored from stack.
Example	Interrupt handler in the lower 64 K. A 20-bit return address is stored on the stack.

```
INTRPT  PUSHM.A   #2,R14    ; Save R14 and R13 (20-bit data)
        ...                 ; Interrupt handler code
        POPM.A    #2,R14    ; Restore R13 and R14 (20-bit data)
        RETI                ; Return to 20-bit address in full memory range
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.38 RLA

*** RLA[.W]** Rotate left arithmetically

*** RLA.B** Rotate left arithmetically

Syntax
```
RLA dst or          RLA.W dst
RLA.B dst
```

Operation C ← MSB ← MSB-1 .... LSB+1 ← LSB ← 0

Emulation
```
ADD dst,dst
ADD.B dst,dst
```

Description The destination operand is shifted left one position as shown in Figure 4-38. The MSB is shifted into the carry bit (C) and the LSB is filled with 0. The RLA instruction acts as a signed multiplication by 2.

An overflow occurs if dst ≥ 04000h and dst < 0C000h before operation is performed; the result has changed sign.

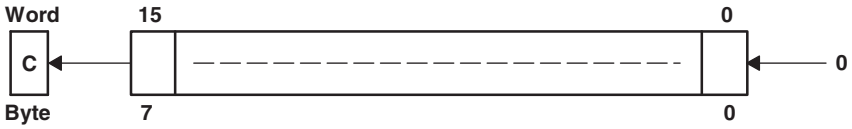


Figure 4-38. Destination Operand—Arithmetic Shift Left

An overflow occurs if dst ≥ 040h and dst < 0C0h before the operation is performed; the result has changed sign.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if result is zero, reset otherwise
- C: Loaded from the MSB
- V: Set if an arithmetic overflow occurs; the initial value is 04000h ≤ dst < 0C000h, reset otherwise
  Set if an arithmetic overflow occurs; the initial value is 040h ≤ dst < 0C0h, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example R7 is multiplied by 2.

```
RLA     R7    ; Shift left R7  (x 2)
```

Example The low byte of R7 is multiplied by 4.

```
RLA.B   R7    ; Shift left low byte of R7  (x 2)
RLA.B   R7    ; Shift left low byte of R7  (x 4)
```

NOTE: RLA substitution

The assembler does not recognize the instructions:

```
RLA  @R5+          RLA.B  @R5+          RLA(.B) @R5
```

They must be substituted by:

```
ADD  @R5+,-2(R5)   ADD.B  @R5+,-1(R5)   ADD(.B) @R5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.39 RLC

*** RLC[.W]** Rotate left through carry

*** RLC.B** Rotate left through carry

Syntax `RLC dst` or `RLC.W dst`

`RLC.B dst`

Operation C ← MSB ← MSB-1 .... LSB+1 ← LSB ← C

Emulation `ADDC dst,dst`

Description The destination operand is shifted left one position as shown in Figure 4-39. The carry bit (C) is shifted into the LSB, and the MSB is shifted into the carry bit (C).

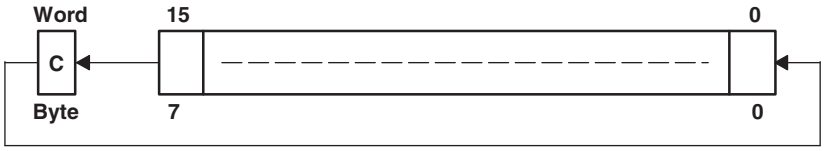


Figure 4-39. Destination Operand—Carry Left Shift

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if result is zero, reset otherwise
- C: Loaded from the MSB
- V: Set if an arithmetic overflow occurs; the initial value is 04000h ≤ dst < 0C000h, reset otherwise
  Set if an arithmetic overflow occurs; the initial value is 040h ≤ dst < 0C0h, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example R5 is shifted left one position.

```
RLC     R5          ; (R5 x 2) + C -> R5
```

Example The input P1IN.1 information is shifted into the LSB of R5.

```
BIT.B   #2,&P1IN    ; Information -> Carry
RLC     R5          ; Carry=P0in.1 -> LSB of R5
```

Example The MEM(LEO) content is shifted left one position.

```
RLC.B   LEO         ; Mem(LEO) x 2 + C -> Mem(LEO)
```

NOTE: RLA substitution

The assembler does not recognize the instructions:

`RLC @R5+` `RLC.B @R5+` `RLC(.B) @R5`

They must be substituted by:

`ADDC @R5+,-2(R5)` `ADDC.B @R5+,-1(R5)` `ADDC(.B) @R5`

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.40 RRA

RRA[.W] Rotate right arithmetically destination word

RRA.B Rotate right arithmetically destination byte

Syntax `RRA.B dst` or `RRA.W dst`

Operation MSB → MSB → MSB–1 → ... LSB+1 → LSB → C

Description The destination operand is shifted right arithmetically by one bit position as shown in Figure 4-40. The MSB retains its value (sign). RRA operates equal to a signed division by 2. The MSB is retained and shifted into the MSB–1. The LSB+1 is shifted into the LSB. The previous LSB is shifted into the carry bit C.

Status Bits
- N: Set if result is negative (MSB = 1), reset otherwise (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Loaded from the LSB
- V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The signed 16-bit number in R5 is shifted arithmetically right one position.

```
RRA     R5              ; R5/2 -> R5
```

Example The signed RAM byte EDE is shifted arithmetically right one position.

```
RRA.B   EDE             ; EDE/2 -> EDE
```

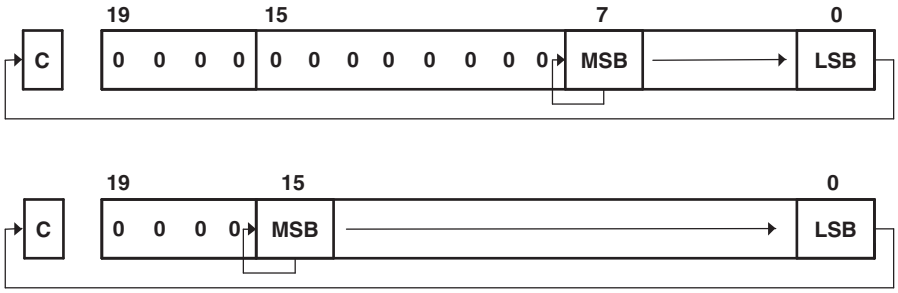


Figure 4-40. Rotate Right Arithmetically RRA.B and RRA.W

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.41 RRC

RRC[.W] Rotate right through carry destination word

RRC.B Rotate right through carry destination byte

Syntax `RRC dst` or `RRC.W dst`

`RRC.B dst`

Operation C → MSB → MSB–1 → ... LSB+1 → LSB → C

Description The destination operand is shifted right by one bit position as shown in Figure 4-41. The carry bit C is shifted into the MSB and the LSB is shifted into the carry bit C.

Status Bits
- N: Set if result is negative (MSB = 1), reset otherwise (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Loaded from the LSB
- V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM word EDE is shifted right one bit position. The MSB is loaded with 1.

```
SETC            ; Prepare carry for MSB
RRC     EDE     ; EDE = EDE >> 1 + 8000h
```

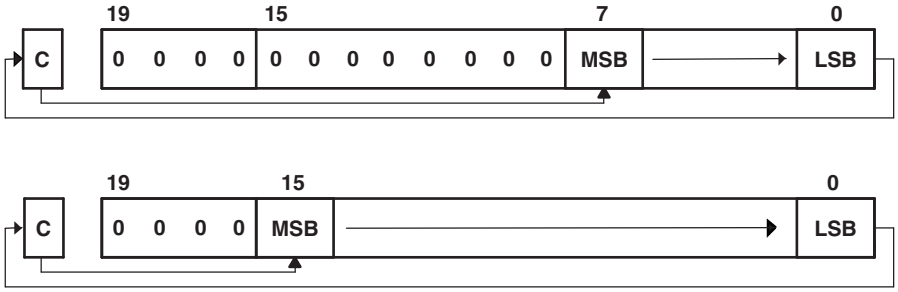


Figure 4-41. Rotate Right Through Carry RRC.B and RRC.W

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.42 SBC

*** SBC[.W]** Subtract borrow (.NOT. carry) from destination

*** SBC.B** Subtract borrow (.NOT. carry) from destination

Syntax `SBC dst` or `SBC.W dst`

`SBC.B dst`

Operation dst + 0FFFFh + C → dst

dst + 0FFh + C → dst

Emulation `SUBC #0,dst`

`SUBC.B #0,dst`

Description The carry bit (C) is added to the destination operand minus one. The previous contents of the destination are lost.

Status Bits
- N: Set if result is negative, reset if positive
- Z: Set if result is zero, reset otherwise
- C: Set if there is a carry from the MSB of the result, reset otherwise
  Set to 1 if no borrow, reset if borrow
- V: Set if an arithmetic overflow occurs, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 16-bit counter pointed to by R13 is subtracted from a 32-bit counter pointed to by R12.

```
SUB     @R13,0(R12)    ; Subtract LSDs
SBC     2(R12)         ; Subtract carry from MSD
```

Example The 8-bit counter pointed to by R13 is subtracted from a 16-bit counter pointed to by R12.

```
SUB.B   @R13,0(R12)    ; Subtract LSDs
SBC.B   1(R12)         ; Subtract carry from MSD
```

NOTE: Borrow implementation

The borrow is treated as a .NOT. carry:

Borrow	Carry Bit
Yes	0
No	1

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.43 SETC

*** SETC** Set carry bit

Syntax SETC

Operation 1 → C

Emulation BIS #1,SR

Description The carry bit (C) is set.

Status Bits
N: Not affected
Z: Not affected
C: Set
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Emulation of the decimal subtraction:
Subtract R5 from R6 decimally.
Assume that R5 = 03987h and R6 = 04137h.

```
DSUB    ADD     #06666h,R5      ; Move content R5 from 0-9 to 6-0Fh
                                ; R5 = 03987h + 06666h = 09FEDh
        INV     R5              ; Invert this (result back to 0-9)
                                ; R5 = .NOT. R5 = 06012h
        SETC                    ; Prepare carry = 1
        DADD    R5,R6           ; Emulate subtraction by addition of:
                                ; (010000h - R5 - 1)
                                ; R6 = R6 + R5 + 1
                                ; R6 = 0150h
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.44 SETN

*** SETN**	Set negative bit
Syntax	`SETN`
Operation	1 → N
Emulation	`BIS #4,SR`
Description	The negative bit (N) is set.
Status Bits	N: Set Z: Not affected C: Not affected V: Not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.45 SETZ

*** SETZ**	Set zero bit
Syntax	`SETZ`
Operation	1 → N
Emulation	`BIS #2,SR`
Description	The zero bit (Z) is set.
Status Bits	N: Not affected Z: Set C: Not affected V: Not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.46 SUB

SUB[.W] Subtract source word from destination word

SUB.B Subtract source byte from destination byte

Syntax `SUB src,dst` or `SUB.W src,dst`

`SUB.B src,dst`

Operation (.not.src) + 1 + dst → dst or dst – src → dst

Description The source operand is subtracted from the destination operand. This is made by adding the 1s complement of the source + 1 to the destination. The source operand is not affected, the result is written to the destination operand.

Status Bits

N: Set if result is negative (src > dst), reset if positive (src ≤ dst)

Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example A 16-bit constant 7654h is subtracted from RAM word EDE.

```
SUB     #7654h,&EDE     ; Subtract 7654h from EDE
```

Example A table word pointed to by R5 (20-bit address) is subtracted from R7. Afterwards, if R7 contains zero, jump to label TONI. R5 is then auto-incremented by 2. R7.19:16 = 0.

```
SUB     @R5+,R7         ; Subtract table number from R7. R5 + 2
JZ      TONI            ; R7 = @R5 (before subtraction)
...                     ; R7 <> @R5 (before subtraction)
```

Example Byte CNT is subtracted from byte R12 points to. The address of CNT is within PC ± 32K. The address R12 points to is in full memory range.

```
SUB.B   CNT,0(R12)      ; Subtract CNT from @R12
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.47 SUBC

SUBC[.W] Subtract source word with carry from destination word

SUBC.B Subtract source byte with carry from destination byte

Syntax `SUBC src,dst` or `SUBC.W src,dst`

`SUBC.B src,dst`

Operation (.not.src) + C + dst → dst or dst – (src – 1) + C → dst

Description The source operand is subtracted from the destination operand. This is done by adding the 1s complement of the source + carry to the destination. The source operand is not affected, the result is written to the destination operand. Used for 32, 48, and 64-bit operands.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example A 16-bit constant 7654h is subtracted from R5 with the carry from the previous instruction. R5.19:16 = 0

```
SUBC.W   #7654h,R5       ; Subtract 7654h + C from R5
```

Example A 48-bit number (3 words) pointed to by R5 (20-bit address) is subtracted from a 48-bit counter in RAM, pointed to by R7. R5 points to the next 48-bit number afterwards. The address R7 points to is in full memory range.

```
SUB      @R5+,0(R7)      ; Subtract LSBs. R5 + 2
SUBC     @R5+,2(R7)      ; Subtract MIDs with C. R5 + 2
SUBC     @R5+,4(R7)      ; Subtract MSBs with C. R5 + 2
```

Example Byte CNT is subtracted from the byte, R12 points to. The carry of the previous instruction is used. The address of CNT is in lower 64 K.

```
SUBC.B   &CNT,0(R12)     ; Subtract byte CNT from @R12
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.2.48 SWPB

SWPB	Swap bytes
Syntax	`SWPB dst`
Operation	dst.15:8 ↔ dst.7:0
Description	The high and the low byte of the operand are exchanged. PC.19:16 bits are cleared in register mode.
Status Bits	Status bits are not affected
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	Exchange the bytes of RAM word EDE (lower 64 K)

```
MOV    #1234h,&EDE      ; 1234h -> EDE
SWPB   &EDE             ; 3412h -> EDE
```

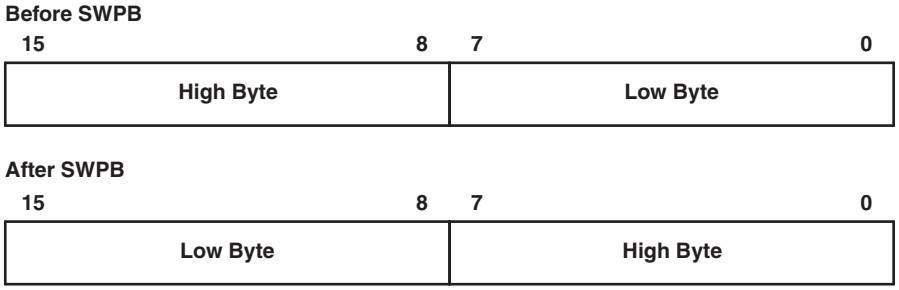


Figure 4-42. Swap Bytes in Memory

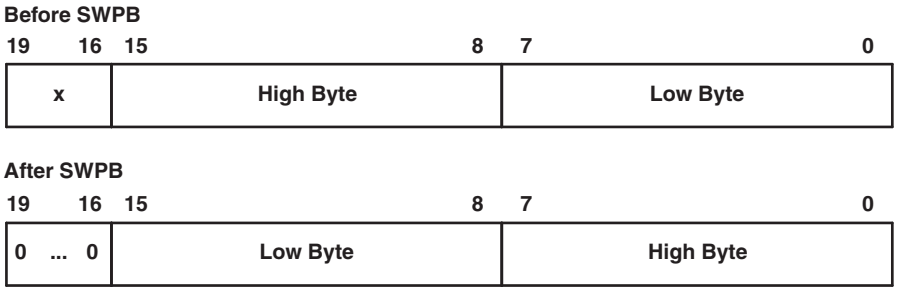


Figure 4-43. Swap Bytes in a Register

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.49 SXT

SXT Extend sign

Syntax `SXT dst`

Operation dst.7 → dst.15:8, dst.7 → dst.19:8 (register mode)

Description Register mode: the sign of the low byte of the operand is extended into the bits Rdst.19:8.

Rdst.7 = 0: Rdst.19:8 = 000h afterwards

Rdst.7 = 1: Rdst.19:8 = FFFh afterwards

Other modes: the sign of the low byte of the operand is extended into the high byte.

dst.7 = 0: high byte = 00h afterwards

dst.7 = 1: high byte = FFh afterwards

Status Bits

N: Set if result is negative, reset otherwise

Z: Set if result is zero, reset otherwise

C: Set if result is not zero, reset otherwise (C = .not.Z)

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The signed 8-bit data in EDE (lower 64 K) is sign extended and added to the 16-bit signed data in R7.

```
MOV.B   &EDE,R5      ; EDE -> R5. 00XXh
SXT     R5           ; Sign extend low byte to R5.19:8
ADD     R5,R7        ; Add signed 16-bit values
```

Example The signed 8-bit data in EDE (PC +32 K) is sign extended and added to the 20-bit data in R7.

```
MOV.B   EDE,R5       ; EDE -> R5. 00XXh
SXT     R5           ; Sign extend low byte to R5.19:8
ADDA    R5,R7        ; Add signed 20-bit values
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.50 TST

*** TST[.W]** Test destination

*** TST.B** Test destination

Syntax `TST dst` or `TST.W dst`

`TST.B dst`

Operation dst + 0FFFFh + 1

dst + 0FFh + 1

Emulation `CMP #0,dst`

`CMP.B #0,dst`

Description The destination operand is compared with zero. The status bits are set according to the result. The destination is not affected.

Status Bits N: Set if destination is negative, reset if positive

Z: Set if destination contains zero, reset otherwise

C: Set

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example R7 is tested. If it is negative, continue at R7NEG; if it is positive but not zero, continue at R7POS.

```
        TST     R7        ; Test R7
        JN      R7NEG     ; R7 is negative
        JZ      R7ZERO    ; R7 is zero
R7POS   ......            ; R7 is positive but not zero
R7NEG   ......            ; R7 is negative
R7ZERO  ......            ; R7 is zero
```

Example The low byte of R7 is tested. If it is negative, continue at R7NEG; if it is positive but not zero, continue at R7POS.

```
        TST.B   R7        ; Test low byte of R7
        JN      R7NEG     ; Low byte of R7 is negative
        JZ      R7ZERO    ; Low byte of R7 is zero
R7POS   ......            ; Low byte of R7 is positive but not zero
R7NEG   .....             ; Low byte of R7 is negative
R7ZERO  ......            ; Low byte of R7 is zero
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.2.51 XOR

XOR[.W] Exclusive OR source word with destination word

XOR.B Exclusive OR source byte with destination byte

Syntax `XOR src,dst` or `XOR.W src,dst`

`XOR.B src,dst`

Operation src .xor. dst → dst

Description The source and destination operands are exclusively ORed. The result is placed into the destination. The source operand is not affected. The previous content of the destination is lost.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if result is not zero, reset otherwise (C = .not. Z)
- V: Set if both operands are negative before execution, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Toggle bits in word CNTR (16-bit data) with information (bit = 1) in address-word TONI. Both operands are located in lower 64 K.

```
XOR     &TONI,&CNTR     ; Toggle bits in CNTR
```

Example A table word pointed to by R5 (20-bit address) is used to toggle bits in R6. R6.19:16 = 0.

```
XOR     @R5,R6          ; Toggle bits in R6
```

Example Reset to zero those bits in the low byte of R7 that are different from the bits in byte EDE. R7.19:8 = 0. The address of EDE is within PC ± 32 K.

```
XOR.B   EDE,R7          ; Set different bits to 1 in R7.
INV.B   R7              ; Invert low byte of R7, high byte is 0h
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3 Extended Instructions

The extended MSP430X instructions give the MSP430X CPU full access to its 20-bit address space. MSP430X instructions require an additional word of op-code called the extension word. All addresses, indexes, and immediate numbers have 20-bit values when preceded by the extension word. The MSP430X extended instructions are listed and described in the following pages.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.1 ADCX

*** ADCX.A** Add carry to destination address-word

*** ADCX.[W]** Add carry to destination word

*** ADCX.B** Add carry to destination byte

Syntax

```
ADCX.A dst
ADCX dst or          ADCX.W dst
ADCX.B dst
```

Operation dst + C → dst

Emulation

```
ADDCX.A #0,dst
ADDCX #0,dst
ADDCX.B #0,dst
```

Description The carry bit (C) is added to the destination operand. The previous contents of the destination are lost.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if there is a carry from the MSB of the result, reset otherwise

V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 40-bit counter, pointed to by R12 and R13, is incremented.

```
INCX.A   @R12     ; Increment lower 20 bits
ADCX.A   @R13     ; Add carry to upper 20 bits
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.2 ADDX

ADDX.A Add source address-word to destination address-word

ADDX.[W] Add source word to destination word

ADDX.B Add source byte to destination byte

Syntax `ADDX.A src,dst`

`ADDX src,dst` or `ADDX.W src,dst`

`ADDX.B src,dst`

Operation src + dst → dst

Description The source operand is added to the destination operand. The previous contents of the destination are lost. Both operands can be located in the full address space.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if there is a carry from the MSB of the result, reset otherwise
- V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Ten is added to the 20-bit pointer CNTR located in two words CNTR (LSBs) and CNTR+2 (MSBs).

```
ADDX.A   #10,CNTR    ; Add 10 to 20-bit pointer
```

Example A table word (16-bit) pointed to by R5 (20-bit address) is added to R6. The jump to label TONI is performed on a carry.

```
ADDX.W   @R5,R6      ; Add table word to R6
JC       TONI        ; Jump if carry
...                  ; No carry
```

Example A table byte pointed to by R5 (20-bit address) is added to R6. The jump to label TONI is performed if no carry occurs. The table pointer is auto-incremented by 1.

```
ADDX.B   @R5+,R6     ; Add table byte to R6. R5 + 1. R6: 000xxh
JNC      TONI        ; Jump if no carry
...                  ; Carry occurred
```

Note: Use ADDA for the following two cases for better code density and execution.

```
ADDX.A   Rsrc,Rdst
ADDX.A   #imm20,Rdst
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.3 ADDCX

ADDCX.A Add source address-word and carry to destination address-word

ADDCX.[W] Add source word and carry to destination word

ADDCX.B Add source byte and carry to destination byte

Syntax `ADDCX.A src,dst`

`ADDCX src,dst` or `ADDCX.W src,dst`

`ADDCX.B src,dst`

Operation src + dst + C → dst

Description The source operand and the carry bit C are added to the destination operand. The previous contents of the destination are lost. Both operands may be located in the full address space.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if there is a carry from the MSB of the result, reset otherwise
- V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Constant 15 and the carry of the previous instruction are added to the 20-bit counter CNTR located in two words.

```
ADDCX.A   #15,&CNTR   ; Add 15 + C to 20-bit CNTR
```

Example A table word pointed to by R5 (20-bit address) and the carry C are added to R6. The jump to label TONI is performed on a carry.

```
ADDCX.W   @R5,R6      ; Add table word + C to R6
JC        TONI        ; Jump if carry
...                   ; No carry
```

Example A table byte pointed to by R5 (20-bit address) and the carry bit C are added to R6. The jump to label TONI is performed if no carry occurs. The table pointer is auto-incremented by 1.

```
ADDCX.B   @R5+,R6     ; Add table byte + C to R6. R5 + 1
JNC       TONI        ; Jump if no carry
...                   ; Carry occurred
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.4 ANDX

ANDX.A Logical AND of source address-word with destination address-word

ANDX.[W] Logical AND of source word with destination word

ANDX.B Logical AND of source byte with destination byte

Syntax

```
ANDX.A src,dst
ANDX src,dst or ANDX.W src,dst
ANDX.B src,dst
```

Operation src .and. dst → dst

Description The source operand and the destination operand are logically ANDed. The result is placed into the destination. The source operand is not affected. Both operands may be located in the full address space.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if the result is not zero, reset otherwise. C = (.not. Z)

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The bits set in R5 (20-bit data) are used as a mask (AAA55h) for the address-word TOM located in two words. If the result is zero, a branch is taken to label TONI.

```
MOVA    #AAA55h,R5     ; Load 20-bit mask to R5
ANDX.A  R5,TOM         ; TOM .and. R5 -> TOM
JZ      TONI           ; Jump if result 0
...                    ; Result > 0
```

or shorter:

```
ANDX.A  #AAA55h,TOM    ; TOM .and. AAA55h -> TOM
JZ      TONI           ; Jump if result 0
```

Example A table byte pointed to by R5 (20-bit address) is logically ANDed with R6. R6.19:8 = 0. The table pointer is auto-incremented by 1.

```
ANDX.B  @R5+,R6        ; AND table byte with R6. R5 + 1
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.5 BICX

BICX.A Clear bits set in source address-word in destination address-word

BICX.[W] Clear bits set in source word in destination word

BICX.B Clear bits set in source byte in destination byte

Syntax `BICX.A src,dst`

`BICX src,dst` or `BICX.W src,dst`

`BICX.B src,dst`

Operation (.not. src) .and. dst → dst

Description The inverted source operand and the destination operand are logically ANDed. The result is placed into the destination. The source operand is not affected. Both operands may be located in the full address space.

Status Bits
N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The bits 19:15 of R5 (20-bit data) are cleared.

```
BICX.A   #0F8000h,R5     ; Clear R5.19:15 bits
```

Example A table word pointed to by R5 (20-bit address) is used to clear bits in R7. R7.19:16 = 0.

```
BICX.W   @R5,R7          ; Clear bits in R7
```

Example A table byte pointed to by R5 (20-bit address) is used to clear bits in output Port1.

```
BICX.B   @R5,&P1OUT      ; Clear I/O port P1 bits
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.6 BISX

BISX.A Set bits set in source address-word in destination address-word

BISX.[W] Set bits set in source word in destination word

BISX.B Set bits set in source byte in destination byte

Syntax

```
BISX.A src,dst
BISX src,dst or BISX.W src,dst
BISX.B src,dst
```

Operation src .or. dst → dst

Description The source operand and the destination operand are logically ORed. The result is placed into the destination. The source operand is not affected. Both operands may be located in the full address space.

Status Bits

N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Bits 16 and 15 of R5 (20-bit data) are set to one.

```
BISX.A    #018000h,R5     ; Set R5.16:15 bits
```

Example A table word pointed to by R5 (20-bit address) is used to set bits in R7.

```
BISX.W   @R5,R7           ; Set bits in R7
```

Example A table byte pointed to by R5 (20-bit address) is used to set bits in output Port1.

```
BISX.B   @R5,&P1OUT       ; Set I/O port P1 bits
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.7 BITX

BITX.A Test bits set in source address-word in destination address-word

BITX.[W] Test bits set in source word in destination word

BITX.B Test bits set in source byte in destination byte

Syntax `BITX.A src,dst`

`BITX src,dst` or `BITX.W src,dst`

`BITX.B src,dst`

Operation src .and. dst → dst

Description The source operand and the destination operand are logically ANDed. The result affects only the status bits. Both operands may be located in the full address space.

Status Bits
- N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if the result is not zero, reset otherwise. C = (.not. Z)
- V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Test if bit 16 or 15 of R5 (20-bit data) is set. Jump to label TONI if so.

```
BITX.A   #018000h,R5     ; Test R5.16:15 bits
JNZ      TONI            ; At least one bit is set
...                      ; Both are reset
```

Example A table word pointed to by R5 (20-bit address) is used to test bits in R7. Jump to label TONI if at least one bit is set.

```
BITX.W   @R5,R7          ; Test bits in R7: C = .not.Z
JC       TONI            ; At least one is set
...                      ; Both are reset
```

Example A table byte pointed to by R5 (20-bit address) is used to test bits in input Port1. Jump to label TONI if no bit is set. The next table byte is addressed.

```
BITX.B   @R5+,&P1IN      ; Test input P1 bits. R5 + 1
JNC      TONI            ; No corresponding input bit is set
...                      ; At least one bit is set
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.8 CLRX

*** CLRX.A** Clear destination address-word
*** CLRX.[W]** Clear destination word
*** CLRX.B** Clear destination byte
Syntax `CLRX.A dst`
`CLRX dst` or `CLRX.W dst`
`CLRX.B dst`
Operation 0 → dst
Emulation `MOVX.A #0,dst`
`MOVX #0,dst`
`MOVX.B #0,dst`
Description The destination operand is cleared.
Status Bits Status bits are not affected.
Mode Bits OSCOFF, CPUOFF, and GIE are not affected.
Example RAM address-word TONI is cleared.

```
CLRX.A   TONI   ; 0 -> TONI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.9 CMPX

CMPX.A Compare source address-word and destination address-word

CMPX.[W] Compare source word and destination word

CMPX.B Compare source byte and destination byte

Syntax `CMPX.A src,dst`

`CMPX src,dst` or `CMPX.W src,dst`

`CMPX.B src,dst`

Operation (.not. src) + 1 + dst or dst – src

Description The source operand is subtracted from the destination operand by adding the 1s complement of the source + 1 to the destination. The result affects only the status bits. Both operands may be located in the full address space.

Status Bits

N: Set if result is negative (src > dst), reset if positive (src ≤ dst)

Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Compare EDE with a 20-bit constant 18000h. Jump to label TONI if EDE equals the constant.

```
CMPX.A   #018000h,EDE     ; Compare EDE with 18000h
JEQ      TONI             ; EDE contains 18000h
...                       ; Not equal
```

Example A table word pointed to by R5 (20-bit address) is compared with R7. Jump to label TONI if R7 contains a lower, signed, 16-bit number.

```
CMPX.W   @R5,R7           ; Compare two signed numbers
JL       TONI             ; R7 < @R5
...                       ; R7 >= @R5
```

Example A table byte pointed to by R5 (20-bit address) is compared to the input in I/O Port1. Jump to label TONI if the values are equal. The next table byte is addressed.

```
CMPX.B   @R5+,&P1IN       ; Compare P1 bits with table. R5 + 1
JEQ      TONI             ; Equal contents
...                       ; Not equal
```

Note: Use CMPA for the following two cases for better density and execution.

```
CMPA     Rsrc,Rdst
CMPA     #imm20,Rdst
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.10 DADCX

*** DADCX.A** Add carry decimally to destination address-word

*** DADCX.[W]** Add carry decimally to destination word

*** DADCX.B** Add carry decimally to destination byte

Syntax

```
DADCX.A dst
DADCX dst or          DADCX.W dst
DADCX.B dst
```

Operation dst + C → dst (decimally)

Emulation

```
DADDX.A #0,dst
DADDX #0,dst
DADDX.B #0,dst
```

Description The carry bit (C) is added decimally to the destination.

Status Bits

- N: Set if MSB of result is 1 (address-word > 79999h, word > 7999h, byte > 79h), reset if MSB is 0
- Z: Set if result is zero, reset otherwise
- C: Set if the BCD result is too large (address-word > 99999h, word > 9999h, byte > 99h), reset otherwise
- V: Undefined

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 40-bit counter, pointed to by R12 and R13, is incremented decimally.

```
DADDX.A   #1,0(R12)     ; Increment lower 20 bits
DADCX.A   0(R13)        ; Add carry to upper 20 bits
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.11 DADDX

DADDX.A Add source address-word and carry decimally to destination address-word

DADDX.[W] Add source word and carry decimally to destination word

DADDX.B Add source byte and carry decimally to destination byte

Syntax

```
DADDX.A src,dst
DADDX src,dst or DADDX.W src,dst
DADDX.B src,dst
```

Operation src + dst + C → dst (decimally)

Description The source operand and the destination operand are treated as two (.B), four (.W), or five (.A) binary coded decimals (BCD) with positive signs. The source operand and the carry bit C are added decimally to the destination operand. The source operand is not affected. The previous contents of the destination are lost. The result is not defined for non-BCD numbers. Both operands may be located in the full address space.

Status Bits

N: Set if MSB of result is 1 (address-word > 79999h, word > 7999h, byte > 79h), reset if MSB is 0.

Z: Set if result is zero, reset otherwise

C: Set if the BCD result is too large (address-word > 99999h, word > 9999h, byte > 99h), reset otherwise

V: Undefined

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Decimal 10 is added to the 20-bit BCD counter DECCNTR located in two words.

```
DADDX.A   #10h,&DECCNTR     ; Add 10 to 20-bit BCD counter
```

Example The eight-digit BCD number contained in 20-bit addresses BCD and BCD+2 is added decimally to an eight-digit BCD number contained in R4 and R5 (BCD+2 and R5 contain the MSDs).

```
CLRC                        ; Clear carry
DADDX.W   BCD,R4            ; Add LSDs
DADDX.W   BCD+2,R5          ; Add MSDs with carry
JC        OVERFLOW          ; Result >99999999: go to error routine
...                         ;    Result ok
```

Example The two-digit BCD number contained in 20-bit address BCD is added decimally to a two-digit BCD number contained in R4.

```
CLRC                        ; Clear carry
DADDX.B   BCD,R4            ; Add BCD to R4 decimally.
                            ; R4: 000ddh
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.12 DECX

*** DECX.A** Decrement destination address-word
*** DECX.[W]** Decrement destination word
*** DECX.B** Decrement destination byte

Syntax
```
DECX.A dst
DECX dst or          DECX.W dst
DECX.B dst
```

Operation dst – 1 → dst

Emulation
```
SUBX.A #1,dst
SUBX #1,dst
SUBX.B #1,dst
```

Description The destination operand is decremented by one. The original contents are lost.

Status Bits
N: Set if result is negative, reset if positive
Z: Set if dst contained 1, reset otherwise
C: Reset if dst contained 0, set otherwise
V: Set if an arithmetic overflow occurs, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM address-word TONI is decremented by one.

```
DECX.A   TONI     ; Decrement TONI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.13 DECDX

*** DECDX.A** Double-decrement destination address-word

*** DECDX.[W]** Double-decrement destination word

*** DECDX.B** Double-decrement destination byte

Syntax

```
DECDX.A dst
DECDX dst or          DECDX.W dst
DECDX.B dst
```

Operation dst – 2 → dst

Emulation

```
SUBX.A #2,dst
SUBX #2,dst
SUBX.B #2,dst
```

Description The destination operand is decremented by two. The original contents are lost.

Status Bits

N: Set if result is negative, reset if positive

Z: Set if dst contained 2, reset otherwise

C: Reset if dst contained 0 or 1, set otherwise

V: Set if an arithmetic overflow occurs, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM address-word TONI is decremented by two.

```
DECDX.A   TONI     ; Decrement TONI
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.14 INCX

*** INCX.A** Increment destination address-word
*** INCX.[W]** Increment destination word
*** INCX.B** Increment destination byte

Syntax
```
INCX.A dst
INCX dst or          INCX.W dst
INCX.B dst
```

Operation dst + 1 → dst

Emulation
```
ADDX.A #1,dst
ADDX #1,dst
ADDX.B #1,dst
```

Description The destination operand is incremented by one. The original contents are lost.

Status Bits
N: Set if result is negative, reset if positive
Z: Set if dst contained 0FFFFFh, reset otherwise
Set if dst contained 0FFFFh, reset otherwise
Set if dst contained 0FFh, reset otherwise
C: Set if dst contained 0FFFFFh, reset otherwise
Set if dst contained 0FFFFh, reset otherwise
Set if dst contained 0FFh, reset otherwise
V: Set if dst contained 07FFFh, reset otherwise
Set if dst contained 07FFFh, reset otherwise
Set if dst contained 07Fh, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM address-wordTONI is incremented by one.

```
INCX.A   TONI     ; Increment TONI (20-bits)
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.15 INCDX

*** INCDX.A** Double-increment destination address-word

*** INCDX.[W]** Double-increment destination word

*** INCDX.B** Double-increment destination byte

Syntax

```
INCDX.A dst
INCDX dst or          INCDX.W dst
INCDX.B dst
```

Operation dst + 2 → dst

Emulation

```
ADDX.A #2,dst
ADDX #2,dst
ADDX.B #2,dst
```

Description The destination operand is incremented by two. The original contents are lost.

Status Bits

N: Set if result is negative, reset if positive

Z: Set if dst contained 0FFFFEh, reset otherwise
Set if dst contained 0FFFEh, reset otherwise
Set if dst contained 0FEh, reset otherwise

C: Set if dst contained 0FFFFEh or 0FFFFFh, reset otherwise
Set if dst contained 0FFFEh or 0FFFFh, reset otherwise
Set if dst contained 0FEh or 0FFh, reset otherwise

V: Set if dst contained 07FFFEh or 07FFFFh, reset otherwise
Set if dst contained 07FFEh or 07FFFh, reset otherwise
Set if dst contained 07Eh or 07Fh, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM byte LEO is incremented by two; PC points to upper memory.

```
INCDX.B   LEO     ; Increment LEO by two
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.16 INVX

*** INVX.A** Invert destination
*** INVX.[W]** Invert destination
*** INVX.B** Invert destination

Syntax
```
INVX.A dst
INVX dst or          INVX.W dst
INVX.B dst
```

Operation .NOT.dst → dst

Emulation
```
XORX.A #0FFFFFh,dst
XORX #0FFFFh,dst
XORX.B #0FFh,dst
```

Description The destination operand is inverted. The original contents are lost.

Status Bits
N: Set if result is negative, reset if positive
Z: Set if dst contained 0FFFFFh, reset otherwise
Set if dst contained 0FFFFh, reset otherwise
Set if dst contained 0FFh, reset otherwise
C: Set if result is not zero, reset otherwise (= .NOT. Zero)
V: Set if initial destination operand was negative, otherwise reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example 20-bit content of R5 is negated (2s complement).

```
INVX.A   R5      ; Invert R5
INCX.A   R5      ; R5 is now negated
```

Example Content of memory byte LEO is negated. PC is pointing to upper memory.

```
INVX.B   LEO     ; Invert LEO
INCX.B   LEO     ; MEM(LEO) is negated
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.17 MOVX

MOVX.A Move source address-word to destination address-word

MOVX.[W] Move source word to destination word

MOVX.B Move source byte to destination byte

Syntax `MOVX.A src,dst`

`MOVX src,dst` or `MOVX.W src,dst`

`MOVX.B src,dst`

Operation src → dst

Description The source operand is copied to the destination. The source operand is not affected. Both operands may be located in the full address space.

Status Bits N: Not affected

Z: Not affected

C: Not affected

V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Move a 20-bit constant 18000h to absolute address-word EDE

```
MOVX.A   #018000h,&EDE          ; Move 18000h to EDE
```

Example The contents of table EDE (word data, 20-bit addresses) are copied to table TOM. The length of the table is 030h words.

```
       MOVA     #EDE,R10               ; Prepare pointer (20-bit address)
Loop   MOVX.W   @R10+,TOM-EDE-2(R10)   ; R10 points to both tables.
                                       ; R10+2
       CMPA     #EDE+60h,R10           ; End of table reached?
       JLO      Loop                   ; Not yet
       ...                             ; Copy completed
```

Example The contents of table EDE (byte data, 20-bit addresses) are copied to table TOM. The length of the table is 020h bytes.

```
       MOVA     #EDE,R10               ; Prepare pointer (20-bit)
       MOV      #20h,R9                ; Prepare counter
Loop   MOVX.W   @R10+,TOM-EDE-2(R10)   ; R10 points to both tables.
                                       ; R10+1
       DEC      R9                     ; Decrement counter
       JNZ      Loop                   ; Not yet done
       ...                             ; Copy completed
```

Ten of the 28 possible addressing combinations of the MOVX.A instruction can use the MOVA instruction. This saves two bytes and code cycles. Examples for the addressing combinations are:

```
MOVX.A   Rsrc,Rdst          MOVA   Rsrc,Rdst        ; Reg/Reg
MOVX.A   #imm20,Rdst        MOVA   #imm20,Rdst      ; Immediate/Reg
MOVX.A   &abs20,Rdst        MOVA   &abs20,Rdst      ; Absolute/Reg
MOVX.A   @Rsrc,Rdst         MOVA   @Rsrc,Rdst       ; Indirect/Reg
MOVX.A   @Rsrc+,Rdst        MOVA   @Rsrc+,Rdst      ; Indirect,Auto/Reg
MOVX.A   Rsrc,&abs20        MOVA   Rsrc,&abs20      ; Reg/Absolute
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The next four replacements are possible only if 16-bit indexes are sufficient for the addressing:

```
MOVX.A   z20(Rsrc),Rdst     MOVA   z16(Rsrc),Rdst   ; Indexed/Reg
MOVX.A   Rsrc,z20(Rdst)     MOVA   Rsrc,z16(Rdst)   ; Reg/Indexed
MOVX.A   symb20,Rdst        MOVA   symb16,Rdst      ; Symbolic/Reg
MOVX.A   Rsrc,symb20        MOVA   Rsrc,symb16      ; Reg/Symbolic
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.18 POPM

POPM.A Restore n CPU registers (20-bit data) from the stack

POPM.[W] Restore n CPU registers (16-bit data) from the stack

Syntax

`POPM.A #n,Rdst` 1 ≤ n ≤ 16

`POPM.W #n,Rdst` or `POPM #n,Rdst` 1 ≤ n ≤ 16

Operation POPM.A: Restore the register values from stack to the specified CPU registers. The SP is incremented by four for each register restored from stack. The 20-bit values from stack (two words per register) are restored to the registers.

POPM.W: Restore the 16-bit register values from stack to the specified CPU registers. The SP is incremented by two for each register restored from stack. The 16-bit values from stack (one word per register) are restored to the CPU registers.

Note : This instruction does not use the extension word.

Description POPM.A: The CPU registers pushed on the stack are moved to the extended CPU registers, starting with the CPU register (Rdst – n + 1). The SP is incremented by (n × 4) after the operation.

POPM.W: The 16-bit registers pushed on the stack are moved back to the CPU registers, starting with CPU register (Rdst – n + 1). The SP is incremented by (n × 2) after the instruction. The MSBs (Rdst.19:16) of the restored CPU registers are cleared.

Status Bits Status bits are not affected, except SR is included in the operation.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Restore the 20-bit registers R9, R10, R11, R12, R13 from the stack

```
POPM.A   #5,R13     ; Restore R9, R10, R11, R12, R13
```

Example Restore the 16-bit registers R9, R10, R11, R12, R13 from the stack.

```
POPM.W   #5,R13     ; Restore R9, R10, R11, R12, R13
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.19 PUSHM

PUSHM.A Save n CPU registers (20-bit data) on the stack

PUSHM.[W] Save n CPU registers (16-bit words) on the stack

Syntax

`PUSHM.A #n,Rdst` 1 ≤ n ≤ 16

`PUSHM.W #n,Rdst` or `PUSHM #n,Rdst` 1 ≤ n ≤ 16

Operation PUSHM.A: Save the 20-bit CPU register values on the stack. The SP is decremented by four for each register stored on the stack. The MSBs are stored first (higher address).

PUSHM.W: Save the 16-bit CPU register values on the stack. The SP is decremented by two for each register stored on the stack.

Description PUSHM.A: The n CPU registers, starting with Rdst backwards, are stored on the stack. The SP is decremented by (n × 4) after the operation. The data (Rn.19:0) of the pushed CPU registers is not affected.

PUSHM.W: The n registers, starting with Rdst backwards, are stored on the stack. The SP is decremented by (n × 2) after the operation. The data (Rn.19:0) of the pushed CPU registers is not affected.

Note : This instruction does not use the extension word.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Save the five 20-bit registers R9, R10, R11, R12, R13 on the stack

```
PUSHM.A   #5,R13     ; Save R13, R12, R11, R10, R9
```

Example Save the five 16-bit registers R9, R10, R11, R12, R13 on the stack

```
PUSHM.W   #5,R13     ; Save R13, R12, R11, R10, R9
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.20 POPX

*** POPX.A** Restore single address-word from the stack

*** POPX.[W]** Restore single word from the stack

*** POPX.B** Restore single byte from the stack

Syntax `POPX.A dst`

`POPX dst` or `POPX.W dst`

`POPX.B dst`

Operation Restore the 8-, 16-, 20-bit value from the stack to the destination. 20-bit addresses are possible. The SP is incremented by two (byte and word operands) and by four (address-word operand).

Emulation `MOVX(.B,.A) @SP+,dst`

Description The item on TOS is written to the destination operand. Register mode, Indexed mode, Symbolic mode, and Absolute mode are possible. The SP is incremented by two or four.

Note: the SP is incremented by two also for byte operations.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Write the 16-bit value on TOS to the 20-bit address &EDE

```
POPX.W   &EDE     ; Write word to address EDE
```

Example Write the 20-bit value on TOS to R9

```
POPX.A   R9     ; Write address-word to R9
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.21 PUSHX

PUSHX.A Save single address-word to the stack

PUSHX.[W] Save single word to the stack

PUSHX.B Save single byte to the stack

Syntax

```
PUSHX.A src
PUSHX src or          PUSHX.W src
PUSHX.B src
```

Operation Save the 8-, 16-, 20-bit value of the source operand on the TOS. 20-bit addresses are possible. The SP is decremented by two (byte and word operands) or by four (address-word operand) before the write operation.

Description The SP is decremented by two (byte and word operands) or by four (address-word operand). Then the source operand is written to the TOS. All seven addressing modes are possible for the source operand.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Save the byte at the 20-bit address &EDE on the stack

```
PUSHX.B   &EDE     ; Save byte at address EDE
```

Example Save the 20-bit value in R9 on the stack.

```
PUSHX.A   R9       ; Save address-word in R9
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.22 RLAM

RLAM.A Rotate left arithmetically the 20-bit CPU register content

RLAM.[W] Rotate left arithmetically the 16-bit CPU register content

Syntax

`RLAM.A #n,Rdst` 1 ≤ n ≤ 4

`RLAM.W #n,Rdst` or `RLAM #n,Rdst` 1 ≤ n ≤ 4

Operation C ← MSB ← MSB-1 .... LSB+1 ← LSB ← 0

Description The destination operand is shifted arithmetically left one, two, three, or four positions as shown in Figure 4-44. RLAM works as a multiplication (signed and unsigned) with 2, 4, 8, or 16. The word instruction RLAM.W clears the bits Rdst.19:16.

Note : This instruction does not use the extension word.

Status Bits

N: Set if result is negative
.A: Rdst.19 = 1, reset if Rdst.19 = 0
.W: Rdst.15 = 1, reset if Rdst.15 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the MSB (n = 1), MSB-1 (n = 2), MSB-2 (n = 3), MSB-3 (n = 4)

V: Undefined

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit operand in R5 is shifted left by three positions. It operates equal to an arithmetic multiplication by 8.

```
RLAM.A   #3,R5      ; R5 = R5 x 8
```

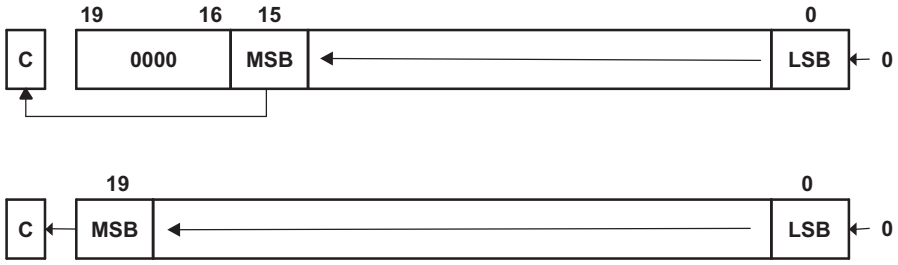


Figure 4-44. Rotate Left Arithmetically—RLAM[.W] and RLAM.A

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.23 RLAX

*** RLAX.A** Rotate left arithmetically address-word

*** RLAX.[W]** Rotate left arithmetically word

*** RLAX.B** Rotate left arithmetically byte

Syntax

```
RLAX.A dst
RLAX dst or          RLAX.W dst
RLAX.B dst
```

Operation C ← MSB ← MSB-1 .... LSB+1 ← LSB ← 0

Emulation

```
ADDX.A dst,dst
ADDX dst,dst
ADDX.B dst,dst
```

Description The destination operand is shifted left one position as shown in Figure 4-45. The MSB is shifted into the carry bit (C) and the LSB is filled with 0. The RLAX instruction acts as a signed multiplication by 2.

Status Bits

N: Set if result is negative, reset if positive

Z: Set if result is zero, reset otherwise

C: Loaded from the MSB

V: Set if an arithmetic overflow occurs: the initial value is 040000h ≤ dst < 0C0000h; reset otherwise

Set if an arithmetic overflow occurs: the initial value is 04000h ≤ dst < 0C000h; reset otherwise

Set if an arithmetic overflow occurs: the initial value is 040h ≤ dst < 0C0h; reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit value in R7 is multiplied by 2

```
RLAX.A   R7     ; Shift left R7 (20-bit)
```

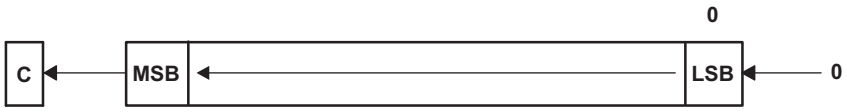


Figure 4-45. Destination Operand-Arithmetic Shift Left

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.24 RLCX

*** RLCX.A** Rotate left through carry address-word

*** RLCX.[W]** Rotate left through carry word

*** RLCX.B** Rotate left through carry byte

Syntax

```
RLCX.A dst
RLCX dst or          RLCX.W dst
RLCX.B dst
```

Operation C ← MSB ← MSB-1 .... LSB+1 ← LSB ← C

Emulation

```
ADDCX.A dst,dst
ADDCX dst,dst
ADDCX.B dst,dst
```

Description The destination operand is shifted left one position as shown in Figure 4-46. The carry bit (C) is shifted into the LSB and the MSB is shifted into the carry bit (C).

Status Bits

N: Set if result is negative, reset if positive

Z: Set if result is zero, reset otherwise

C: Loaded from the MSB

V: Set if an arithmetic overflow occurs: the initial value is 040000h ≤ dst < 0C0000h; reset otherwise

Set if an arithmetic overflow occurs: the initial value is 04000h ≤ dst < 0C000h; reset otherwise

Set if an arithmetic overflow occurs: the initial value is 040h ≤ dst < 0C0h; reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit value in R5 is shifted left one position.

```
RLCX.A   R5     ; (R5 x 2) + C -> R5
```

Example The RAM byte LEO is shifted left one position. PC is pointing to upper memory.

```
RLCX.B   LEO    ; RAM(LEO) x 2 + C -> RAM(LEO)
```

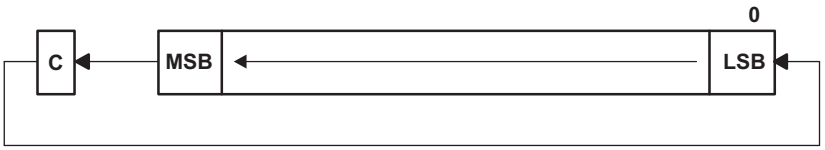


Figure 4-46. Destination Operand-Carry Left Shift

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.25 RRAM

RRAM.A Rotate right arithmetically the 20-bit CPU register content

RRAM.[W] Rotate right arithmetically the 16-bit CPU register content

Syntax

`RRAM.A #n,Rdst` 1 ≤ n ≤ 4

`RRAM.W #n,Rdst` or `RRAM #n,Rdst` 1 ≤ n ≤ 4

Operation MSB → MSB → MSB–1 ... LSB+1 → LSB → C

Description The destination operand is shifted right arithmetically by one, two, three, or four bit positions as shown in Figure 4-47. The MSB retains its value (sign). RRAM operates equal to a signed division by 2, 4, 8, or 16. The MSB is retained and shifted into MSB-1. The LSB+1 is shifted into the LSB, and the LSB is shifted into the carry bit C. The word instruction RRAM.W clears the bits Rdst.19:16.

Note : This instruction does not use the extension word.

Status Bits

N: Set if result is negative
.A: Rdst.19 = 1, reset if Rdst.19 = 0
.W: Rdst.15 = 1, reset if Rdst.15 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the LSB (n = 1), LSB+1 (n = 2), LSB+2 (n = 3), or LSB+3 (n = 4)

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The signed 20-bit number in R5 is shifted arithmetically right two positions.

```
RRAM.A   #2,R5          ; R5/4 -> R5
```

Example The signed 20-bit value in R15 is multiplied by 0.75. (0.5 + 0.25) × R15.

```
PUSHM.A   #1,R15        ; Save extended R15 on stack
RRAM.A    #1,R15        ; R15 y 0.5 -> R15
ADDX.A    @SP+,R15      ; R15 y 0.5 + R15 = 1.5 y R15 -> R15
RRAM.A    #1,R15        ; (1.5 y R15) y 0.5 = 0.75 y R15 -> R15
```

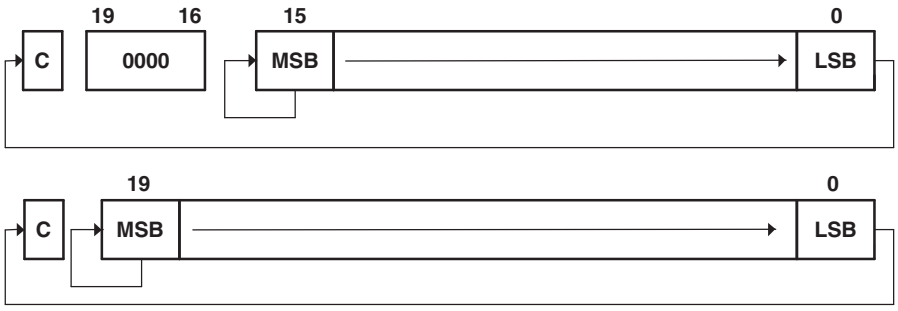


Figure 4-47. Rotate Right Arithmetically RRAM[.W] and RRAM.A

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.26 RRAX

RRAX.A Rotate right arithmetically the 20-bit operand

RRAX.[W] Rotate right arithmetically the 16-bit operand

RRAX.B Rotate right arithmetically the 8-bit operand

Syntax

```
RRAX.A Rdst
RRAX.W Rdst
RRAX Rdst
RRAX.B Rdst
RRAX.A dst
RRAX dst or          RRAX.W dst
RRAX.B dst
```

Operation MSB → MSB → MSB–1 ... LSB+1 → LSB → C

Description Register mode for the destination: the destination operand is shifted right by one bit position as shown in Figure 4-48. The MSB retains its value (sign). The word instruction RRAX.W clears the bits Rdst.19:16, the byte instruction RRAX.B clears the bits Rdst.19:8. The MSB retains its value (sign), the LSB is shifted into the carry bit. RRAX here operates equal to a signed division by 2.

All other modes for the destination: the destination operand is shifted right arithmetically by one bit position as shown in Figure 4-49. The MSB retains its value (sign), the LSB is shifted into the carry bit. RRAX here operates equal to a signed division by 2. All addressing modes, with the exception of the Immediate mode, are possible in the full memory.

Status Bits

N: Set if result is negative, reset if positive
.A: dst.19 = 1, reset if dst.19 = 0
.W: dst.15 = 1, reset if dst.15 = 0
.B: dst.7 = 1, reset if dst.7 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the LSB

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The signed 20-bit number in R5 is shifted arithmetically right four positions.

```
RPT      #4
RRAX.A   R5        ; R5/16 -> R5
```

Example The signed 8-bit value in EDE is multiplied by 0.5.

Copyright © 2009–2013, Texas Instruments Incorporated

```
RRAX.B    &EDE        ; EDE/2 -> EDE
```

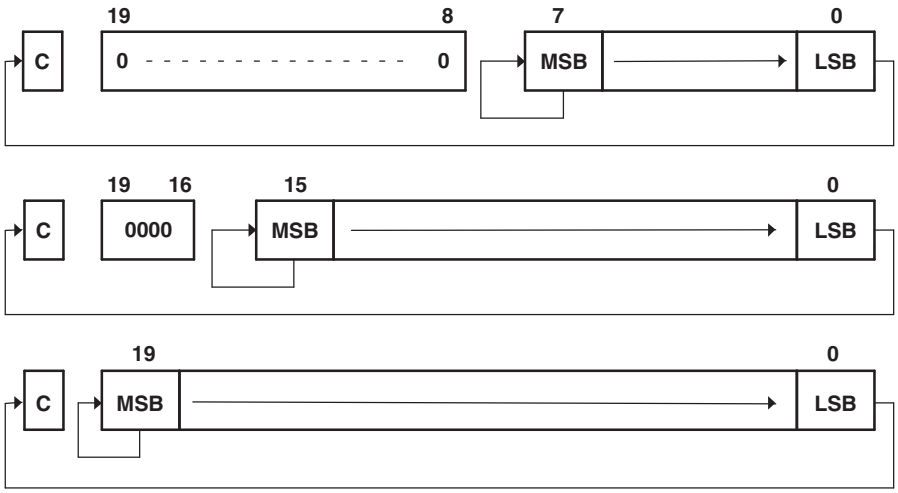


Figure 4-48. Rotate Right Arithmetically RRAX(.B,.A) – Register Mode

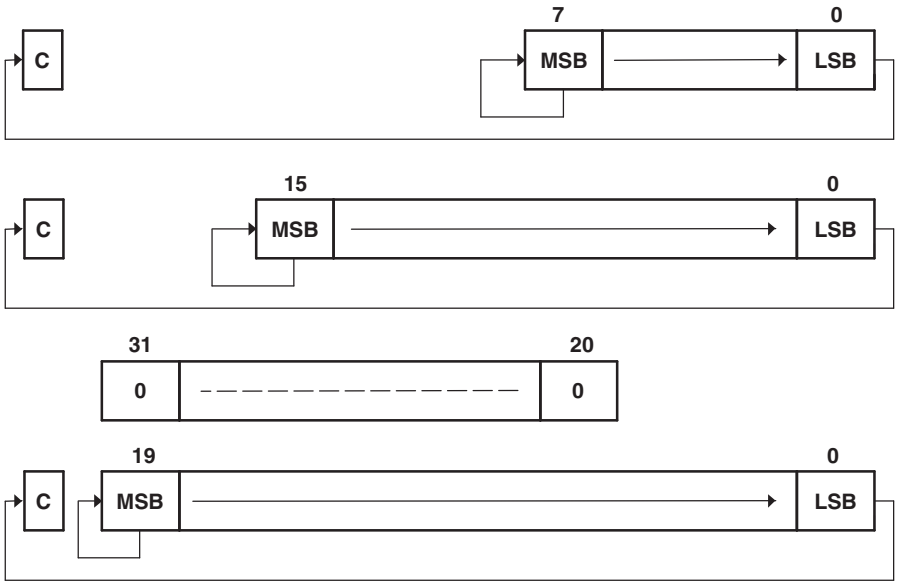


Figure 4-49. Rotate Right Arithmetically RRAX(.B,.A) – Non-Register Mode

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.27 RRCM

RRCM.A	Rotate right through carry the 20-bit CPU register content	
RRCM.[W]	Rotate right through carry the 16-bit CPU register content	
Syntax	`RRCM.A #n,Rdst`	1 ≤ n ≤ 4
	`RRCM.W #n,Rdst` or `RRCM #n,Rdst`	1 ≤ n ≤ 4
Operation	C → MSB → MSB–1 ... LSB+1 → LSB → C	
Description	The destination operand is shifted right by one, two, three, or four bit positions as shown in Figure 4-50. The carry bit C is shifted into the MSB, the LSB is shifted into the carry bit. The word instruction RRCM.W clears the bits Rdst.19:16.	
	Note : This instruction does not use the extension word.	
Status Bits	N: Set if result is negative	
	.A: Rdst.19 = 1, reset if Rdst.19 = 0	
	.W: Rdst.15 = 1, reset if Rdst.15 = 0	
	Z: Set if result is zero, reset otherwise	
	C: Loaded from the LSB (n = 1), LSB+1 (n = 2), LSB+2 (n = 3), or LSB+3 (n = 4)	
	V: Reset	
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Example The address-word in R5 is shifted right by three positions. The MSB–2 is loaded with 1.

```
SETC                ; Prepare carry for MSB-2
RRCM.A   #3,R5      ; R5 = R5 » 3 + 20000h
```

Example The word in R6 is shifted right by two positions. The MSB is loaded with the LSB. The MSB–1 is loaded with the contents of the carry flag.

```
RRCM.W   #2,R6      ; R6 = R6 » 2. R6.19:16 = 0
```

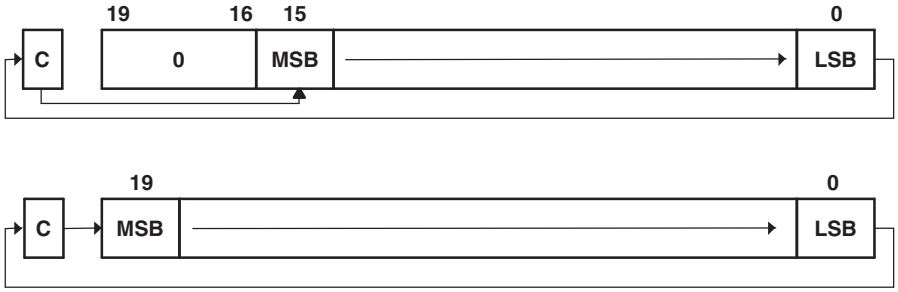


Figure 4-50. Rotate Right Through Carry RRCM[.W] and RRCM.A

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.28 RRCX

RRCX.A Rotate right through carry the 20-bit operand

RRCX.[W] Rotate right through carry the 16-bit operand

RRCX.B Rotate right through carry the 8-bit operand

Syntax

```
RRCX.A Rdst
RRCX.W Rdst
RRCX Rdst
RRCX.B Rdst

RRCX.A dst
RRCX dst or          RRCX.W dst
RRCX.B dst
```

Operation C → MSB → MSB–1 ... LSB+1 → LSB → C

Description Register mode for the destination: the destination operand is shifted right by one bit position as shown in Figure 4-51. The word instruction RRCX.W clears the bits Rdst.19:16, the byte instruction RRCX.B clears the bits Rdst.19:8. The carry bit C is shifted into the MSB, the LSB is shifted into the carry bit.

All other modes for the destination: the destination operand is shifted right by one bit position as shown in Figure 4-52. The carry bit C is shifted into the MSB, the LSB is shifted into the carry bit. All addressing modes, with the exception of the Immediate mode, are possible in the full memory.

Status Bits

N: Set if result is negative
- .A: dst.19 = 1, reset if dst.19 = 0
- .W: dst.15 = 1, reset if dst.15 = 0
- .B: dst.7 = 1, reset if dst.7 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the LSB

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit operand at address EDE is shifted right by one position. The MSB is loaded with 1.

```
SETC              ; Prepare carry for MSB
RRCX.A   EDE      ; EDE = EDE » 1 + 80000h
```

Example The word in R6 is shifted right by 12 positions.

Copyright © 2009–2013, Texas Instruments Incorporated

```
RPT     #12
RRCX.W  R6      ; R6 = R6 » 12. R6.19:16 = 0
```

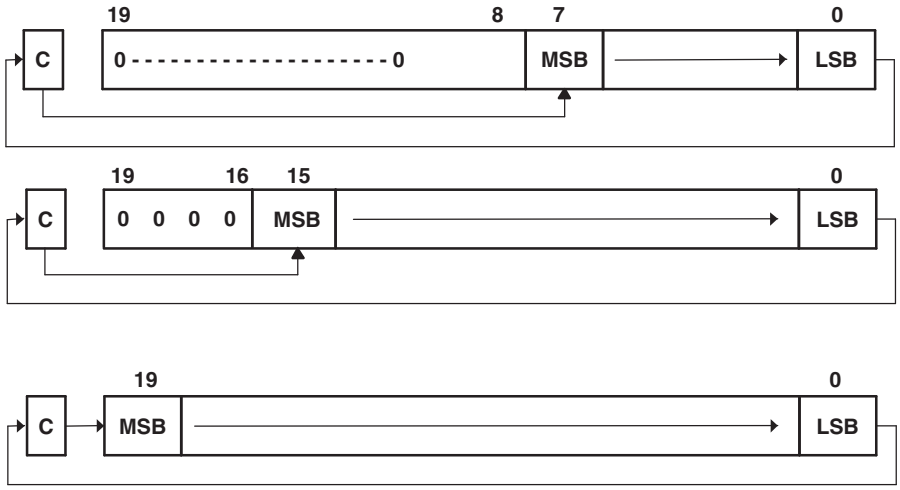


Figure 4-51. Rotate Right Through Carry RRCX(.B,.A) – Register Mode

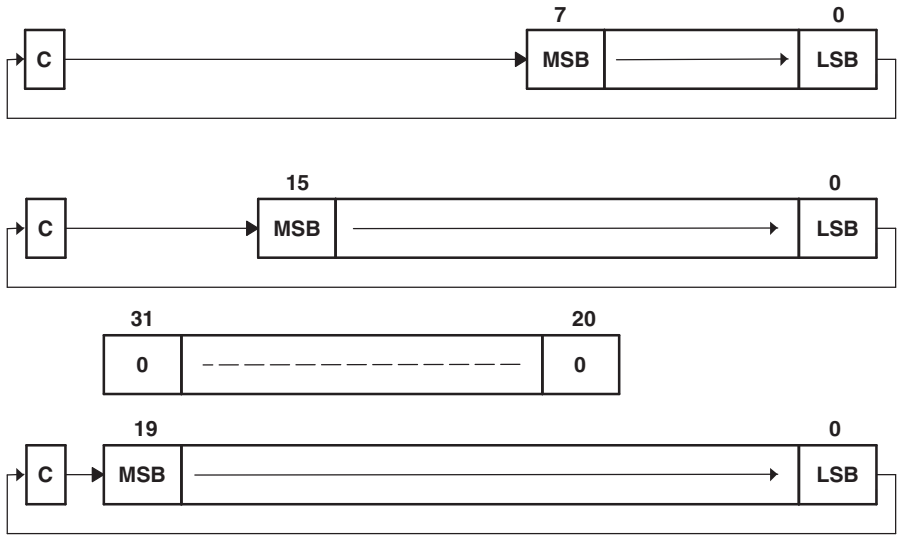


Figure 4-52. Rotate Right Through Carry RRCX(.B,.A) – Non-Register Mode

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.29 RRUM

RRUM.A Rotate right through carry the 20-bit CPU register content

RRUM.[W] Rotate right through carry the 16-bit CPU register content

Syntax

`RRUM.A #n,Rdst` 1 ≤ n ≤ 4

`RRUM.W #n,Rdst` or `RRUM #n,Rdst` 1 ≤ n ≤ 4

Operation 0 → MSB → MSB–1 ... LSB+1 → LSB → C

Description The destination operand is shifted right by one, two, three, or four bit positions as shown in Figure 4-53. Zero is shifted into the MSB, the LSB is shifted into the carry bit. RRUM works like an unsigned division by 2, 4, 8, or 16. The word instruction RRUM.W clears the bits Rdst.19:16.

Note : This instruction does not use the extension word.

Status Bits

N: Set if result is negative
.A: Rdst.19 = 1, reset if Rdst.19 = 0
.W: Rdst.15 = 1, reset if Rdst.15 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the LSB (n = 1), LSB+1 (n = 2), LSB+2 (n = 3), or LSB+3 (n = 4)

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The unsigned address-word in R5 is divided by 16.

```
RRUM.A   #4,R5      ; R5 = R5 » 4. R5/16
```

Example The word in R6 is shifted right by one bit. The MSB R6.15 is loaded with 0.

```
RRUM.W   #1,R6      ; R6 = R6/2. R6.19:15 = 0
```

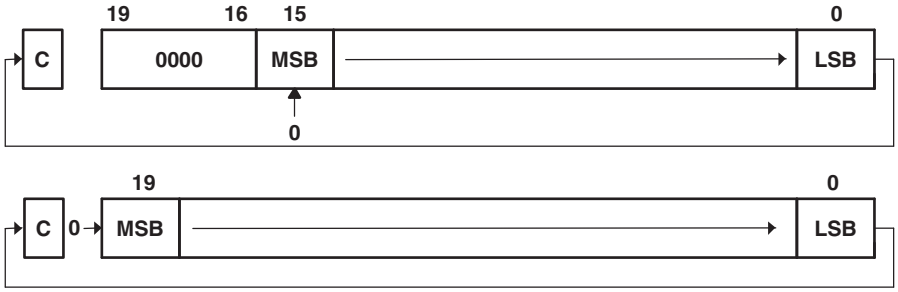


Figure 4-53. Rotate Right Unsigned RRUM[.W] and RRUM.A

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.30 RRUX

RRUX.A Shift right unsigned the 20-bit CPU register content

RRUX.[W] Shift right unsigned the 16-bit CPU register content

RRUX.B Shift right unsigned the 8-bit CPU register content

Syntax

```
RRUX.A Rdst
RRUX.W Rdst
RRUX Rdst
RRUX.B Rdst
```

Operation C=0 → MSB → MSB–1 ... LSB+1 → LSB → C

Description RRUX is valid for register mode only: the destination operand is shifted right by one bit position as shown in Figure 4-54. The word instruction RRUX.W clears the bits Rdst.19:16. The byte instruction RRUX.B clears the bits Rdst.19:8. Zero is shifted into the MSB, the LSB is shifted into the carry bit.

Status Bits

N: Set if result is negative
- .A: dst.19 = 1, reset if dst.19 = 0
- .W: dst.15 = 1, reset if dst.15 = 0
- .B: dst.7 = 1, reset if dst.7 = 0

Z: Set if result is zero, reset otherwise

C: Loaded from the LSB

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The word in R6 is shifted right by 12 positions.

```
RPT      #12
RRUX.W   R6       ; R6 = R6 » 12. R6.19:16 = 0
```

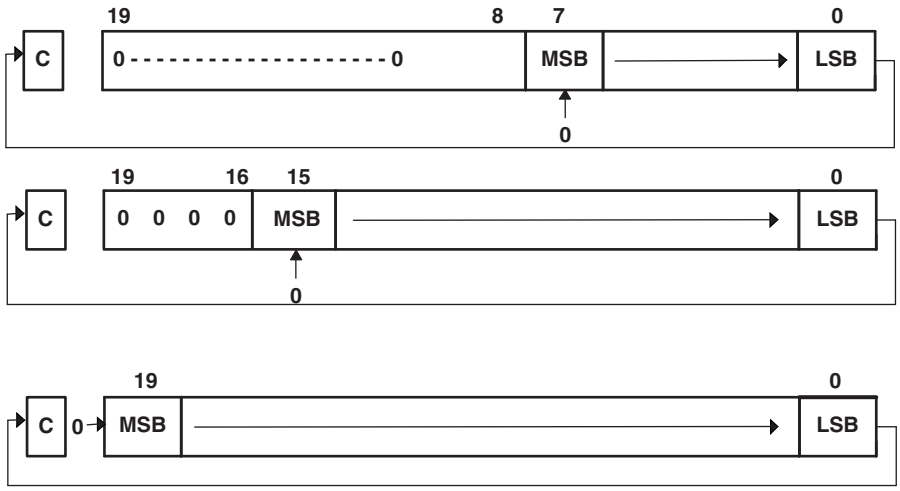


Figure 4-54. Rotate Right Unsigned RRUX(.B,.A) – Register Mode

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.31 SBCX

*** SBCX.A** Subtract borrow (.NOT. carry) from destination address-word

*** SBCX.[W]** Subtract borrow (.NOT. carry) from destination word

*** SBCX.B** Subtract borrow (.NOT. carry) from destination byte

Syntax

```
SBCX.A dst
SBCX dst or          SBCX.W dst
SBCX.B dst
```

Operation

dst + 0FFFFFh + C → dst

dst + 0FFFFh + C → dst

dst + 0FFh + C → dst

Emulation

```
SBCX.A #0,dst
SBCX #0,dst
SBCX.B #0,dst
```

Description The carry bit (C) is added to the destination operand minus one. The previous contents of the destination are lost.

Status Bits

N: Set if result is negative, reset if positive

Z: Set if result is zero, reset otherwise

C: Set if there is a carry from the MSB of the result, reset otherwise
Set to 1 if no borrow, reset if borrow

V: Set if an arithmetic overflow occurs, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 8-bit counter pointed to by R13 is subtracted from a 16-bit counter pointed to by R12.

```
SUBX.B   @R13,0(R12)      ; Subtract LSDs
SBCX.B   1(R12)           ; Subtract carry from MSD
```

NOTE: Borrow implementation

The borrow is treated as a .NOT. carry:

Borrow	Carry Bit
Yes	0
No	1

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.32 SUBX

SUBX.A Subtract source address-word from destination address-word

SUBX.[W] Subtract source word from destination word

SUBX.B Subtract source byte from destination byte

Syntax

```
SUBX.A src,dst
SUBX src,dst or SUBX.W src,dst
SUBX.B src,dst
```

Operation (.not. src) + 1 + dst → dst or dst – src → dst

Description The source operand is subtracted from the destination operand. This is done by adding the 1s complement of the source + 1 to the destination. The source operand is not affected. The result is written to the destination operand. Both operands may be located in the full address space.

Status Bits

N: Set if result is negative (src > dst), reset if positive (src ≤ dst)

Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example A 20-bit constant 87654h is subtracted from EDE (LSBs) and EDE+2 (MSBs).

```
SUBX.A   #87654h,EDE     ; Subtract 87654h from EDE+2|EDE
```

Example A table word pointed to by R5 (20-bit address) is subtracted from R7. Jump to label TONI if R7 contains zero after the instruction. R5 is auto-incremented by two. R7.19:16 = 0.

```
SUBX.W   @R5+,R7         ; Subtract table number from R7. R5 + 2
JZ       TONI            ; R7 = @R5 (before subtraction)
...                      ; R7 <> @R5 (before subtraction)
```

Example Byte CNT is subtracted from the byte R12 points to in the full address space. Address of CNT is within PC ± 512 K.

```
SUBX.B   CNT,0(R12)      ; Subtract CNT from @R12
```

Note: Use SUBA for the following two cases for better density and execution.

```
SUBX.A   Rsrc,Rdst
SUBX.A   #imm20,Rdst
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.33 SUBCX

SUBCX.A Subtract source address-word with carry from destination address-word

SUBCX.[W] Subtract source word with carry from destination word

SUBCX.B Subtract source byte with carry from destination byte

Syntax

```
SUBCX.A src,dst
```

```
SUBCX src,dst or SUBCX.W src,dst
```

```
SUBCX.B src,dst
```

Operation (.not. src) + C + dst → dst or dst – (src – 1) + C → dst

Description The source operand is subtracted from the destination operand. This is made by adding the 1s complement of the source + carry to the destination. The source operand is not affected, the result is written to the destination operand. Both operands may be located in the full address space.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if there is a carry from the MSB, reset otherwise

V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow).

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example A 20-bit constant 87654h is subtracted from R5 with the carry from the previous instruction.

```
SUBCX.A   #87654h,R5      ; Subtract 87654h + C from R5
```

Example A 48-bit number (3 words) pointed to by R5 (20-bit address) is subtracted from a 48-bit counter in RAM, pointed to by R7. R5 auto-increments to point to the next 48-bit number.

```
SUBX.W    @R5+,0(R7)      ; Subtract LSBs. R5 + 2
SUBCX.W   @R5+,2(R7)      ; Subtract MIDs with C. R5 + 2
SUBCX.W   @R5+,4(R7)      ; Subtract MSBs with C. R5 + 2
```

Example Byte CNT is subtracted from the byte R12 points to. The carry of the previous instruction is used. 20-bit addresses.

```
SUBCX.B   &CNT,0(R12)     ; Subtract byte CNT from @R12
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.34 SWPBX

SWPBX.A Swap bytes of lower word

SWPBX.[W] Swap bytes of word

Syntax `SWPBX.A dst`

`SWPBX dst` or `SWPBX.W dst`

Operation dst.15:8 ↔ dst.7:0

Description Register mode: Rn.15:8 are swapped with Rn.7:0. When the .A extension is used, Rn.19:16 are unchanged. When the .W extension is used, Rn.19:16 are cleared.

Other modes: When the .A extension is used, bits 31:20 of the destination address are cleared, bits 19:16 are left unchanged, and bits 15:8 are swapped with bits 7:0. When the .W extension is used, bits 15:8 are swapped with bits 7:0 of the addressed word.

Status Bits Status bits are not affected.

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Exchange the bytes of RAM address-word EDE

```
MOVX.A    #23456h,&EDE      ; 23456h -> EDE
SWPBX.A   EDE               ; 25634h -> EDE
```

Example Exchange the bytes of R5

```
MOVA      #23456h,R5        ; 23456h -> R5
SWPBX.W   R5                ; 05634h -> R5
```

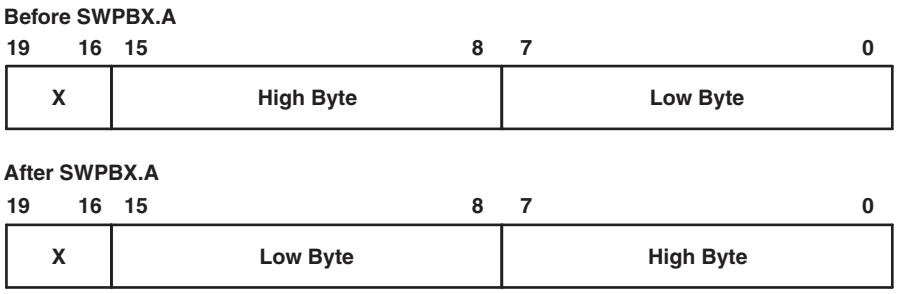


Figure 4-55. Swap Bytes SWPBX.A Register Mode

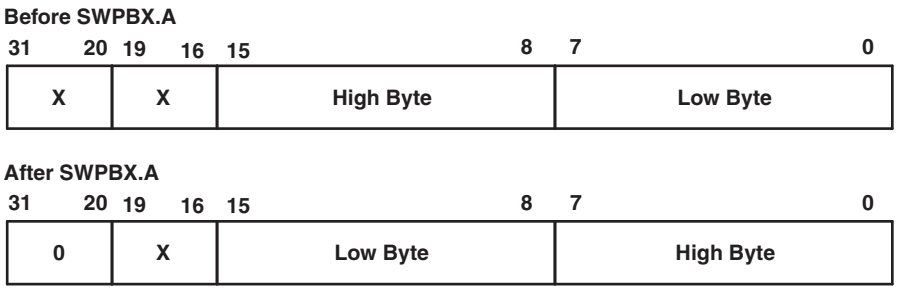


Figure 4-56. Swap Bytes SWPBX.A In Memory

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

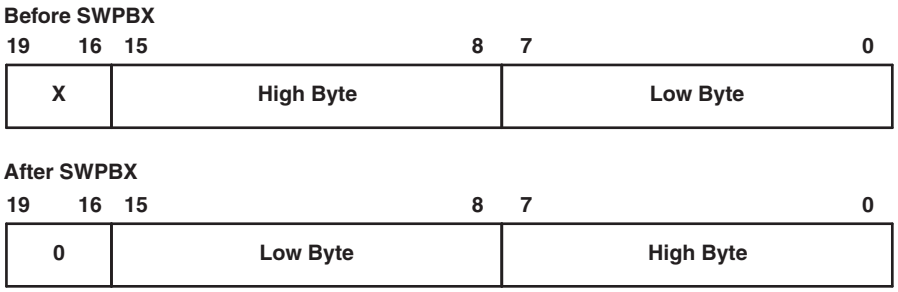


Figure 4-57. Swap Bytes SWPBX[.W] Register Mode

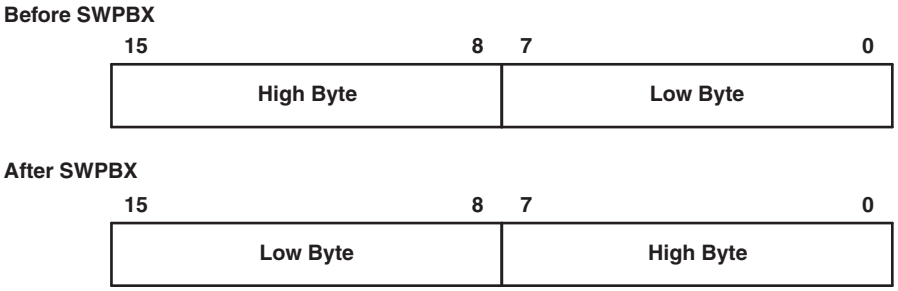


Figure 4-58. Swap Bytes SWPBX[.W] In Memory

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.35 SXTX

SXTX.A Extend sign of lower byte to address-word

SXTX.[W] Extend sign of lower byte to word

Syntax
```
SXTX.A dst
SXTX dst or          SXTX.W dst
```

Operation dst.7 → dst.15:8, Rdst.7 → Rdst.19:8 (Register mode)

Description Register mode: The sign of the low byte of the operand (Rdst.7) is extended into the bits Rdst.19:8.

Other modes: SXTX.A: the sign of the low byte of the operand (dst.7) is extended into dst.19:8. The bits dst.31:20 are cleared.

SXTX[.W]: the sign of the low byte of the operand (dst.7) is extended into dst.15:8.

Status Bits
- N: Set if result is negative, reset otherwise
- Z: Set if result is zero, reset otherwise
- C: Set if result is not zero, reset otherwise (C = .not.Z)
- V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The signed 8-bit data in EDE.7:0 is sign extended to 20 bits: EDE.19:8. Bits 31:20 located in EDE+2 are cleared.

```
SXTX.A      &EDE            ; Sign extended EDE -> EDE+2/EDE
```

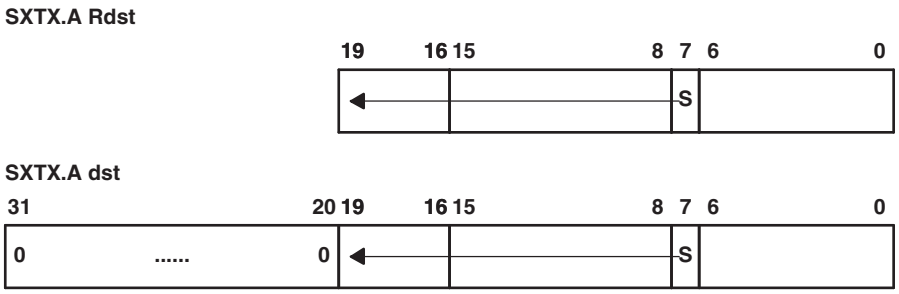


Figure 4-59. Sign Extend SXTX.A

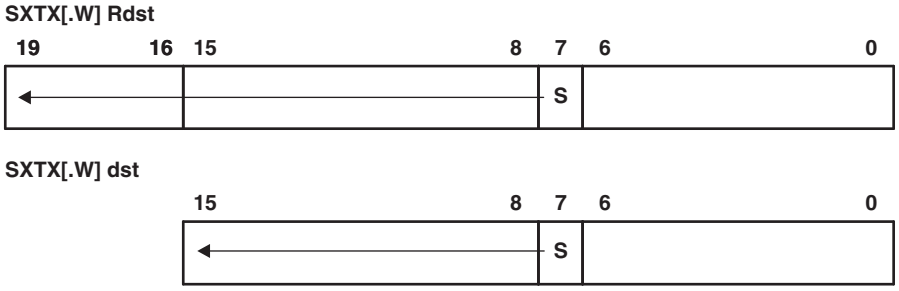


Figure 4-60. Sign Extend SXTX[.W]

Copyright © 2009–2013, Texas Instruments Incorporated

4.6.3.36 TSTX

*** TSTX.A** Test destination address-word

*** TSTX.[W]** Test destination word

*** TSTX.B** Test destination byte

Syntax

```
TSTX.A dst
TSTX dst or          TSTX.W dst
TSTX.B dst
```

Operation

dst + 0FFFFFh + 1

dst + 0FFFFh + 1

dst + 0FFh + 1

Emulation

```
CMPX.A #0,dst
CMPX #0,dst
CMPX.B #0,dst
```

Description The destination operand is compared with zero. The status bits are set according to the result. The destination is not affected.

Status Bits

N: Set if destination is negative, reset if positive

Z: Set if destination contains zero, reset otherwise

C: Set

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example RAM byte LEO is tested; PC is pointing to upper memory. If it is negative, continue at LEONEG; if it is positive but not zero, continue at LEOPOS.

```
            TSTX.B   LEO          ; Test LEO
            JN       LEONEG       ; LEO is negative
            JZ       LEOZERO      ; LEO is zero
LEOPOS      ......                ; LEO is positive but not zero
LEONEG      ......                ; LEO is negative
LEOZERO     ......                ; LEO is zero
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.3.37 XORX

XORX.A Exclusive OR source address-word with destination address-word

XORX.[W] Exclusive OR source word with destination word

XORX.B Exclusive OR source byte with destination byte

Syntax

`XORX.A src,dst`

`XORX src,dst` or `XORX.W src,dst`

`XORX.B src,dst`

Operation src .xor. dst → dst

Description The source and destination operands are exclusively ORed. The result is placed into the destination. The source operand is not affected. The previous contents of the destination are lost. Both operands may be located in the full address space.

Status Bits

N: Set if result is negative (MSB = 1), reset if positive (MSB = 0)

Z: Set if result is zero, reset otherwise

C: Set if result is not zero, reset otherwise (carry = .not. Zero)

V: Set if both operands are negative (before execution), reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Toggle bits in address-word CNTR (20-bit data) with information in address-word TONI (20-bit address)

```
XORX.A   TONI,&CNTR      ; Toggle bits in CNTR
```

Example A table word pointed to by R5 (20-bit address) is used to toggle bits in R6.

```
XORX.W   @R5,R6          ; Toggle bits in R6. R6.19:16 = 0
```

Example Reset to zero those bits in the low byte of R7 that are different from the bits in byte EDE (20-bit address)

```
XORX.B   EDE,R7          ; Set different bits to 1 in R7
INV.B    R7              ; Invert low byte of R7. R7.19:8 = 0.
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4 Address Instructions

MSP430X address instructions are instructions that support 20-bit operands but have restricted addressing modes. The addressing modes are restricted to the Register mode and the Immediate mode, except for the MOVA instruction. Restricting the addressing modes removes the need for the additional extension-word op-code improving code density and execution time. The MSP430X address instructions are listed and described in the following pages.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.1 ADDA

ADDA Add 20-bit source to a 20-bit destination register

Syntax
```
ADDA Rsrc,Rdst
ADDA #imm20,Rdst
```

Operation src + Rdst → Rdst

Description The 20-bit source operand is added to the 20-bit destination CPU register. The previous contents of the destination are lost. The source operand is not affected.

Status Bits
- N: Set if result is negative (Rdst.19 = 1), reset if positive (Rdst.19 = 0)
- Z: Set if result is zero, reset otherwise
- C: Set if there is a carry from the 20-bit result, reset otherwise
- V: Set if the result of two positive operands is negative, or if the result of two negative numbers is positive, reset otherwise

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example R5 is increased by 0A4320h. The jump to TONI is performed if a carry occurs.

```
ADDA    #0A4320h,R5      ; Add A4320h to 20-bit R5
JC      TONI             ; Jump on carry
...                      ; No carry occurred
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.4.2 BRA

*** BRA** Branch to destination

Syntax `BRA dst`

Operation dst → PC

Emulation `MOVA dst,PC`

Description An unconditional branch is taken to a 20-bit address anywhere in the full address space. All seven source addressing modes can be used. The branch instruction is an address-word instruction. If the destination address is contained in a memory location X, it is contained in two ascending words: X (LSBs) and (X + 2) (MSBs).

Status Bits
N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Examples Examples for all addressing modes are given.

Immediate mode: Branch to label EDE located anywhere in the 20-bit address space or branch directly to address.

```
BRA     #EDE          ; MOVA   #imm20,PC
BRA     #01AA04h
```

Symbolic mode: Branch to the 20-bit address contained in addresses EXEC (LSBs) and EXEC+2 (MSBs). EXEC is located at the address (PC + X) where X is within +32 K. Indirect addressing.

```
BRA     EXEC          ; MOVA   z16(PC),PC
```

Note: If the 16-bit index is not sufficient, a 20-bit index may be used with the following instruction.

```
MOVX.A  EXEC,PC       ; 1M byte range with 20-bit index
```

Absolute mode: Branch to the 20-bit address contained in absolute addresses EXEC (LSBs) and EXEC+2 (MSBs). Indirect addressing.

```
BRA     &EXEC         ; MOVA   &abs20,PC
```

Register mode: Branch to the 20-bit address contained in register R5. Indirect R5.

```
BRA     R5            ; MOVA   R5,PC
```

Indirect mode: Branch to the 20-bit address contained in the word pointed to by register R5 (LSBs). The MSBs have the address (R5 + 2). Indirect, indirect R5.

```
BRA     @R5           ; MOVA   @R5,PC
```


Copyright © 2009–2013, Texas Instruments Incorporated

Indirect, Auto-Increment mode: Branch to the 20-bit address contained in the words pointed to by register R5 and increment the address in R5 afterwards by 4. The next time the software flow uses R5 as a pointer, it can alter the program execution due to access to the next address in the table pointed to by R5. Indirect, indirect R5.

```
BRA      @R5+          ; MOVA   @R5+,PC. R5 + 4
```

Indexed mode: Branch to the 20-bit address contained in the address pointed to by register (R5 + X) (for example, a table with addresses starting at X). (R5 + X) points to the LSBs, (R5 + X + 2) points to the MSBs of the address. X is within R5 + 32 K. Indirect, indirect (R5 + X).

```
BRA      X(R5)         ; MOVA   z16(R5),PC
```

Note: If the 16-bit index is not sufficient, a 20-bit index X may be used with the following instruction:

```
MOVX.A   X(R5),PC      ; 1M byte range with 20-bit index
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.4.3 CALLA

CALLA Call a subroutine

Syntax `CALLA dst`

Operation
dst → tmp 20-bit dst is evaluated and stored
SP – 2 → SP
PC.19:16 → @SP updated PC with return address to TOS (MSBs)
SP – 2 → SP
PC.15:0 → @SP updated PC to TOS (LSBs)
tmp → PC saved 20-bit dst to PC

Description A subroutine call is made to a 20-bit address anywhere in the full address space. All seven source addressing modes can be used. The call instruction is an address-word instruction. If the destination address is contained in a memory location X, it is contained in two ascending words, X (LSBs) and (X + 2) (MSBs). Two words on the stack are needed for the return address. The return is made with the instruction RETA.

Status Bits
N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Examples Examples for all addressing modes are given.

Immediate mode: Call a subroutine at label EXEC or call directly an address.

```
CALLA   #EXEC          ; Start address EXEC
CALLA   #01AA04h       ; Start address 01AA04h
```

Symbolic mode: Call a subroutine at the 20-bit address contained in addresses EXEC (LSBs) and EXEC+2 (MSBs). EXEC is located at the address (PC + X) where X is within +32 K. Indirect addressing.

```
CALLA   EXEC           ; Start address at @EXEC. z16(PC)
```

Absolute mode: Call a subroutine at the 20-bit address contained in absolute addresses EXEC (LSBs) and EXEC+2 (MSBs). Indirect addressing.

```
CALLA   &EXEC          ; Start address at @EXEC
```

Register mode: Call a subroutine at the 20-bit address contained in register R5. Indirect R5.

```
CALLA   R5             ; Start address at @R5
```

Indirect mode: Call a subroutine at the 20-bit address contained in the word pointed to by register R5 (LSBs). The MSBs have the address (R5 + 2). Indirect, indirect R5.

```
CALLA   @R5            ; Start address at @R5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Indirect, Auto-Increment mode: Call a subroutine at the 20-bit address contained in the words pointed to by register R5 and increment the 20-bit address in R5 afterwards by 4. The next time the software flow uses R5 as a pointer, it can alter the program execution due to access to the next word address in the table pointed to by R5. Indirect, indirect R5.

```
CALLA   @R5+          ; Start address at @R5. R5 + 4
```

Indexed mode: Call a subroutine at the 20-bit address contained in the address pointed to by register (R5 + X); for example, a table with addresses starting at X. (R5 + X) points to the LSBs, (R5 + X + 2) points to the MSBs of the word address. X is within R5 + 32 K. Indirect, indirect (R5 + X).

```
CALLA   X(R5)         ; Start address at @(R5+X). z16(R5)
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.4 CLRA

*** CLRA**	Clear 20-bit destination register
Syntax	`CLRA Rdst`
Operation	0 → Rdst
Emulation	`MOVA #0,Rdst`
Description	The destination register is cleared.
Status Bits	Status bits are not affected.
Example	The 20-bit value in R10 is cleared.

```
CLRA   R10      ; 0 -> R10
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.5 CMPA

CMPA Compare the 20-bit source with a 20-bit destination register

Syntax
```
CMPA Rsrc,Rdst
CMPA #imm20,Rdst
```

Operation (.not. src) + 1 + Rdst or Rdst – src

Description The 20-bit source operand is subtracted from the 20-bit destination CPU register. This is made by adding the 1s complement of the source + 1 to the destination register. The result affects only the status bits.

Status Bits
- N: Set if result is negative (src > dst), reset if positive (src ≤ dst)
- Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)
- C: Set if there is a carry from the MSB, reset otherwise
- V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example A 20-bit immediate operand and R6 are compared. If they are equal, the program continues at label EQUAL.

```
CMPA    #12345h,R6      ; Compare R6 with 12345h
JEQ     EQUAL           ; R6 = 12345h
...                     ; Not equal
```

Example The 20-bit values in R5 and R6 are compared. If R5 is greater than (signed) or equal to R6, the program continues at label GRE.

```
CMPA    R6,R5           ; Compare R6 with R5 (R5 - R6)
JGE     GRE             ; R5 >= R6
...                     ; R5 < R6
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.6 DECDA

*** DECDA**	Double-decrement 20-bit destination register
Syntax	`DECDA Rdst`
Operation	Rdst – 2 → Rdst
Emulation	`SUBA #2,Rdst`
Description	The destination register is decremented by two. The original contents are lost.
Status Bits	N: Set if result is negative, reset if positive Z: Set if Rdst contained 2, reset otherwise C: Reset if Rdst contained 0 or 1, set otherwise V: Set if an arithmetic overflow occurs, otherwise reset
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The 20-bit value in R5 is decremented by 2.

```
DECDA   R5      ; Decrement R5 by two
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.7 INCDA

*** INCDA**	Double-increment 20-bit destination register
Syntax	`INCDA Rdst`
Operation	Rdst + 2 → Rdst
Emulation	`ADDA #2,Rdst`
Description	The destination register is incremented by two. The original contents are lost.
Status Bits	N: Set if result is negative, reset if positive Z: Set if Rdst contained 0FFFFEh, reset otherwise Set if Rdst contained 0FFFEh, reset otherwise Set if Rdst contained 0FEh, reset otherwise C: Set if Rdst contained 0FFFFEh or 0FFFFFh, reset otherwise Set if Rdst contained 0FFFEh or 0FFFFh, reset otherwise Set if Rdst contained 0FEh or 0FFh, reset otherwise V: Set if Rdst contained 07FFFEh or 07FFFFh, reset otherwise Set if Rdst contained 07FFEh or 07FFFh, reset otherwise Set if Rdst contained 07Eh or 07Fh, reset otherwise
Mode Bits	OSCOFF, CPUOFF, and GIE are not affected.
Example	The 20-bit value in R5 is incremented by two.

```
INCDA   R5      ; Increment R5 by two
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.4.8 MOVA

MOVA Move the 20-bit source to the 20-bit destination

Syntax

```
MOVA Rsrc,Rdst
MOVA #imm20,Rdst
MOVA z16(Rsrc),Rdst
MOVA EDE,Rdst
MOVA &abs20,Rdst
MOVA @Rsrc,Rdst
MOVA @Rsrc+,Rdst
MOVA Rsrc,z16(Rdst)
MOVA Rsrc,&abs20
```

Operation src → Rdst

Rsrc → dst

Description The 20-bit source operand is moved to the 20-bit destination. The source operand is not affected. The previous content of the destination is lost.

Status Bits

N: Not affected

Z: Not affected

C: Not affected

V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Examples Copy 20-bit value in R9 to R8

```
MOVA   R9,R8          ; R9 -> R8
```

Write 20-bit immediate value 12345h to R12

```
MOVA   #12345h,R12    ; 12345h -> R12
```

Copy 20-bit value addressed by (R9 + 100h) to R8. Source operand in addresses (R9 + 100h) LSBs and (R9 + 102h) MSBs.

```
MOVA   100h(R9),R8       ; Index: + 32 K. 2 words transferred
```

Move 20-bit value in 20-bit absolute addresses EDE (LSBs) and EDE+2 (MSBs) to R12

```
MOVA   &EDE,R12          ; &EDE -> R12. 2 words transferred
```

Move 20-bit value in 20-bit addresses EDE (LSBs) and EDE+2 (MSBs) to R12. PC index ± 32 K.

```
MOVA   EDE,R12           ; EDE -> R12. 2 words transferred
```

Copy 20-bit value R9 points to (20 bit address) to R8. Source operand in addresses @R9 LSBs and @(R9 + 2) MSBs.

```
MOVA   @R9,R8            ; @R9 -> R8. 2 words transferred
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Copy 20-bit value R9 points to (20 bit address) to R8. R9 is incremented by four afterwards. Source operand in addresses @R9 LSBs and @(R9 + 2) MSBs.

```
MOVA    @R9+,R8          ; @R9 -> R8. R9 + 4. 2 words transferred.
```

Copy 20-bit value in R8 to destination addressed by (R9 + 100h). Destination operand in addresses @(R9 + 100h) LSBs and @(R9 + 102h) MSBs.

```
MOVA    R8,100h(R9)      ; Index: +- 32 K. 2 words transferred
```

Move 20-bit value in R13 to 20-bit absolute addresses EDE (LSBs) and EDE+2 (MSBs)

```
MOVA    R13,&EDE         ; R13 -> EDE. 2 words transferred
```

Move 20-bit value in R13 to 20-bit addresses EDE (LSBs) and EDE+2 (MSBs). PC index ± 32 K.

```
MOVA    R13,EDE          ; R13 -> EDE. 2 words transferred
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.9 RETA

*** RETA** Return from subroutine

Syntax `RETA`

Operation
@SP → PC.15:0 LSBs (15:0) of saved PC to PC.15:0
SP + 2 → SP
@SP → PC.19:16 MSBs (19:16) of saved PC to PC.19:16
SP + 2 → SP

Emulation `MOVA @SP+,PC`

Description The 20-bit return address information, pushed onto the stack by a CALLA instruction, is restored to the PC. The program continues at the address following the subroutine call. The SR bits SR.11:0 are not affected. This allows the transfer of information with these bits.

Status Bits
N: Not affected
Z: Not affected
C: Not affected
V: Not affected

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example Call a subroutine SUBR from anywhere in the 20-bit address space and return to the address after the CALLA

```
        CALLA     #SUBR       ; Call subroutine starting at SUBR
        ...                   ; Return by RETA to here
SUBR    PUSHM.A   #2,R14      ; Save R14 and R13 (20 bit data)
        ...                   ; Subroutine code
        POPM.A    #2,R14      ; Restore R13 and R14 (20 bit data)
        RETA                  ; Return (to full address space)
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

4.6.4.10 SUBA

SUBA Subtract 20-bit source from 20-bit destination register

Syntax
```
SUBA Rsrc,Rdst
SUBA #imm20,Rdst
```

Operation (.not.src) + 1 + Rdst → Rdst or Rdst – src → Rdst

Description The 20-bit source operand is subtracted from the 20-bit destination register. This is made by adding the 1s complement of the source + 1 to the destination. The result is written to the destination register, the source is not affected.

Status Bits
- N: Set if result is negative (src > dst), reset if positive (src ≤ dst)
- Z: Set if result is zero (src = dst), reset otherwise (src ≠ dst)
- C: Set if there is a carry from the MSB (Rdst.19), reset otherwise
- V: Set if the subtraction of a negative source operand from a positive destination operand delivers a negative result, or if the subtraction of a positive source operand from a negative destination operand delivers a positive result, reset otherwise (no overflow)

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit value in R5 is subtracted from R6. If a carry occurs, the program continues at label TONI.

```
SUBA  R5,R6     ; R6 - R5 -> R6
JC    TONI      ; Carry occurred
...             ; No carry
```


Copyright © 2009–2013, Texas Instruments Incorporated

4.6.4.11 TSTA

*** TSTA** Test 20-bit destination register

Syntax `TSTA Rdst`

Operation dst + 0FFFFFh + 1

dst + 0FFFFh + 1

dst + 0FFh + 1

Emulation `CMPA #0,Rdst`

Description The destination register is compared with zero. The status bits are set according to the result. The destination register is not affected.

Status Bits N: Set if destination register is negative, reset if positive

Z: Set if destination register contains zero, reset otherwise

C: Set

V: Reset

Mode Bits OSCOFF, CPUOFF, and GIE are not affected.

Example The 20-bit value in R7 is tested. If it is negative, continue at R7NEG; if it is positive but not zero, continue at R7POS.

```
            TSTA   R7          ; Test R7
            JN     R7NEG       ; R7 is negative
            JZ     R7ZERO      ; R7 is zero
R7POS       ......             ; R7 is positive but not zero
R7NEG       ......             ; R7 is negative
R7ZERO      ......             ; R7 is zero
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Flash Memory Controller

This chapter describes the operation of the flash memory controller.

Topic **Page**

5.1 Flash Memory Introduction 289
5.2 Flash Memory Segmentation 290
5.3 Flash Memory Operation 292
5.4 FCTL Registers 307

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.1 Flash Memory Introduction

The flash memory is byte, word, and long-word addressable and programmable. The flash memory module has an integrated controller that controls programming and erase operations. The module contains three registers, a timing generator, and a voltage generator to supply program and erase voltages. The cumulative high-voltage time must not be exceeded, and each 32-bit word can be written not more than four times (in byte, word, or long word write modes) before another erase cycle (see device-specific data sheet for details).

The flash memory features include:

- Internal programming voltage generation
- Byte, word (2 bytes), and long (4 bytes) programmable
- Ultralow power operation
- Segment erase, bank erase (device specific), and mass erase
- Marginal 0 and marginal 1 read modes
- Each bank (device specific) can be erased individually while program execution can proceed in a different flash bank.

NOTE: Bank operations are not supported on all devices. See the device-specific data sheet for banks supported and their respective sizes.

The block diagram of the flash memory and controller is shown in Figure 5-1.

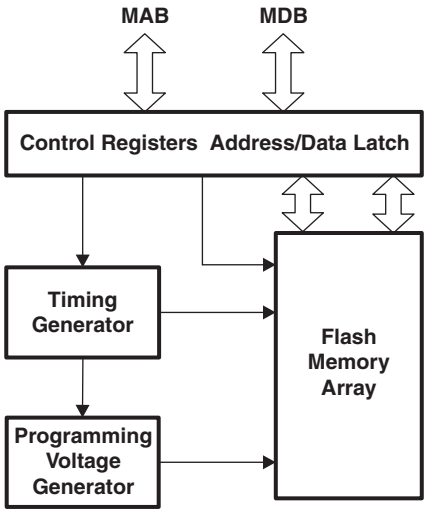


Figure 5-1. Flash Memory Module Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.2 Flash Memory Segmentation

The flash main memory is partitioned into 512-byte segments. Single bits, bytes, or words can be written to flash memory, but a segment is the smallest size of the flash memory that can be erased.

The flash memory is partitioned into main and information memory sections. There is no difference in the operation of the main and information memory sections. Code and data can be located in either section. The difference between the sections is the segment size.

There are four information memory segments, A through D. Each information memory segment contains 128 bytes and can be erased individually.

The bootstrap loader (BSL) memory consists of four segments, A through D. Each BSL memory segment contains 512 bytes and can be erased individually.

The main memory segment size is 512 byte. See the device-specific data sheet for the start and end addresses of each bank, when available, and for the complete memory map of a device.

Figure 5-2 shows the flash segmentation using an example of 256-KB flash that has four banks of 64 KB (segments A through D) and information memory.

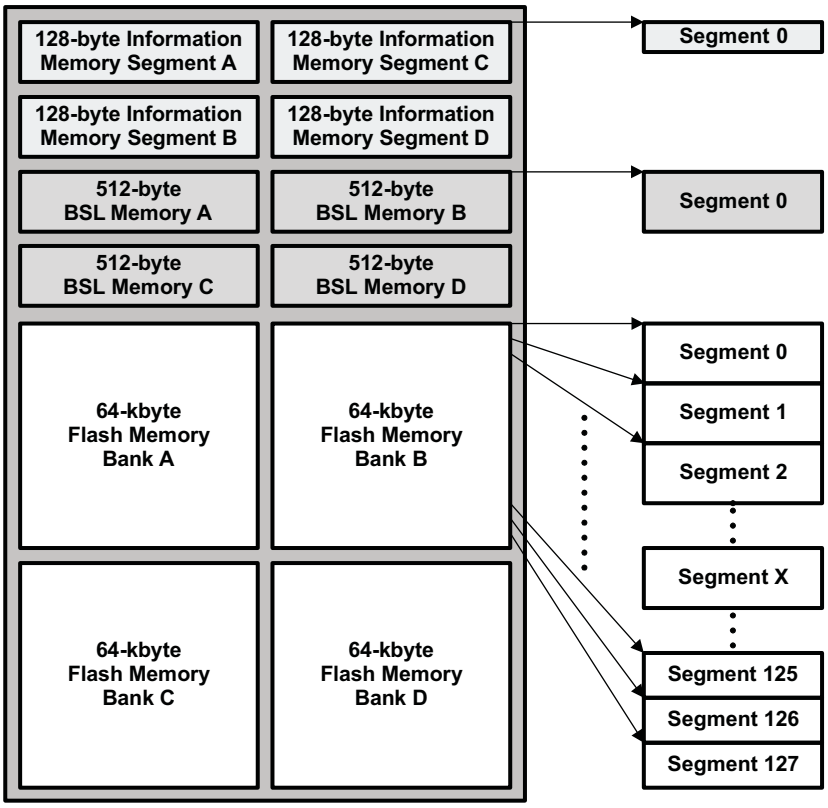


Figure 5-2. 256-KB Flash Memory Segments Example

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.2.1 Segment A

Segment A of the information memory is locked separately from all other segments with the LOCKA bit. If LOCKA = 1, segment A cannot be written or erased, and all information memory is protected from being segment erased. If LOCKA = 0, segment A can be erased and written like any other flash memory segment.

The state of the LOCKA bit is toggled when a 1 is written to it. Writing a 0 to LOCKA has no effect. This allows existing flash programming routines to be used unchanged.

```
; Unlock Info Memory
   MOV      #FWPW,&FCTL4             ; Clear LOCKINFO, if set
; Unlock SegmentA
   BIT      #LOCKA,&FCTL3            ; Test LOCKA
   JZ       SEGA_UNLOCKED            ; Already unlocked?
   MOV      #FWPW+LOCKA,&FCTL3       ; No, unlock SegmentA
SEGA_UNLOCKED                       ; Yes, continue
; SegmentA is unlocked


; Lock SegmentA
   BIT      #LOCKA,&FCTL3            ; Test LOCKA
   JNZ      SEGA_LOCKED              ; Already locked?
   MOV      #FWPW+LOCKA,&FCTL3       ; No, lock SegmentA
SEGA_LOCKED                         ; Yes, continue
; SegmentA is locked
; Lock Info Memory
   MOV      #FWPW+LOCKINFO,&FCTL4    ; Set LOCKINFO
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3 Flash Memory Operation

The default mode of the flash memory is read mode. In read mode, the flash memory is not being erased or written, the flash timing generator and voltage generator are off, and the memory operates identically to ROM.

Read and fetch while erase – The flash memory allows execution of a program from flash while a different flash bank is erased. Data reads are also possible from any flash bank not being erased.

NOTE: Read and fetch while erase

The read and fetch while erase feature is available in flash memory configurations where more than one flash bank is available. If there is one flash bank available, holding the complete flash program memory, the read from the program memory and information memory and BSL memory during the erase is not provided. Table 5-1 summarizes which flash operations are supported for devices that support read and fetch while erasing.

Table 5-1. Supported Simultaneous Code Execution and Flash Operations

Flash Operation	Simultaneous Code Execution	
	Within Flash	Within RAM
Bank Erase	Supported Executed code must not reside in the bank to be erased	Supported
Segment Erase	Not Supported	Supported
Byte, word, long-word write	Not supported	Supported

Flash memory is in-system programmable (ISP) without the need for additional external voltage. The CPU can program the flash memory. The flash memory write and erase modes are selected by the BLKWRT, WRT, MERAS, and ERASE bits and are:

- Byte, word, or long-word (32-bit) write
- Block write
- Segment erase
- Bank erase (only main memory)
- Mass erase (all main memory banks)
- Read during bank erase (except for the one currently read from)

Reading or writing to flash memory while it is busy programming or erasing (page, mass, or bank) from the same bank is prohibited. Any flash erase or programming can be initiated from within flash memory or RAM.

5.3.1 Erasing Flash Memory

The logical value of an erased flash memory bit is 1. Each bit can be programmed from 1 to 0 individually, but to reprogram from 0 to 1 requires an erase cycle. The smallest amount of flash that can be erased is one segment. There are three erase modes selected by the ERASE and MERAS bits listed in Table 5-2.

Table 5-2. Erase Modes

MERAS	ERASE	Erase Mode
0	1	Segment erase
1	0	Bank erase (of one bank) selected by the dummy write address(1)
1	1	Mass erase (all memory banks are erased. Information memory A to D and BSL segments A to D are not erased)

(1) Bank operations are not supported on all devices. See the device-specific data sheet for support of bank operations.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.1.1 Erase Cycle

An erase cycle is initiated by a dummy write to the address range of the segment to be erased. The dummy write starts the erase operation and is required for all erase operations including mass erase. Figure 5-3 shows the erase cycle timing. The BUSY bit is set immediately after the dummy write and remains set throughout the erase cycle. BUSY, MERAS, and ERASE are automatically cleared when the cycle completes. No additional dummy write access should be made while the control bits are cleared, otherwise, ACCVIFG is set. The mass erase cycle timing is not dependent on the amount of flash memory present on a device. Erase cycle times are equivalent for all devices.

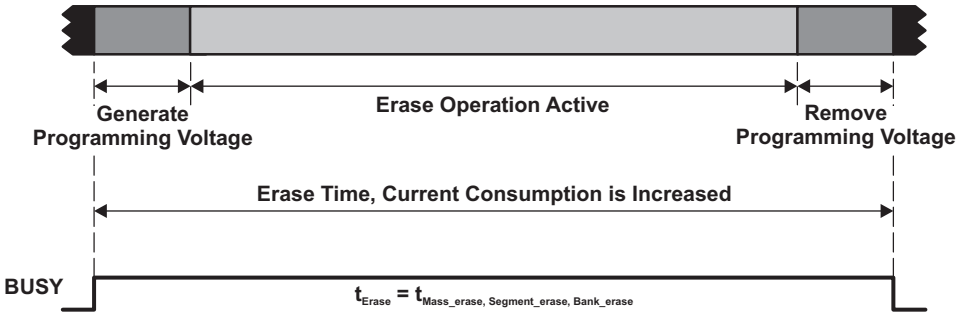


Figure 5-3. Erase Cycle Timing

5.3.1.2 Erasing Main Memory

The main memory consists of one or more banks. Each bank can be erased individually (bank erase). All main memory banks can be erased in the mass erase mode.

5.3.1.3 Erasing Information Memory or BSL Flash Segments

The information memory A to D and the BSL segments A to D can only be erased in segment erase mode. They are not erased during a bank erase or a mass erase. Erasing is only possible by first clearing the LOCKINFO bit.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.1.4 Initiating Erase From Flash

An erase cycle can be initiated from within flash memory. During a bank erase, code can be executed from flash or RAM. The executed code cannot be located in a bank to be erased.

For any segment erase, the CPU is held until the erase cycle completes regardless of the bank the code resides in. After the segment erase cycle ends, the CPU resumes code execution with the instruction following the dummy write.

When initiating an erase cycle from within flash memory, it is possible to erase the code needed for execution after the erase operation. If this occurs, CPU execution is unpredictable after the erase cycle.

The flow to initiate an erase from flash is shown in Figure 5-4.

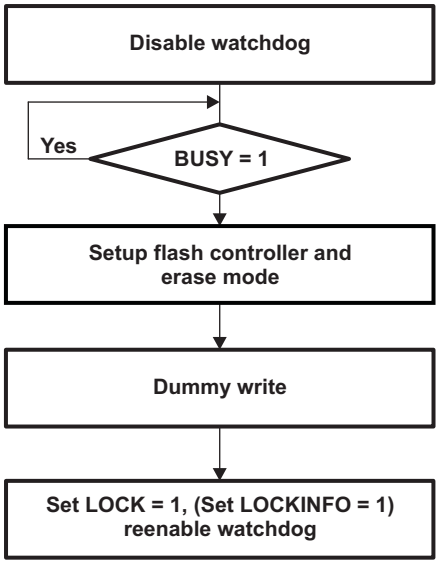


Figure 5-4. Erase Cycle From Flash

```
; Segment Erase from flash.
; Assumes Program Memory. Information memory or BSL
; requires LOCKINFO to be cleared as well.
; Assumes ACCVIE = NMIIE = OFIE = 0.
    MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
L1  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L1                         ; Loop while busy
    MOV   #FWPW,&FCTL3               ; Clear LOCK
    MOV   #FWPW+ERASE,&FCTL1         ; Enable segment erase
    CLR   &0FC10h                    ; Dummy write
L2  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L2                         ; Loop while busy
    MOV   #FWPW+LOCK,&FCTL3          ; Done, set LOCK
    ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.1.5 Initiating Erase From RAM

An erase cycle can be initiated from RAM. In this case, the CPU is not held and continues to execute code from RAM. The mass erase (all main memory banks) operation is initiated while executing from RAM. The BUSY bit is used to determine the end of the erase cycle. If the flash is busy completing a bank erase, flash addresses of a different bank can be used to read data or to fetch instructions. While the flash is BUSY, starting an erase cycle or a programming cycle causes an access violation, ACCIFG is set to 1, and the result of the erase operation is unpredictable.

The flow to initiate an erase from flash from RAM is shown in Figure 5-5.

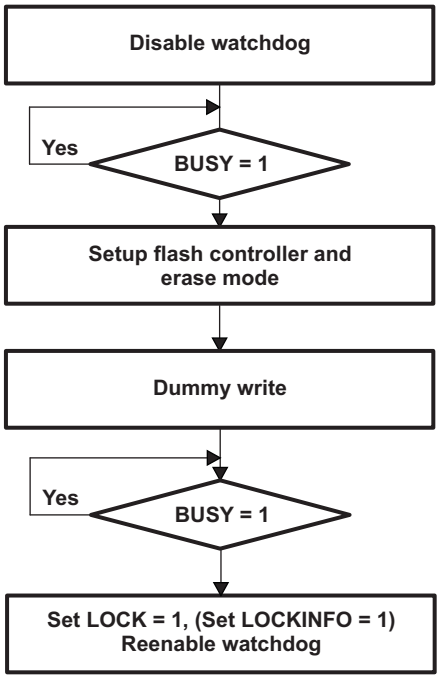


Figure 5-5. Erase Cycle From RAM

```
; segment Erase from RAM.
; Assumes Program Memory. Information memory or BSL
; requires LOCKINFO to be cleared as well.
; Assumes ACCVIE = NMIIE = OFIE = 0.
    MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
L1  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L1                         ; Loop while busy
    MOV   #FWPW,&FCTL3               ; Clear LOCK
    MOV   #FWPW+ERASE,&FCTL1         ; Enable page erase
    CLR   &0FC10h                    ; Dummy write
L2  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L2                         ; Loop while busy
    MOV   #FWPW+LOCK,&FCTL3          ; Done, set LOCK
    ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2 Writing Flash Memory

The write modes, selected by the WRT and BLKWRT bits, are listed in Table 5-3.

Table 5-3. Write Modes

BLKWRT	WRT	Write Mode
0	1	Byte or word write
1	0	Long-word write
1	1	Long-word block write

The write modes use a sequence of individual write instructions. Using the long-word write mode is approximately twice as fast as the byte or word mode. Using the long-word block write mode is approximately four times faster than byte or word mode, because the voltage generator remains on for the complete block write, and long-words are written in parallel. Any instruction that modifies a destination can be used to modify a flash location in either byte or word write mode, long-word write mode, or block long-word write mode.

The BUSY bit is set while the write operation is active and cleared when the operation completes. If the write operation is initiated from RAM, the CPU must not access flash while BUSY is set to 1. Otherwise, an access violation occurs, ACCVIFG is set, and the flash write is unpredictable.

5.3.2.1 Byte or Word Write

A byte or word write operation can be initiated from within flash memory or from RAM. When initiating from within flash memory, the CPU is held while the write completes. After the write completes, the CPU resumes code execution with the instruction following the write access. The byte, word, and long-word write timing is shown in Figure 5-6. Byte, word, and long-word write times are identical.

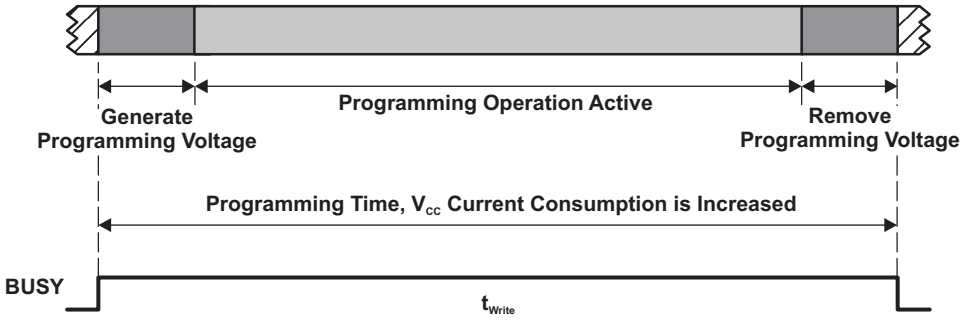


Figure 5-6. Byte, Word, and Long-Word Write Timing

When a byte or word write is executed from RAM, the CPU continues to execute code from RAM. The BUSY bit must be zero before the CPU accesses flash again, otherwise an access violation occurs, ACCVIFG is set, and the write result is unpredictable.

In any write mode, the internally-generated programming voltage is applied to the complete 128-byte block. The cumulative programming time, t_{CPT}, must not be exceeded for any block. Each byte, word, or long-word write adds to the cumulative program time of a segment. If the maximum cumulative program time is reached or exceeded, the segment must be erased. Further programming or using the data returns unpredictable results (see the device-specific data sheet for specifications).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.2 Initiating Byte or Word Write From Flash

The flow to initiate a byte or word write from flash is shown in Figure 5-7.

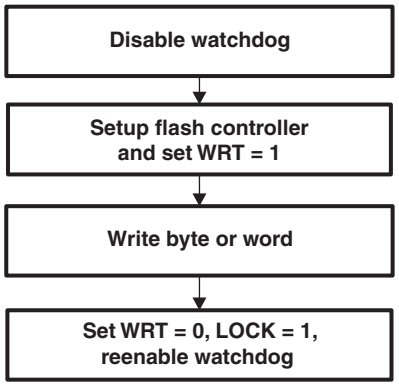


Figure 5-7. Initiating a Byte or Word Write From Flash

```
; Byte or word write from flash.
; Assumes 0x0FF1E is already erased
; Assumes ACCVIE = NMIIE = OFIE = 0.
   MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
   MOV   #FWPW,&FCTL3               ; Clear LOCK
   MOV   #FWPW+WRT,&FCTL1           ; Enable write
   MOV   #0123h,&0FF1Eh             ; 0123h -> 0x0FF1E
   MOV   #FWPW,&FCTL1               ; Done. Clear WRT
   MOV   #FWPW+LOCK,&FCTL3          ; Set LOCK
   ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.3 Initiating Byte or Word Write From RAM

The flow to initiate a byte or word write from RAM is shown in Figure 5-8.

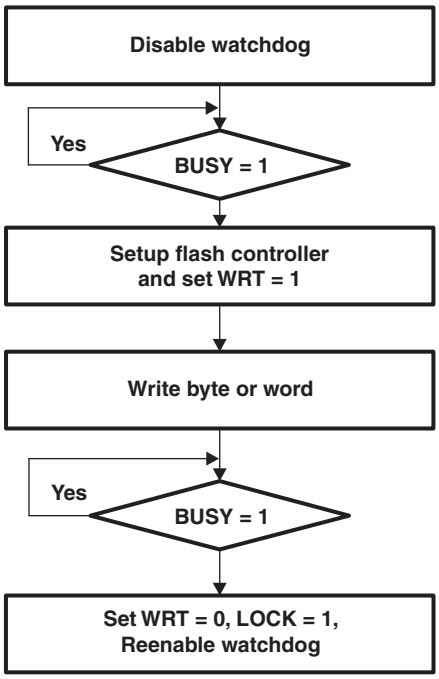


Figure 5-8. Initiating a Byte or Word Write From RAM

```
; Byte or word write from RAM.
; Assumes 0x0FF1E is already erased
; Assumes ACCVIE = NMIIE = OFIE = 0.
    MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
L1  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L1                         ; Loop while busy
    MOV   #FWPW,&FCTL3               ; Clear LOCK
    MOV   #FWPW+WRT,&FCTL1           ; Enable write
    MOV   #0123h,&0FF1Eh             ; 0123h -> 0x0FF1E
L2  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L2                         ; Loop while busy
    MOV   #FWPW,&FCTL1               ; Clear WRT
    MOV   #FWPW+LOCK,&FCTL3          ; Set LOCK
    ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.4 Long-Word Write

A long-word write operation can be initiated from within flash memory or from RAM. The BUSY bit is set to 1 after 32 bits are written to the flash controller and the programming cycle starts. When initiating from within flash memory, the CPU is held while the write completes. After the write completes, the CPU resumes code execution with the instruction following the write access. The long-word write timing is shown in Figure 5-6.

A long-word consists of four consecutive bytes aligned to at 32-bit address (only the lower two address bits are different). The bytes can be written in any order or any combination of bytes and words. If a byte or word is written more than once, the last data written to the four bytes are stored into the flash memory.

If a write to a flash address outside of the 32-bit address happens before all four bytes are available, the data written so far is discarded, and the latest byte or word written defines the new 32-bit aligned address.

When 32 bits are available, the write cycle is executed. When executing from RAM, the CPU continues to execute code. The BUSY bit must be zero before the CPU accesses flash again, otherwise an access violation occurs, ACCVIFG is set, and the write result is unpredictable.

In long-word write mode, the internally-generated programming voltage is applied to a complete 128-byte block. The cumulative programming time, t_{CPT}, must not be exceeded for any block. Each write adds to the cumulative program time of a segment. If the maximum cumulative program time is reached or exceeded, the segment must be erased. Further programming or using the data returns unpredictable results.

With each write, the amount of time the block is subjected to the programming voltage accumulates. If the cumulative programming time is reached or exceeded, the block must be erased before further programming or use (see the device-specific data sheet for specifications).

5.3.2.5 Initiating Long-Word Write From Flash

The flow to initiate a long-word write from flash is shown in Figure 5-9.

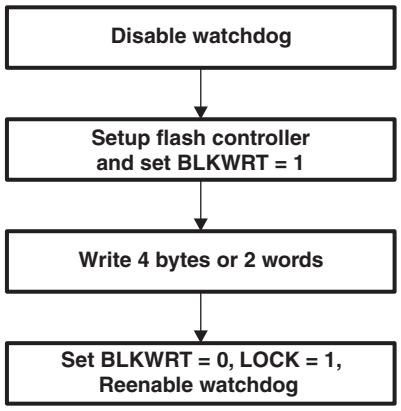


Figure 5-9. Initiating Long-Word Write From Flash

```
; Long-word write from flash.
; Assumes 0x0FF1C and 0x0FF1E is already erased
; Assumes ACCVIE = NMIIE = OFIE = 0.
   MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
   MOV   #FWPW,&FCTL3               ; Clear LOCK
   MOV   #FWPW+BLKWRT,&FCTL1        ; Enable 2-word write
   MOV   #0123h,&0FF1Ch             ; 0123h -> 0x0FF1C
   MOV   #45676h,&0FF1Eh            ; 04567h -> 0x0FF1E
   MOV   #FWPW,&FCTL1               ; Done. Clear BLKWRT
   MOV   #FWPW+LOCK,&FCTL3          ; Set LOCK
   ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.6 Initiating Long-Word Write From RAM

The flow to initiate a long-word write from RAM is shown in Figure 5-10.

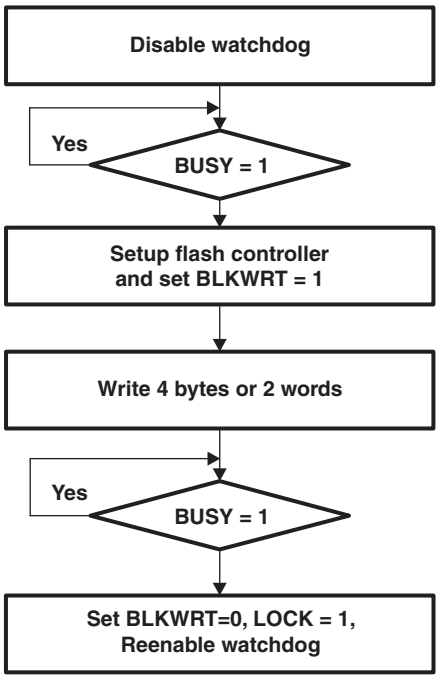


Figure 5-10. Initiating Long-Word Write from RAM

```
; Two 16-bit word writes from RAM.
; Assumes 0x0FF1C and 0x0FF1E is already erased
; Assumes ACCVIE = NMIIE = OFIE = 0.
    MOV   #WDTPW+WDTHOLD,&WDTCTL     ; Disable WDT
L1  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L1                         ; Loop while busy
    MOV   #FWPW,&FCTL3               ; Clear LOCK
    MOV   #FWPW+BLKWRT,&FCTL1        ; Enable write
    MOV   #0123h,&0FF1Ch             ; 0123h -> 0x0FF1C
    MOV   #4567h,&0FF1Eh             ; 4567h -> 0x0FF1E
L2  BIT   #BUSY,&FCTL3               ; Test BUSY
    JNZ   L2                         ; Loop while busy
    MOV   #FWPW,&FCTL1               ; Clear WRT
    MOV   #FWPW+LOCK,&FCTL3          ; Set LOCK
    ...                              ; Re-enable WDT?
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.7 Block Write

The block write can be used to accelerate the flash write process when many sequential bytes or words need to be programmed. The flash programming voltage remains on for the duration of writing the 128-byte row. The cumulative programming time, t_{CPT}, must not be exceeded for any row during a block write. Only long-word writes are possible using block write mode.

A block write cannot be initiated from within flash memory. The block write must be initiated from RAM. The BUSY bit remains set throughout the duration of the block write. The WAIT bit must be checked between writing four bytes, or two words, to the block. When WAIT is set, then four bytes, or two 16-bit words, of the block can be written. When writing successive blocks, the BLKWRT bit must be cleared after the current block is completed. BLKWRT can be set initiating the next block write after the required flash recovery time given by t_{END}. BUSY is cleared following each block write completion, indicating the next block can be written. Figure 5-11 shows the block write timing. The first long-word write requires $t_{Block,0}$ and the last long-write requires $t_{Block,N}$. All other blocks require $t_{Block,1-(N-1)}$.

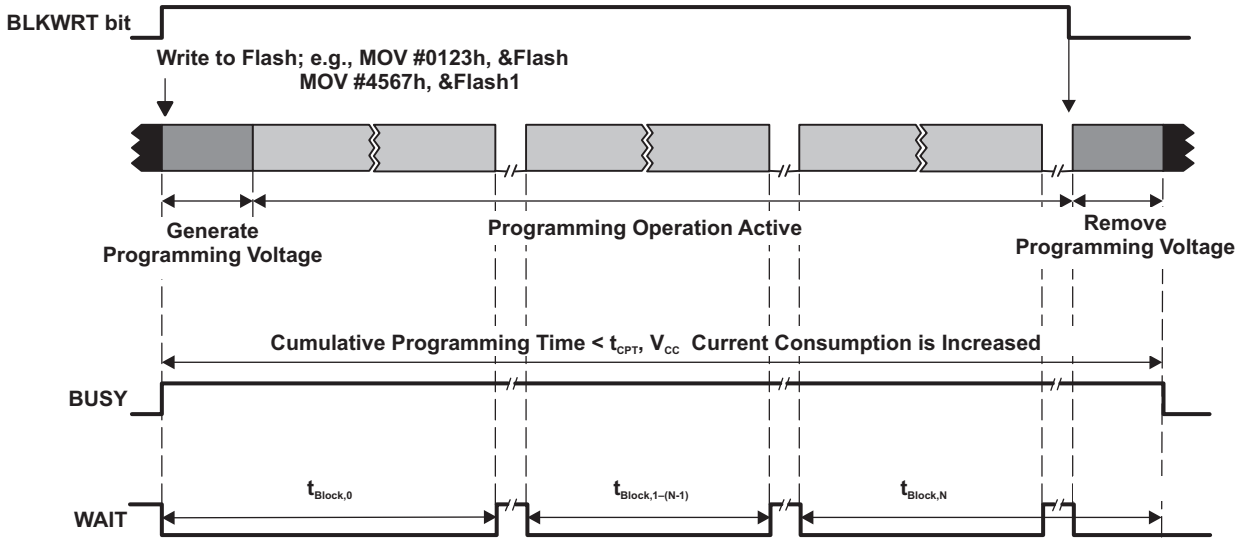


Figure 5-11. Block-Write Cycle Timing

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.2.8 Block Write Flow and Example

A block write flow is shown in Figure 5-12 and the following code example.

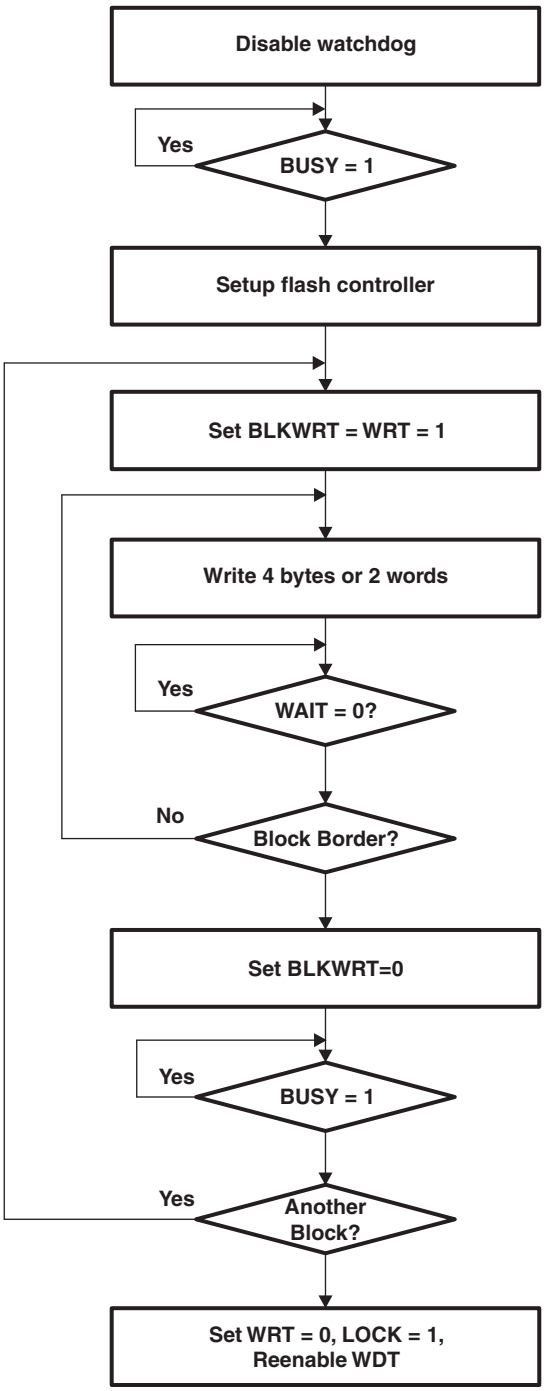


Figure 5-12. Block Write Flow

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.


```
; Write one block starting at 0F000h.
; Must be executed from RAM, Assumes Flash is already erased.
; Assumes ACCVIE = NMIIE = OFIE = 0.
    MOV    #32,R5                      ; Use as write counter
    MOV    #0F000h,R6                  ; Write pointer
    MOV    #WDTPW+WDTHOLD,&WDTCTL      ; Disable WDT
L1  BIT    #BUSY,&FCTL3                ; Test BUSY
    JNZ    L1                          ; Loop while busy
    MOV    #FWPW,&FCTL3                ; Clear LOCK
    MOV    #FWPW+BLKWRT+WRT,&FCTL1     ; Enable block write
L2  MOV    Write_Value1,0(R6)          ; Write 1st location
    MOV    Write_Value2,2(R6)          ; Write 2nd word
L3  BIT    #WAIT,&FCTL3                ; Test WAIT
    JZ     L3                          ; Loop while WAIT=0
    INCD   R6                          ; Point to next words
    INCD   R6                          ; Point to next words
    DEC    R5                          ; Decrement write counter
    JNZ    L2                          ; End of block?
    MOV    #FWPW,&FCTL1                ; Clear WRT, BLKWRT
L4  BIT    #BUSY,&FCTL3                ; Test BUSY
    JNZ    L4                          ; Loop while busy
    MOV    #FWPW+LOCK,&FCTL3           ; Set LOCK
    ...                                ; Re-enable WDT if needed
```

5.3.3 Flash Memory Access During Write or Erase

When a write or an erase operation is initiated from RAM while BUSY = 1, the CPU may not write to any flash location. Otherwise, an access violation occurs, ACCVIFG is set, and the result is unpredictable. ACCVIFG is also set if a Flash write or erase access is attempted without any Flash write or erase mode selected first.

When a write operation is initiated from within flash memory, the CPU continues code execution with the next instruction fetch after the write cycle completed (BUSY = 0).

The op-code 3FFFh is the JMP PC instruction. This causes the CPU to loop until the flash operation is finished. When the operation is finished and BUSY = 0, the flash controller allows the CPU to fetch the op-code and program execution resumes.

The flash access conditions while BUSY = 1 are listed in Table 5-4.

Table 5-4. Flash Access While Flash is Busy (BUSY = 1)

Flash Operation	Flash Access	WAIT	Result
Bank erase	Read	0	From the erased bank: ACCVIFG = 0. 03FFFh is the value read. From any other flash location: ACCVIFG = 0. Valid read.
	Write	0	ACCVIFG = 1. Write is ignored.
	Instruction fetch	0	From the erased bank: ACCVIFG = 0. CPU fetches 03FFFh. This is the JMP PC instruction. From any other flash location: ACCVIFG = 0. Valid instruction fetch.
Segment erase	Read	0	ACCVIFG = 0: 03FFFh is the value read.
	Write	0	ACCVIFG = 1: Write is ignored.
	Instruction fetch	0	ACCVIFG = 0: CPU fetches 03FFFh. This is the JMP PC instruction.
Word or byte write or long-word write	Read	0	ACCVIFG = 0: 03FFFh is the value read.
	Write	0	ACCVIFG = 1: Write is ignored.
	Instruction fetch	0	ACCVIFG = 0: CPU fetches 03FFFh. This is the JMP PC instruction.
Block write	Any	0	ACCVIFG = 1: LOCK = 1, block write is exited.
	Read	1	ACCVIFG = 0: 03FFFh is the value read.
	Write	1	ACCVIFG = 0: Valid write
	Instruction fetch	1	ACCVIFG = 1: LOCK = 1, block write is exited

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Interrupts are automatically disabled during any flash operation.

The watchdog timer (in watchdog mode) should be disabled before a flash erase cycle. A reset aborts the erase and the result is unpredictable. After the erase cycle has completed, the watchdog may be reenabled.

5.3.4 Stopping Write or Erase Cycle

Any write or erase operation can be stopped before its normal completion by setting the emergency exit bit EMEX. Setting the EMEX bit stops the active operation and resets the flash controller. All flash operations cease, the flash returns to read mode, and all bits in the FCTL1 register are reset. The LOCK bit of FCTL3 is set. The result of the intended operation is unpredictable.

5.3.4.1 EMEX With Single Bank Flash Memory

For devices with single bank flash memories, write and erase operations initiated from flash, the CPU is held until the flash operation completes. Therefore it is not possible to perform an emergency exit by the EMEX bit. The emergency exit of write or erase operations initiated from RAM can be performed using the EMEX bit. The BUSY bit is used to determine the end of the emergency exit cycle. The user must ensure that code execution does not continue until the BUSY bit is cleared by the flash controller.

5.3.4.2 EMEX With Multiple Bank Flash Memory

For devices with multiple bank flash memories, write and segment erase operations initiated from flash, regardless of which bank the code resides in, the CPU is held until the flash operation completes. Therefore it is not possible to perform an emergency exit by the EMEX bit. For bank erase, there is a possibility to perform an EMEX if the bank being erased is not where the code resides. The BUSY bit is used to determine the end of the emergency exit cycle. The user must ensure that code execution does not continue until the BUSY bit is cleared by the flash controller.

The emergency exit of write or any erase operations initiated from RAM can be performed using the EMEX bit. The BUSY bit is used to determine the end of the emergency exit cycle. The user must ensure that code execution does not continue until the BUSY bit is cleared by the flash controller.

5.3.5 Checking Flash Memory

The result of a programming cycle of the flash memory can be checked by calculating and storing a checksum (CRC) of parts or the complete flash memory content. The CRC module can be used for this purpose (see the device-specific data sheet). During the runtime of the system, the known checksums can be recalculated and compared with the expected values stored in the flash memory. The program checking the flash memory content is executed in RAM.

To get an early indication of weak memory cells, reading the flash can be done in combination with the device-specific marginal read modes. The marginal read modes are controlled by the FCTL4.MRG0 and FCTL4.MRG1 register bits if available (device specific). During marginal read mode, marginally programmed flash memory bit locations can be detected. One method for identifying such memory locations would be to periodically perform a checksum calculation over a section of flash memory (for example, a flash segment) and repeating this procedure with the marginal read mode enabled. If they do not match, it could indicate an insufficiently programmed flash memory location. It is possible to refresh the affected Flash memory segment by disabling marginal read mode, copying to RAM, erasing the flash segment, and writing back to it from RAM.

The program checking the flash memory contents must be executed from RAM. Executing code from flash automatically disables the marginal read mode. The marginal read modes are controlled by the MRG0 and MRG1 register bits. Setting MRG1 is used to detect insufficiently programmed flash cells containing a "1" (erased bits). Setting MRG0 is used to detect insufficiently programmed flash cells containing a "0" (programmed bits). Only one of these bits should be set at a time. Therefore, a full marginal read check requires two passes of checking the flash memory content’s integrity. During marginal read mode, the flash access speed (MCLK) must be limited to 1 MHz (see the device-specific data sheet).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.6 Configuring and Accessing the Flash Memory Controller

The FCTLx registers are 16-bit password-protected read and write registers. Any read or write access must use word instructions, and write accesses must include the write password 0A5h in the upper byte. Any write to any FCTLx register with a value other than 0A5h in the upper byte is a password violation, sets the KEYV flag, and triggers a PUC system reset. Any read of any FCTLx registers reads 096h in the upper byte.

Any write to FCTL1 during an erase or byte, word, double-word write operation is an access violation and sets ACCVIFG. Writing to FCTL1 is allowed in block write mode when WAIT = 1, but writing to FCTL1 in block write mode when WAIT = 0 is an access violation and sets ACCVIFG.

Any write to FCTL2 (this register is currently not implemented) when BUSY = 1 is an access violation.

Any FCTLx register may be read when BUSY = 1. A read does not cause an access violation.

5.3.7 Flash Memory Controller Interrupts

The flash controller has two interrupt sources, KEYV and ACCVIFG. ACCVIFG is set when an access violation occurs. When the ACCVIE bit is reenabled after a flash write or erase, a set ACCVIFG flag generates an interrupt request. The ACCVIE bit resides in the Special Function Register, SFRIE1 (see the SYS chapter for details). ACCVIFG sources the NMI interrupt vector, so it is not necessary for GIE to be set for ACCVIFG to request an interrupt. ACCVIFG may also be checked by software to determine if an access violation occurred. ACCVIFG must be reset by software.

The password violation flag, KEYV, is set when any of the flash control registers are written with an incorrect password. When this occurs, a PUC is generated immediately, resetting the device.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.3.8 Programming Flash Memory Devices

There are three options for programming a flash device. All options support in-system programming.

- Program via JTAG
- Program via the BSL
- Program via a custom solution

5.3.8.1 Programming Flash Memory Via JTAG

Devices can be programmed via the JTAG port. The JTAG interface requires four signals (five signals on 20- and 28-pin devices), ground, and optionally VCC and $\overline{\text{RST}}$/NMI.

The JTAG port is protected with a fuse. Blowing the fuse completely disables the JTAG port and is not reversible. Further access to the device via JTAG is not possible For more details, see the *MSP430 Programming Via the JTAG Interface User's Guide* (SLAU320).

5.3.8.2 Programming Flash Memory Via Bootstrap Loader (BSL)

Every flash device contains a BSL. The BSL enables users to read or program the flash memory or RAM using a UART serial interface. Access to the flash memory via the BSL is protected by a 256-bit user-defined password. For more details, see the *MSP430 Programming Via the Bootstrap Loader User's Guide* (SLAU319).

5.3.8.3 Programming Flash Memory Via Custom Solution

The ability of the MSP430 CPU to write to its own flash memory allows for in-system and external custom programming solutions as shown in Figure 5-13. The user can choose to provide data through any means available (for example, UART or SPI). User-developed software can receive the data and program the flash memory. Because this type of solution is developed by the user, it can be completely customized to fit the application needs for programming, erasing, or updating the flash memory.

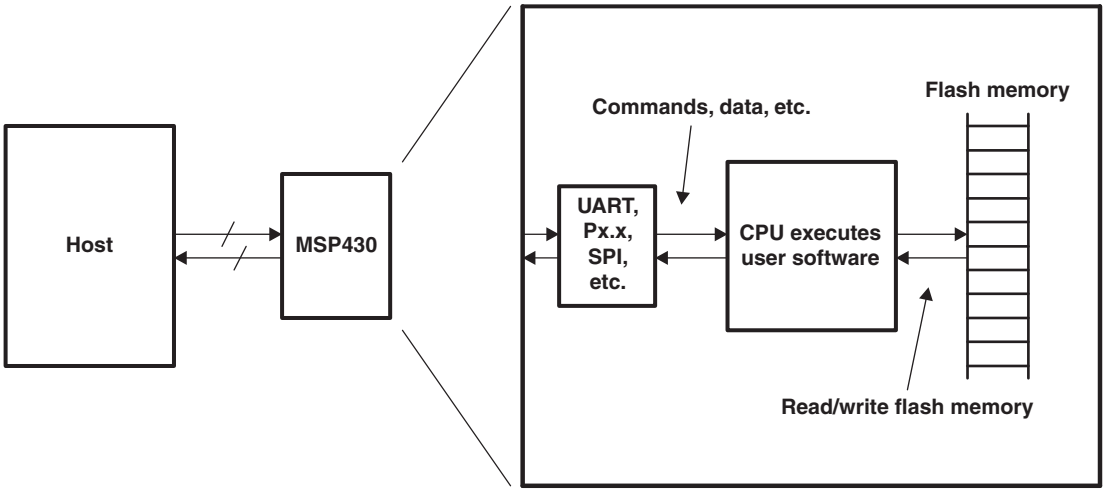


Figure 5-13. User-Developed Programming Solution

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.4 FCTL Registers

The flash memory controller (FCTL) registers are listed in Table 5-5. The base address can be found in the device-specific data sheet. The address offset is given in Table 5-5.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 5-5. FCTL Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	FCTL1	Flash Memory Control 1	Read/write	Word	9600h	Section 5.4.1
00h	FCTL1_L		Read/Write	Byte	00h	
01h	FCTL1_H		Read/Write	Byte	96h	
04h	FCTL3	Flash Memory Control 3	Read/write	Word	9658h	Section 5.4.2
04h	FCTL3_L		Read/Write	Byte	58h	
05h	FCTL3_H		Read/Write	Byte	96h	
06h	FCTL4	Flash Memory Control 4	Read/write	Word	9600h	Section 5.4.3
06h	FCTL4_L		Read/Write	Byte	00h	
07h	FCTL4_H		Read/Write	Byte	96h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.4.1 FCTL1 Register

Flash Memory Control 1 Register

Figure 5-14. FCTL1 Register

15	14	13	12	11	10	9	8
FRPW/FWPW							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

7	6	5	4	3	2	1	0
BLKWRT	WRT	SWRT	Reserved		MERAS	ERASE	Reserved
rw-0	rw-0	rw-0	r-0	r-0	rw-0	rw-0	r-0

Table 5-6. FCTL1 Register Description

Bit	Field	Type	Reset	Description
15-8	FRPW/FWPW	RW	96h	FCTL password. Always read as 096h. Must be written as 0A5h or a PUC is generated.
7	BLKWRT	RW	0h	Block write. BLKWRT and WRT are used together to select the write mode. The values shown below are for BLKWRT-WRT. 0-0 = Reserved 0-1 = Byte or word write 1-0 = Long-word write 1-1 = Long-word block write
6	WRT	RW	0h	Write. BLKWRT and WRT are used together to select the write mode. The values shown below are for BLKWRT-WRT. 0-0 = Reserved 0-1 = Byte or word write 1-0 = Long-word write 1-1 = Long-word block write
5	SWRT	RW	0h	Smart write. If this bit is set, the program time is shortened. The programming quality has to be checked by marginal read modes.
4-3	Reserved	R	0h	Reserved. Always reads as 0.
2	MERAS			Mass erase. MERAS and ERASE are used together to select the erase mode. MERAS and ERASE are automatically reset when EMEX is set or a flash erase operation has completed. The values shown below are for MERAS-ERASE. 0-0 = No erase 0-1 = Segment erase 1-0 = Bank erase (erase of one bank) 1-1 = Mass erase (erase all flash memory banks)
1	ERASE			Erase. MERAS and ERASE are used together to select the erase mode. MERAS and ERASE are automatically reset when EMEX is set or a flash erase operation has completed. The values shown below are for MERAS-ERASE. 0-0 = No erase 0-1 = Segment erase 1-0 = Bank erase (erase of one bank) 1-1 = Mass erase (erase all flash memory banks)
0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.4.2 FCTL3 Register

Flash Memory Control 3 Register

Figure 5-15. FCTL3 Register

15	14	13	12	11	10	9	8
FRPW/FWPW							
7	6	5	4	3	2	1	0
Reserved	LOCKA	EMEX	LOCK	WAIT	ACCVIFG	KEYV	BUSY
r-0	rw-1	rw-0	rw-1	r-1	rw-0	rw-(0)	rw-0

Table 5-7. FCTL3 Register Description

Bit	Field	Type	Reset	Description
15-8	FRPW/FWPW	RW	96h	FCTLx password. Always read as 096h. Must be written as 0A5h or a PUC is generated.
7	Reserved	R	0h	Reserved. Always reads as 0.
6	LOCKA	RW	1h	Segment A lock. Write a 1 to this bit to change its state. Writing 0 has no effect. 0b = Segment A of the information memory is unlocked and can be written or erased in segment erase mode. 1b = Segment A of the information memory is locked and can not be written or erased in segment erase mode.
5	EMEX	RW	0h	Emergency exit. Setting this bit stops any erase or write operation. The LOCK bit is set. 0b = No emergency exit 1b = Emergency exit
4	LOCK	RW	1h	Lock. This bit unlocks the flash memory for writing or erasing. The LOCK bit can be set any time during a byte or word write or erase operation, and the operation completes normally. In the block write mode, if the LOCK bit is set while BLKWRT = WAIT = 1, BLKWRT and WAIT are reset and the mode ends normally. 0b = Unlocked 1b = Locked
3	WAIT	R	1h	Wait. Indicates the flash memory is being written to. 0b = Flash memory is not ready for the next byte or word write. 1b = Flash memory is ready for the next byte or word write.
2	ACCVIFG	RW	0h	Access violation interrupt flag 0b = No interrupt pending 1b = Interrupt pending
1	KEYV	RW	0h	Flash password violation. This bit indicates an incorrect FCTLx password was written to any flash control register and generates a PUC when set. KEYV must be reset with software. 0b = FCTLx password was written correctly. 1b = FCTLx password was written incorrectly.
0	BUSY	RW	0h	Busy. This bit indicates if the flash is currently busy erasing or programming. 0b = Not busy 1b = Busy

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

5.4.3 FCTL4 Register

Flash Memory Control 4 Register

Figure 5-16. FCTL4 Register

15	14	13	12	11	10	9	8
FRPW/FWPW							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

7	6	5	4	3	2	1	0
LOCKINFO	Reserved	MRG1	MRG0	Reserved			VPE
rw-0	r-0	rw-0	rw-0	r-0	r-0	r-0	rw-0

Table 5-8. FCTL4 Register Description

Bit	Field	Type	Reset	Description
15-8	FRPW/FWPW	RW	96h	FCTLx password. Always reads as 096h. Must be written as 0A5h or a PUC is generated.
7	LOCKINFO	RW	0h	Lock information memory. If set, the information memory cannot be erased in segment erase mode and cannot be written to.
6	Reserved	R	0h	Reserved. Always reads as 0.
5	MRG1	RW	0h	Marginal read 1 mode. This bit enables the marginal 1 read mode. The marginal read 1 bit is valid for reads from the flash memory only. During a fetch cycle, the marginal mode is turned off automatically. If both MRG1 and MRG0 are set, MRG1 is active and MRG0 is ignored. 0b = Marginal 1 read mode is disabled. 1b = Marginal 1 read mode is enabled.
4	MRG0	RW	0h	Marginal read 0 mode. This bit enables the marginal 0 read mode. The marginal read 1 bit is valid for reads from the flash memory only. During a fetch cycle, the marginal mode is turned off automatically. If both MRG1 and MRG0 are set, MRG1 is active and MRG0 is ignored. 0b = Marginal 0 read mode is disabled. 1b = Marginal 0 read mode is enabled.
3-1	Reserved	R	0h	Reserved. Always reads as 0.
0	VPE	RW	0h	Voltage changed during program error. This bit is set by hardware and can only be cleared by software. If DVCC changed significantly during programming, this bit is set to indicate an invalid result. The ACCVIFG bit is set if VPE is set.

Copyright © 2009–2013, Texas Instruments Incorporated

5.4.4 SFRIE1 Register

Interrupt Enable 1 Register

Figure 5-17. SFRIE1 Register

15	14	13	12	11	10	9	8

7	6	5	4	3	2	1	0
		ACCVIE					
		rw-0					

Table 5-9. SFRIE1 Register Description

Bit	Field	Type	Reset	Description
15-6				These bits may be used by other modules (see the device-specific data sheet and the SYS chapter for details).
5	ACCVIE	RW	0h	Flash memory access violation interrupt enable. This bit enables the ACCVIFG interrupt. Because other bits in SFRIE1 may be used for other modules, it is recommended to set or clear this bit using BIS or BIC instructions, rather than MOV or CLR instructions. See the SYS chapter for more details. 0b = Interrupt not enabled 1b = Interrupt enabled
4-0				These bits may be used by other modules (see the device-specific data sheet and the SYS chapter for details).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

RAM Controller (RAMCTL)

The RAM controller (RAMCTL) allows control of the operation of the RAM.

Topic **Page**

6.1 RAM Controller (RAMCTL) Introduction 313
6.2 RAMCTL Operation 313
6.3 RAMCTL Registers 314

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

6.1 RAM Controller (RAMCTL) Introduction

The RAMCTL provides access to the different power modes of the RAM. The RAMCTL allows the ability to reduce the leakage current while the CPU is off. The RAM can also be switched off. In retention mode, the RAM content is saved while the RAM content is lost in off mode. The RAM is partitioned in sectors, typically of 4KB (sector) size. See the device-specific data sheet for actual block allocation and size. Each sector is controlled by the RAM controller RAM Sector Off control bit (RCRSyOFF) of the RAMCTL Control 0 register (RCCTL0). The RCCTL0 register is protected with a key. Only if the correct key is written during a word write, the RCCTL0 register content can be modified. Byte write accesses or write accesses with a wrong key are ignored.

6.2 RAMCTL Operation

Active mode

In active mode, the RAM can be read and written at any time. If a RAM address of a sector must hold data, the whole sector cannot be switched off.

Low-power modes

In all low-power modes, the CPU is switched off. As soon as the CPU is switched off, the RAM enters retention mode to reduce the leakage current.

RAM off mode

Each sector can be turned off independently of each other by setting the respective RCRSyOFF bit to 1. Reading from a switched off RAM sector returns 0 as data. All data previously stored into a switched off RAM sector is lost and cannot be read, even if the sector is turned on again.

Stack pointer

The program stack is located in RAM. Sectors holding the stack must not be turned off if an interrupt has to be executed, or a low-power mode is entered.

USB buffer memory

On devices with USB, the USB buffer memory is located in RAM. Sector 7 is used for this purpose. RCRS7OFF can be set to switch off this memory if it is not required for USB operation or is not being utilized in normal operation.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

6.3 RAMCTL Registers

The RAMCTL module register is listed in Table 6-1. The base address can be found in the device-specific data sheet. The address offset is given in Table 6-1.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 6-1. RAMCTL Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	RCCTL0	RAM Controller Control 0	Read/write	Word	6900h	Section 6.3.1
00h	RCCTL0_L		Read/write	Byte	00h	
01h	RCCTL0_H		Read/write	Byte	69h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

6.3.1 RCCTL0 Register

RAM Controller Control 0 Register

Figure 6-1. RCCTL0 Register

15	14	13	12	11	10	9	8
RCKEY							
rw-0	rw-1	rw-1	rw-0	rw-1	rw-0	rw-0	rw-1
7	6	5	4	3	2	1	0
RCRS7OFF	RCRS6OFF	RCRS5OFF	RCRS4OFF	RCRS3OFF	RCRS2OFF	RCRS1OFF	RCRS0OFF
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 6-2. RCCTL0 Register Description

Bit	Field	Type	Reset	Description
15-8	RCKEY	RW	69h	RAM controller key. Always read as 69h. Must be written as 5Ah, otherwise the RAMCTL write is ignored.
7	RCRS7OFF	RW	0h	RAM controller RAM sector 7 off. Setting the bit to 1 turns off the RAM sector 7. All data of the RAM sector 7 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
6	RCRS6OFF	RW	0h	RAM controller RAM sector 6 off. Setting the bit to 1 turns off the RAM sector 6. All data of the RAM sector 6 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
5	RCRS5OFF	RW	0h	RAM controller RAM sector 5 off. Setting the bit to 1 turns off the RAM sector 5. All data of the RAM sector 5 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
4	RCRS4OFF	RW	0h	RAM controller RAM sector 4 off. Setting the bit to 1 turns off the RAM sector 4. All data of the RAM sector 4 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
3	RCRS3OFF	RW	0h	RAM controller RAM sector 3 off. Setting the bit to 1 turns off the RAM sector 3. All data of the RAM sector 3 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
2	RCRS2OFF	RW	0h	RAM controller RAM sector 2 off. Setting the bit to 1 turns off the RAM sector 2. All data of the RAM sector 2 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
1	RCRS1OFF	RW	0h	RAM controller RAM sector 1 off. Setting the bit to 1 turns off the RAM sector 1. All data of the RAM sector 1 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.
0	RCRS0OFF	RW	0h	RAM controller RAM sector 0 off. Setting the bit to 1 turns off the RAM sector 0. All data of the RAM sector 0 is lost. See the device-specific data sheet to find the the number of RAM sectors available along with their respective address ranges and sizes.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Direct Memory Access (DMA) Controller Module

The direct memory access (DMA) controller module transfers data from one address to another without CPU intervention. This chapter describes the operation of the DMA controller.

Topic **Page**

7.1 Direct Memory Access (DMA) Introduction 317
7.2 DMA Operation 319
7.3 DMA Registers 330

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.1 Direct Memory Access (DMA) Introduction

The DMA controller transfers data from one address to another, without CPU intervention, across the entire address range. For example, the DMA controller can move data from the ADC conversion memory to RAM.

Devices that contain a DMA controller may have up to eight DMA channels available. Therefore, depending on the number of DMA channels available, some features described in this chapter are not applicable to all devices. See the device-specific data sheet for number of channels supported.

Using the DMA controller can increase the throughput of peripheral modules. It can also reduce system power consumption by allowing the CPU to remain in a low-power mode, without having to awaken to move data to or from a peripheral.

DMA controller features include:

- Up to eight independent transfer channels
- Configurable DMA channel priorities
- Requires only two MCLK clock cycles per transfer
- Byte or word and mixed byte/word transfer capability
- Block sizes up to 65535 bytes or words
- Configurable transfer trigger selections
- Selectable-edge or level-triggered transfer
- Four addressing modes
- Single, block, or burst-block transfer modes

The DMA controller block diagram is shown in Figure 7-1.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

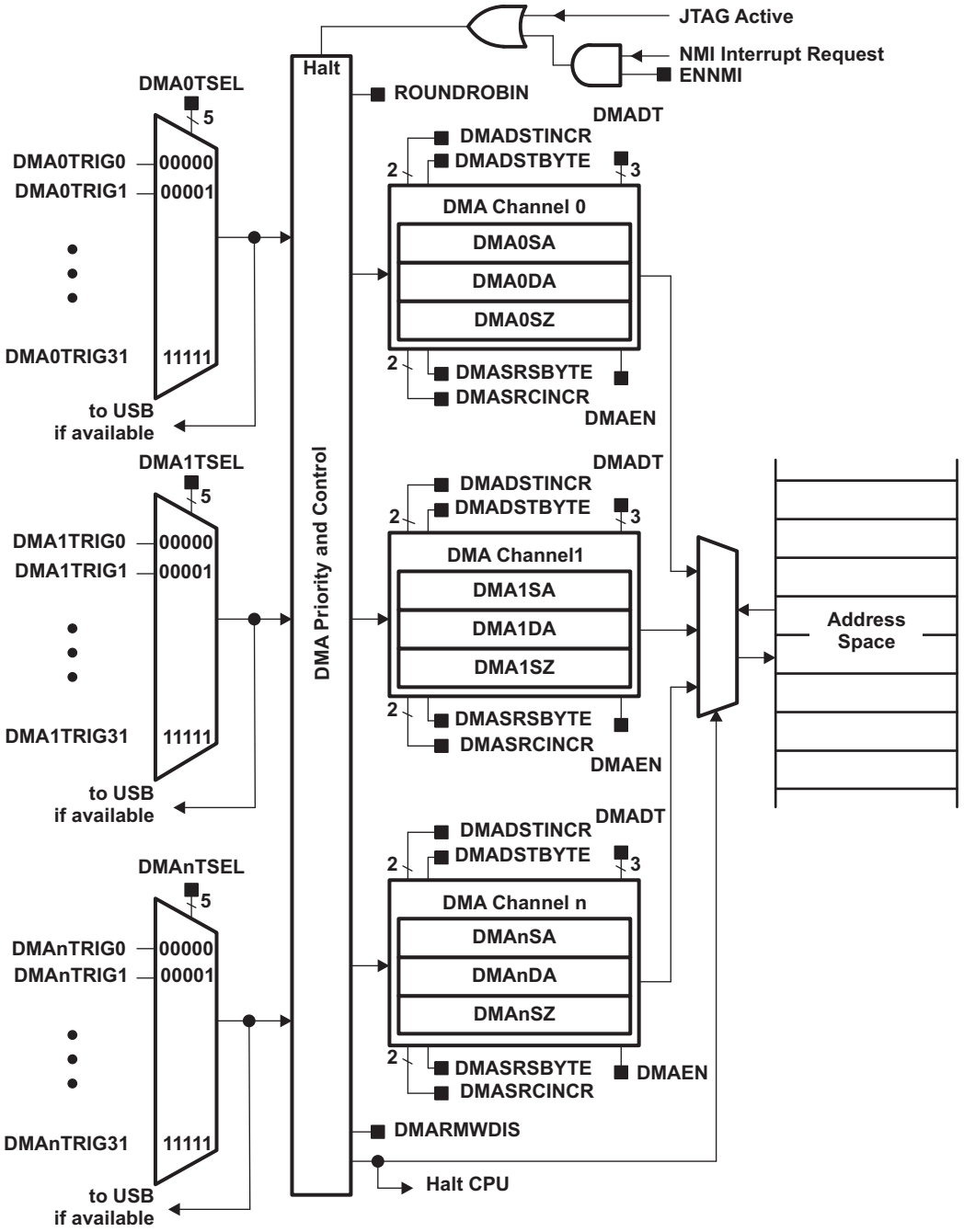


Figure 7-1. DMA Controller Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.2 DMA Operation

The DMA controller is configured with user software. The setup and operation of the DMA is discussed in the following sections.

7.2.1 DMA Addressing Modes

The DMA controller has four addressing modes. The addressing mode for each DMA channel is independently configurable. For example, channel 0 may transfer between two fixed addresses, while channel 1 transfers between two blocks of addresses. The addressing modes are shown in Figure 7-2. The addressing modes are:

- Fixed address to fixed address
- Fixed address to block of addresses
- Block of addresses to fixed address
- Block of addresses to block of addresses

The addressing modes are configured with the DMASRCINCR and DMADSTINCR control bits. The DMASRCINCR bits select if the source address is incremented, decremented, or unchanged after each transfer. The DMADSTINCR bits select if the destination address is incremented, decremented, or unchanged after each transfer.

Transfers may be byte to byte, word to word, byte to word, or word to byte. When transferring word to byte, only the lower byte of the source-word transfers. When transferring byte to word, the upper byte of the destination-word is cleared when the transfer occurs.

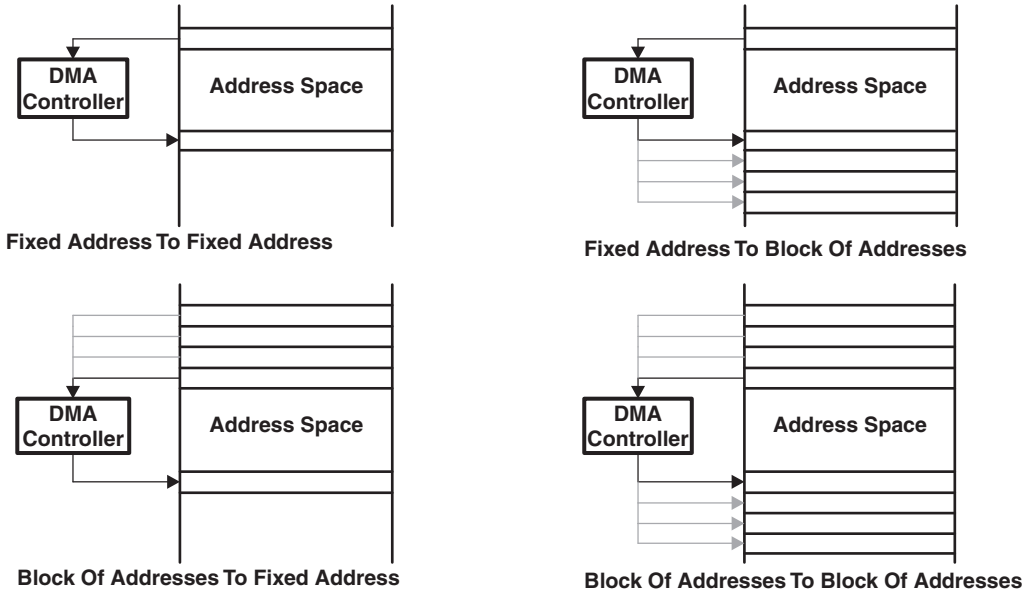


Figure 7-2. DMA Addressing Modes

7.2.2 DMA Transfer Modes

The DMA controller has six transfer modes selected by the DMADT bits as listed in Table 7-1. Each channel is individually configurable for its transfer mode. For example, channel 0 may be configured in single transfer mode, while channel 1 is configured for burst-block transfer mode, and channel 2 operates in repeated block mode. The transfer mode is configured independently from the addressing mode. Any addressing mode can be used with any transfer mode.

Two types of data can be transferred selectable by the DMAxCTL DSTBYTE and SRCBYTE fields. The source and/or destination location can be either byte or word data. It is also possible to transfer byte to byte, word to word, or any combination.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 7-1. DMA Transfer Modes

DMADT	Transfer Mode	Description
000	Single transfer	Each transfer requires a trigger. DMAEN is automatically cleared when DMAxSZ transfers have been made.
001	Block transfer	A complete block is transferred with one trigger. DMAEN is automatically cleared at the end of the block transfer.
010, 011	Burst-block transfer	CPU activity is interleaved with a block transfer. DMAEN is automatically cleared at the end of the burst-block transfer.
100	Repeated single transfer	Each transfer requires a trigger. DMAEN remains enabled.
101	Repeated block transfer	A complete block is transferred with one trigger. DMAEN remains enabled.
110, 111	Repeated burst-block transfer	CPU activity is interleaved with a block transfer. DMAEN remains enabled.

7.2.2.1 Single Transfer

In single transfer mode, each byte/word transfer requires a separate trigger. The single transfer state diagram is shown in Figure 7-3.

The DMAxSZ register is used to define the number of transfers to be made. The DMADSTINCR and DMASRCINCR bits select if the destination address and the source address are incremented or decremented after each transfer. If DMAxSZ = 0, no transfers occur.

The DMAxSA, DMAxDA, and DMAxSZ registers are copied into temporary registers. The temporary values of DMAxSA and DMAxDA are incremented or decremented after each transfer. The DMAxSZ register is decremented after each transfer. When the DMAxSZ register decrements to zero, it is reloaded from its temporary register and the corresponding DMAIFG flag is set. When DMADT = {0}, the DMAEN bit is cleared automatically when DMAxSZ decrements to zero and must be set again for another transfer to occur.

In repeated single transfer mode, the DMA controller remains enabled with DMAEN = 1, and a transfer occurs every time a trigger occurs.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

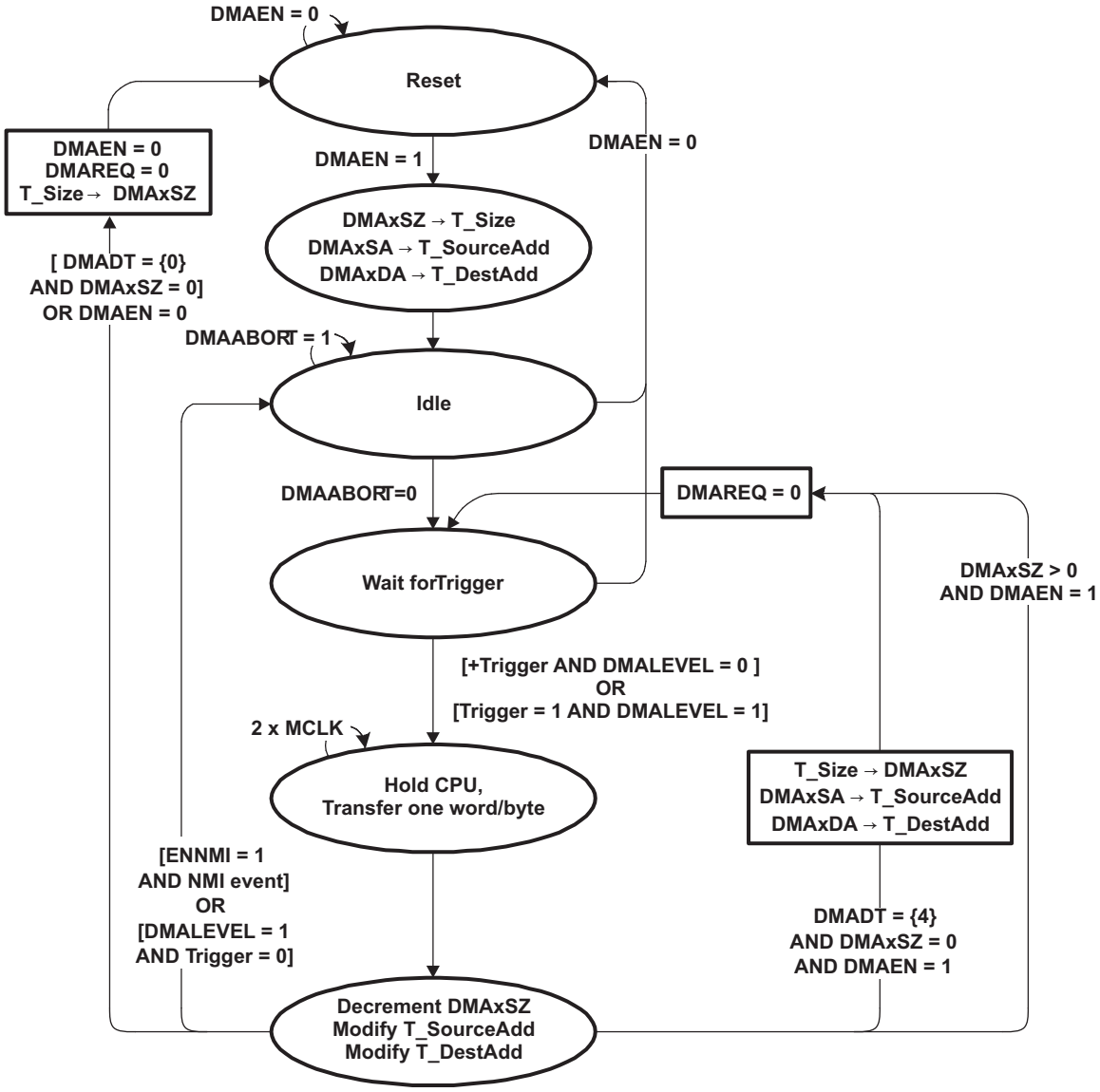


Figure 7-3. DMA Single Transfer State Diagram

7.2.2.2 Block Transfer

In block transfer mode, a transfer of a complete block of data occurs after one trigger. When DMADT = {1} ,the DMAEN bit is cleared after the completion of the block transfer and must be set again before another block transfer can be triggered. After a block transfer has been triggered, further trigger signals occurring during the block transfer are ignored. The block transfer state diagram is shown in Figure 7-4.

The DMAxSZ register is used to define the size of the block, and the DMADSTINCR and DMASRCINCR bits select if the destination address and the source address are incremented or decremented after each transfer of the block. If DMAxSZ = 0, no transfers occur.

The DMAxSA, DMAxDA, and DMAxSZ registers are copied into temporary registers. The temporary values of DMAxSA and DMAxDA are incremented or decremented after each transfer in the block. The DMAxSZ register is decremented after each transfer of the block and shows the number of transfers remaining in the block. When the DMAxSZ register decrements to zero, it is reloaded from its temporary register and the corresponding DMAIFG flag is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

During a block transfer, the CPU is halted until the complete block has been transferred. The block transfer takes 2 × MCLK × DMAxSZ clock cycles to complete. CPU execution resumes with its previous state after the block transfer is complete.

In repeated block transfer mode, the DMAEN bit remains set after completion of the block transfer. The next trigger after the completion of a repeated block transfer triggers another block transfer.

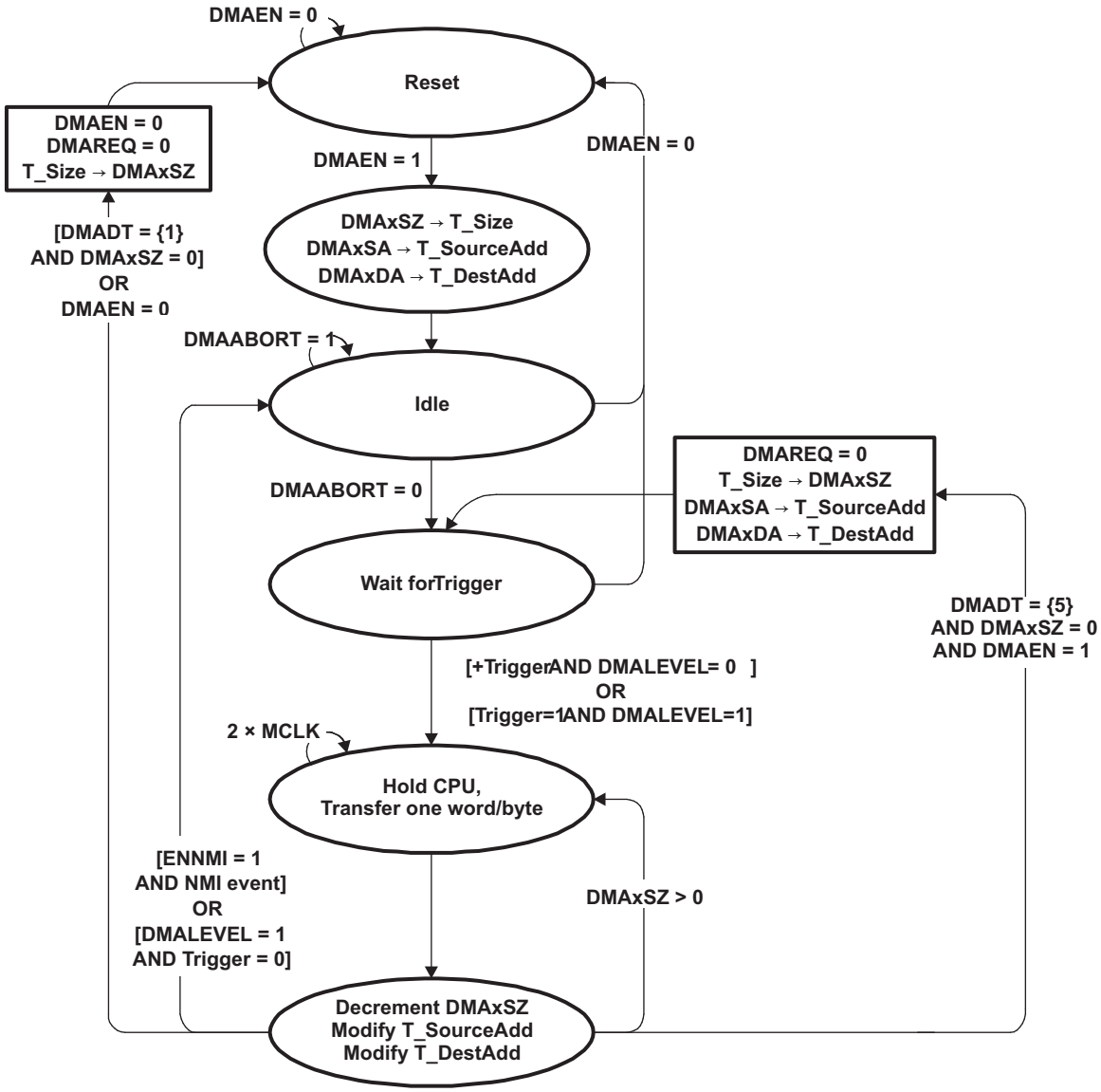


Figure 7-4. DMA Block Transfer State Diagram

7.2.2.3 Burst-Block Transfer

In burst-block mode, transfers are block transfers with CPU activity interleaved. The CPU executes two MCLK cycles after every four byte/word transfers of the block, resulting in 20% CPU execution capacity. After the burst-block, CPU execution resumes at 100% capacity and the DMAEN bit is cleared. DMAEN must be set again before another burst-block transfer can be triggered. After a burst-block transfer has been triggered, further trigger signals occurring during the burst-block transfer are ignored. The burst-block transfer state diagram is shown in Figure 7-5.

The DMAxSZ register is used to define the size of the block, and the DMADSTINCR and DMASRCINCR bits select if the destination address and the source address are incremented or decremented after each transfer of the block. If DMAxSZ = 0, no transfers occur.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The DMAxSA, DMAxDA, and DMAxSZ registers are copied into temporary registers. The temporary values of DMAxSA and DMAxDA are incremented or decremented after each transfer in the block. The DMAxSZ register is decremented after each transfer of the block and shows the number of transfers remaining in the block. When the DMAxSZ register decrements to zero, it is reloaded from its temporary register and the corresponding DMAIFG flag is set.

In repeated burst-block mode, the DMAEN bit remains set after completion of the burst-block transfer and no further trigger signals are required to initiate another burst-block transfer. Another burst-block transfer begins immediately after completion of a burst-block transfer. In this case, the transfers must be stopped by clearing the DMAEN bit, or by an (non)maskable interrupt (NMI) when ENNMI is set. In repeated burst-block mode the CPU executes at 20% capacity continuously until the repeated burst-block transfer is stopped.

Copyright © 2009–2013, Texas Instruments Incorporated

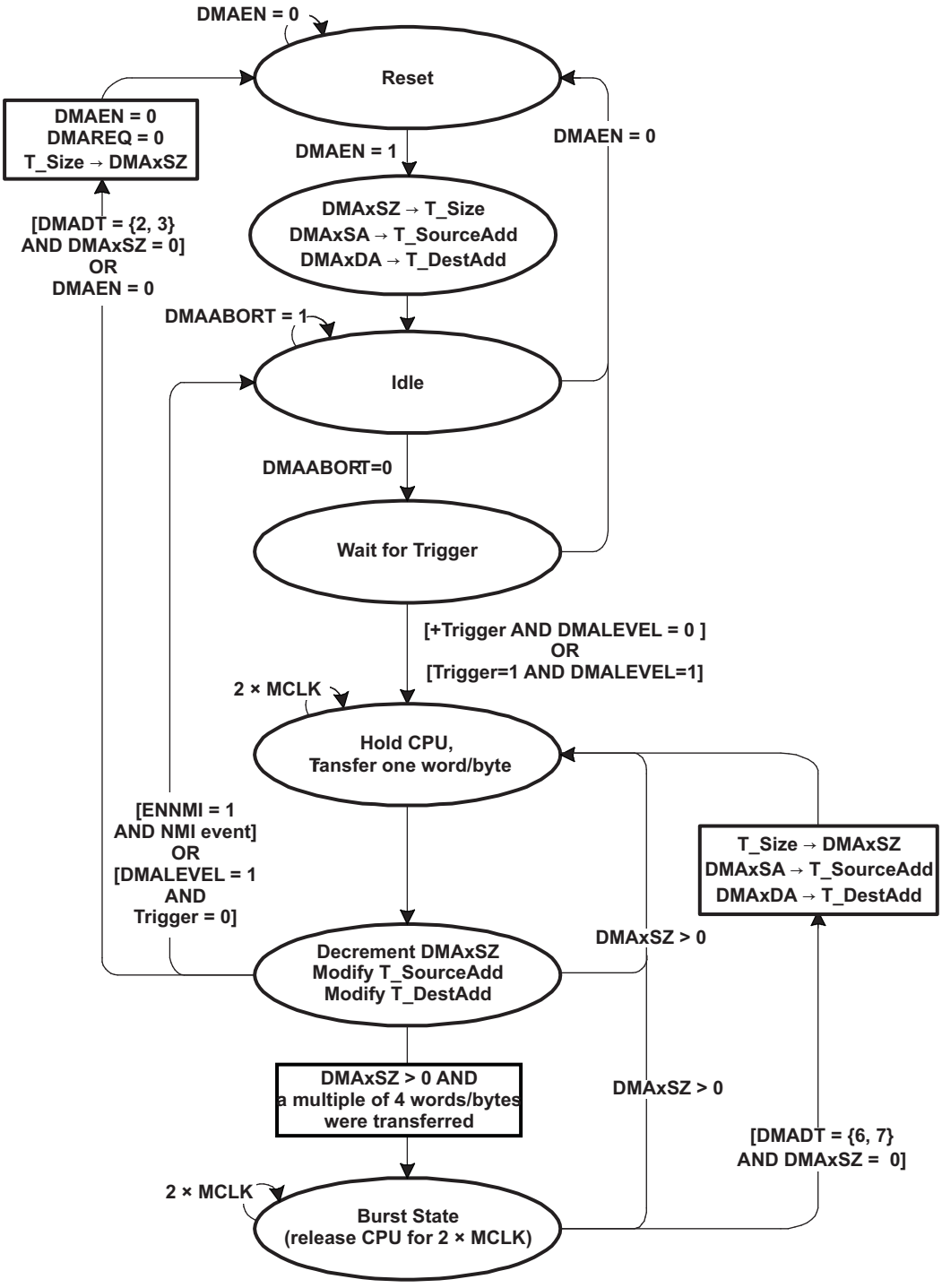


Figure 7-5. DMA Burst-Block Transfer State Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.2.3 Initiating DMA Transfers

Each DMA channel is independently configured for its trigger source with the DMAxTSEL. The DMAxTSEL bits should be modified only when the DMACTLx DMAEN bit is 0. Otherwise, unpredictable DMA triggers may occur.Table 7-2 describes the trigger operation for each type of module. See the device-specific data sheet for the list of triggers available, along with their respective DMAxTSEL values.

When selecting the trigger, the trigger must not have already occurred, or the transfer does not take place.

NOTE: DMA trigger selection and USB

On devices that contain a USB module, the triggers selection from DMA channels 0, 1, or 2 can be used for the USB time stamp event selection (see the USB module description for further details).

7.2.3.1 Edge-Sensitive Triggers

When DMALEVEL = 0, edge-sensitive triggers are used, and the rising edge of the trigger signal initiates the transfer. In single-transfer mode, each transfer requires its own trigger. When using block or burst-block modes, only one trigger is required to initiate the block or burst-block transfer.

7.2.3.2 Level-Sensitive Triggers

When DMALEVEL = 1, level-sensitive triggers are used. For proper operation, level-sensitive triggers can only be used when external trigger DMAE0 is selected as the trigger. DMA transfers are triggered as long as the trigger signal is high and the DMAEN bit remains set.

The trigger signal must remain high for a block or burst-block transfer to complete. If the trigger signal goes low during a block or burst-block transfer, the DMA controller is held in its current state until the trigger goes back high or until the DMA registers are modified by software. If the DMA registers are not modified by software, when the trigger signal goes high again, the transfer resumes from where it was when the trigger signal went low.

When DMALEVEL = 1, transfer modes selected when DMADT = {0, 1, 2, 3} are recommended because the DMAEN bit is automatically reset after the configured transfer.

7.2.4 Halting Executing Instructions for DMA Transfers

The DMARMWDIS bit controls when the CPU is halted for DMA transfers. When DMARMWDIS = 0, the CPU is halted immediately and the transfer begins when a trigger is received. In this case, it is possible that CPU read-modify-write operations can be interrupted by a DMA transfer. When DMARMWDIS = 1, the CPU finishes the currently executing read-modify-write operation before the DMA controller halts the CPU and the transfer begins (see Table 7-2).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 7-2. DMA Trigger Operation

Module	Operation
DMA	A transfer is triggered when the DMAREQ bit is set. The DMAREQ bit is automatically reset when the transfer starts. A transfer is triggered when the DMAxIFG flag is set. DMA0IFG triggers channel 1, DMA1IFG triggers channel 2, and DMA2IFG triggers channel 0. None of the DMAxIFG flags are automatically reset when the transfer starts. A transfer is triggered by the external trigger DMAE0.
Timer_A	A transfer is triggered when the TAxCCR0 CCIFG flag is set. The TAxCCR0 CCIFG flag is automatically reset when the transfer starts. If the TAxCCR0 CCIE bit is set, the TAxCCR0 CCIFG flag dies not trigger a transfer. A transfer is triggered when the TAxCCR2 CCIFG flag is set. The TAxCCR2 CCIFG flag is automatically reset when the transfer starts. If the TAxCCR2 CCIE bit is set, the TAxCCR2 CCIFG flag does not trigger a transfer.
Timer_B	A transfer is triggered when the TBxCCR0 CCIFG flag is set. The TBxCCR0 CCIFG flag is automatically reset when the transfer starts. If the TBxCCR0 CCIE bit is set, the TBxCCR0 CCIFG flag does not trigger a transfer. A transfer is triggered when the TBxCCR2 CCIFG flag is set. The TBxCCR2 CCIFG flag is automatically reset when the transfer starts. If the TBxCCR2 CCIE bit is set, the TBxCCR2 CCIFG flag does not trigger a transfer.
USCI_Ax	A transfer is triggered when USCI_Ax receives new data. UCAxRXIFG is automatically reset when the transfer starts. If UCAxRXIE is set, the UCAxRXIFG does not trigger a transfer. A transfer is triggered when USCI_Ax is ready to transmit new data. UCAxTXIFG is automatically reset when the transfer starts. If UCAxTXIE is set, the UCAxTXIFG does not trigger a transfer.
USCI_Bx	A transfer is triggered when USCI_Bx receives new data. UCBxRXIFG is automatically reset when the transfer starts. If UCBxRXIE is set, the UCBxRXIFG does not trigger a transfer. A transfer is triggered when USCI_Bx is ready to transmit new data. UCBxTXIFG is automatically reset when the transfer starts. If UCBxTXIE is set, the UCBxTXIFG does not trigger a transfer.
DAC12_A	A transfer is triggered when the DAC12_xCTL0 DAC12IFG flag is set. The DAC12_xCTL0 DAC12IFG flag is automatically cleared when the transfer starts. If the DAC12_xCTL0 DAC12IE bit is set, the DAC12_xCTL0 DAC12IFG flag does not trigger a transfer.
ADC12_A	A transfer is triggered by an ADC12IFG flag. When single-channel conversions are performed, the corresponding ADC12IFG is the trigger. When sequences are used, the ADC12IFG for the last conversion in the sequence is the trigger. A transfer is triggered when the conversion is completed and the ADC12IFG is set. Setting the ADC12IFG with software does not trigger a transfer. All ADC12IFG flags are automatically reset when the associated ADC12MEMx register is accessed by the DMA controller.
MPY	A transfer is triggered when the hardware multiplier is ready for a new operand.
Reserved	No transfer is triggered.

7.2.5 Stopping DMA Transfers

There are two ways to stop DMA transfers in progress:

- A single, block, or burst-block transfer may be stopped with an NMI, if the ENNMI bit is set in register DMACTL1.
- A burst-block transfer may be stopped by clearing the DMAEN bit.

7.2.6 DMA Channel Priorities

The default DMA channel priorities are DMA0 through DMA7. If two or three triggers happen simultaneously or are pending, the channel with the highest priority completes its transfer (single, block, or burst-block transfer) first, then the second priority channel, then the third priority channel. Transfers in progress are not halted if a higher-priority channel is triggered. The higher-priority channel waits until the transfer in progress completes before starting.

The DMA channel priorities are configurable with the ROUNDROBIN bit. When the ROUNDROBIN bit is set, the channel that completes a transfer becomes the lowest priority. The *order* of the priority of the channels always stays the same, DMA0-DMA1-DMA2, for example, for three channels. When the ROUNDROBIN bit is cleared, the channel priority returns to the default priority.

DMA Priority	Transfer Occurs	New DMA Priority
DMA0-DMA1-DMA2	DMA1	DMA2-DMA0-DMA1
DMA2-DMA0-DMA1	DMA2	DMA0-DMA1-DMA2
DMA0-DMA1-DMA2	DMA0	DMA1-DMA2-DMA0

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.2.7 DMA Transfer Cycle Time

The DMA controller requires one or two MCLK clock cycles to synchronize before each single transfer or complete block or burst-block transfer. Each byte/word transfer requires two MCLK cycles after synchronization, and one cycle of wait time after the transfer. Because the DMA controller uses MCLK, the DMA cycle time is dependent on the MSP430 operating mode and clock system setup.

If the MCLK source is active but the CPU is off, the DMA controller uses the MCLK source for each transfer, without reenabling the CPU. If the MCLK source is off, the DMA controller temporarily restarts MCLK, sourced with DCOCLK, for the single transfer or complete block or burst-block transfer. The CPU remains off and after the transfer completes, MCLK is turned off. The maximum DMA cycle time for all operating modes is shown in Table 7-3.

Table 7-3. Maximum Single-Transfer DMA Cycle Time

CPU Operating Mode Clock Source	Maximum DMA Cycle Time
Active mode MCLK = DCOCLK	4 MCLK cycles
Active mode MCLK = LFXT1CLK	4 MCLK cycles
Low-power mode LPM0/1 MCLK = DCOCLK	5 MCLK cycles
Low-power mode LPM3/4 MCLK = DCOCLK	5 MCLK cycles + 5 µs(1)
Low-power mode LPM0/1 MCLK = LFXT1CLK	5 MCLK cycles
Low-power mode LPM3 MCLK = LFXT1CLK	5 MCLK cycles
Low-power mode LPM4 MCLK = LFXT1CLK	5 MCLK cycles + 5 µs(1)

(1) The additional 5 µs are needed to start the DCOCLK. It is the $t_{(LPMx)}$ parameter in the data sheet.

7.2.8 Using DMA With System Interrupts

DMA transfers are not interruptible by system interrupts. System interrupts remain pending until the completion of the transfer. NMIs can interrupt the DMA controller if the ENNMI bit is set.

System interrupt service routines are interrupted by DMA transfers. If an interrupt service routine or other routine must execute with no interruptions, the DMA controller should be disabled prior to executing the routine.

7.2.9 DMA Controller Interrupts

Each DMA channel has its own DMAIFG flag. Each DMAIFG flag is set in any mode when the corresponding DMAxSZ register counts to zero. If the corresponding DMAIE and GIE bits are set, an interrupt request is generated.

All DMAIFG flags are prioritized, with DMA0IFG being the highest, and combined to source a single interrupt vector. The highest-priority enabled interrupt generates a number in the DMAIV register. This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled DMA interrupts do not affect the DMAIV value.

Any access, read or write, of the DMAIV register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. For example, assume that DMA0 has the highest priority. If the DMA0IFG and DMA2IFG flags are set when the interrupt service routine accesses the DMAIV register, DMA0IFG is reset automatically. After the RETI instruction of the interrupt service routine is executed, the DMA2IFG generates another interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.2.9.1 DMAIV Software Example

The following software example shows the recommended use of DMAIV and the handling overhead for an eight channel DMA controller. The DMAIV value is added to the PC to automatically jump to the appropriate routine.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself.

```
;Interrupt handler for DMAxIFG                    Cycles

DMA_HND      ...              ; Interrupt latency          6
         ADD      &DMAIV,PC   ; Add offset to Jump table   3
         RETI                 ; Vector  0: No interrupt    5
         JMP      DMA0_HND    ; Vector  2: DMA channel 0   2
         JMP      DMA1_HND    ; Vector  4: DMA channel 1   2
         JMP      DMA2_HND    ; Vector  6: DMA channel 2   2
         JMP      DMA3_HND    ; Vector  8: DMA channel 3   2
         JMP      DMA4_HND    ; Vector 10: DMA channel 4   2
         JMP      DMA5_HND    ; Vector 12: DMA channel 5   2
         JMP      DMA6_HND    ; Vector 14: DMA channel 6   2
         JMP      DMA7_HND    ; Vector 16: DMA channel 7   2

DMA7_HND                     ; Vector 16: DMA channel 7
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA6_HND                     ; Vector 14: DMA channel 6
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA5_HND                     ; Vector 12: DMA channel 5
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA4_HND                     ; Vector 10: DMA channel 4
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA3_HND                     ; Vector 8: DMA channel 3
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA2_HND                     ; Vector 6: DMA channel 2
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA1_HND                     ; Vector 4: DMA channel 1
         ...                 ; Task starts here
         RETI                ; Back to main program       5

DMA0_HND                     ; Vector 2: DMA channel 0
         ...                 ; Task starts here
         RETI                ; Back to main program       5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.2.10 Using the USCI_B I²C Module With the DMA Controller

The USCI_B I²C module provides two trigger sources for the DMA controller. The USCI_B I²C module can trigger a transfer when new I²C data is received and the when the transmit data is needed.

7.2.11 Using ADC12 With the DMA Controller

MSP430 devices with an integrated DMA controller can automatically move data from any ADC12MEMx register to another location. DMA transfers are done without CPU intervention and independently of any low-power modes. The DMA controller increases throughput of the ADC12 module, and enhances low-power applications allowing the CPU to remain off while data transfers occur.

DMA transfers can be triggered from any ADC12IFG flag. When CONSEQx = {0,2}, the ADC12IFG flag for the ADC12MEMx used for the conversion can trigger a DMA transfer. When CONSEQx = {1,3}, the ADC12IFG flag for the last ADC12MEMx in the sequence can trigger a DMA transfer. Any ADC12IFG flag is automatically cleared when the DMA controller accesses the corresponding ADC12MEMx.

7.2.12 Using DAC12 With the DMA Controller

MSP430 devices with an integrated DMA controller can automatically move data to the DAC12_xDAT register. DMA transfers are done without CPU intervention and independently of any low-power modes. The DMA controller increases throughput to the DAC12 module, and enhances low-power applications allowing the CPU to remain off while data transfers occur.

Applications requiring periodic waveform generation can benefit from using the DMA controller with the DAC12. For example, an application that produces a sinusoidal waveform may store the sinusoid values in a table. The DMA controller can continuously and automatically transfer the values to the DAC12 at specific intervals creating the sinusoid with zero CPU execution. The DAC12_xCTL DAC12IFG flag is automatically cleared when the DMA controller accesses the DAC12_xDAT register.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3 DMA Registers

The DMA module registers are listed in Table 7-4. The base addresses can be found in the device-specific data sheet. Each channel starts at its respective base address. The address offsets are listed in Table 7-4.

Table 7-4. DMA Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	DMACTL0	DMA Control 0	Read/write	Word	0000h	Section 7.3.1
02h	DMACTL1	DMA Control 1	Read/write	Word	0000h	Section 7.3.2
04h	DMACTL2	DMA Control 2	Read/write	Word	0000h	Section 7.3.3
06h	DMACTL3	DMA Control 3	Read/write	Word	0000h	Section 7.3.4
08h	DMACTL4	DMA Control 4	Read/write	Word	0000h	Section 7.3.5
0Eh	DMAIV	DMA Interrupt Vector	Read only	Word	0000h	Section 7.3.10
00h	DMA0CTL	DMA Channel 0 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA0SA	DMA Channel 0 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA0DA	DMA Channel 0 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA0SZ	DMA Channel 0 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA1CTL	DMA Channel 1 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA1SA	DMA Channel 1 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA1DA	DMA Channel 1 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA1SZ	DMA Channel 1 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA2CTL	DMA Channel 2 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA2SA	DMA Channel 2 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA2DA	DMA Channel 2 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA2SZ	DMA Channel 2 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA3CTL	DMA Channel 3 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA3SA	DMA Channel 3 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA3DA	DMA Channel 3 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA3SZ	DMA Channel 3 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA4CTL	DMA Channel 4 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA4SA	DMA Channel 4 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA4DA	DMA Channel 4 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA4SZ	DMA Channel 4 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA5CTL	DMA Channel 5 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA5SA	DMA Channel 5 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA5DA	DMA Channel 5 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA5SZ	DMA Channel 5 Transfer Size	Read/write	Word	undefined	Section 7.3.9
00h	DMA6CTL	DMA Channel 6 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA6SA	DMA Channel 6 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA6DA	DMA Channel 6 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA6SZ	DMA Channel 6 Transfer Size	Read/write	Word	undefined	Section 7.3.9

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 7-4. DMA Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	DMA7CTL	DMA Channel 7 Control	Read/write	Word	0000h	Section 7.3.6
02h	DMA7SA	DMA Channel 7 Source Address	Read/write	Word, double word	undefined	Section 7.3.7
06h	DMA7DA	DMA Channel 7 Destination Address	Read/write	Word, double word	undefined	Section 7.3.8
0Ah	DMA7SZ	DMA Channel 7 Transfer Size	Read/write	Word	undefined	Section 7.3.9

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.1 DMACTL0 Register

DMA Control 0 Register

Figure 7-6. DMACTL0 Register

15	14	13	12	11	10	9	8
Reserved			DMA1TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Reserved			DMA0TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 7-5. DMACTL0 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-8	DMA1TSEL	RW	0h	DMA 1 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA1TRIG0 00001b = DMA1TRIG1 00010b = DMA1TRIG2 ⋮ 11110b = DMA1TRIG30 11111b = DMA1TRIG31
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-0	DMA0TSEL	RW	0h	DMA 0 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA0TRIG0 00001b = DMA0TRIG1 00010b = DMA0TRIG2 ⋮ 11110b = DMA0TRIG30 11111b = DMA0TRIG31

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.2 DMACTL1 Register

DMA Control 1 Register

Figure 7-7. DMACTL1 Register

15	14	13	12	11	10	9	8
Reserved			DMA3TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Reserved			DMA2TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 7-6. DMACTL1 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-8	DMA3TSEL	RW	0h	DMA 3 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA3TRIG0 00001b = DMA3TRIG1 00010b = DMA3TRIG2 ⋮ 11110b = DMA3TRIG30 11111b = DMA3TRIG31
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-0	DMA2TSEL	RW	0h	DMA 2 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA2TRIG0 00001b = DMA2TRIG1 00010b = DMA2TRIG2 ⋮ 11110b = DMA2TRIG30 11111b = DMA2TRIG31

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.3 DMACTL2 Register

DMA Control 2 Register

Figure 7-8. DMACTL2 Register

15	14	13	12	11	10	9	8
Reserved			DMA5TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Reserved			DMA4TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 7-7. DMACTL2 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-8	DMA5TSEL	RW	0h	DMA 5 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA5TRIG0 00001b = DMA5TRIG1 00010b = DMA5TRIG2 ⋮ 11110b = DMA5TRIG30 11111b = DMA5TRIG31
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-0	DMA4TSEL	RW	0h	DMA 4 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA4TRIG0 00001b = DMA4TRIG1 00010b = DMA4TRIG2 ⋮ 11110b = DMA4TRIG30 11111b = DMA4TRIG31

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.4 DMACTL3 Register

DMA Control 3 Register

Figure 7-9. DMACTL3 Register

15	14	13	12	11	10	9	8
Reserved			DMA7TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Reserved			DMA6TSEL				
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 7-8. DMACTL3 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-8	DMA7TSEL	RW	0h	DMA 7 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA7TRIG0 00001b = DMA7TRIG1 00010b = DMA7TRIG2 ⋮ 11110b = DMA7TRIG30 11111b = DMA7TRIG31
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-0	DMA6TSEL	RW	0h	DMA 6 trigger select. These bits select the DMA transfer trigger. See the device-specific data sheet for number of channels and trigger assignment. 00000b = DMA6TRIG0 00001b = DMA6TRIG1 00010b = DMA6TRIG2 ⋮ 11110b = DMA6TRIG30 11111b = DMA6TRIG31

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.5 DMACTL4 Register

DMA Control 4 Register

Figure 7-10. DMACTL4 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved					DMARMWDIS	ROUNDROBIN	ENNMI
r0	r0	r0	r0	r0	rw-(0)	rw-(0)	rw-(0)

Table 7-9. DMACTL4 Register Description

Bit	Field	Type	Reset	Description
15-3	Reserved	R	0h	Reserved. Always reads as 0.
2	DMARMWDIS	RW	0h	Read-modify-write disable. When set, this bit inhibits any DMA transfers from occurring during CPU read-modify-write operations. 0b = DMA transfers can occur during read-modify-write CPU operations. 1b = DMA transfers inhibited during read-modify-write CPU operations
1	ROUNDROBIN	RW	0h	Round robin. This bit enables the round-robin DMA channel priorities. 0b = DMA channel priority is DMA0-DMA1-DMA2 - ..... -DMA7. 1b = DMA channel priority changes with each transfer.
0	ENNMI	RW	0h	Enable NMI. This bit enables the interruption of a DMA transfer by an NMI. When an NMI interrupts a DMA transfer, the current transfer is completed normally, further transfers are stopped and DMAABORT is set. 0b = NMI does not interrupt DMA transfer. 1b = NMI interrupts a DMA transfer.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.6 DMAxCTL Register

DMA Channel x Control Register

Figure 7-11. DMAxCTL Register

15	14	13	12	11	10	9	8
Reserved	DMADT			DMADSTINCR		DMASRCINCR	
r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
DMADSTBYTE	DMASRCBYTE	DMALEVEL	DMAEN	DMAIFG	DMAIE	DMAABORT	DMAREQ
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 7-10. DMAxCTL Register Description

Bit	Field	Type	Reset	Description
15	Reserved	R	0h	Reserved. Always reads as 0.
14-12	DMADT	RW	0h	DMA transfer mode 000b = Single transfer 001b = Block transfer 010b = Burst-block transfer 011b = Burst-block transfer 100b = Repeated single transfer 101b = Repeated block transfer 110b = Repeated burst-block transfer 111b = Repeated burst-block transfer
11-10	DMADSTINCR	RW	0h	DMA destination increment. This bit selects automatic incrementing or decrementing of the destination address after each byte or word transfer. When DMADSTBYTE = 1, the destination address increments/decrements by one. When DMADSTBYTE = 0, the destination address increments/decrements by two. The DMAxDA is copied into a temporary register and the temporary register is incremented or decremented. DMAxDA is not incremented or decremented. 00b = Destination address is unchanged. 01b = Destination address is unchanged. 10b = Destination address is decremented. 11b = Destination address is incremented.
9-8	DMASRCINCR	RW	0h	DMA source increment. This bit selects automatic incrementing or decrementing of the source address for each byte or word transfer. When DMASRCBYTE = 1, the source address increments/decrements by one. When DMASRCBYTE = 0, the source address increments/decrements by two. The DMAxSA is copied into a temporary register and the temporary register is incremented or decremented. DMAxSA is not incremented or decremented. 00b = Source address is unchanged. 01b = Source address is unchanged. 10b = Source address is decremented. 11b = Source address is incremented.
7	DMADSTBYTE	RW	0h	DMA destination byte. This bit selects the destination as a byte or word. 0b = Word 1b = Byte
6	DMASRCBYTE	RW	0h	DMA source byte. This bit selects the source as a byte or word. 0b = Word 1b = Byte
5	DMALEVEL	RW	0h	DMA level. This bit selects between edge-sensitive and level-sensitive triggers. 0b = Edge sensitive (rising edge) 1b = Level sensitive (high level)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 7-10. DMAxCTL Register Description (continued)

Bit	Field	Type	Reset	Description
4	DMAEN	RW	0h	DMA enable 0b = Disabled 1b = Enabled
3	DMAIFG	RW	0h	DMA interrupt flag 0b = No interrupt pending 1b = Interrupt pending
2	DMAIE	RW	0h	DMA interrupt enable 0b = Disabled 1b = Enabled
1	DMAABORT	RW	0h	DMA abort. This bit indicates if a DMA transfer was interrupt by an NMI. 0b = DMA transfer not interrupted 1b = DMA transfer interrupted by NMI
0	DMAREQ	RW	0h	DMA request. Software-controlled DMA start. DMAREQ is reset automatically. 0b = No DMA start 1b = Start DMA

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.7 DMAxSA Register

DMA Channel x Source Address Register

Figure 7-12. DMAxSA Register

31	30	29	28	27	26	25	24
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
23	22	21	20	19	18	17	16
Reserved				DMAxSA			
r0	r0	r0	r0	rw	rw	rw	rw
15	14	13	12	11	10	9	8
DMAxSA							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
DMAxSA							
rw	rw	rw	rw	rw	rw	rw	rw

Table 7-11. DMAxSA Register Description

Bit	Field	Type	Reset	Description
31-20	Reserved	R	0h	Reserved. Always reads as 0.
19-0	DMAxSA	RW	undefined	DMA source address. The source address register points to the DMA source address for single transfers or the first source address for block transfers. The source address register remains unchanged during block and burst-block transfers. There are two words for the DMAxSA register. Bits 31-20 are reserved and always read as zero. Reading or writing bits 19-16 requires the use of extended instructions. When writing to DMAxSA with word instructions, bits 19-16 are cleared.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.8 DMAxDA Register

DMA Channel x Destination Address Register

Figure 7-13. DMAxDA Register

31	30	29	28	27	26	25	24
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
23	22	21	20	19	18	17	16
Reserved				DMAxDA			
r0	r0	r0	r0	rw	rw	rw	rw
15	14	13	12	11	10	9	8
DMAxDA							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
DMAxDA							
rw	rw	rw	rw	rw	rw	rw	rw

Table 7-12. DMAxDA Register Description

Bit	Field	Type	Reset	Description
31-20	Reserved	R	0h	Reserved. Always reads as 0.
19-0	DMAxDA	RW	undefined	DMA destination address. The destination address register points to the DMA destination address for single transfers or the first destination address for block transfers. The destination address register remains unchanged during block and burst-block transfers. There are two words for the DMAxDA register. Bits 31-20 are reserved and always read as zero. Reading or writing bits 19-16 requires the use of extended instructions. When writing to DMAxDA with word instructions, bits 19-16 are cleared.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.9 DMAxSZ Register

DMA Channel x Size Address Register

Figure 7-14. DMAxSZ Register

15	14	13	12	11	10	9	8
DMAxSZ							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
DMAxSZ							
rw	rw	rw	rw	rw	rw	rw	rw

Table 7-13. DMAxSZ Register Description

Bit	Field	Type	Reset	Description
15-0	DMAxSZ	RW	undefined	DMA size. The DMA size register defines the number of byte/word data per block transfer. DMAxSZ register decrements with each word or byte transfer. When DMAxSZ decrements to 0, it is immediately and automatically reloaded with its previously initialized value. 00000h = Transfer is disabled. 00001h = One byte or word is transferred. 00002h = Two bytes or words are transferred. ⋮ 0FFFFh = 65535 bytes or words are transferred.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

7.3.10 DMAIV Register

DMA Interrupt Vector Register

Figure 7-15. DMAIV Register

15	14	13	12	11	10	9	8
DMAIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
DMAIV							
r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 7-14. DMAIV Register Description

Bit	Field	Type	Reset	Description
15-0	DMAIV	R	0h	DMA interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: DMA channel 0; Interrupt Flag: DMA0IFG; Interrupt Priority: Highest 04h = Interrupt Source: DMA channel 1; Interrupt Flag: DMA1IFG 06h = Interrupt Source: DMA channel 2; Interrupt Flag: DMA2IFG 08h = Interrupt Source: DMA channel 3; Interrupt Flag: DMA3IFG 0Ah = Interrupt Source: DMA channel 4; Interrupt Flag: DMA4IFG 0Ch = Interrupt Source: DMA channel 5; Interrupt Flag: DMA5IFG 0Eh = Interrupt Source: DMA channel 6; Interrupt Flag: DMA6IFG 10h = Interrupt Source: DMA channel 7; Interrupt Flag: DMA7IFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Digital I/O Module

This chapter describes the operation of the digital I/O ports in all devices.

Topic **Page**

8.1	Digital I/O Introduction	344
8.2	Digital I/O Operation	345
8.3	I/O Configuration and LPMx.5 Low-Power Modes	348
8.4	Digital I/O Registers	350

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.1 Digital I/O Introduction

The digital I/O features include:

- Independently programmable individual I/Os
- Any combination of input or output
- Individually configurable P1 and P2 interrupts. Some devices may include additional port interrupts.
- Independent input and output data registers
- Individually configurable pullup or pulldown resistors

Devices within the family may have up to twelve digital I/O ports implemented (P1 to P11 and PJ). Most ports contain eight I/O lines; however, some ports may contain less (see the device-specific data sheet for ports available). Each I/O line is individually configurable for input or output direction, and each can be individually read or written. Each I/O line is individually configurable for pullup or pulldown resistors, as well as, configurable drive strength, full or reduced. PJ contains only four I/O lines.

Ports P1 and P2 always have interrupt capability. Each interrupt for the P1 and P2 I/O lines can be individually enabled and configured to provide an interrupt on a rising or falling edge of an input signal. All P1 I/O lines source a single interrupt vector P1IV, and all P2 I/O lines source a different, single interrupt vector P2IV. On some devices, additional ports with interrupt capability may be available (see the device-specific data sheet for details) and contain their own respective interrupt vectors.

Individual ports can be accessed as byte-wide ports or can be combined into word-wide ports and accessed via word formats. Port pairs P1 and P2, P3 and P4, P5 and P6, and so on, are associated with the names PA, PB, PC, and so on, respectively. All port registers are handled in this manner with this naming convention except for the interrupt vector registers; for example, PAIV does not exist for P1IV and P2IV.

When writing to port PA with word operations, all 16 bits are written to the port. When writing to the lower byte of the PA port using byte operations, the upper byte remains unchanged. Similarly, writing to the upper byte of the PA port using byte instructions leaves the lower byte unchanged. When writing to a port that contains less than the maximum number of bits possible, the unused bits are a "don't care". Ports PB, PC, PD, PE, and PF behave similarly.

Reading of the PA port using word operations causes all 16 bits to be transferred to the destination. Reading the lower or upper byte of the PA port (P1 or P2) and storing to memory using byte operations causes only the lower or upper byte to be transferred to the destination, respectively. Reading of the PA port and storing to a general-purpose register using byte operations causes the byte transferred to be written to the least significant byte of the register. The upper significant byte of the destination register is cleared automatically. Ports PB, PC, PD, PE, and PF behave similarly. When reading from ports that contain less than the maximum bits possible, unused bits are read as zeros (similarly for port PJ).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.2 Digital I/O Operation

The digital I/O are configured with user software. The setup and operation of the digital I/O are discussed in the following sections.

8.2.1 Input Registers (PxIN)

Each bit in each PxIN register reflects the value of the input signal at the corresponding I/O pin when the pin is configured as I/O function. These registers are read only.

- Bit = 0: Input is low
- Bit = 1: Input is high

NOTE: Writing to read-only registers PxIN

Writing to these read-only registers results in increased current consumption while the write attempt is active.

8.2.2 Output Registers (PxOUT)

Each bit in each PxOUT register is the value to be output on the corresponding I/O pin when the pin is configured as I/O function, output direction.

- Bit = 0: Output is low
- Bit = 1: Output is high

If the pin is configured as I/O function, input direction and the pullup/pulldown resistor are enabled; the corresponding bit in the PxOUT register selects pullup or pulldown.

- Bit = 0: Pin is pulled down
- Bit = 1: Pin is pulled up

8.2.3 Direction Registers (PxDIR)

Each bit in each PxDIR register selects the direction of the corresponding I/O pin, regardless of the selected function for the pin. PxDIR bits for I/O pins that are selected for other functions must be set as required by the other function.

- Bit = 0: Port pin is switched to input direction
- Bit = 1: Port pin is switched to output direction

8.2.4 Pullup or Pulldown Resistor Enable Registers (PxREN)

Each bit in each PxREN register enables or disables the pullup/pulldown resistor of the corresponding I/O pin. The corresponding bit in the PxOUT register selects if the pin contains a pullup or pulldown.

- Bit = 0: Pullup/pulldown resistor disabled
- Bit = 1: Pullup/pulldown resistor enabled

Table 8-1 summarizes the usage of PxDIR, PxREN, and PxOUT for proper I/O configuration.

Table 8-1. I/O Configuration

PxDIR	PxREN	PxOUT	I/O Configuration
0	0	x	Input
0	1	0	Input with pulldown resistor
0	1	1	Input with pullup resistor
1	x	x	Output

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.2.5 Output Drive Strength Registers (PxDS)

Each bit in each PxDS register selects either full drive or reduced drive strength. Default is reduced drive strength.

- Bit = 0: Reduced drive strength
- Bit = 1: Full drive strength

NOTE: Drive strength and EMI

All outputs default to reduced drive strength to reduce EMI. Using full drive strength can result in increased EMI.

8.2.6 Function Select Registers (PxSEL)

Port pins are often multiplexed with other peripheral module functions. See the device-specific data sheet to determine pin functions. Each PxSEL bit is used to select the pin function – I/O port or peripheral module function.

- Bit = 0: I/O Function is selected for the pin
- Bit = 1: Peripheral module function is selected for the pin

Setting PxSEL = 1 does not automatically set the pin direction. Other peripheral module functions may require the PxDIR bits to be configured according to the direction needed for the module function. See the pin schematics in the device-specific data sheet.

NOTE: P1 and P2 interrupts are disabled when PxSEL = 1

When any PxSEL bit is set, the corresponding pin's interrupt function is disabled. Therefore, signals on these pins does not generate P1 or P2 interrupts, regardless of the state of the corresponding P1IE or P2IE bit.

When a port pin is selected as an input to a peripheral, the input signal to the peripheral is a latched representation of the signal at the device pin. While its corresponding PxSEL = 1, the internal input signal follows the signal at the pin. However, if its PxSEL = 0, the input to the peripheral maintains the value of the input signal at the device pin before its corresponding PxSEL bit was reset.

8.2.7 Port Interrupts

Each pin in ports P1 and P2 has interrupt capability, configured with the PxIFG, PxIE, and PxIES registers. On some devices, additional ports have interrupt capability (see the device-specific data sheet). All P1 interrupt flags are prioritized, with P1IFG.0 being the highest, and combined to source a single interrupt vector. The highest priority enabled interrupt generates a number in the P1IV register. This number can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled P1 interrupts do not affect the P1IV value. The same functionality exists for P2. The PxIV registers are word access only. Some devices may contain additional port interrupts besides P1 and P2. See the device specific data sheet to determine which port interrupts are available.

Each PxIFG bit is the interrupt flag for its corresponding I/O pin and is set when the selected input signal edge occurs at the pin. All PxIFG interrupt flags request an interrupt when their corresponding PxIE bit and the GIE bit are set. Software can also set each PxIFG flag, providing a way to generate a software-initiated interrupt.

- Bit = 0: No interrupt is pending
- Bit = 1: An interrupt is pending

Only transitions, not static levels, cause interrupts. If any PxIFG flag becomes set during a Px interrupt service routine, or is set after the RETI instruction of a Px interrupt service routine is executed, the set PxIFG flag generates another interrupt. This ensures that each transition is acknowledged.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: PxIFG flags when changing PxOUT, PxDIR, or PxREN

Writing to P1OUT, P1DIR, P1REN, P2OUT, P2DIR, or P2REN can result in setting the corresponding P1IFG or P2IFG flags.

Any access (read or write) of the P1IV register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. For example, assume that P1IFG.0 has the highest priority. If the P1IFG.0 and P1IFG.2 flags are set when the interrupt service routine accesses the P1IV register, P1IFG.0 is reset automatically. After the RETI instruction of the interrupt service routine is executed, the P1IFG.2 generates another interrupt.

Port P2 interrupts behave similarly, and source a separate single interrupt vector and utilize the P2IV register.

Port Interrupt Software Example

The following software example shows the recommended use of P1IV and the handling overhead. The P1IV value is added to the PC to automatically jump to the appropriate routine. The P2IV is similar.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself.

```
;Interrupt handler for P1                                  Cycles
P1_HND   ...                ; Interrupt latency              6
         ADD      &P1IV,PC  ; Add offset to Jump table       3
         RETI               ; Vector  0: No interrupt        5
         JMP      P1_0_HND  ; Vector  2: Port 1 bit 0        2
         JMP      P1_1_HND  ; Vector  4: Port 1 bit 1        2
         JMP      P1_2_HND  ; Vector  6: Port 1 bit 2        2
         JMP      P1_3_HND  ; Vector  8: Port 1 bit 3        2
         JMP      P1_4_HND  ; Vector 10: Port 1 bit 4        2
         JMP      P1_5_HND  ; Vector 12: Port 1 bit 5        2
         JMP      P1_6_HND  ; Vector 14: Port 1 bit 6        2
         JMP      P1_7_HND  ; Vector 16: Port 1 bit 7        2

P1_7_HND                   ; Vector 16: Port 1 bit 7
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_6_HND                   ; Vector 14: Port 1 bit 6
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_5_HND                   ; Vector 12: Port 1 bit 5
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_4_HND                   ; Vector 10: Port 1 bit 4
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_3_HND                   ; Vector 8: Port 1 bit 3
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_2_HND                   ; Vector 6: Port 1 bit 2
         ...               ; Task starts here
         RETI              ; Back to main program           5

P1_1_HND                   ; Vector 4: Port 1 bit 1
         ...               ; Task starts here
         RETI              ; Back to main program           5
P1_0_HND                   ; Vector 2: Port 1 bit 0
         ...               ; Task starts here
         RETI              ; Back to main program           5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Interrupt Edge Select Registers (PxIES)

Each PxIES bit selects the interrupt edge for the corresponding I/O pin.

- Bit = 0: Respective PxIFG flag is set with a low-to-high transition
- Bit = 1: Respective PxIFG flag is set with a high-to-low transition

NOTE: Writing to PxIES

Writing to P1IES or P2IES for each corresponding I/O can result in setting the corresponding interrupt flags.

PxIES	PxIN	PxIFG
0 → 1	0	May be set
0 → 1	1	Unchanged
1 → 0	0	Unchanged
1 → 0	1	May be set

Interrupt Enable Registers (PxIE)

Each PxIE bit enables the associated PxIFG interrupt flag.

- Bit = 0: The interrupt is disabled
- Bit = 1: The interrupt is enabled

8.2.8 Configuring Unused Port Pins

Unused I/O pins should be configured as I/O function, output direction, and left unconnected on the PC board, to prevent a floating input and reduce power consumption. The value of the PxOUT bit is don't care, because the pin is unconnected. Alternatively, the integrated pullup/pulldown resistor can be enabled by setting the PxREN bit of the unused pin to prevent the floating input. See the SYS chapter for termination of unused pins.

NOTE: Configuring port J and shared JTAG pins:

Application should ensure that port PJ is configured properly to prevent a floating input. Because port PJ is shared with the JTAG function, floating inputs may not be noticed when in an emulation environment. Port J is initialized to high-impedance inputs by default.

8.3 I/O Configuration and LPMx.5 Low-Power Modes

NOTE: The LPMx.5 low-power modes may not be available on all devices. The LPM4.5 power mode allows for lowest power consumption and no clocks are available. The LPM3.5 power mode allows for RTC mode operation at the lowest power consumption available. See the SYS chapter for details; also see the device-specific datasheet for LPMx.5 low-power modes that are available. With respect to the digital I/O, this section is applicable for both LPM3.5 and LPM4.5.

The regulator of the Power Management Module (PMM) is disabled upon entering LPMx.5 (LPM3.5 or LPM4.5), which causes all I/O register configurations to be lost. Because the I/O register configurations are lost, the configuration of I/O pins must be handled differently to ensure that all pins in the application behave in a controlled manner upon entering and exiting LPMx.5. Properly setting the I/O pins is critical to achieving the lowest possible power consumption in LPMx.5, as well as preventing any possible uncontrolled input or output I/O state in the application. The application has complete control of the I/O pin conditions preventing the possibility of unwanted spurious activity upon entry and exit from LPMx.5. The detailed flow for entering and exiting LPMx.5 with respect to the I/O operation is as follows:

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

1. Set all I/Os to general purpose I/Os and configure as needed. Each I/O can be set to input high impedance, input with pulldown, input with pullup, output high (low or high drive strength), or output low (low or high drive strength). It is critical that no inputs are left floating in the application, otherwise excess current may be drawn in LPMx.5. Configuring the I/O in this manner ensures that each pin is in a safe condition prior to entering LPMx.5.

   Optionally, configure input interrupt pins for wake-up from LPMx.5. To wake the device from LPMx.5, a general-purpose I/O port must contain an input port with interrupt capability. Not all devices include wakeup from LPMx.5 via I/O, and not all inputs with interrupt capability offer wakeup from LPMx.5. See the device-specific data sheet for availability. To configure a port to wake up the device, it should be configured properly prior to entering LPMx.5. Each port should be configured as general-purpose input. Pulldowns or pullups can be applied if required. Setting the PxIES bit of the corresponding register determines the edge transition that wakes the device. Lastly, the PxIE for the port must be enabled, as well as the general interrupt enable.

   NOTE: It is not possible to wakeup from LPMx.5 if its respective interrupt flag is already asserted. It is recommended that the respective flag be cleared prior to entering LPMx.5. It is also recommended that GIE = 1 be set prior to entry into LPMx.5. Any pending flags in this case could then be serviced prior to LPMx.5 entry.

   Although it is recommended to set GIE = 1 prior to entering LPMx.5, it is not required. Device wakeup from LPMx.5 with an enabled wakeup function will still cause the device to wake up from LPMx.5 even with GIE = 0. If GIE = 0 prior to LPMx.5, additional care may be required. Should the respective interrupt event should occur during LPMx.5 entry, the device may not recognize this or any future interrupt wakeup event on this function.

2. Enter LPMx.5 with LPMx.5 entry sequence, enable general interrupts for wake-up:

```
MOV.B #PMMPW_H, &PMMCTL0_H                  ; Open PMM registers for write
BIS.B #PMMREGOFF, &PMMCTL0_L                ;
BIS   #GIE+CPUOFF+OSCOFF+SCG1+SCG0,SR       ; Enter LPMx.5 when PMMREGOFF is set
```

3. Upon entry into LPMx.5, LOCKLPM5 residing in PM5CTL0 of the PMM module is set automatically. The I/O pin states are held and locked based on the settings prior to LPMx.5 entry. Note that only the pin conditions are retained. All other port configuration register settings such as PxDIR, PxREN, PxOUT, PxDS, PxIES, and PxIE contents are lost.
4. An LPMx.5 wakeup event (for example, an edge on a configured wakeup input pin) starts the BOR entry sequence together with the regulator. All peripheral registers are set to their default conditions. Upon exit from LPMx.5, the I/O pins remain locked while LOCKLPM5 remains set. Keeping the I/O pins locked ensures that all pin conditions remain stable upon entering the active mode regardless of the default I/O register settings.
5. Once in active mode, the I/O configuration and I/O interrupt configuration that was not retained during LPMx.5 should be restored to the values prior to entering LPMx.5. It is recommended to reconfigure the PxIES and PxIE to their previous settings to prevent a false port interrupt from occurring. The LOCKLPM5 bit can then be cleared, which releases the I/O pin conditions and I/O interrupt configuration. Any changes to the port configuration registers while LOCKLPM5 is set, have no effect on the I/O pins.
6. After enabling the I/O interrupts, the I/O interrupt that caused the wakeup can be serviced indicated by the PxIFG flags. These flags can be used directly, or the corresponding PxIV register may be used. Note that the PxIFG flag cannot be cleared until the LOCKLPM5 bit has been cleared.

   NOTE: It is possible that multiple events occurred on various ports. In these cases, multiple PxIFG flags will be set, and it cannot be determined which port has caused the I/O wakeup.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4 Digital I/O Registers

The digital I/O registers are listed in Table 8-2. The base addresses can be found in the device-specific data sheet. Each port grouping begins at its base address. The address offsets are given in Table 8-2.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 8-2. Digital I/O Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
0Eh	P1IV	Port 1 Interrupt Vector	Read only	Word	0000h	Section 8.4.1
0Eh	P1IV_L		Read only	Byte	00h	
0Fh	P1IV_H		Read only	Byte	00h	
1Eh	P2IV	Port 2 Interrupt Vector	Read only	Word	0000h	Section 8.4.2
1Eh	P2IV_L		Read only	Byte	00h	
1Fh	P2IV_H		Read only	Byte	00h	
00h	P1IN or PAIN_L	Port 1 Input	Read only	Byte		Section 8.4.9
02h	P1OUT or PAOUT_L	Port 1 Output	Read/write	Byte	undefined	Section 8.4.10
04h	P1DIR or PADIR_L	Port 1 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P1REN or PAREN_L	Port 1 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P1DS or PADS_L	Port 1 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P1SEL or PASEL_L	Port 1 Port Select	Read/write	Byte	00h	Section 8.4.14
18h	P1IES or PAIES_L	Port 1 Interrupt Edge Select	Read/write	Byte	undefined	Section 8.4.3
1Ah	P1IE or PAIE_L	Port 1 Interrupt Enable	Read/write	Byte	00h	Section 8.4.4
1Ch	P1IFG or PAIFG_L	Port 1 Interrupt Flag	Read/write	Byte	00h	Section 8.4.5
01h	P2IN or PAIN_H	Port 2 Input	Read only	Byte		Section 8.4.9
03h	P2OUT or PAOUT_H	Port 2 Output	Read/write	Byte	undefined	Section 8.4.10
05h	P2DIR or PADIR_H	Port 2 Direction	Read/write	Byte	00h	Section 8.4.11
07h	P2REN or PAREN_H	Port 2 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
09h	P2DS or PADS_H	Port 2 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Bh	P2SEL or PASEL_H	Port 2 Port Select	Read/write	Byte	00h	Section 8.4.14
19h	P2IES or PAIES_H	Port 2 Interrupt Edge Select	Read/write	Byte	undefined	Section 8.4.6
1Bh	P2IE or PAIE_H	Port 2 Interrupt Enable	Read/write	Byte	00h	Section 8.4.7
1Dh	P2IFG or PAIFG_H	Port 2 Interrupt Flag	Read/write	Byte	00h	Section 8.4.8
00h	P3IN or PBIN_L	Port 3 Input	Read only	Byte		Section 8.4.9
02h	P3OUT or PBOUT_L	Port 3 Output	Read/write	Byte	undefined	Section 8.4.10

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 8-2. Digital I/O Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
04h	P3DIR or PBDIR_L	Port 3 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P3REN or PBREN_L	Port 3 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P3DS or PBDS_L	Port 3 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P3SEL or PBSEL_L	Port 3 Port Select	Read/write	Byte	00h	Section 8.4.14
01h	P4IN or PBIN_H	Port 4 Input	Read only	Byte		Section 8.4.9
03h	P4OUT or PBOUT_H	Port 4 Output	Read/write	Byte	undefined	Section 8.4.10
05h	P4DIR or PBDIR_H	Port 4 Direction	Read/write	Byte	00h	Section 8.4.11
07h	P4REN or PBREN_H	Port 4 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
09h	P4DS or PBDS_H	Port 4 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Bh	P4SEL or PBSEL_H	Port 4 Port Select	Read/write	Byte	00h	Section 8.4.14
00h	P5IN or PCIN_L	Port 5 Input	Read only	Byte		Section 8.4.9
02h	P5OUT or PCOUT_L	Port 5 Output	Read/write	Byte	undefined	Section 8.4.10
04h	P5DIR or PCDIR_L	Port 5 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P5REN or PCREN_L	Port 5 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P5DS or PCDS_L	Port 5 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P5SEL or PCSEL_L	Port 5 Port Select	Read/write	Byte	00h	Section 8.4.14
01h	P6IN or PCIN_H	Port 6 Input	Read only	Byte		Section 8.4.9
03h	P6OUT or PCOUT_H	Port 6 Output	Read/write	Byte	undefined	Section 8.4.10
05h	P6DIR or PCDIR_H	Port 6 Direction	Read/write	Byte	00h	Section 8.4.11
07h	P6REN or PCREN_H	Port 6 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
09h	P6DS or PCDS_H	Port 6 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Bh	P6SEL or PCSEL_H	Port 6 Port Select	Read/write	Byte	00h	Section 8.4.14
00h	P7IN or PDIN_L	Port 7 Input	Read only	Byte		Section 8.4.9
02h	P7OUT or PDOUT_L	Port 7 Output	Read/write	Byte	undefined	Section 8.4.10
04h	P7DIR or PDDIR_L	Port 7 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P7REN or PDREN_L	Port 7 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P7DS or PDDS_L	Port 7 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P7SEL or PDSEL_L	Port 7 Port Select	Read/write	Byte	00h	Section 8.4.14

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 8-2. Digital I/O Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
01h	P8IN or PDIN_H	Port 8 Input	Read only	Byte		Section 8.4.9
03h	P8OUT or PDOUT_H	Port 8 Output	Read/write	Byte	undefined	Section 8.4.10
05h	P8DIR or PDDIR_H	Port 8 Direction	Read/write	Byte	00h	Section 8.4.11
07h	P8REN or PDREN_H	Port 8 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
09h	P8DS or PDDS_H	Port 8 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Bh	P8SEL or PDSEL_H	Port 8 Port Select	Read/write	Byte	00h	Section 8.4.14
00h	P9IN or PEIN_L	Port 9 Input	Read only	Byte		Section 8.4.9
02h	P9OUT or PEOUT_L	Port 9 Output	Read/write	Byte	undefined	Section 8.4.10
04h	P9DIR or PEDIR_L	Port 9 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P9REN or PEREN_L	Port 9 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P9DS or PEDS_L	Port 9 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P9SEL or PESEL_L	Port 9 Port Select	Read/write	Byte	00h	Section 8.4.14
01h	P10IN or PEIN_H	Port 10 Input	Read only	Byte		Section 8.4.9
03h	P10OUT or PEOUT_H	Port 10 Output	Read/write	Byte	undefined	Section 8.4.10
05h	P10DIR or PEDIR_H	Port 10 Direction	Read/write	Byte	00h	Section 8.4.11
07h	P10REN or PEREN_H	Port 10 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
09h	P10DS or PEDS_H	Port 10 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Bh	P10SEL or PESEL_H	Port 10 Port Select	Read/write	Byte	00h	Section 8.4.14
00h	P11IN or PFIN_L	Port 11 Input	Read only	Byte		Section 8.4.9
02h	P11OUT or PFOUT_L	Port 11 Output	Read/write	Byte	undefined	Section 8.4.10
04h	P11DIR or PFDIR_L	Port 11 Direction	Read/write	Byte	00h	Section 8.4.11
06h	P11REN or PFREN_L	Port 11 Resistor Enable	Read/write	Byte	00h	Section 8.4.12
08h	P11DS or PFDS_L	Port 11 Drive Strength	Read/write	Byte	00h	Section 8.4.13
0Ah	P11SEL or PFSEL_L	Port 11 Port Select	Read/write	Byte	00h	Section 8.4.14
00h	PAIN	Port A Input	Read only	Word		
00h	PAIN_L		Read only	Byte		
01h	PAIN_H		Read only	Byte		
02h	PAOUT	Port A Output	Read/write	Word	undefined	
02h	PAOUT_L		Read/write	Byte	undefined	
03h	PAOUT_H		Read/write	Byte	undefined	
04h	PADIR	Port A Direction	Read/write	Word	0000h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 8-2. Digital I/O Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
04h	PADIR_L		Read/write	Byte	00h	
05h	PADIR_H		Read/write	Byte	00h	
06h	PAREN	Port A Resistor Enable	Read/write	Word	0000h	
06h	PAREN_L		Read/write	Byte	00h	
07h	PAREN_H		Read/write	Byte	00h	
08h	PADS	Port A Drive Strength	Read/write	Word	0000h	
08h	PADS_L		Read/write	Byte	00h	
09h	PADS_H		Read/write	Byte	00h	
0Ah	PASEL	Port A Port Select	Read/write	Word	0000h	
0Ah	PASEL_L		Read/write	Byte	00h	
0Bh	PASEL_H		Read/write	Byte	00h	
18h	PAIES	Port A Interrupt Edge Select	Read/write	Word	undefined	
18h	PAIES_L		Read/write	Byte	undefined	
19h	PAIES_H		Read/write	Byte	undefined	
1Ah	PAIE	Port A Interrupt Enable	Read/write	Word	0000h	
1Ah	PAIE_L		Read/write	Byte	00h	
1Bh	PAIE_H		Read/write	Byte	00h	
1Ch	PAIFG	Port A Interrupt Flag	Read/write	Word	0000h	
1Ch	PAIFG_L		Read/write	Byte	00h	
1Dh	PAIFG_H		Read/write	Byte	00h	
00h	PBIN	Port B Input	Read only	Word		
00h	PBIN_L		Read only	Byte		
01h	PBIN_H		Read only	Byte		
02h	PBOUT	Port B Output	Read/write	Word	undefined	
02h	PBOUT_L		Read/write	Byte	undefined	
03h	PBOUT_H		Read/write	Byte	undefined	
04h	PBDIR	Port B Direction	Read/write	Word	0000h	
04h	PBDIR_L		Read/write	Byte	00h	
05h	PBDIR_H		Read/write	Byte	00h	
06h	PBREN	Port B Resistor Enable	Read/write	Word	0000h	
06h	PBREN_L		Read/write	Byte	00h	
07h	PBREN_H		Read/write	Byte	00h	
08h	PBDS	Port B Drive Strength	Read/write	Word	0000h	
08h	PBDS_L		Read/write	Byte	00h	
09h	PBDS_H		Read/write	Byte	00h	
0Ah	PBSEL	Port B Port Select	Read/write	Word	0000h	
0Ah	PBSEL_L		Read/write	Byte	00h	
0Bh	PBSEL_H		Read/write	Byte	00h	
00h	PCIN	Port C Input	Read only	Word		
00h	PCIN_L		Read only	Byte		
01h	PCIN_H		Read only	Byte		
02h	PCOUT	Port C Output	Read/write	Word	undefined	
02h	PCOUT_L		Read/write	Byte	undefined	
03h	PCOUT_H		Read/write	Byte	undefined	
04h	PCDIR	Port C Direction	Read/write	Word	0000h	
04h	PCDIR_L		Read/write	Byte	00h	
05h	PCDIR_H		Read/write	Byte	00h	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 8-2. Digital I/O Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
06h	PCREN	Port C Resistor Enable	Read/write	Word	0000h	
06h	PCREN_L		Read/write	Byte	00h	
07h	PCREN_H		Read/write	Byte	00h	
08h	PCDS	Port C Drive Strength	Read/write	Word	0000h	
08h	PCDS_L		Read/write	Byte	00h	
09h	PCDS_H		Read/write	Byte	00h	
0Ah	PCSEL	Port C Port Select	Read/write	Word	0000h	
0Ah	PCSEL_L		Read/write	Byte	00h	
0Bh	PCSEL_H		Read/write	Byte	00h	
00h	PDIN	Port D Input	Read only	Word		
00h	PDIN_L		Read only	Byte		
01h	PDIN_H		Read only	Byte		
02h	PDOUT	Port D Output	Read/write	Word	undefined	
02h	PDOUT_L		Read/write	Byte	undefined	
03h	PDOUT_H		Read/write	Byte	undefined	
04h	PDDIR	Port D Direction	Read/write	Word	0000h	
04h	PDDIR_L		Read/write	Byte	00h	
05h	PDDIR_H		Read/write	Byte	00h	
06h	PDREN	Port D Resistor Enable	Read/write	Word	0000h	
06h	PDREN_L		Read/write	Byte	00h	
07h	PDREN_H		Read/write	Byte	00h	
08h	PDDS	Port D Drive Strength	Read/write	Word	0000h	
08h	PDDS_L		Read/write	Byte	00h	
09h	PDDS_H		Read/write	Byte	00h	
0Ah	PDSEL	Port D Port Select	Read/write	Word	0000h	
0Ah	PDSEL_L		Read/write	Byte	00h	
0Bh	PDSEL_H		Read/write	Byte	00h	
00h	PEIN	Port E Input	Read only	Word		
00h	PEIN_L		Read only	Byte		
01h	PEIN_H		Read only	Byte		
02h	PEOUT	Port E Output	Read/write	Word	undefined	
02h	PEOUT_L		Read/write	Byte	undefined	
03h	PEOUT_H		Read/write	Byte	undefined	
04h	PEDIR	Port E Direction	Read/write	Word	0000h	
04h	PEDIR_L		Read/write	Byte	00h	
05h	PEDIR_H		Read/write	Byte	00h	
06h	PEREN	Port E Resistor Enable	Read/write	Word	0000h	
06h	PEREN_L		Read/write	Byte	00h	
07h	PEREN_H		Read/write	Byte	00h	
08h	PEDS	Port E Drive Strength	Read/write	Word	0000h	
08h	PEDS_L		Read/write	Byte	00h	
09h	PEDS_H		Read/write	Byte	00h	
0Ah	PESEL	Port E Port Select	Read/write	Word	0000h	
0Ah	PESEL_L		Read/write	Byte	00h	
0Bh	PESEL_H		Read/write	Byte	00h	
00h	PFIN	Port F Input	Read only	Word		
00h	PFIN_L		Read only	Byte		

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 8-2. Digital I/O Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
01h	PFIN_H		Read only	Byte		
02h	PFOUT	Port F Output	Read/write	Word	undefined	
02h	PFOUT_L		Read/write	Byte	undefined	
03h	PFOUT_H		Read/write	Byte	undefined	
04h	PFDIR	Port F Direction	Read/write	Word	0000h	
04h	PFDIR_L		Read/write	Byte	00h	
05h	PFDIR_H		Read/write	Byte	00h	
06h	PFREN	Port F Resistor Enable	Read/write	Word	0000h	
06h	PFREN_L		Read/write	Byte	00h	
07h	PFREN_H		Read/write	Byte	00h	
08h	PFDS	Port F Drive Strength	Read/write	Word	0000h	
08h	PFDS_L		Read/write	Byte	00h	
09h	PFDS_H		Read/write	Byte	00h	
0Ah	PFSEL	Port F Port Select	Read/write	Word	0000h	
0Ah	PFSEL_L		Read/write	Byte	00h	
0Bh	PFSEL_H		Read/write	Byte	00h	
00h	PJIN	Port J Input	Read only	Word		
00h	PJIN_L		Read only	Byte		
01h	PJIN_H		Read only	Byte		
02h	PJOUT	Port J Output	Read/write	Word	undefined	
02h	PJOUT_L		Read/write	Byte	undefined	
03h	PJOUT_H		Read/write	Byte	undefined	
04h	PJDIR	Port J Direction	Read/write	Word	0000h	
04h	PJDIR_L		Read/write	Byte	00h	
05h	PJDIR_H		Read/write	Byte	00h	
06h	PJREN	Port J Resistor Enable	Read/write	Word	0000h	
06h	PJREN_L		Read/write	Byte	00h	
07h	PJREN_H		Read/write	Byte	00h	
08h	PJDS	Port J Drive Strength	Read/write	Word	0000h	
08h	PJDS_L		Read/write	Byte	00h	
09h	PJDS_H		Read/write	Byte	00h	

Copyright © 2009–2013, Texas Instruments Incorporated

8.4.1 P1IV Register

Port 1 Interrupt Vector Register

Figure 8-1. P1IV Register

15	14	13	12	11	10	9	8
P1IV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
P1IV							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 8-3. P1IV Register Description

Bit	Field	Type	Reset	Description
15-0	P1IV	R	0h	Port 1 interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Port 1.0 interrupt; Interrupt Flag: P1IFG.0; Interrupt Priority: Highest 04h = Interrupt Source: Port 1.1 interrupt; Interrupt Flag: P1IFG.1 06h = Interrupt Source: Port 1.2 interrupt; Interrupt Flag: P1IFG.2 08h = Interrupt Source: Port 1.3 interrupt; Interrupt Flag: P1IFG.3 0Ah = Interrupt Source: Port 1.4 interrupt; Interrupt Flag: P1IFG.4 0Ch = Interrupt Source: Port 1.5 interrupt; Interrupt Flag: P1IFG.5 0Eh = Interrupt Source: Port 1.6 interrupt; Interrupt Flag: P1IFG.6 10h = Interrupt Source: Port 1.7 interrupt; Interrupt Flag: P1IFG.7; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4.2 P2IV Register

Port 2 Interrupt Vector Register

Figure 8-2. P2IV Register

15	14	13	12	11	10	9	8
P2IV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
P2IV							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 8-4. P2IV Register Description

Bit	Field	Type	Reset	Description
15-0	P2IV	R	0h	Port 2 interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Port 2.0 interrupt; Interrupt Flag: P2IFG.0; Interrupt Priority: Highest 04h = Interrupt Source: Port 2.1 interrupt; Interrupt Flag: P2IFG.1 06h = Interrupt Source: Port 2.2 interrupt; Interrupt Flag: P2IFG.2 08h = Interrupt Source: Port 2.3 interrupt; Interrupt Flag: P2IFG.3 0Ah = Interrupt Source: Port 2.4 interrupt; Interrupt Flag: P2IFG.4 0Ch = Interrupt Source: Port 2.5 interrupt; Interrupt Flag: P2IFG.5 0Eh = Interrupt Source: Port 2.6 interrupt; Interrupt Flag: P2IFG.6 10h = Interrupt Source: Port 2.7 interrupt; Interrupt Flag: P2IFG.7; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4.3 P1IES Register

Port 1 Interrupt Edge Select Register

Figure 8-3. P1IES Register

7	6	5	4	3	2	1	0
P1IES							
rw	rw	rw	rw	rw	rw	rw	rw

Table 8-5. P1IES Register Description

Bit	Field	Type	Reset	Description
7-0	P1IES	RW	undefined	Port 1 interrupt edge select 0b = P1IFG flag is set with a low-to-high transition. 1b = P1IFG flag is set with a high-to-low transition.

8.4.4 P1IE Register

Port 1 Interrupt Enable Register

Figure 8-4. P1IE Register

7	6	5	4	3	2	1	0
P1IE							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-6. P1IE Register Description

Bit	Field	Type	Reset	Description
7-0	P1IE	RW	0h	Port 1 interrupt enable 0b = Corresponding port interrupt disabled 1b = Corresponding port interrupt enabled

8.4.5 P1IFG Register

Port 1 Interrupt Flag Register

Figure 8-5. P1IFG Register

7	6	5	4	3	2	1	0
P1IFG							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-7. P1IFG Register Description

Bit	Field	Type	Reset	Description
7-0	P1IFG	RW	0h	Port 1 interrupt flag 0b = No interrupt is pending 1b = Interrupt is pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4.6 P2IES Register

Port 2 Interrupt Edge Select Register

Figure 8-6. P2IES Register

7	6	5	4	3	2	1	0
P2IES							
rw	rw	rw	rw	rw	rw	rw	rw

Table 8-8. P2IES Register Description

Bit	Field	Type	Reset	Description
7-0	P2IES	RW	undefined	Port 2 interrupt edge select 0b = P2IFG flag is set with a low-to-high transition. 1b = P2IFG flag is set with a high-to-low transition.

8.4.7 P2IE Register

Port 2 Interrupt Enable Register

Figure 8-7. P2IE Register

7	6	5	4	3	2	1	0
P2IE							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-9. P2IE Register Description

Bit	Field	Type	Reset	Description
7-0	P2IE	RW	0h	Port 2 interrupt enable 0b = Corresponding port interrupt disabled 1b = Corresponding port interrupt enabled

8.4.8 P2IFG Register

Port 2 Interrupt Flag Register

Figure 8-8. P2IFG Register

7	6	5	4	3	2	1	0
P2IFG							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-10. P2IFG Register Description

Bit	Field	Type	Reset	Description
7-0	P2IFG	RW	0h	Port 2 interrupt flag 0b = No interrupt is pending 1b = Interrupt is pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4.9 PxIN Register

Port x Input Register

Figure 8-9. PxIN Register

7	6	5	4	3	2	1	0
PxIN							
r	r	r	r	r	r	r	r

Table 8-11. PxIN Register Description

Bit	Field	Type	Reset	Description
7-0	PxIN	R	undefined	Port x input. Read only.

8.4.10 PxOUT Register

Port x Output Register

Figure 8-10. PxOUT Register

7	6	5	4	3	2	1	0
PxOUT							
rw	rw	rw	rw	rw	rw	rw	rw

Table 8-12. PxOUT Register Description

Bit	Field	Type	Reset	Description
7-0	PxOUT	RW	undefined	Port x output When I/O configured to output mode: 0b = Output is low 1b = Output is high When I/O configured to input mode and pullups/pulldowns enabled: 0b = Pulldown selected 1b = Pullup selected

8.4.11 PxDIR Register

Port x Direction Register

Figure 8-11. PxDIR Register

7	6	5	4	3	2	1	0
PxDIR							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-13. PxDIR Register Description

Bit	Field	Type	Reset	Description
7-0	PxDIR	RW	0h	Port x direction 0b = Port configured as input 1b = Port configured as output

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

8.4.12 PxREN Register

Port x Pullup/Pulldown Resistor Enable Registers

Figure 8-12. PxREN Register

7	6	5	4	3	2	1	0
PxREN							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-14. PxREN Register Description

Bit	Field	Type	Reset	Description
7-0	PxREN	RW	0h	Port x pullup/pulldown resistor enable. When respective port is configured as input, setting this bit will enable the pullup or pulldown. See Table 8-1 0b = Pullup or pulldown disabled. 1b = Pullup or pulldown enabled.

8.4.13 PxDS Register

Port x Drive Strength Register

Figure 8-13. PxDS Register

7	6	5	4	3	2	1	0
PxDS							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-15. PxDS Register Description

Bit	Field	Type	Reset	Description
7-0	PxDS	RW	0h	Port x drive strength 0b = Reduced output drive strength 1b = Full output drive strength

8.4.14 PxSEL Register

REGISTERDESCRIPTION

Figure 8-14. PxSEL Register

7	6	5	4	3	2	1	0
PxSEL							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 8-16. PxSEL Register Description

Bit	Field	Type	Reset	Description
7-0	PxSEL	RW	0h	Port x function selection 0b = I/O function is selected 1b = Peripheral module function is selected

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Port Mapping Controller

The port mapping controller allows a flexible mapping of digital functions to port pins. This chapter describes the port mapping controller.

Topic **Page**

9.1 Port Mapping Controller Introduction 363
9.2 Port Mapping Controller Operation 363
9.3 Port Mapping Controller Registers 365

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

9.1 Port Mapping Controller Introduction

The port mapping controller allows the flexible and reconfigurable mapping of digital functions to port pins.

The port mapping controller features are:

- Configuration protected by write access key.
- Default mapping provided for each port pin (device-dependent, the device pinout in the device-specific data sheet).
- Mapping can be reconfigured during runtime.
- Each output signal can be mapped to several output pins.

9.2 Port Mapping Controller Operation

The port mapping is configured with user software. The setup is discussed in the following sections.

9.2.1 Access

To enable write access to any of the port mapping controller registers, the correct key must be written into the PMAPKEYID register. The PMAPKEYID register always reads 096A5h. Writing the key 02D52h grants write access to all port mapping controller registers. Read access is always possible.

If an invalid key is written while write access is granted, any further write accesses are prevented. It is recommended that the application completes mapping configuration by writing an invalid key.

There is a timeout counter implemented that is incremented with each (assembler) instruction, and when it counts to 32, the write access is locked again. Any access to the port mapping controller registers resets the counter. Interrupts should be disabled during the configuration process or the application should take precautions that the execution of an interrupt service routine does not accidentally cause a permanent lock of the port mapping registers; for example, by using the reconfiguration capability (see Section 9.2.2).

The access status is reflected in the PMAPLOCK bit.

By default, the port mapping controller allows only one configuration after PUC. A second attempt to enable write access by writing the correct key is ignored, and the registers remain locked. A PUC is required to disable the permanent lock again. If it is necessary to reconfigure the mapping during runtime, the PMAPRECFG bit must be set during the first write access timeslot. If PMAPRECFG is cleared during later configuration sessions, no more configuration sessions are possible.

9.2.2 Mapping

For each port pin, Px.y, on ports providing the mapping functionality, a mapping register, PxMAPy, is available. Setting this register to a certain value maps a module's input and output signals to the respective port pin Px.y. The port pin itself is switched from a general purpose I/O to the selected peripheral/secondary function by setting the corresponding PxSEL.y bit to 1. If the input or the output function of the module is used, it is typically defined by the setting the PxDIR.y bit. If PxDIR.y = 0, the pin is an input, if PxDIR.y = 1, the pin is an output. There are also peripherals (for example, the USCI module) that control the direction or even other functions of the pin (for example, open drain), and these options are documented in the mapping table.

With the port mapping functionality the output of a module can be mapped to multiple pins. Also the input of a module can receive inputs from multiple pins. When mapping multiple inputs onto one function, care needs to be taken because the input signals are logically ORed together without applying any priority; therefore, a logic one on any of the inputs results in a logic one at the module. If the PxSEL.y bit is 0, the corresponding input signal is a logic zero.

The mapping is device-dependent; see the device-specific data sheet for available functions and specific values. The use of mapping mnemonics to abstract the underlying PxMAPy values is recommended to allow simple portability between different devices. Table 9-1 shows some examples for mapping mnemonics of some common peripherals.

All mappable port pins provide the function PM_ANALOG (0FFh). Setting the port mapping register PxMAPy to PM_ANALOG together with PxSEL.y = 1 disables the output driver and the input Schmitt-trigger, to prevent parasitic cross currents when applying analog signals.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 9-1. Examples for Port Mapping Mnemonics and Functions

PxMAPy Mnemonic	Input Pin Function With PxSEL.y = 1 and PxDIR.y = 0	Output Pin Function With PxSEL.y = 1 and PxDIR.y = 1
PM_NONE	None	DVSS
PM_ACLK	None	ACLK
PM_MCLK	None	MCLK
PM_SMCLK	None	SMCLK
PM_TA0CLK	Timer_A0 clock input	DVSS
PM_TA0CCR0A	Timer_A0 CCR0 capture input CCI0A	TA0 CCR0 compare output Out0
PM_TA0CCR1A	Timer_A0 CCR1 capture input CCI1A	TA0 CCR1 compare output Out1
PM_TA0CCR2A	Timer_A0 CCR2 capture input CCI2A	TA0 CCR2 compare output Out2
PM_TA0CCR3A	Timer_A0 CCR3 capture input CCI3A	TA0 CCR3 compare output Out3
PM_TA0CCR4A	Timer_A0 CCR4 capture input CCI4A	TA0 CCR4 compare output Out4
PM_TA1CLK	Timer_A1 clock input	DVSS
PM_TA1CCR0A	Timer_A1 CCR0 capture input CCI0A	TA1 CCR0 compare output Out0
PM_TA1CCR1A	Timer_A1 CCR1 capture input CCI1A	TA1 CCR1 compare output Out1
PM_TA1CCR2A	Timer_A1 CCR2 capture input CCI2A	TA1 CCR2 compare output Out2
PM_TBCLK	Timer_B clock input	DVSS
PM_TBOUTH	Timer_B outputs high impedance	DVSS
PM_TBCCR0A	Timer_B CCR0 capture input CCI0A	TB CCR0 compare output Out0 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR1A	Timer_B CCR1 capture input CCI1A	TB CCR1 compare output Out1 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR2A	Timer_B CCR2 capture input CCI2A	TB CCR2 compare output Out2 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR3A	Timer_B CCR3 capture input CCI3A	TB CCR3 compare output Out3 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR4A	Timer_B CCR4 capture input CCI4A	TB CCR4 compare output Out4 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR5A	Timer_B CCR5 capture input CCI3A	TB CCR5 compare output Out5 [direction controlled by Timer_B (TBOUTH)]
PM_TBCCR6A	Timer_B CCR6 capture input CCI4A	TB CCR6 compare output Out6 [direction controlled by Timer_B (TBOUTH)]
PM_UCA0RXD	USCI_A0 UART RXD (direction controlled by USCI - input)	
PM_UCA0SOMI	USCI_A0 SPI slave out master in (direction controlled by USCI)	
PM_UCA0TXD	USCI_A0 UART TXD (direction controlled by USCI - output)	
PM_UCA0SIMO	USCI_A0 SPI slave in master out (direction controlled by USCI)	
PM_UCA0CLK	USCI_A0 clock input/output (direction controlled by USCI)	
PM_UCA0STE	USCI_A0 SPI slave transmit enable (direction controlled by USCI)	
PM_UCB0SOMI	USCI_B0 SPI slave out master in (direction controlled by USCI)	
PM_UCB0SCL	USCI_B0 I2C clock (open drain and direction controlled by USCI	
PM_UCB0SIMO	USCI_B0 SPI slave in master out (direction controlled by USCI)	
PM_UCB0SDA	USCI_B0 I2C data (open drain and direction controlled by USCI)	
PM_UCB0CLK	USCI_B0 clock input/output (direction controlled by USCI)	
PM_UCB0STE	USCI_B0 SPI slave transmit enable (direction controlled by USCI)	
PM_ANALOG	Disables the output driver and the input Schmitt-trigger to prevent parasitic cross currents when applying analog signals	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

9.3 Port Mapping Controller Registers

The control register for the port mapping controller are listed in Table 9-2. The mapping registers are listed in Table 9-3. The mapping registers can also be accessed as words, as shown in Table 9-4.

Table 9-2. Port Mapping Control Registers

Offset	Acronym	Register Name	Type	Reset
00h	PMAPKEYID	Port mapping key register	Read/write	Reset with PUC
02h	PMAPCTL	Port mapping control register	Read/write	Reset with PUC

Table 9-3. Port Mapping Registers for Port Px – Byte Access

Offset	Acronym	Register Name	Type	Reset
00h	PxMAP0	Port Px.0 mapping register	Read/write	Device dependent
01h	PxMAP1	Port Px.1 mapping register	Read/write	Device dependent
02h	PxMAP2	Port Px.2 mapping register	Read/write	Device dependent
03h	PxMAP3	Port Px.3 mapping register	Read/write	Device dependent
04h	PxMAP4	Port Px.4 mapping register	Read/write	Device dependent
05h	PxMAP5	Port Px.5 mapping register	Read/write	Device dependent
06h	PxMAP6	Port Px.6 mapping register	Read/write	Device dependent
07h	PxMAP7	Port Px.7 mapping register	Read/write	Device dependent

Table 9-4. Port Mapping Registers for Port Px – Word Access

Offset	Acronym	Register Name	Type	Reset
00h	PxMAP01	Port Px.0/Port Px.1 mapping register	Read/write	Device dependent
02h	PxMAP23	Port Px.2/Port Px.3 mapping register	Read/write	Device dependent
04h	PxMAP45	Port Px.4/Port Px.5 mapping register	Read/write	Device dependent
06h	PxMAP67	Port Px.6/Port Px.7 mapping register	Read/write	Device dependent

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

9.3.1 PMAPKEYID Register

Port Mapping Key Register

Figure 9-1. PMAPKEYID Register

15	14	13	12	11	10	9	8
PMAPKEYx							
7	6	5	4	3	2	1	0
PMAPKEYx							

Table 9-5. PMAPKEYID Register Description

Bit	Field	Type	Reset	Description
15-0	PMAPKEYx	RW	96A5h	Port write access key. Always reads 096A5h. Must be written 02D52h for write access to the port mapping registers.

9.3.2 PMAPCTL Register

Port Mapping Control Register

Figure 9-2. PMAPCTL Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved						PMAPRECFG	PMAPLOCKED
r0	r0	r0	r0	r0	r0	rw-0	r-1

Table 9-6. PMAPCTL Register Description

Bit	Field	Type	Reset	Description
15-2	Reserved	R	0h	Reserved. Always reads as 0.
1	PMAPRECFG	RW	0h	Port mapping reconfiguration control bit 0b = Configuration allowed only once 1b = Allow reconfiguration of port mapping
0	PMAPLOCKED	R	1h	Port mapping lock bit. Read only 0b = Access to mapping registers is granted 1b = Access to mapping registers is locked

9.3.3 PxMAPy Register

Port Px.y Mapping Register

Figure 9-3. PxMAPy Register

7	6	5	4	3	2	1	0
PMAPx							
rw-0(1)	rw-0(1)	rw-0(1)	rw-0(1)	rw-0(1)	rw-0(1)	rw-0(1)	rw-0(1)

(1) If not all bits are required to decode all provided functions, the unused bits are r0.

Table 9-7. PxMAPy Register Description

Bit	Field	Type	Reset	Description
7-0	PMAPx	RW	0h	Selects secondary port function. Settings are device-dependent; see the device-specific data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Cyclic Redundancy Check (CRC) Module

The cyclic redundancy check (CRC) module provides a signature for a given data sequence. This chapter describes the operation and use of the CRC module.

NOTE: The CRC module on the CC430F613x, CC430F612x, and CC430F513x devices does not support the bit-wise reverse feature described in this module description. Registers CRCDIRB and CRCRESR, along with their respective functionality, are not available.

Topic **Page**

10.1 Cyclic Redundancy Check (CRC) Module Introduction 368
10.2 CRC Standard and Bit Order 368
10.3 CRC Checksum Generation 369
10.4 CRC Registers 372

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.1 Cyclic Redundancy Check (CRC) Module Introduction

The CRC module produces a signature for a given sequence of data values. The signature is generated through a feedback path from data bits 0, 4, 11, and 15 (see Figure 10-1). The CRC signature is based on the polynomial given in the CRC-CCITT-BR polynomial (see Equation 10) .

$$f(x) = x^{16} + x^{12} + x^{5} + 1 \quad (10)$$

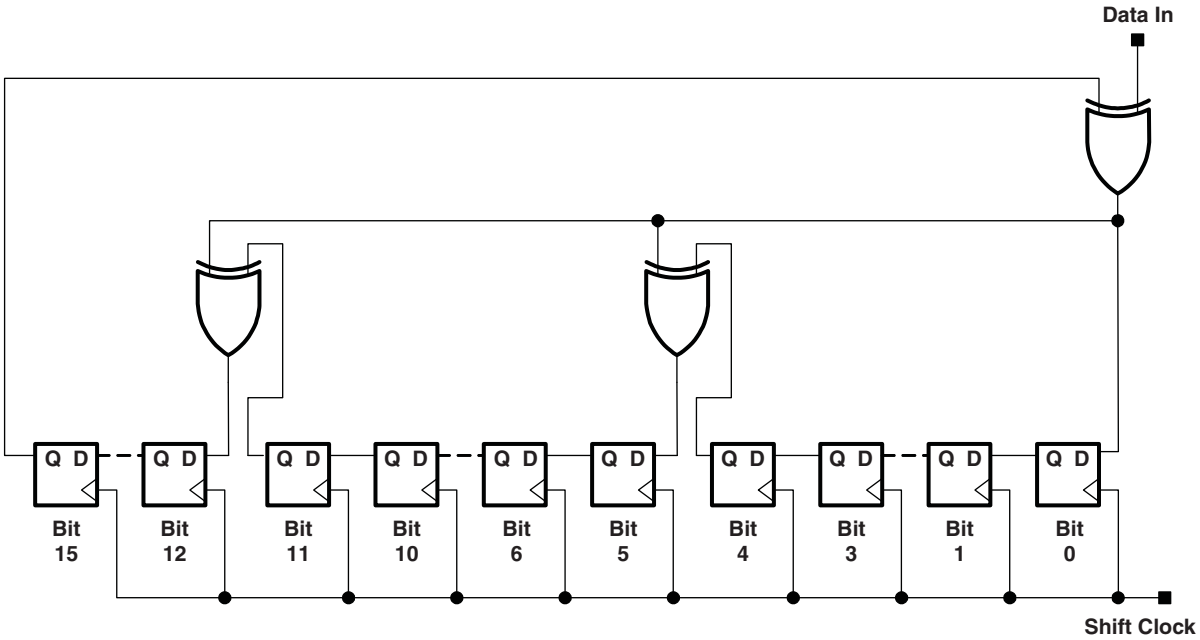


Figure 10-1. LFSR Implementation of CRC-CCITT Standard, Bit 0 is the MSB of the Result

Identical input data sequences result in identical signatures when the CRC is initialized with a fixed seed value, whereas different sequences of input data, in general, result in different signatures.

10.2 CRC Standard and Bit Order

The definitions of the various CRC standards were done in the era of main frame computers, and by convention bit 0 was treated as the MSB. Today, as in most microcontrollers such as the MSP430, bit 0 normally denotes the LSB. In Figure 10-1, the bit convention shown is as given in the original standards i.e. bit 0 is the MSB. The fact that bit 0 is treated for some as LSB, and for others as MSB, continues to cause confusion. The CRC16 module therefore provides a bit reversed register pair for CRC16 operations to support both conventions.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.3 CRC Checksum Generation

The CRC generator is first initialized by writing a 16-bit word (seed) to the CRC Initialization and Result (CRCINIRES) register. Any data that should be included into the CRC calculation must be written to the CRC Data Input (CRCDI or CRCDIRB) register in the same order that the original CRC signature was calculated. The actual signature can be read from the CRCINIRES register to compare the computed checksum with the expected checksum.

Signature generation describes a method on how the result of a signature operation can be calculated. The calculated signature, which is computed by an external tool, is called checksum in the following text. The checksum is stored in the product's memory and is used to check the correctness of the CRC operation result.

10.3.1 CRC Implementation

To allow parallel processing of the CRC, the linear feedback shift register (LFSR) functionality is implemented with an XOR tree. This implementation shows the identical behavior as the LFSR approach after 8 bits of data are shifted in when the LSB is 'shifted' in first. The generation of a signature calculation has to be started by writing a seed to the CRCINIRES register to initialize the register. Software or hardware (for example, DMA) can transfer data to the CRCDI or CRCDIRB register (for example, from memory). The value in CRCDI or CRCDIRB is then included into the signature, and the result is available in the signature result registers at the next read access (CRCINIRES and CRCRESR). The signature can be generated using word or byte data.

If a word data is processed, the lower byte at the even address is used at the first clock (MCLK) cycle. During the second clock cycle, the higher byte is processed. Thus, it takes two clock cycles to process word data, while it takes only one clock (MCLK) cycle to process byte data.

Data bytes written to CRCDIRB in word mode or the data byte in byte mode are bit-wise reversed before the CRC engine adds them to the signature. The bits among each byte are reversed. Data bytes written to CRCDI in word mode or the data byte in byte mode are not bit reversed before use by the CRC engine.

If the Check Sum itself (with reversed bit order) is included into the CRC operation (as data written to CRCDI or CRCDIRB), the result in the CRCINIRES and CRCRESR registers must be zero.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

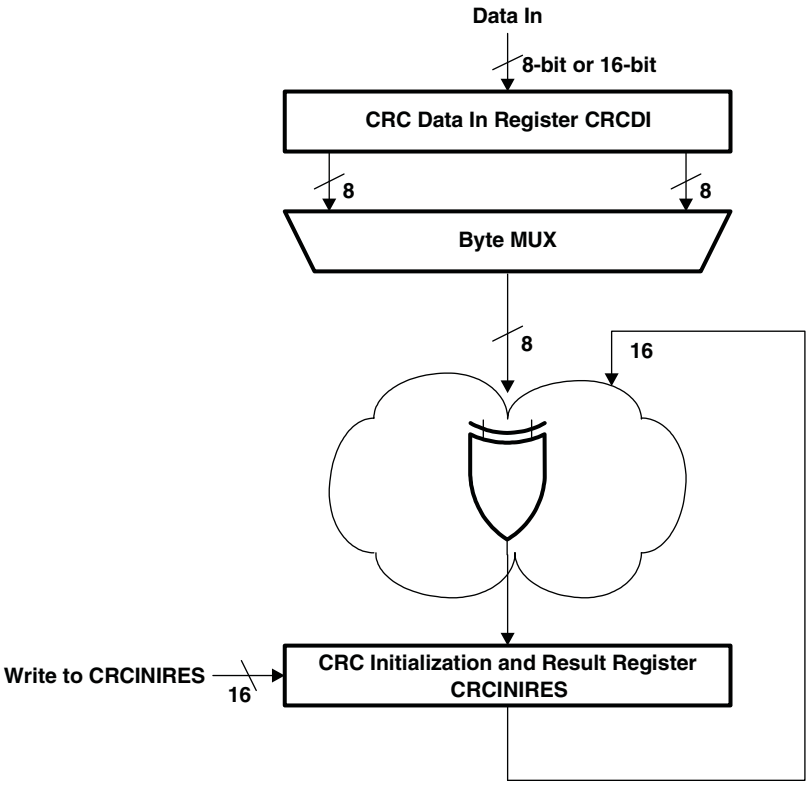


Figure 10-2. Implementation of CRC-CCITT Using the CRCDI and CRCINIRES Registers

10.3.2 Assembler Examples

10.3.2.1 General Assembler Example

This example demonstrates the operation of the on-chip CRC:

```
   ...
   PUSH   R4                ; Save registers
   PUSH   R5
   MOV    #StartAddress,R4  ; StartAddress < EndAddress
   MOV    #EndAddress,R5
   MOV    &INIT, &CRCINIRES ; INIT to CRCINIRES
L1 MOV    @R4+,&CRCDI       ; Item to Data In register
   CMP    R5,R4             ; End address reached?
   JLO    L1                ; No
   MOV    &Check_Sum,&CRCDI ; Yes, Include checksum
   TST    &CRCINIRES        ; Result = 0?
   JNZ    CRC_ERROR         ; No, CRCRES <> 0: error
   ...                      ; Yes, CRCRES=0:
                            ; information ok.
   POP    R5                ; Restore registers
   POP    R4
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.3.2.2 Reference Data Sequence

The details of the implemented CRC algorithm is shown by the following data sequences using word or byte accesses and the CRC data-in as well as the CRC data-in reverse byte registers:

```
   ...
mov     #0FFFFh,&CRCINIRES  ; initialize CRC
mov.b   #00031h,&CRCDI_L    ; "1"
mov.b   #00032h,&CRCDI_L    ; "2"
mov.b   #00033h,&CRCDI_L    ; "3"
mov.b   #00034h,&CRCDI_L    ; "4"
mov.b   #00035h,&CRCDI_L    ; "5"
mov.b   #00036h,&CRCDI_L    ; "6"
mov.b   #00037h,&CRCDI_L    ; "7"
mov.b   #00038h,&CRCDI_L    ; "8"
mov.b   #00039h,&CRCDI_L    ; "9"

cmp     #089F6h,&CRCINIRES  ; compare result
                            ; CRCRESR contains 06F91h
jeq     &Success            ; no error
br      &Error              ; to error handler
```

```
mov     #0FFFFh,&CRCINIRES  ; initialize CRC
mov.w   #03231h,&CRCDI      ; "1" & "2"
mov.w   #03433h,&CRCDI      ; "3" & "4"
mov.w   #03635h,&CRCDI      ; "5" & "6"
mov.w   #03837h,&CRCDI      ; "7" & "8"
mov.b   #039h,  &CRCDI_L    ; "9"

cmp     #089F6h,&CRCINIRES  ; compare result
                              ; CRCRESR contains 06F91h
jeq     &Success            ; no error
br      &Error              ; to error handler
```

```
   ...
mov     #0FFFFh,&CRCINIRES  ; initialize CRC
mov.b   #00031h,&CRCDIRB_L  ; "1"
mov.b   #00032h,&CRCDIRB_L  ; "2"
mov.b   #00033h,&CRCDIRB_L  ; "3"
mov.b   #00034h,&CRCDIRB_L  ; "4"
mov.b   #00035h,&CRCDIRB_L  ; "5"
mov.b   #00036h,&CRCDIRB_L  ; "6"
mov.b   #00037h,&CRCDIRB_L  ; "7"
mov.b   #00038h,&CRCDIRB_L  ; "8"
mov.b   #00039h,&CRCDIRB_L  ; "9"

cmp     #029B1h,&CRCINIRES  ; compare result
                            ; CRCRESR contains 08D94h
jeq     &Success            ; no error
br      &Error              ; to error handler
```

```
...
mov     #0FFFFh,&CRCINIRES  ; initialize CRC
mov.w   #03231h,&CRCDIRB    ; "1" & "2"
mov.w   #03433h,&CRCDIRB    ; "3" & "4"
mov.w   #03635h,&CRCDIRB    ; "5" & "6"
mov.w   #03837h,&CRCDIRB    ; "7" & "8"
mov.b   #039h,  &CRCDIRB_L  ; "9"

cmp     #029B1h,&CRCINIRES  ; compare result
                            ; CRCRESR contains 08D94h
jeq     &Success            ; no error
br      &Error              ; to error handler
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.4 CRC Registers

The CRC module registers are listed in Table 10-1. The base address can be found in the device-specific data sheet. The address offset is given in Table 10-1.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 10-1. CRC Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
0000h	CRCDI	CRC Data In	Read/write	Word	0000h	Section 10.4.1
0000h	CRCDI_L		Read/write	Byte	00h	
0001h	CRCDI_H		Read/write	Byte	00h	
0002h	CRCDIRB	CRC Data In Reverse Byte(1)	Read/write	Word	0000h	Section 10.4.2
0002h	CRCDIRB_L		Read/write	Byte	00h	
0003h	CRCDIRB_H		Read/write	Byte	00h	
0004h	CRCINIRES	CRC Initialization and Result	Read/write	Word	FFFFh	Section 10.4.3
0004h	CRCINIRES_L		Read/write	Byte	FFh	
0005h	CRCINIRES_H		Read/write	Byte	FFh	
0006h	CRCRESR	CRC Result Reverse(1)	Read only	Word	FFFFh	Section 10.4.4
0006h	CRCRESR_L		Read/write	Byte	FFh	
0007h	CRCRESR_H		Read/write	Byte	FFh	

(1) Not available on MSP430F543x and MSP430F541x non-A versions.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.4.1 CRCDI Register

CRC Data In Register

Figure 10-3. CRCDI Register

15	14	13	12	11	10	9	8
CRCDI							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
CRCDI							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 10-2. CRCDI Register Description

Bit	Field	Type	Reset	Description
15-0	CRCDI	RW	0h	CRC data in. Data written to the CRCDI register is included to the present signature in the CRCINIRES register according to the CRC-CCITT standard.

10.4.2 CRCDIRB Register

CRC Data In Reverse Register

Figure 10-4. CRCDIRB Register

15	14	13	12	11	10	9	8
CRCDIRB							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
CRCDIRB							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 10-3. CRCDIRB Register Description

Bit	Field	Type	Reset	Description
15-0	CRCDIRB	RW	0h	CRC data in reverse byte. Data written to the CRCDIRB register is included to the present signature in the CRCINIRES and CRCRESR registers according to the CRC-CCITT standard. Reading the register returns the register CRCDI content.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

10.4.3 CRCINIRES Register

CRC Initialization and Result Register

Figure 10-5. CRCINIRES Register

15	14	13	12	11	10	9	8
CRCINIRES							
rw-1	rw-1	rw-1	rw-1	rw-1	rw-1	rw-1	rw-1
7	6	5	4	3	2	1	0
CRCINIRES							
rw-1	rw-1	rw-1	rw-1	rw-1	rw-1	rw-1	rw-1

Table 10-4. CRCINIRES Register Description

Bit	Field	Type	Reset	Description
15-0	CRCINIRES	RW	FFFFh	CRC initialization and result. This register holds the current CRC result (according to the CRC-CCITT standard). Writing to this register initializes the CRC calculation with the value written to it. The value just written can be read from CRCINIRES register.

10.4.4 CRCRESR Register

CRC Reverse Result Register

Figure 10-6. CRCRESR Register

15	14	13	12	11	10	9	8
CRCRESR							
r-1	r-1	r-1	r-1	r-1	r-1	r-1	r-1
7	6	5	4	3	2	1	0
CRCRESR							
r-1	r-1	r-1	r-1	r-1	r-1	r-1	r-1

Table 10-5. CRCRESR Register Description

Bit	Field	Type	Reset	Description
15-0	CRCRESR	R	FFFFh	CRC reverse result. This register holds the current CRC result (according to the CRC-CCITT standard). The order of bits is reversed (for example, CRCINIRES[15] = CRCRESR[0]) compared to the order of bits in the CRCINIRES register (see example code).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

AES Accelerator

The AES accelerator module performs AES128 encryption or decryption in hardware. This chapter describes the AES accelerator.

Topic **Page**

11.1 AES Accelerator Introduction 376
11.2 AES Accelerator Operation 377
11.3 AES_ACCEL Registers 382

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.1 AES Accelerator Introduction

The AES accelerator module performs encryption and decryption of 128-bit data with 128-bit keys according to the advanced encryption standard (AES) (FIPS PUB 197) in hardware.

The AES accelerator features are:

- Encryption and decryption according to AES FIPS PUB 197 with 128-bit key
- On-the-fly key expansion for encryption and decryption
- Off-line key generation for decryption
- Byte and word access to key, input, and output data
- AES ready interrupt flag

The AES accelerator block diagram is shown in Figure 11-1.

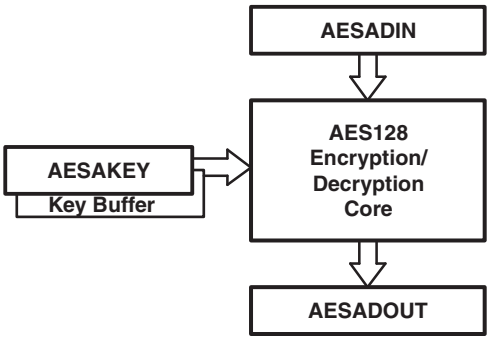


Figure 11-1. AES Accelerator Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.2 AES Accelerator Operation

The AES accelerator is configured with user software. The setup and operation is discussed in the following sections.

Internally, the AES algorithm's operations are performed on a two-dimensional array of bytes called the State. For AES-128, the State consists of four rows of bytes, each containing four bytes. The input is assigned to the State array as illustrated in Figure 11-2, with in[0] being the first data byte written into the AES accelerator data input register, AESADIN. The encrypt or decrypt operations are then conducted on the State array, after which its final values can be read from the output with out[0] being the first data byte read from the AES accelerator data output register, AESADOUT.

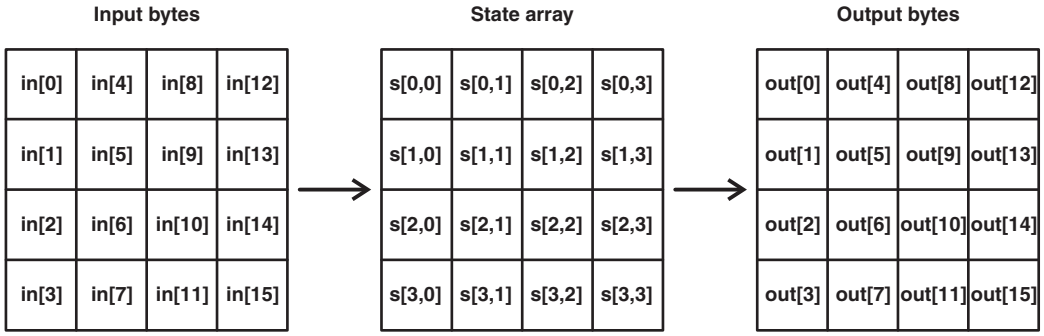


Figure 11-2. AES State Array Input and Output

The module allows word and byte access to all data registers, AESAKEY, AESADIN, and AESADOUT. Word and byte access should not be mixed while reading from or writing into one of the registers. However, it is possible to write one of the registers using byte access and another using word access.

NOTE: Access Restrictions

While the AES accelerator is busy (AESBUSY = 1), AESADOUT always reads as zero, the AESDOUTCNTx counter, the AESDOUTRD flag, and the AESDINWR flag are reset, any attempt to change AESOPx, AESDINWR, or AESKEYWR is ignored, and writing to AESAKEY or AESADIN aborts the current operation, the complete module is reset (except for AESRDYIE and AESOPx), and the AES error flag AESERRFG is set.

AESADIN and AESAKEY are write-only registers and always read as zero.

Writing data into AESADIN influences the content of the corresponding output data; for example, writing in[0] alters out[0], writing in[1] alters out[1], etc., but interleafed operation is possible; for example, first reading out[0], then writing in[0], and continuing with reading out[1], writing in[1], etc.

NOTE: When using a code debugger, the AES module does not stop its operation when program code is halted or single stepped.

Copyright © 2009–2013, Texas Instruments Incorporated

11.2.1 Encryption

Figure 11-3 shows the encryption process with the cipher being a series of transformations that converts the plaintext written into the AESADIN register to a ciphertext that can be read from the AESADOUT register using the cipher key provided via the AESAKEY register.

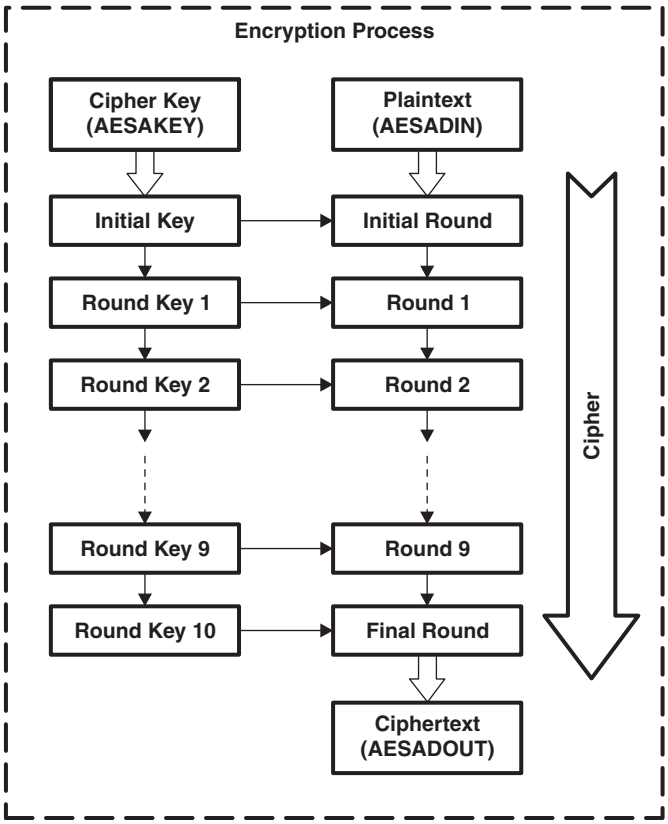


Figure 11-3. AES-128 Encryption Process

The steps to perform encryption are:

1. Set AESOPx = 00 to select encryption. Changing the AESOPx bits clears the AESKEYWR flag, and a new key must be loaded in the next step.
2. Load the 128-bit key into AESAKEY or set the AESKEYWR flag by software, if the key from a previous operation should be used. When all 16 bytes are written, the AESKEYWR flag indicates completion.

   If a key was loaded previously without changing AESOPx, the AESKEYWR flag is cleared with the first write access to AESAKEY. Loading the key mist be completed before the next step is performed.
3. Load 128-bit data into AESADIN, or set the AESDINWR flag by software if the output data from a previous operation should be encrypted. When all 16 bytes are written, the AESDINWR flag indicates completion. The module starts encrypting the presented data when AESDINWR = 1.
4. While the AES module is performing encryption, the AESBUSY bit is 1. The encryption takes 167 MCLK clock cycles. After its completion, the AESRDYIFG is set, and the result can be read from AESADOUT. When all 16 bytes are read, the AESDOUTRD flag indicates completion.

   The AESRDYIFG flag is cleared when reading AESADOUT or writing to AESAKEY or AESADIN.
5. If additional data should be encrypted with the same key loaded in step 2, new data can be written into AESADIN after the results of the operation on the previous data were read from AESADOUT. When an additional 16 data bytes are written, the module automatically starts the encryption using the key loaded in step 2.

   When using the output feedback (OFB) cipher block chaining mode, setting the AESDINWR flag is sufficient to trigger the next encryption, and the module starts the encryption automatically using the output data from the previous encryption as input data.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.2.2 Decryption

Figure 11-4 shows the decryption process with the inverse cipher being a series of transformations that convert the ciphertext written into the AESADIN register to a plaintext that can be read from the AESADOUT register using the cipher key provided via the AESAKEY register.

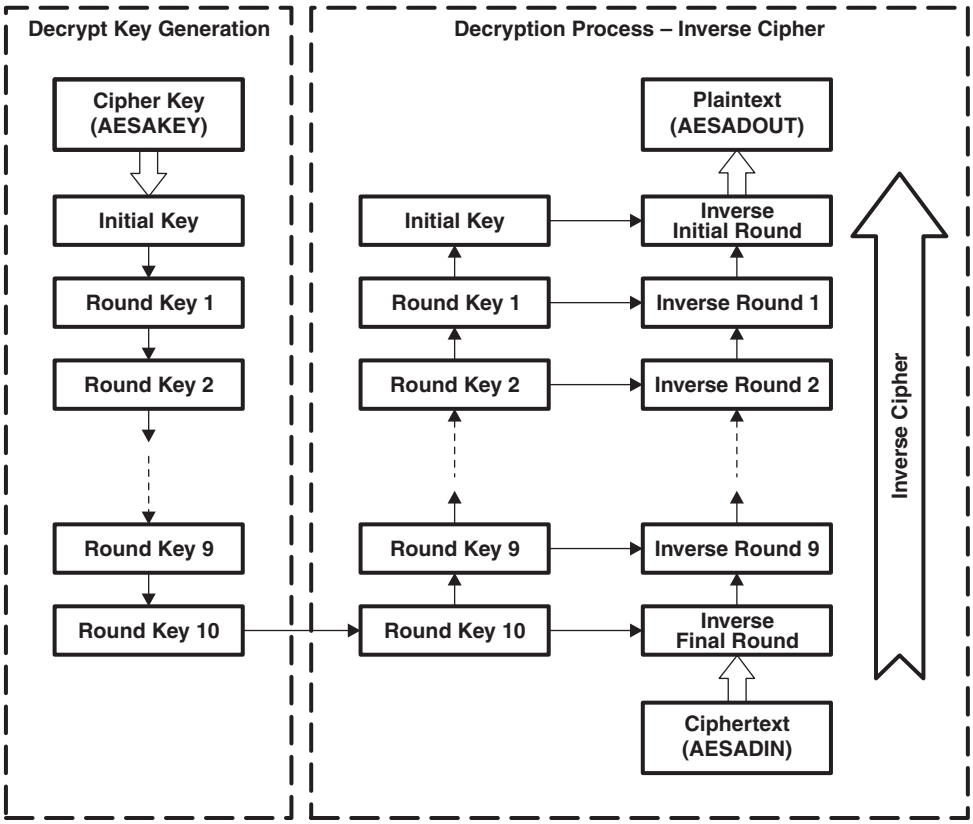


Figure 11-4. AES-128 Decryption Process using AESOPx = 01

The steps to perform decryption are:

1. Set AESOPx = 01 to select decryption using the same key used for encryption. Set AESOPx = 11 if the first-round key required for decryption (round key 10) is already generated and is loaded in step 2. Changing the AESOPx bits clears the AESKEYWR flag, and a new key must be loaded in step 2.
2. Load the 128-bit key into AESAKEY, or set the AESKEYWR flag by software, if the key from a previous operation should be used. When all 16 bytes are written, the AESKEYWR flag indicates completion.

   If a key was loaded previously without changing AESOPx, the AESKEYWR flag is cleared with the first write access to AESAKEY. Loading the key must be completed before the next step is performed.
3. Load 128-bit data into AESADIN or set the AESDINWR flag by software if the output data from a previous operation should be decrypted. When all 16 bytes are written, the AESDINWR flag indicates completion. The module starts decrypting the presented data as soon as AESDINWR = 1.
4. While the AES module is performing decryption, the AESBUSY bit is 1. The decryption takes 214 MCLK clock cycles with AESOPx = 01 and 167 MCLK clock cycles with AESOPx = 11. After its completion, the AESRDYIFG is set, and the result can be read from AESADOUT. When all 16 bytes are read the AESDOUTRD flag indicates completion.

   The AESRDYIFG flag is cleared when reading AESADOUT or writing to AESAKEY or AESADIN.
5. If additional data should be decrypted with the same key loaded in step 2, new data can be written into AESADIN after the results of the operation on the previous data were read from AESADOUT. When additional 16 data bytes are written, the module automatically starts the decryption using the key loaded in step 2.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.2.3 Decryption Key Generation

Figure 11-5 shows the decryption process with a pregenerated decryption key. In this case, the decryption key is calculated first with AESOPx = 10, then the precalculated key can be used together with the decryption operation AESOPx = 11.

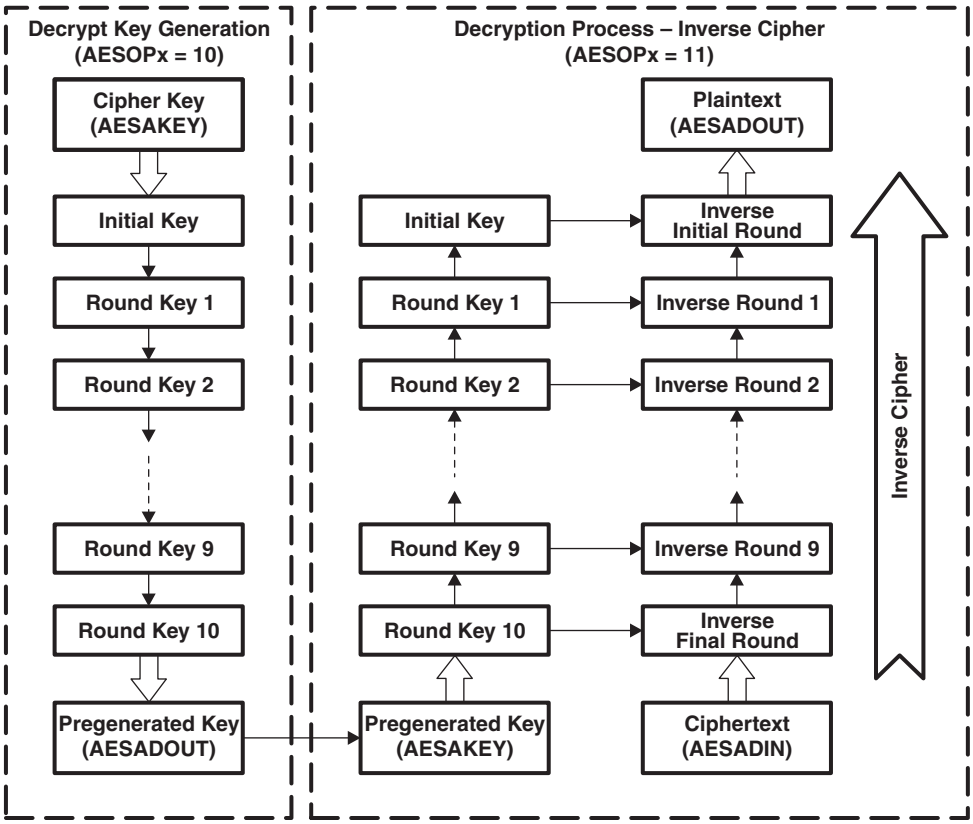


Figure 11-5. AES-128 Decryption Process using AESOPx = 10 and 11

To generate the decryption key independent from the actual decryption, the following steps are required:

1. Set AESOPx = 10 to select decryption key generation. Changing the AESOPx bits clears the AESKEYWR flag, and a new key must be loaded in step 2.
2. Load the 128-bit key into AESAKEY, or set the AESKEYWR flag by software if the key from a previous operation should be used. When all 16 bytes are written, the AESKEYWR flag indicates completion.

   The generation of the first round key required for decryption is started immediately.
3. While the AES module is performing the key generation, the AESBUSY bit is 1. It takes 52 CPU clock cycles to complete the key generation. After its completion, the AESRDYIFG is set, and the result can be read from AESADOUT. When all 16 bytes are read, the AESDOUTRD flag indicates completion.

   The AESRDYIFG flag is cleared when reading AESADOUT or writing to AESAKEY or AESADIN.
4. If data should be decrypted with the generated key, AESOPx must be set to 11. Then the generated key must be loaded or, if it was just generated with AESOPx = 10, it is sufficient to set the AESKEYWR flag by software to indicate that the key is already valid. Afterward, the steps described in Section 11.2.2 to load the data, etc., must be followed.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.2.4 Using the AES Accelerator With Low-Power Modes

The AES accelerator module provides automatic clock activation for MCLK for use with low-power modes. When the AES accelerator is busy, it automatically activates MCLK, regardless of the control-bit settings for the clock source. The clock remains active until the AES accelerator completes its operation.

11.2.5 AES Accelerator Interrupts

The AESRDYIFG interrupt flag is set when the AES module completes the selected operation on the provided data. An interrupt request is generated if AESRDYIE and GIE are also set. AESRDYIFG is automatically reset if the AES interrupt is serviced, if AESADOUT is read, or if AESADIN or AESAKEY are written. AESRDYIFG is reset after a PUC or with AESSWRST = 1. AESRDYIE is reset after a PUC but is not reset by AESSWRST = 1.

11.2.6 Implementing Block Cipher Modes

All block cipher modes can be implemented using the AES accelerator together with software. A separate application report describes the block cipher modes together with their implementation in software.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.3 AES_ACCEL Registers

The AES Accelerator registers are listed in Table 11-1.

Table 11-1. AES_ACCEL Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
000h	AESACTL0	AES accelerator control register 0	Read/write	Word	00h	Section 11.3.1
002h	AESACTL1	AES accelerator control register 1	Read/write	Word	00h	Section 11.3.2
004h	AESASTAT	AES accelerator status register	Read only	Word	00h	Section 11.3.3
006h	AESAKEY	AES accelerator key register	Read/write	Word	00h	Section 11.3.4
008h	AESADIN	AES accelerator data in register	Read/write	Word	00h	Section 11.3.5
00Ah	AESADOUT	AES accelerator data out register	Read/write	Word	00h	Section 11.3.6
00Ch	AESAXDIN	AES accelerator XORed data in register	Read/write	Word	00h	Section 11.3.7
00Eh	AESAXIN	AES accelerator XORed data in register (no trigger)	Read/write	Word	00h	Section 11.3.8

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.3.1 AESACTL0 Register

AES accelerator control register 0

AESACTL0 is shown in Figure 11-6 and described in Table 11-2.

Figure 11-6. AESACTL0 Register

15	14	13	12	11	10	9	8
AESCMEN	Reserved		AESRDYIE	AESERRFG	Reserved		AESRDYIFG
rw-0	r0	r0	rw-0	rw-0	r0	r0	rw-0
7	6	5	4	3	2	1	0
AESSWRST	AESCMx		Reserved	AESKLx		AESOPx	
rw-0	r0	r0	r0	rw-0	rw-0	rw-0	rw-0

Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0.

Table 11-2. AESACTL0 Register Description

Bit	Field	Type	Reset	Description
15	AESCMEN	RW	0h	AESCMEN enables the support of the ciphermodes ECB, CBC, OFB and CFB together with the DMA. Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0. 0 = No DMA triggers are generated 1 = DMA ciphermode support operation is enabled and the corresponding DMA triggers are generated.
14-13	Reserved	R	0h	Reserved
12	AESRDYIE	RW	0h	AES ready interrupt enable. AESRDYIE is not reset by AESSWRST = 1. 0 = Interrupt disabled 1 = Interrupt enabled
11	AESERRFG	RW	0h	AES error flag. AESAKEY or AESADIN were written while an AES operation was in progress. The bit must be cleared by software. 0 = No error 1 = Error occurred
10-9	Reserved	R	0h	Reserved
8	AESRDYIFG	RW	0h	AES ready interrupt flag. Set when the selected AES operation was completed and the result can be read from AESADOUT. Automatically cleared when AESADOUT is read or AESAKEY or AESADIN is written. 0 = No interrupt pending 1 = Interrupt pending
7	AESSWRST	RW	0h	AES software reset. Immediately resets the complete AES accelerator module even when busy except for the AESRDYIE, the AESKLx and the AESOPx bits. It also clears the (internal) state memory. The AESSWRST bit is automatically reset and is always read as zero. 0 = No reset 1 = Reset AES accelerator module
6-5	AESCMx	R	0h	AES cipher mode select. These bits are ignored for AESCMEN=0. Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0. 00 = ECB 01 = CBC 10 = OFB 11 = CFB
4	Reserved	R	0h	Reserved

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 11-2. AESACTL0 Register Description (continued)

Bit	Field	Type	Reset	Description
3-2	AESKLx	RW	0h	AES key length. These bits define which of the 3 AES standards is performed. The AESKLx bits are not reset by AESSWRST = 1. Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0. 00 = AES128. The keysize is 128 bit. 01 = AES192. The keysize is 192 bit. 10 = AES256. The keysize is 256 bit. 11 = Reserved
1-0	AESOPx	RW	0h	AES operation. The AESOPx bits are not reset by AESSWRST = 1. Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0. 00 = Encryption 01 = Decryption. The provided key is the same key used for encryption. 10 = Generate first round key required for decryption. 11 = Decryption. The provided key is the first round key required for decryption.

11.3.2 AESACTL1 Register

AES Accelerator Control Register 1

AESACTL1 is shown in Figure 11-7 and described in Table 11-3.

Figure 11-7. AESACTL1 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
AESBLKCNTx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0.

Table 11-3. AESACTL1 Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0	Reserved. Always reads 0.
7-0	AESBLKCNTx	RW	0	Cipher Block Counter. Number of blocks to be encrypted or decrypted with block cipher modes enabled (AESCMEN=1). Ignored if AESCMEN=0. The block counter decrements with each performed encryption or decryption. Writes are ignored when AESCMEN = 1 and AESBLKCNTx > 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.3.3 AESASTAT Register

AES Accelerator Status Register

AESASTAT is shown in Figure 11-8 and described in Table 11-4.

Figure 11-8. AESASTAT Register

15	14	13	12	11	10	9	8
AESDOUTCNTx				AESDINCNTx			
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
7	6	5	4	3	2	1	0
AESKEYCNTx				AESDOUTRD	AESDINWR	AEKEYWR	AESBUSY
r-0	r-0	r-0	r-0	r-0	rw-0	rw-0	r-0

Table 11-4. AESASTAT Register Description

Bit	Field	Type	Reset	Description
15-12	AESDOUTCNTx	R	0h	Bytes read via AESADOUT. Reset when AESDOUTRD is reset. If AESDOUTCNTx = 0 and AESDOUTRD = 0, no bytes were read. If AESDOUTCNTx = 0 and AESDOUTRD = 1, all bytes were read.
11-8	AESDINCNTx	R	0h	Bytes written via AESADIN, AESAXDIN or AESAXIN. Reset when AESDINWR is reset. If AESDINCNTx = 0 and AESDINWR = 0, no bytes were written. If AESDINCNTx = 0 and AESDINWR = 1, all bytes were written.
7-4	AESKEYCNTx	R	0h	Bytes written via AESAKEY for AESKLx=00, words written via AESAKEY if AESKLx=01,10,11. Reset when AESKEYWR is reset. If AESKEYCNTx = 0 and AESKEYWR = 0, no bytes were written. If AESKEYCNTx = 0 and AESKEYWR = 1, all bytes were written.
3	AESDOUTRD	R	0h	All 16 bytes read from AESADOUT. AESDOUTRD is reset by PUC, AESSWRST, an error condition, changing AESOPx, changing AESKLx, when the AES accelerator is busy, and when the output data is read again. 0 = Not all bytes read 1 = All bytes read
2	AESDINWR	RW	0h	All 16 bytes written to AESADIN, AESAXDIN or AESAXIN. This bit can be modified by software only if AESCMEN=0. Changing its state by software also resets the AESDINCNTx bits. AESDINWR is reset by PUC, AESSWRST, an error condition, changing AESOPx, changing AESKLx, the start to (over)write the data, and when the AES accelerator is busy. Because it is reset when AESOPx or AESKLx is changed it can be set by software again to indicate that the current data is still valid. 0 = Not all bytes written 1 = All bytes written
1	AESKEYWR	RW	0h	All 16 bytes written to AESAKEY. This bit can be modified by software but it must not be reset by software (1→0) if AESCMEN=1. Changing its state by software also resets the AESKEYCNTx bits. AESKEYWR is reset by PUC, AESSWRST, an error condition, changing AESOPx, changing AESKLx, and the start to (over)write a new key. Because it is reset when AESOPx is changed it can be set by software again to indicate that the loaded key is still valid. 0 = Not all bytes written 1 = All bytes written
0	AESBUSY	R	0h	AES accelerator module busy; encryption, decryption, or key generation in progress. 0 = Not busy 1 = Busy

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

11.3.4 AESAKEY Register

AES Accelerator Key Register

AESAKEY is shown in Figure 11-9 and described in Table 11-5.

Figure 11-9. AESAKEY Register

15	14	13	12	11	10	9	8
AESKEY1x (Key Byte n+1)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0
7	6	5	4	3	2	1	0
AESKEY0x (Key Byte n)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0

Table 11-5. AESAKEY Register Description

Bit	Field	Type	Reset	Description
15-8	AESKEY1x	W	0	AES key byte n+1 when AESAKEY is written as word. Do not use these bits for byte access. Do not mix word and byte access. Always reads as zero. The key is reset by PUC or by AESSWRST = 1.
7-0	AESKEY0x	W	0	AES key byte n when AESAKEY is written as word. AES next key byte when AESAKEY_L is written as byte. Do not mix word and byte access. Always reads as zero. The key is reset by PUC or by AESSWRST = 1.

Copyright © 2009–2013, Texas Instruments Incorporated

11.3.5 AESADIN Register

AES Accelerator Data In Register

AESADIN is shown in Figure 11-10 and described in Table 11-6.

Figure 11-10. AESADIN Register

15	14	13	12	11	10	9	8
AESDIN1x (DIN Byte n+1)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0
7	6	5	4	3	2	1	0
AESDIN0x (DIN Byte n)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0

Table 11-6. AESADIN Register Description

Bit	Field	Type	Reset	Description
15-8	AESDIN1x	W	0	AES data in byte n+1 when AESADIN is written as word. Do not use these bits for byte access. Do not mix word and byte access. Always reads as zero.
7-0	AESDIN0x	W	0	AES data in byte n when AESADIN is written as word. AES next data in byte when AESADIN_L is written as byte. Do not mix word and byte access. Always reads as zero.

11.3.6 AESADOUT Register

AES Accelerator Data Out Register

AESADOUT is shown in Figure 11-11 and described in Table 11-7.

Figure 11-11. AESADOUT Register

15	14	13	12	11	10	9	8
AESDOUT1x (DOUT Byte n+1)							
7	6	5	4	3	2	1	0
AESDOUT0x (DOUT Byte n)							

Table 11-7. AESADOUT Register Description

Bit	Field	Type	Reset	Description
15-8	AESDOUT1x	R	0	AES data out byte n+1 when AESADOUT is read as word. Do not use these bits for byte access. Do not mix word and byte access.
7-0	AESDOUT0x	R	0	AES data out byte n when AESADOUT is read as word. AES next data out byte when AESADOUT_L is read as byte. Do not mix word and byte access.

Copyright © 2009–2013, Texas Instruments Incorporated

11.3.7 AESAXDIN Register

AES accelerator XORed data in register

AESAXDIN is shown in Figure 11-12 and described in Table 11-8.

Figure 11-12. AESAXDIN Register

15	14	13	12	11	10	9	8
AESXDIN1x (XDIN Byte n+1)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0
7	6	5	4	3	2	1	0
AESXDIN0x (XDIN Byte n)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0

Table 11-8. AESAXDIN Register Description

Bit	Field	Type	Reset	Description
15-8	AESXDIN1x	W	0	AES data in byte n+1 when AESAXDIN is written as word. Do not use these bits for byte access. Do not mix word and byte access. Always reads as zero.
7-0	AESXDIN0x	W	0	AES data in byte n when AESAXDIN is written as word. AES next data in byte when AESAXDIN_L is written as byte. Do not mix word and byte access. Always reads as zero.

11.3.8 AESAXIN Register

AES accelerator XORed data in register (no trigger)

AESAXIN is shown in Figure 11-13 and described in Table 11-9.

Figure 11-13. AESAXIN Register

15	14	13	12	11	10	9	8
AESXIN1x (DIN Byte n+1)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0
7	6	5	4	3	2	1	0
AESXIN0x (DIN Byte n)							
w-0	w-0	w-0	w-0	w-0	w-0	w-0	w-0

Table 11-9. AESAXIN Register Description

Bit	Field	Type	Reset	Description
15-8	AESXIN1x	W	0	AES data in byte n+1 when AESAXIN is written as word. Do not use these bits for byte access. Do not mix word and byte access. Always reads as zero.
7-0	AESXIN0x	W	0	AES data in byte n when AESAXIN is written as word. AES next data in byte when AESAXIN_L is written as byte. Do not mix word and byte access. Always reads as zero.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Watchdog Timer (WDT_A)

The watchdog timer is a 32-bit timer that can be used as a watchdog or as an interval timer. This chapter describes the watchdog timer. The enhanced watchdog timer, WDT_A, is implemented in all devices.

Topic **Page**

12.1 WDT_A Introduction 390
12.2 WDT_A Operation 392
12.3 WDT_A Registers 394

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

12.1 WDT_A Introduction

The primary function of the watchdog timer (WDT_A) module is to perform a controlled system restart after a software problem occurs. If the selected time interval expires, a system reset is generated. If the watchdog function is not needed in an application, the module can be configured as an interval timer and can generate interrupts at selected time intervals.

Features of the watchdog timer module include:

- Eight software-selectable time intervals
- Watchdog mode
- Interval mode
- Password-protected access to Watchdog Timer Control (WDTCTL) register
- Selectable clock source
- Can be stopped to conserve power
- Clock fail-safe feature

The watchdog timer block diagram is shown in Figure 12-1.

NOTE: Watchdog timer powers up active.

After a PUC, the WDT_A module is automatically configured in the watchdog mode with an initial ~32-ms reset interval using the SMCLK. The user must setup or halt the WDT_A prior to the expiration of the initial reset interval.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

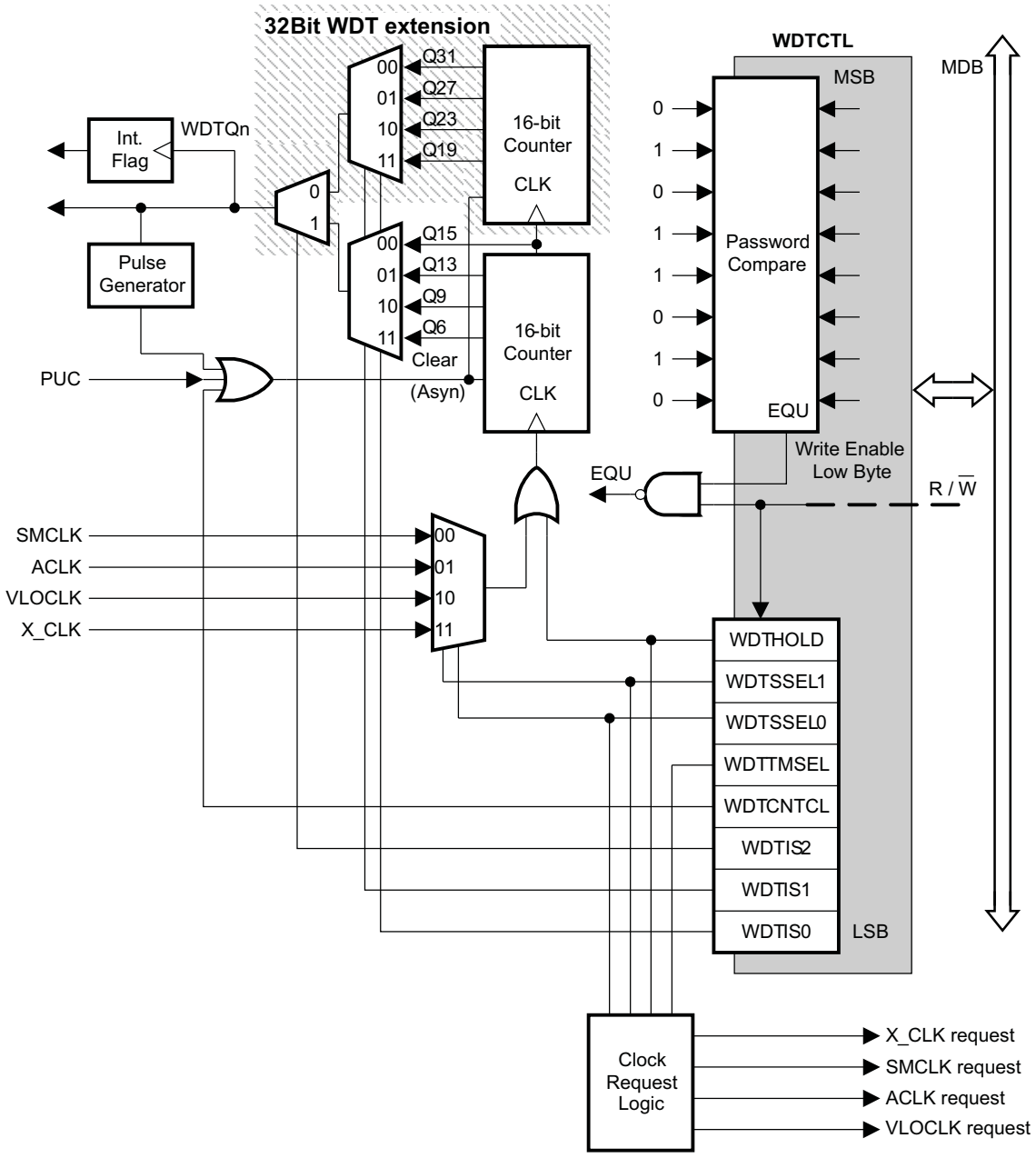


Figure 12-1. Watchdog Timer Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

12.2 WDT_A Operation

The watchdog timer module can be configured as either a watchdog or interval timer with the WDTCTL register. WDTCTL is a 16-bit password-protected read/write register. Any read or write access must use word instructions and write accesses must include the write password 05Ah in the upper byte. Any write to WDTCTL with any value other than 05Ah in the upper byte is a password violation and triggers a PUC system reset, regardless of timer mode. Any read of WDTCTL reads 069h in the upper byte. Byte reads on WDTCTL high or low part result in the value of the low byte. Writing byte wide to upper or lower parts of WDTCTL results in a PUC.

12.2.1 Watchdog Timer Counter (WDTCNT)

The WDTCNT is a 32-bit up counter that is not directly accessible by software. The WDTCNT is controlled and its time intervals are selected through the Watchdog Timer Control (WDTCTL) register. The WDTCNT can be sourced from SMCLK, ACLK, VLOCLK, or X_CLK on some devices. The clock source is selected with the WDTSSEL bits. The timer interval is selected with the WDTIS bits.

12.2.2 Watchdog Mode

After a PUC condition, the WDT module is configured in the watchdog mode with an initial ~32-ms reset interval using the SMCLK. The user must setup, halt, or clear the watchdog timer prior to the expiration of the initial reset interval or another PUC is generated. When the watchdog timer is configured to operate in watchdog mode, either writing to WDTCTL with an incorrect password, or expiration of the selected time interval triggers a PUC. A PUC resets the watchdog timer to its default condition.

12.2.3 Interval Timer Mode

Setting the WDTTMSEL bit to 1 selects the interval timer mode. This mode can be used to provide periodic interrupts. In interval timer mode, the WDTIFG flag is set at the expiration of the selected time interval. A PUC is not generated in interval timer mode at expiration of the selected timer interval, and the WDTIFG enable bit WDTIE remains unchanged.

When the WDTIE bit and the GIE bit are set, the WDTIFG flag requests an interrupt. The WDTIFG interrupt flag is automatically reset when its interrupt request is serviced, or it may be reset by software. The interrupt vector address in interval timer mode is different from that in watchdog mode.

NOTE: Modifying the watchdog timer

The watchdog timer interval should be changed together with WDTCNTCL = 1 in a single instruction to avoid an unexpected immediate PUC or interrupt. The watchdog timer should be halted before changing the clock source to avoid a possible incorrect interval.

12.2.4 Watchdog Timer Interrupts

The watchdog timer uses two bits in the SFRs for interrupt control:

- WDT interrupt flag, WDTIFG, located in SFRIFG1.0
- WDT interrupt enable, WDTIE, located in SFRIE1.0

When using the watchdog timer in the watchdog mode, the WDTIFG flag sources a reset vector interrupt.The WDTIFG will self clear upon a watchdog timeout event. The SYSRSTIV can be read to determine if the reset was caused by a watchdog timeout event.

When using the watchdog timer in interval timer mode, the WDTIFG flag is set after the selected time interval and requests a watchdog timer interval timer interrupt if the WDTIE and the GIE bits are set. The interval timer interrupt vector is different from the reset vector used in watchdog mode. In interval timer mode, the WDTIFG flag is reset automatically when the interrupt is serviced, or can be reset with software.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

12.2.5 Clock Fail-Safe Feature

The WDT_A provides a fail-safe clocking feature, ensuring the clock to the WDT_A cannot be disabled while in watchdog mode. This means that the low-power modes may be affected by the choice for the WDT_A clock.

If SMCLK or ACLK fails as the WDT_A clock source, VLOCLK is automatically selected as the WDT_A clock source.

When the WDT_A module is used in interval timer mode, there is no fail-safe feature within WDT_A for the clock source.

12.2.6 Operation in Low-Power Modes

The devices have several low-power modes. Different clock signals are available in different low-power modes. The requirements of the application and the type of clocking that is used determine how the WDT_A should be configured. For example, the WDT_A should not be configured in watchdog mode with a clock source that is originally sourced from DCO, XT1 in high-frequency mode, or XT2 via SMCLK or ACLK, if the user wants to use low-power mode 3. In this case, SMCLK or ACLK would remain enabled, increasing the current consumption of LPM3. When the watchdog timer is not required, the WDTHOLD bit can be used to hold the WDTCNT, reducing power consumption.

12.2.7 Software Examples

Any write operation to WDTCTL must be a word operation with 05Ah (WDTPW) in the upper byte:

```
; Periodically clear an active watchdog
MOV #WDTPW+WDTIS2+WDTIS1+WDTCNTCL,&WDTCTL
;
; Change watchdog timer interval
MOV #WDTPW+WDTCNTCL+SSEL,&WDTCTL
;
; Stop the watchdog
MOV #WDTPW+WDTHOLD,&WDTCTL
;
; Change WDT to interval timer mode, clock/8192 interval
MOV #WDTPW+WDTCNTCL+WDTTMSEL+WDTIS2+WDTIS0,&WDTCTL
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

12.3 WDT_A Registers

The watchdog timer module registers are listed in Table 12-1. The base address for the watchdog timer module registers and special function registers (SFRs) can be found in device-specific data sheets. The address offset is given in Table 12-1.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 12-1. WDT_A Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
0Ch	WDTCTL	Watchdog Timer Control	Read/write	Word	6904h	Section 12.3.1
0Ch	WDTCTL_L		Read/write	Byte	04h	
0Dh	WDTCTL_H		Read/write	Byte	69h	

Copyright © 2009–2013, Texas Instruments Incorporated

12.3.1 WDTCTL Register

Watchdog Timer Control Register

Figure 12-2. WDTCTL Register

15	14	13	12	11	10	9	8
WDTPW							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

7	6	5	4	3	2	1	0
WDTHOLD	WDTSSEL		WDTTMSEL	WDTCNTCL	WDTIS		
rw-0	rw-0	rw-0	rw-0	r0(w)	rw-1	rw-0	rw-0

Table 12-2. WDTCTL Register Description

Bit	Field	Type	Reset	Description
15-8	WDTPW	RW	69h	Watchdog timer password. Always read as 069h. Must be written as 5Ah; if any other value is written, a PUC is generated.
7	WDTHOLD	RW	0h	Watchdog timer hold. This bit stops the watchdog timer. Setting WDTHOLD = 1 when the WDT is not in use conserves power. 0b = Watchdog timer is not stopped. 1b = Watchdog timer is stopped.
6-5	WDTSSEL	RW	0h	Watchdog timer clock source select 00b = SMCLK 01b = ACLK 10b = VLOCLK 11b = X_CLK; VLOCLK in devices that do not support X_CLK
4	WDTTMSEL	RW	0h	Watchdog timer mode select 0b = Watchdog mode 1b = Interval timer mode
3	WDTCNTCL	RW	0h	Watchdog timer counter clear. Setting WDTCNTCL = 1 clears the count value to 0000h. WDTCNTCL is automatically reset. 0b = No action 1b = WDTCNT = 0000h
2-0	WDTIS	RW	4h	Watchdog timer interval select. These bits select the watchdog timer interval to set the WDTIFG flag and/or generate a PUC. 000b = Watchdog clock source /(2^31) (18h:12m:16s at 32.768 kHz) 001b = Watchdog clock source /(2^27) (01h:08m:16s at 32.768 kHz) 010b = Watchdog clock source /(2^23) (00h:04m:16s at 32.768 kHz) 011b = Watchdog clock source /(2^19) (00h:00m:16s at 32.768 kHz) 100b = Watchdog clock source /(2^15) (1 s at 32.768 kHz) 101b = Watchdog clock source /(2^13) (250 ms at 32.768 kHz) 110b = Watchdog clock source /(2^9) (15.625 ms at 32.768 kHz) 111b = Watchdog clock source /(2^6) (1.95 ms at 32.768 kHz)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Timer_A

Timer_A is a 16-bit timer/counter with multiple capture/compare registers. There can be multiple Timer_A modules on a given device (see the device-specific data sheet). This chapter describes the operation and use of the Timer_A module.

Topic **Page**

13.1 Timer_A Introduction **397**
13.2 Timer_A Operation **399**
13.3 Timer_A Registers **411**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.1 Timer_A Introduction

Timer_A is a 16-bit timer/counter with up to seven capture/compare registers. Timer_A can support multiple capture/compares, PWM outputs, and interval timing. Timer_A also has extensive interrupt capabilities. Interrupts may be generated from the counter on overflow conditions and from each of the capture/compare registers.

Timer_A features include:

- Asynchronous 16-bit timer/counter with four operating modes
- Selectable and configurable clock source
- Up to seven configurable capture/compare registers
- Configurable outputs with pulse width modulation (PWM) capability
- Asynchronous input and output latching
- Interrupt vector register for fast decoding of all Timer_A interrupts

The block diagram of Timer_A is shown in Figure 13-1.

NOTE: Use of the word *count*

Count is used throughout this chapter. It means the counter must be in the process of counting for the action to take place. If a particular value is directly written to the counter, an associated action does not take place.

NOTE: Nomenclature

There may be multiple instantiations of Timer_A on a given device. The prefix TAx is used, where x is a greater than equal to zero indicating the Timer_A instantiation. For devices with one instantiation, x = 0. The suffix n, where n = 0 to 6, represents the specific capture/compare registers associated with the Timer_A instantiation.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

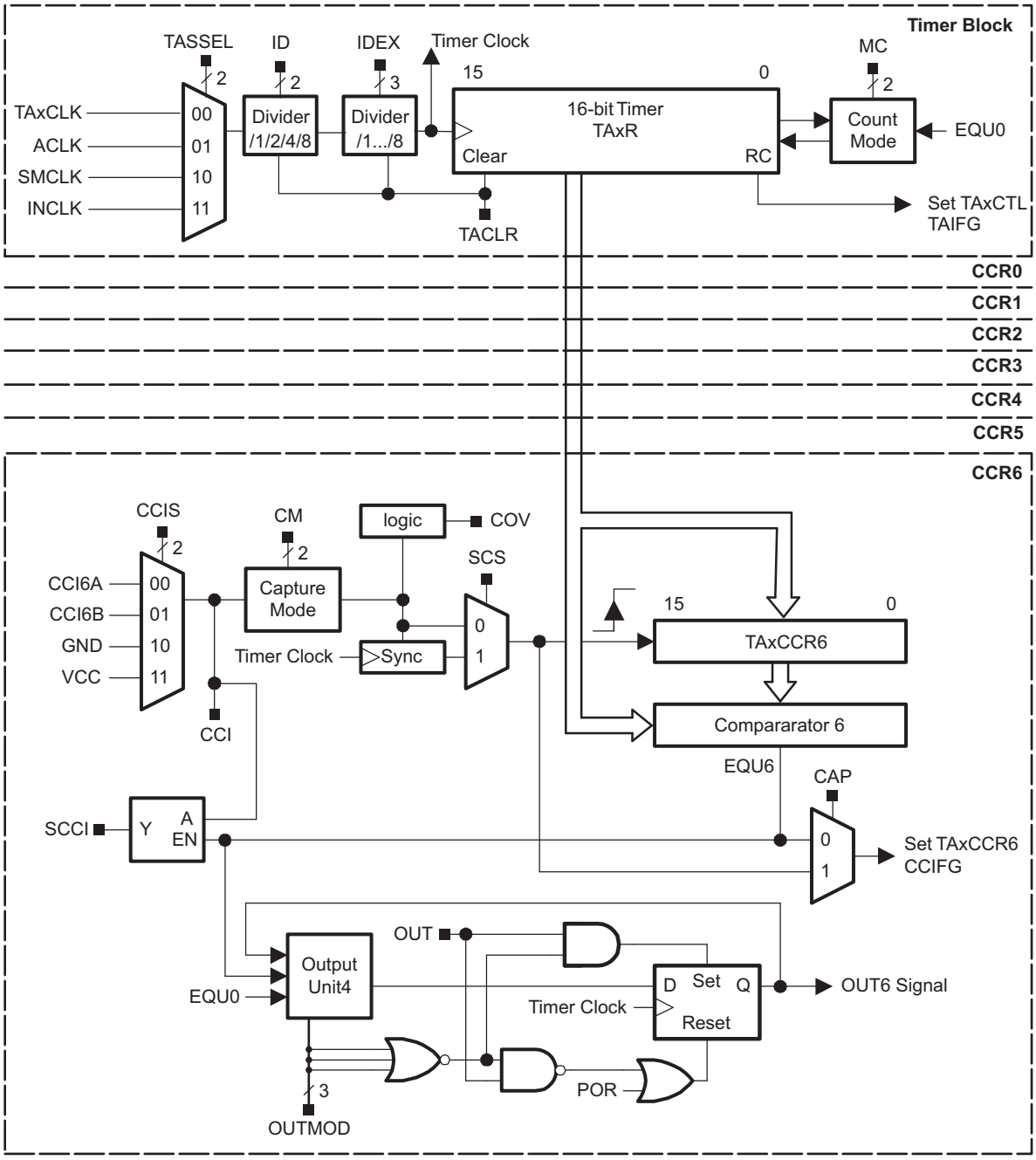


Figure 13-1. Timer_A Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2 Timer_A Operation

The Timer_A module is configured with user software. The setup and operation of Timer_A are discussed in the following sections.

13.2.1 16-Bit Timer Counter

The 16-bit timer/counter register, TAxR, increments or decrements (depending on mode of operation) with each rising edge of the clock signal. TAxR can be read or written with software. Additionally, the timer can generate an interrupt when it overflows.

TAxR may be cleared by setting the TACLR bit. Setting TACLR also clears the clock divider and count direction for up/down mode.

NOTE: Modifying Timer_A registers

It is recommended to stop the timer before modifying its operation (with exception of the interrupt enable, interrupt flag, and TACLR) to avoid errant operating conditions.

When the timer clock is asynchronous to the CPU clock, any read from TAxR should occur while the timer is not operating or the results may be unpredictable. Alternatively, the timer may be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Any write to TAxR takes effect immediately.

13.2.1.1 Clock Source Select and Divider

The timer clock can be sourced from ACLK, SMCLK, or externally via TAxCLK or INCLK. The clock source is selected with the TASSEL bits. The selected clock source may be passed directly to the timer or divided by 2, 4, or 8, using the ID bits. The selected clock source can be further divided by 2, 3, 4, 5, 6, 7, or 8 using the TAIDEX bits. The timer clock divider logic is reset when TACLR is set.

NOTE: Timer_A dividers

After programming ID or TAIDEX bits, set the TACLR bit. This clears the contents of TAxR and resets the clock divider logic to a defined state. The clock dividers are implemented as down counters. Therefore, when the TACLR bit is cleared, the timer clock immediately begins clocking at the first rising edge of the Timer_A clock source selected with the TASSEL bits and continues clocking at the divider settings set by the ID and TAIDEX bits.

13.2.2 Starting the Timer

The timer may be started or restarted in the following ways:

- The timer counts when MC > { 0 } and the clock source is active.
- When the timer mode is either up or up/down, the timer may be stopped by writing 0 to TAxCCR0. The timer may then be restarted by writing a nonzero value to TAxCCR0. In this scenario, the timer starts incrementing in the up direction from zero.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.3 Timer Mode Control

The timer has four modes of operation: stop, up, continuous, and up/down (see Table 13-1). The operating mode is selected with the MC bits.

Table 13-1. Timer Modes

MC	Mode	Description
00	Stop	The timer is halted.
01	Up	The timer repeatedly counts from zero to the value of TAxCCR0
10	Continuous	The timer repeatedly counts from zero to 0FFFFh.
11	Up/down	The timer repeatedly counts from zero up to the value of TAxCCR0 and back down to zero.

13.2.3.1 Up Mode

The up mode is used if the timer period must be different from 0FFFFh counts. The timer repeatedly counts up to the value of compare register TAxCCR0, which defines the period (see Figure 13-2). The number of timer counts in the period is TAxCCR0 + 1. When the timer value equals TAxCCR0, the timer restarts counting from zero. If up mode is selected when the timer value is greater than TAxCCR0, the timer immediately restarts counting from zero.

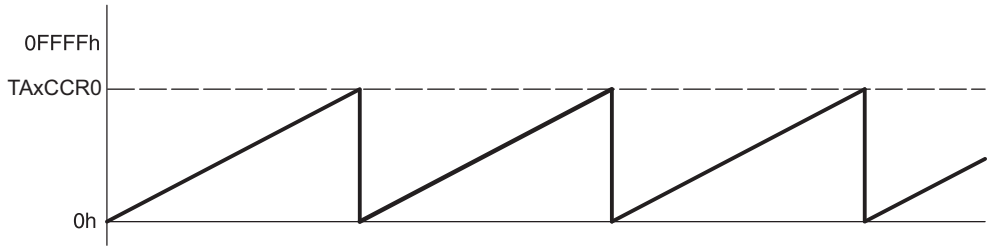


Figure 13-2. Up Mode

The TAxCCR0 CCIFG interrupt flag is set when the timer *counts* to the TAxCCR0 value. The TAIFG interrupt flag is set when the timer *counts* from TAxCCR0 to zero. Figure 13-3 shows the flag set cycle.

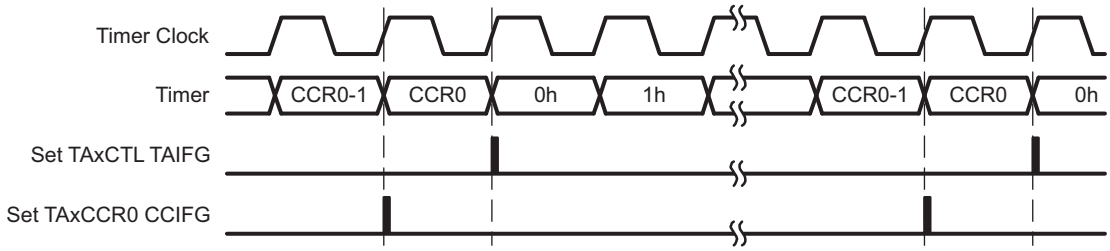


Figure 13-3. Up Mode Flag Setting

13.2.3.1.1 Changing Period Register TAxCCR0

When changing TAxCCR0 while the timer is running, if the new period is greater than or equal to the old period or greater than the current count value, the timer counts up to the new period. If the new period is less than the current count value, the timer rolls to zero. However, one additional count may occur before the counter rolls to zero.

Copyright © 2009–2013, Texas Instruments Incorporated

13.2.3.2 Continuous Mode

In the continuous mode, the timer repeatedly counts up to 0FFFFh and restarts from zero as shown in Figure 13-4. The capture/compare register TAxCCR0 works the same way as the other capture/compare registers.

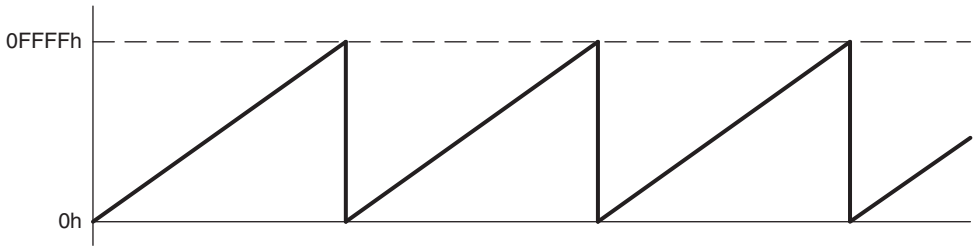


Figure 13-4. Continuous Mode

The TAIFG interrupt flag is set when the timer *counts* from 0FFFFh to zero. Figure 13-5 shows the flag set cycle.

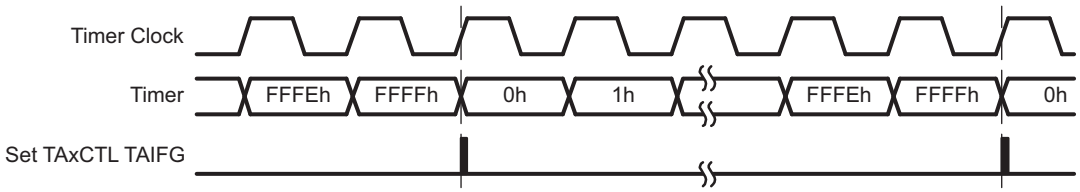


Figure 13-5. Continuous Mode Flag Setting

13.2.3.3 Use of Continuous Mode

The continuous mode can be used to generate independent time intervals and output frequencies. Each time an interval is completed, an interrupt is generated. The next time interval is added to the TAxCCRn register in the interrupt service routine. Figure 13-6 shows two separate time intervals, t_0 and t_1, being added to the capture/compare registers. In this usage, the time interval is controlled by hardware, not software, without impact from interrupt latency. Up to n (where n = 0 to 6), independent time intervals or output frequencies can be generated using capture/compare registers.

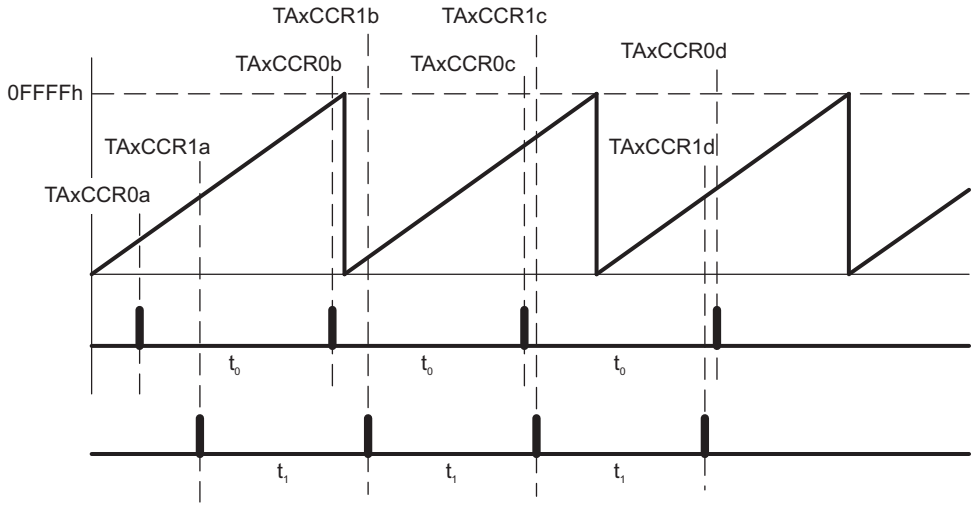


Figure 13-6. Continuous Mode Time Intervals

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Time intervals can be produced with other modes as well, where TAxCCR0 is used as the period register. Their handling is more complex since the sum of the old TAxCCRn data and the new period can be higher than the TAxCCR0 value. When the previous TAxCCRn value plus t_x is greater than the TAxCCR0 data, the TAxCCR0 value must be subtracted to obtain the correct time interval.

13.2.3.4 Up/Down Mode

The up/down mode is used if the timer period must be different from 0FFFFh counts, and if symmetrical pulse generation is needed. The timer repeatedly counts up to the value of compare register TAxCCR0 and back down to zero (see Figure 13-7). The period is twice the value in TAxCCR0.

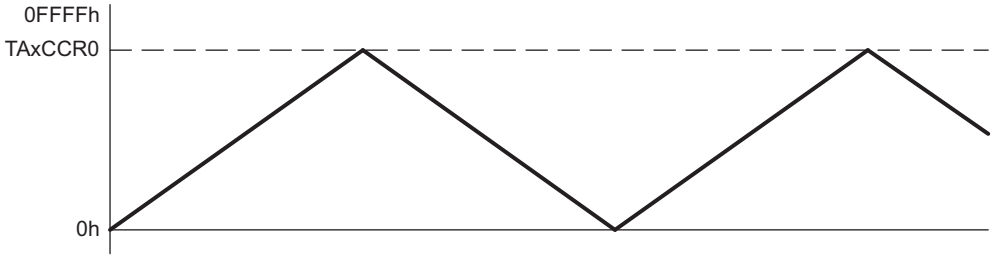


Figure 13-7. Up/Down Mode

The count direction is latched. This allows the timer to be stopped and then restarted in the same direction it was counting before it was stopped. If this is not desired, the TACLR bit must be set to clear the direction. The TACLR bit also clears the TAxR value and the timer clock divider.

In up/down mode, the TAxCCR0 CCIFG interrupt flag and the TAIFG interrupt flag are set only once during a period, separated by one-half the timer period. The TAxCCR0 CCIFG interrupt flag is set when the timer *counts* from TAxCCR0-1 to TAxCCR0, and TAIFG is set when the timer completes *counting* down from 0001h to 0000h. Figure 13-8 shows the flag set cycle.

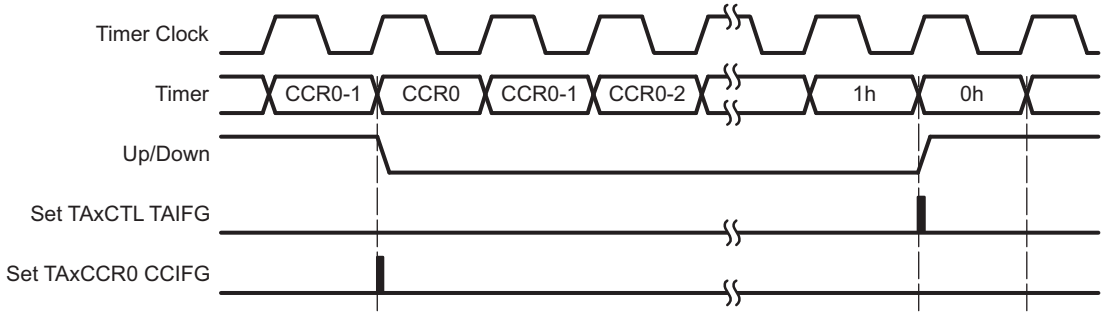


Figure 13-8. Up/Down Mode Flag Setting

13.2.3.4.1 Changing Period Register TAxCCR0

When changing TAxCCR0 while the timer is running and counting in the down direction, the timer continues its descent until it reaches zero. The new period takes effect after the counter counts down to zero.

When the timer is counting in the up direction, and the new period is greater than or equal to the old period or greater than the current count value, the timer counts up to the new period before counting down.

When the timer is counting in the up direction and the new period is less than the current count value, the timer begins counting down. However, one additional count may occur before the counter begins counting down.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.3.5 Use of Up/Down Mode

The up/down mode supports applications that require dead times between output signals (see section *Timer_A Output Unit*). For example, to avoid overload conditions, two outputs driving an H-bridge must never be in a high state simultaneously. In the example shown in Figure 13-9, the t_{dead} is:

$t_{dead} = t_{timer} \times (TAxCCR1 - TAxCCR2)$

Where:

t_{dead} = Time during which both outputs need to be inactive
t_{timer} = Cycle time of the timer clock
TAxCCRn = Content of capture/compare register n

The TAxCCRn registers are not buffered. They update immediately when written to. Therefore, any required dead time is not maintained automatically.

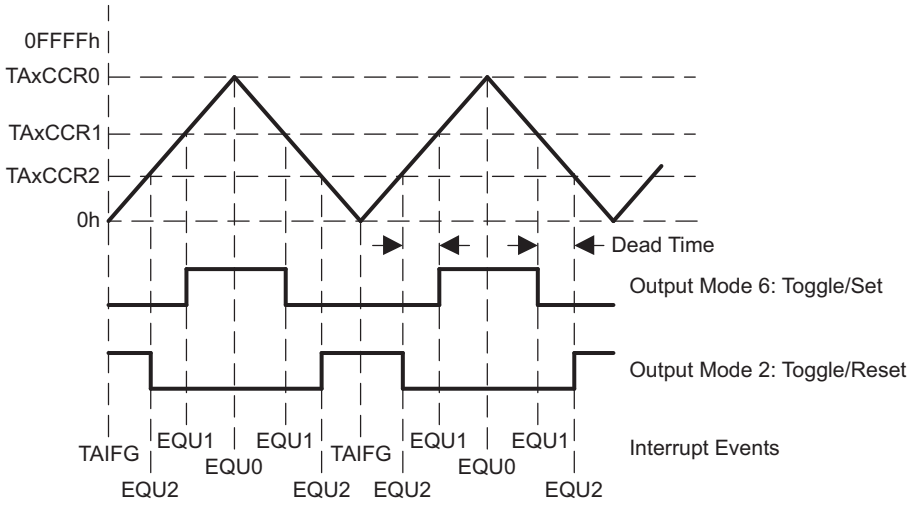


Figure 13-9. Output Unit in Up/Down Mode

13.2.4 Capture/Compare Blocks

Up to seven identical capture/compare blocks, TAxCCRn (where n = 0 to 7), are present in Timer_A. Any of the blocks may be used to capture the timer data or to generate time intervals.

13.2.4.1 Capture Mode

The capture mode is selected when CAP = 1. Capture mode is used to record time events. It can be used for speed computations or time measurements. The capture inputs CCIxA and CCIxB are connected to external pins or internal signals and are selected with the CCIS bits. The CM bits select the capture edge of the input signal as rising, falling, or both. A capture occurs on the selected edge of the input signal. If a capture occurs:

- The timer value is copied into the TAxCCRn register.
- The interrupt flag CCIFG is set.

The input signal level can be read at any time via the CCI bit. Devices may have different signals connected to CCIxA and CCIxB. See the device-specific data sheet for the connections of these signals.

The capture signal can be asynchronous to the timer clock and cause a race condition. Setting the SCS bit synchronizes the capture with the next timer clock. Setting the SCS bit to synchronize the capture signal with the timer clock is recommended (see Figure 13-10).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

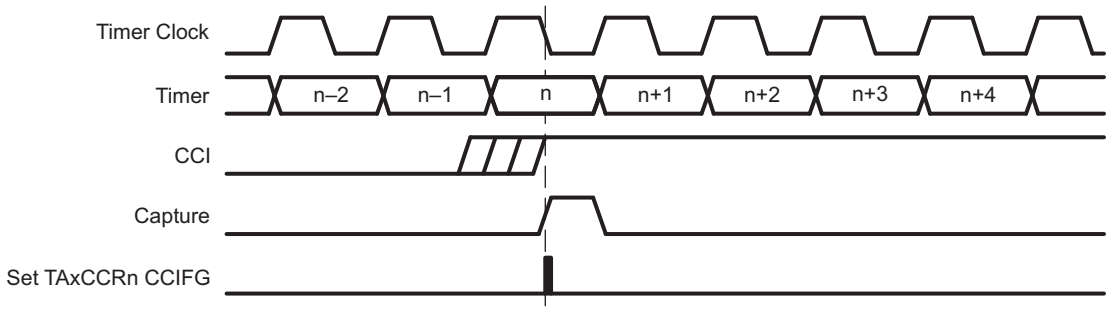


Figure 13-10. Capture Signal (SCS = 1)

NOTE: Changing Capture Inputs

Changing capture inputs while in capture mode may cause unintended capture events. To avoid this scenario, capture inputs should only be changed when capture mode is disabled (CM = {0} or CAP = 0).

Overflow logic is provided in each capture/compare register to indicate if a second capture was performed before the value from the first capture was read. Bit COV is set when this occurs as shown in Figure 13-11. COV must be reset with software.

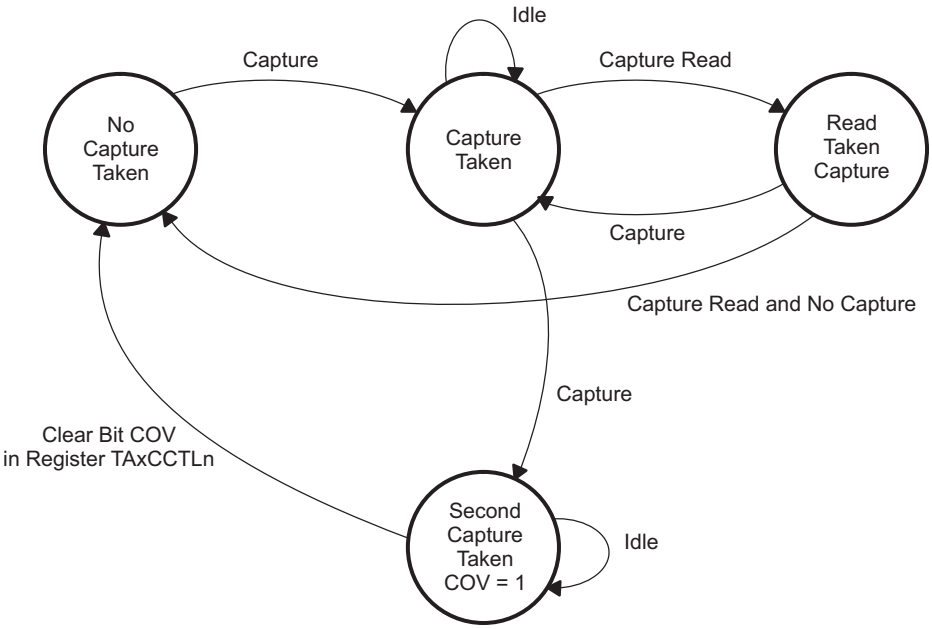


Figure 13-11. Capture Cycle

Copyright © 2009–2013, Texas Instruments Incorporated

13.2.4.1.1 Capture Initiated by Software

Captures can be initiated by software. The CMx bits can be set for capture on both edges. Software then sets CCIS1 = 1 and toggles bit CCIS0 to switch the capture signal between V_{CC} and GND, initiating a capture each time CCIS0 changes state:

```
MOV  #CAP+SCS+CCIS1+CM_3,&TA0CCTL1  ; Setup TA0CCTL1, synch. capture mode
                                    ; Event trigger on both edges of capture input.
XOR  #CCIS0,&TA0CCTL1               ; TA0CCR1 = TA0R
```

NOTE: Capture Initiated by Software

In general, changing capture inputs while in capture mode may cause unintended capture events. For this scenario, switching the capture input between VCC and GND, disabling the capture mode is not required.

13.2.4.2 Compare Mode

The compare mode is selected when CAP = 0. The compare mode is used to generate PWM output signals or interrupts at specific time intervals. When TAxR *counts* to the value in a TAxCCRn, where n represents the specific capture/compare register.

- Interrupt flag CCIFG is set.
- Internal signal EQUn = 1.
- EQUn affects the output according to the output mode.
- The input signal CCI is latched into SCCI.

13.2.5 Output Unit

Each capture/compare block contains an output unit. The output unit is used to generate output signals, such as PWM signals. Each output unit has eight operating modes that generate signals based on the EQU0 and EQUn signals.

13.2.5.1 Output Modes

The output modes are defined by the OUTMOD bits and are described in Table 13-2. The OUTn signal is changed with the rising edge of the timer clock for all modes except mode 0. Output modes 2, 3, 6, and 7 are not useful for output unit 0 because EQUn = EQU0.

Table 13-2. Output Modes

OUTMODx	Mode	Description
000	Output	The output signal OUTn is defined by the OUT bit. The OUTn signal updates immediately when OUT is updated.
001	Set	The output is set when the timer *counts* to the TAxCCRn value. It remains set until a reset of the timer, or until another output mode is selected and affects the output.
010	Toggle/Reset	The output is toggled when the timer *counts* to the TAxCCRn value. It is reset when the timer *counts* to the TAxCCR0 value.
011	Set/Reset	The output is set when the timer *counts* to the TAxCCRn value. It is reset when the timer *counts* to the TAxCCR0 value.
100	Toggle	The output is toggled when the timer *counts* to the TAxCCRn value. The output period is double the timer period.
101	Reset	The output is reset when the timer *counts* to the TAxCCRn value. It remains reset until another output mode is selected and affects the output.
110	Toggle/Set	The output is toggled when the timer *counts* to the TAxCCRn value. It is set when the timer *counts* to the TAxCCR0 value.
111	Reset/Set	The output is reset when the timer *counts* to the TAxCCRn value. It is set when the timer *counts* to the TAxCCR0 value.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.5.1.1 Output Example—Timer in Up Mode

The OUTn signal is changed when the timer *counts* up to the TAxCCRn value and rolls from TAxCCR0 to zero, depending on the output mode. An example is shown in Figure 13-12 using TAxCCR0 and TAxCCR1.

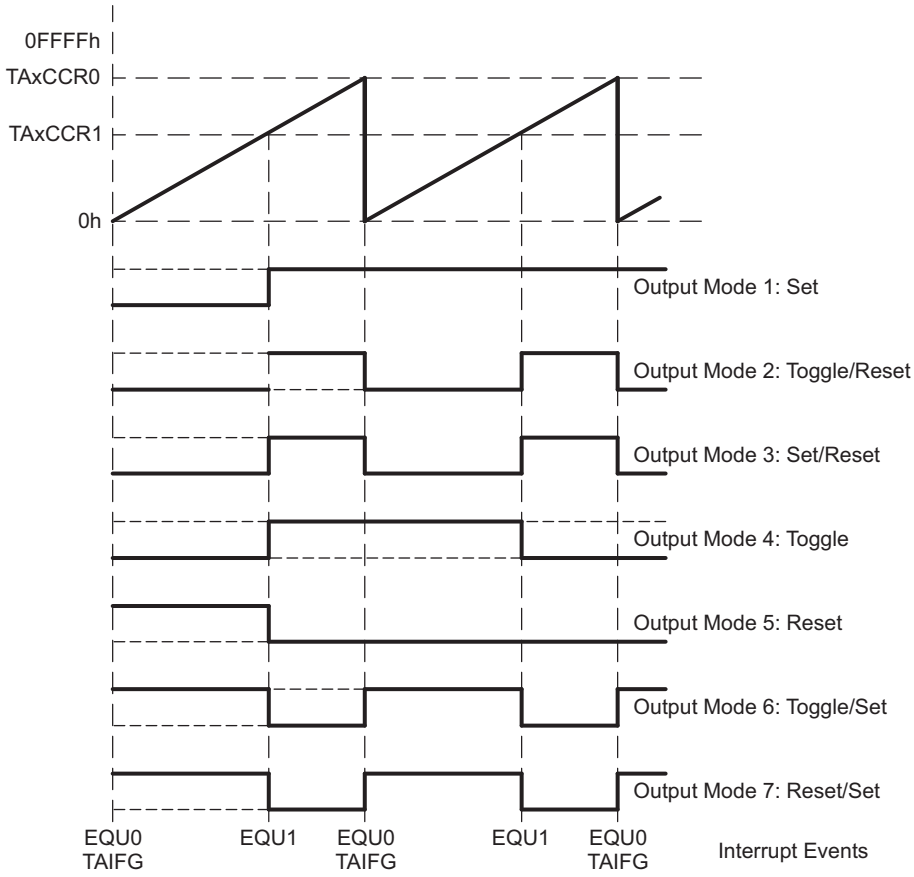


Figure 13-12. Output Example – Timer in Up Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.5.1.2 Output Example – Timer in Continuous Mode

The OUTn signal is changed when the timer reaches the TAxCCRn and TAxCCR0 values, depending on the output mode. An example is shown in Figure 13-13 using TAxCCR0 and TAxCCR1.

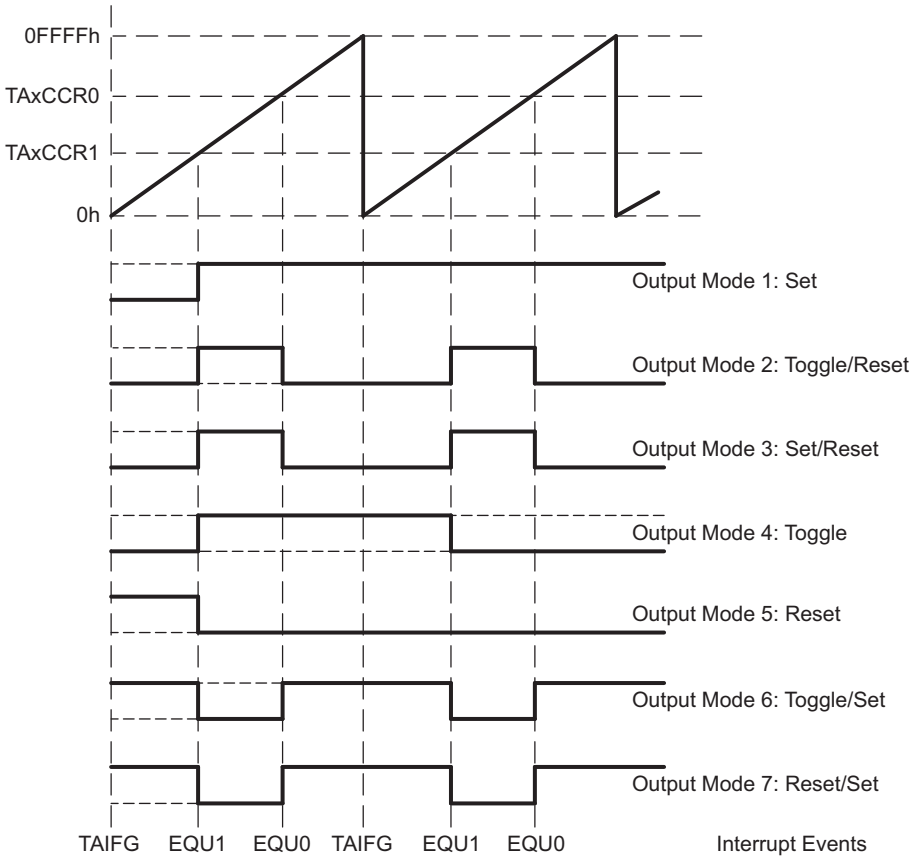


Figure 13-13. Output Example – Timer in Continuous Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.5.1.3 Output Example – Timer in Up/Down Mode

The OUTn signal changes when the timer equals TAxCCRn in either count direction and when the timer equals TAxCCR0, depending on the output mode. An example is shown in Figure 13-14 using TAxCCR0 and TAxCCR2.

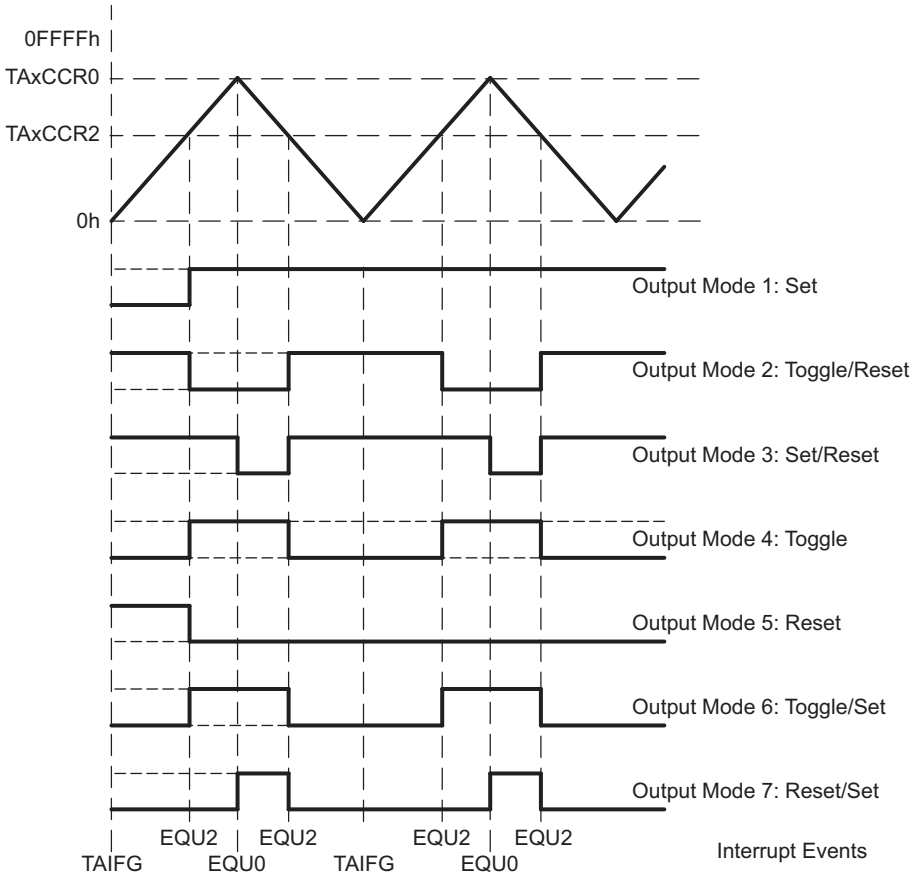


Figure 13-14. Output Example – Timer in Up/Down Mode

NOTE: Switching between output modes

When switching between output modes, one of the OUTMOD bits should remain set during the transition, unless switching to mode 0. Otherwise, output glitching can occur, because a NOR gate decodes output mode 0. A safe method for switching between output modes is to use output mode 7 as a transition state:

```
BIS   #OUTMOD_7,&TA0CCTL1      ; Set output mode=7
BIC   #OUTMOD,&TA0CCTL1       ; Clear unwanted bits
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.2.6 Timer_A Interrupts

Two interrupt vectors are associated with the 16-bit Timer_A module:

- TAxCCR0 interrupt vector for TAxCCR0 CCIFG
- TAxIV interrupt vector for all other CCIFG flags and TAIFG

In capture mode, any CCIFG flag is set when a timer value is captured in the associated TAxCCRn register. In compare mode, any CCIFG flag is set if TAxR *counts* to the associated TAxCCRn value. Software may also set or clear any CCIFG flag. All CCIFG flags request an interrupt when their corresponding CCIE bit and the GIE bit are set.

13.2.6.1 TAxCCR0 Interrupt

The TAxCCR0 CCIFG flag has the highest Timer_A interrupt priority and has a dedicated interrupt vector as shown in Figure 13-15. The TAxCCR0 CCIFG flag is automatically reset when the TAxCCR0 interrupt request is serviced.

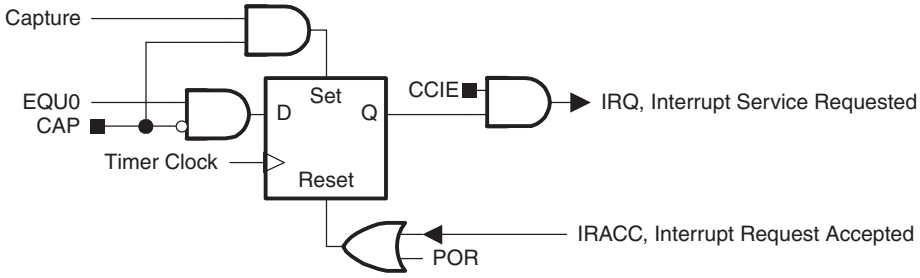


Figure 13-15. Capture/Compare TAxCCR0 Interrupt Flag

13.2.6.2 TAxIV, Interrupt Vector Generator

The TAxCCRy CCIFG flags and TAIFG flags are prioritized and combined to source a single interrupt vector. The interrupt vector register TAxIV is used to determine which flag requested an interrupt.

The highest-priority enabled interrupt generates a number in the TAxIV register (see register description). This number can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled Timer_A interrupts do not affect the TAxIV value.

Any access, read or write, of the TAxIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. For example, if the TAxCCR1 and TAxCCR2 CCIFG flags are set when the interrupt service routine accesses the TAxIV register, TAxCCR1 CCIFG is reset automatically. After the RETI instruction of the interrupt service routine is executed, the TAxCCR2 CCIFG flag generates another interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

13.2.6.2.1 TAxIV Software Example

The following software example shows the recommended use of TAxIV and the handling overhead. The TAxIV value is added to the PC to automatically jump to the appropriate routine. The example assumes a single instantiation of the largest timer configuration available.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself. The latencies are:

- Capture/compare block TA0CCR0: 11 cycles
- Capture/compare blocks TA0CCR1, TA0CCR2, TA0CCR3, TA0CCR4, TA0CCR5, TA0CCR6: 16 cycles
- Timer overflow TA0IFG: 14 cycles

```
; Interrupt handler for TA0CCR0 CCIFG.                    Cycles
CCIFG_0_HND
;        ...         ; Start of handler Interrupt latency    6
         RETI                                                5

; Interrupt handler for TA0IFG, TA0CCR1 through TA0CCR6 CCIFG.

TA0_HND      ...               ; Interrupt latency           6
         ADD      &TA0IV,PC    ; Add offset to Jump table    3
         RETI                  ; Vector  0: No interrupt     5
         JMP      CCIFG_1_HND  ; Vector  2: TA0CCR1          2
         JMP      CCIFG_2_HND  ; Vector  4: TA0CCR2          2
         JMP      CCIFG_3_HND  ; Vector  6: TA0CCR3          2
         JMP      CCIFG_4_HND  ; Vector  8: TA0CCR4          2
         JMP      CCIFG_5_HND  ; Vector 10: TA0CCR5          2
         JMP      CCIFG_6_HND  ; Vector 12: TA0CCR6          2

TA0IFG_HND                    ; Vector 14: TA0IFG Flag
         ...                  ; Task starts here
         RETI                                               5

CCIFG_6_HND                   ; Vector 12: TA0CCR6
         ...                  ; Task starts here
         RETI                 ; Back to main program        5

CCIFG_5_HND                   ; Vector 10: TA0CCR5
         ...                  ; Task starts here
         RETI                 ; Back to main program        5

CCIFG_4_HND                   ; Vector 8: TA0CCR4
         ...                  ; Task starts here
         RETI                 ; Back to main program        5

CCIFG_3_HND                   ; Vector 6: TA0CCR3
         ...                  ; Task starts here
         RETI                 ; Back to main program        5

CCIFG_2_HND                   ; Vector 4: TA0CCR2
         ...                  ; Task starts here
         RETI                 ; Back to main program        5

CCIFG_1_HND                   ; Vector 2: TA0CCR1
         ...                  ; Task starts here
         RETI                 ; Back to main program        5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3 Timer_A Registers

Timer_A registers are listed in Table 13-3 for the largest configuration available. The base address can be found in the device-specific data sheet.

Table 13-3. Timer_A Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	TAxCTL	Timer_Ax Control	Read/write	Word	0000h	Section 13.3.1
02h	TAxCCTL0	Timer_Ax Capture/Compare Control 0	Read/write	Word	0000h	Section 13.3.3
04h	TAxCCTL1	Timer_Ax Capture/Compare Control 1	Read/write	Word	0000h	Section 13.3.3
06h	TAxCCTL2	Timer_Ax Capture/Compare Control 2	Read/write	Word	0000h	Section 13.3.3
08h	TAxCCTL3	Timer_Ax Capture/Compare Control 3	Read/write	Word	0000h	Section 13.3.3
0Ah	TAxCCTL4	Timer_Ax Capture/Compare Control 4	Read/write	Word	0000h	Section 13.3.3
0Ch	TAxCCTL5	Timer_Ax Capture/Compare Control 5	Read/write	Word	0000h	Section 13.3.3
0Eh	TAxCCTL6	Timer_Ax Capture/Compare Control 6	Read/write	Word	0000h	Section 13.3.3
10h	TAxR	Timer_Ax Counter	Read/write	Word	0000h	Section 13.3.2
12h	TAxCCR0	Timer_Ax Capture/Compare 0	Read/write	Word	0000h	Section 13.3.4
14h	TAxCCR1	Timer_Ax Capture/Compare 1	Read/write	Word	0000h	Section 13.3.4
16h	TAxCCR2	Timer_Ax Capture/Compare 2	Read/write	Word	0000h	Section 13.3.4
18h	TAxCCR3	Timer_Ax Capture/Compare 3	Read/write	Word	0000h	Section 13.3.4
1Ah	TAxCCR4	Timer_Ax Capture/Compare 4	Read/write	Word	0000h	Section 13.3.4
1Ch	TAxCCR5	Timer_Ax Capture/Compare 5	Read/write	Word	0000h	Section 13.3.4
1Eh	TAxCCR6	Timer_Ax Capture/Compare 6	Read/write	Word	0000h	Section 13.3.4
2Eh	TAxIV	Timer_Ax Interrupt Vector	Read only	Word	0000h	Section 13.3.5
20h	TAxEX0	Timer_Ax Expansion 0	Read/write	Word	0000h	Section 13.3.6

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3.1 TAxCTL Register

Timer_Ax Control Register

Figure 13-16. TAxCTL Register

15	14	13	12	11	10	9	8
Reserved						TASSEL	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ID		MC		Reserved	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	w-(0)	rw-(0)	rw-(0)

Table 13-4. TAxCTL Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	RW	0h	Reserved
9-8	TASSEL	RW	0h	Timer_A clock source select 00b = TAxCLK 01b = ACLK 10b = SMCLK 11b = INCLK
7-6	ID	RW	0h	Input divider. These bits along with the TAIDEX bits select the divider for the input clock. 00b = /1 01b = /2 10b = /4 11b = /8
5-4	MC	RW	0h	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power. 00b = Stop mode: Timer is halted 01b = Up mode: Timer counts up to TAxCCR0 10b = Continuous mode: Timer counts up to 0FFFFh 11b = Up/down mode: Timer counts up to TAxCCR0 then down to 0000h
3	Reserved	RW	0h	Reserved
2	TACLR	RW	0h	Timer_A clear. Setting this bit resets TAxR, the timer clock divider logic, and the count direction. The TACLR bit is automatically reset and is always read as zero.
1	TAIE	RW	0h	Timer_A interrupt enable. This bit enables the TAIFG interrupt request. 0b = Interrupt disabled 1b = Interrupt enabled
0	TAIFG	RW	0h	Timer_A interrupt flag 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3.2 TAxR Register

Timer_Ax Counter Register

Figure 13-17. TAxR Register

15	14	13	12	11	10	9	8
TAxR							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
TAxR							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 13-5. TAxR Register Description

Bit	Field	Type	Reset	Description
15-0	TAxR	RW	0h	Timer_A register. The TAxR register is the count of Timer_A.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3.3 TAxCCTLn Register

Timer_Ax Capture/Compare Control n Register

Figure 13-18. TAxCCTLn Register

15	14	13	12	11	10	9	8
CM		CCIS		SCS	SCCI	Reserved	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r-(0)	r-(0)	rw-(0)
7	6	5	4	3	2	1	0
OUTMOD			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

Table 13-6. TAxCCTLn Register Description

Bit	Field	Type	Reset	Description
15-14	CM	RW	0h	Capture mode 00b = No capture 01b = Capture on rising edge 10b = Capture on falling edge 11b = Capture on both rising and falling edges
13-12	CCIS	RW	0h	Capture/compare input select. These bits select the TAxCCR0 input signal. See the device-specific data sheet for specific signal connections. 00b = CCIxA 01b = CCIxB 10b = GND 11b = VCC
11	SCS	RW	0h	Synchronize capture source. This bit is used to synchronize the capture input signal with the timer clock. 0b = Asynchronous capture 1b = Synchronous capture
10	SCCI	RW	0h	Synchronized capture/compare input. The selected CCI input signal is latched with the EQUx signal and can be read via this bit.
9	Reserved	R	0h	Reserved. Reads as 0.
8	CAP	RW	0h	Capture mode 0b = Compare mode 1b = Capture mode
7-5	OUTMOD	RW	0h	Output mode. Modes 2, 3, 6, and 7 are not useful for TAxCCR0 because EQUx = EQU0. 000b = OUT bit value 001b = Set 010b = Toggle/reset 011b = Set/reset 100b = Toggle 101b = Reset 110b = Toggle/set 111b = Reset/set
4	CCIE	RW	0h	Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag. 0b = Interrupt disabled 1b = Interrupt enabled
3	CCI	R	0h	Capture/compare input. The selected input signal can be read by this bit.
2	OUT	RW	0h	Output. For output mode 0, this bit directly controls the state of the output. 0b = Output low 1b = Output high

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 13-6. TAxCCTLn Register Description (continued)

Bit	Field	Type	Reset	Description
1	COV	RW	0h	Capture overflow. This bit indicates a capture overflow occurred. COV must be reset with software. 0b = No capture overflow occurred 1b = Capture overflow occurred
0	CCIFG	RW	0h	Capture/compare interrupt flag 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3.4 TAxCCRn Register

Timer_A Capture/Compare n Register

Figure 13-19. TAxCCRn Register

15	14	13	12	11	10	9	8
TAxCCRn							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
TAxCCRn							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 13-7. TAxCCRn Register Description

Bit	Field	Type	Reset	Description
15-0	TAxCCR0	RW	0h	Compare mode: TAxCCRn holds the data for the comparison to the timer value in the Timer_A Register, TAR. Capture mode: The Timer_A Register, TAR, is copied into the TAxCCRn register when a capture is performed.

13.3.5 TAxIV Register

Timer_Ax Interrupt Vector Register

Figure 13-20. TAxIV Register

15	14	13	12	11	10	9	8
TAIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
TAIV							
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r0

Table 13-8. TAxIV Register Description

Bit	Field	Type	Reset	Description
15-0	TAIV	R	0h	Timer_A interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Capture/compare 1; Interrupt Flag: TAxCCR1 CCIFG; Interrupt Priority: Highest 04h = Interrupt Source: Capture/compare 2; Interrupt Flag: TAxCCR2 CCIFG 06h = Interrupt Source: Capture/compare 3; Interrupt Flag: TAxCCR3 CCIFG 08h = Interrupt Source: Capture/compare 4; Interrupt Flag: TAxCCR4 CCIFG 0Ah = Interrupt Source: Capture/compare 5; Interrupt Flag: TAxCCR5 CCIFG 0Ch = Interrupt Source: Capture/compare 6; Interrupt Flag: TAxCCR6 CCIFG 0Eh = Interrupt Source: Timer overflow; Interrupt Flag: TAxCTL TAIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

13.3.6 TAxEX0 Register

Timer_Ax Expansion 0 Register

Figure 13-21. TAxEX0 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved					TAIDEX(1)		
r0	r0	r0	r0	r0	rw-(0)	rw-(0)	rw-(0)

(1) After programming TAIDEX bits and configuration of the timer, set TACLR bit to ensure proper reset of the timer divider logic.

Table 13-9. TAxEX0 Register Description

Bit	Field	Type	Reset	Description
15-3	Reserved	R	0h	Reserved. Reads as 0.
2-0	TAIDEX	RW	0h	Input divider expansion. These bits along with the ID bits select the divider for the input clock. 000b = Divide by 1 001b = Divide by 2 010b = Divide by 3 011b = Divide by 4 100b = Divide by 5 101b = Divide by 6 110b = Divide by 7 111b = Divide by 8

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Real-Time Clock (RTC) Overview

14.1 RTC Overview

Table 14-1. RTC Overview

Feature	RTC_A	RTC_D LPM3.5, Calendar and Counter Mode
Calendar Mode	Yes	Yes
Counter Mode	Yes	Yes
Programmable Alarms	Yes	Yes
Password Protected Calendar Registers	No	No
Input Clocks	ALCK, SMCLK	32-kHz crystal oscillator
LPM3.5 Support	No	Yes
Offset Calibration Register	Yes	Yes
Temperature Compensation Register	No	No
Frequency Adjustment Range	-2.035ppm × 63 ≈ -128ppm +4.069ppm × 63 ≈ +256 ppm	-2.17ppm × 59 ≈ -128ppm +4.34ppm × 59 ≈ +256 ppm
Frequency Adjustment Steps	-2.035 ppm, +4.069 pmm	-2.17 ppm, +4.34 pmm
Temperature Compensation	With software, manipulating offset calibration value	With software, manipulating offset calibration value
Calibration and Compensation Period	64 min	60 min
BCD to Binary Conversion	Integrated for Calendar Mode	Integrated for Calendar Mode plus separate conversion registers

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Real-Time Clock (RTC_A)

The Real-Time Clock (RTC_A) module provides clock counters with a calendar, a flexible programmable alarm, and calibration. This chapter describes the RTC_A module.

Topic **Page**

15.1 RTC_A Introduction **420**
15.2 RTC_A Operation **422**
15.3 RTC_A Registers **428**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.1 RTC_A Introduction

The RTC_A module provides a real-time clock and calendar function that can also be configured as a general-purpose counter.

RTC_A features include:

- Configurable for real-time clock with calendar function or general-purpose counter
- Provides seconds, minutes, hours, day of week, day of month, month, and year in real-time clock with calendar function
- Interrupt capability
- Selectable BCD or binary format in real-time clock mode
- Programmable alarms in real-time clock mode
- Calibration logic for time offset correction in real-time clock mode

The RTC_A block diagram is shown in Figure 15-1.

NOTE: Real-time clock initialization

Most RTC_A module registers have no initial condition. These registers must be configured by user software before use.

Copyright © 2009–2013, Texas Instruments Incorporated

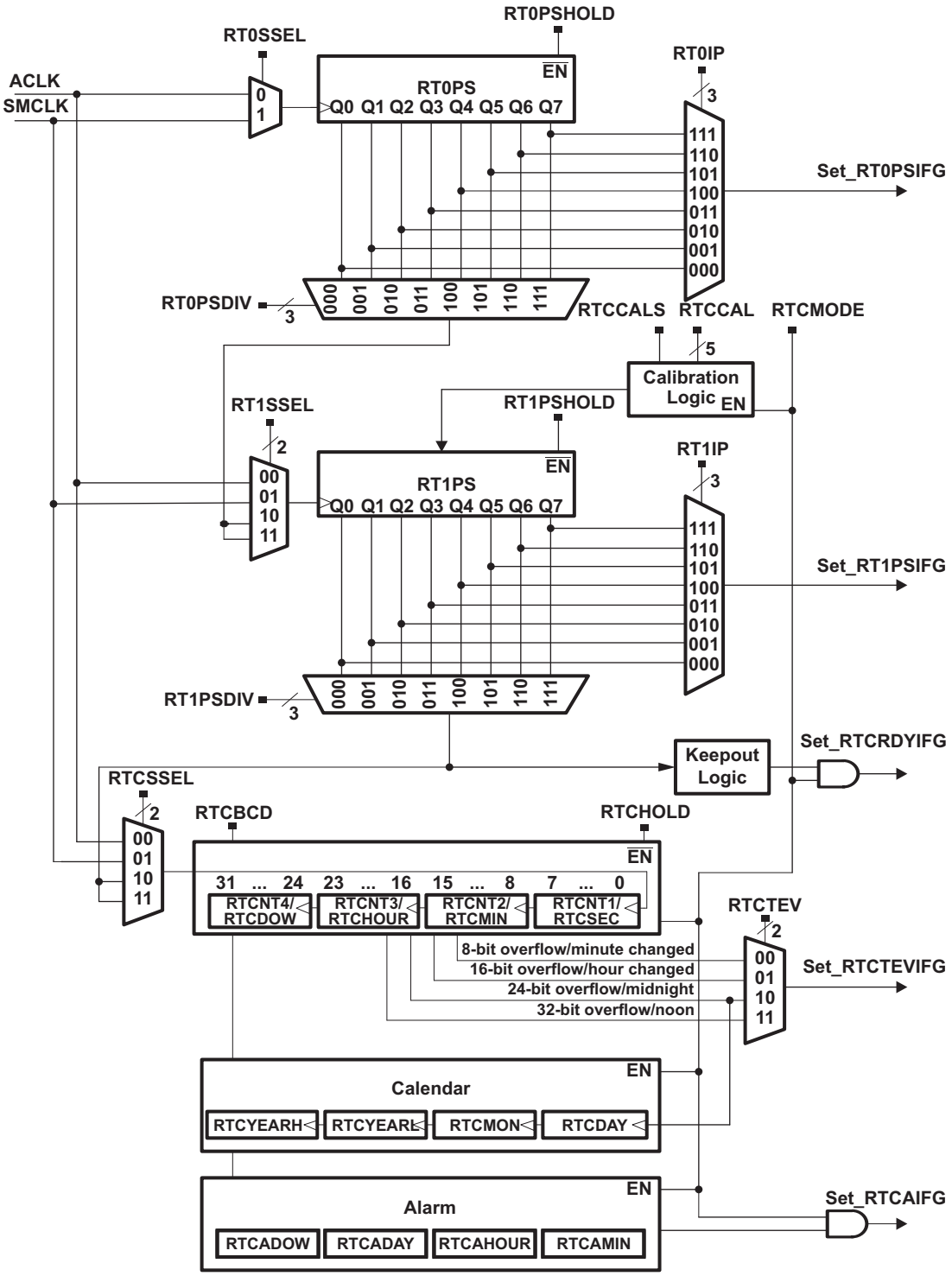


Figure 15-1. RTC_A

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.2 RTC_A Operation

The RTC_A module can be configured as a real-time clock with calendar function (calendar mode) or as a 32-bit general purpose counter (counter mode) with the RTCMODE bit.

15.2.1 Counter Mode

Counter mode is selected when RTCMODE is reset. In this mode, a 32-bit counter is provided that is directly accessible by software. Switching from calendar mode to counter mode resets the count value (RTCNT1, RTCNT2, RTCNT3, RTCNT4), as well as the prescale counters (RT0PS, RT1PS).

The clock to increment the counter can be sourced from ACLK, SMCLK, or prescaled versions of ACLK or SMCLK. Prescaled versions of ACLK or SMCLK are sourced from the prescale dividers (RT0PS and RT1PS). RT0PS and RT1PS output /2, /4, /8, 16, /32, /64, /128, and /256 versions of ACLK and SMCLK, respectively. The output of RT0PS can be cascaded with RT1PS. The cascaded output can be used as a clock source input to the 32-bit counter.

Four individual 8-bit counters are cascaded to provide the 32-bit counter. This provides 8-bit, 16-bit, 24-bit, or 32-bit overflow intervals of the counter clock. The RTCTEV bits select the respective trigger event. An RTCTEV event can trigger an interrupt by setting the RTCTEVIE bit. Each counter, RTCNT1 through RTCNT4, is individually accessible and may be written to.

RT0PS and RT1PS can be configured as two 8-bit counters or cascaded into a single 16-bit counter. RT0PS and RT1PS can be halted on an individual basis by setting their respective RT0PSHOLD and RT1PSHOLD bits. When RT0PS is cascaded with RT1PS, setting RT0PSHOLD causes both RT0PS and RT1PS to be halted. The 32-bit counter can be halted several ways depending on the configuration. If the 32-bit counter is sourced directly from ACLK or SMCLK, it can be halted by setting RTCHOLD. If it is sourced from the output of RT1PS, it can be halted by setting RT1PSHOLD or RTCHOLD. Finally, if it is sourced from the cascaded outputs of RT0PS and RT1PS, it can be halted by setting RT0PSHOLD, RT1PSHOLD, or RTCHOLD.

NOTE: Accessing the RTCNT1, RTCNT2, RTCNT3, RTCNT4, RT0PS, RT1PS registers

When the counter clock is asynchronous to the CPU clock, any read from any RTCNT1, RTCNT2, RTCNT3, RTCNT4, RT0PS, or RT1PS register should occur while the counter is not operating. Otherwise, the results may be unpredictable. Alternatively, the counter may be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Any write to these registers takes effect immediately.

15.2.2 Calendar Mode

Calendar mode is selected when RTCMODE is set. In calendar mode, the RTC_A module provides seconds, minutes, hours, day of week, day of month, month, and year in selectable BCD or hexadecimal format. The calendar includes a leap-year algorithm that considers all years evenly divisible by four as leap years. This algorithm is accurate from the year 1901 through 2099.

15.2.2.1 Real-Time Clock and Prescale Dividers

The prescale dividers, RT0PS and RT1PS, are automatically configured to provide a 1-s clock interval for the RTC_A. RT0PS is sourced from ACLK. ACLK must be set to 32768 Hz (nominal) for proper RTC_A calendar operation. RT1PS is cascaded with the output ACLK/256 of RT0PS. The RTC_A is sourced with the /128 output of RT1PS, thereby providing the required 1-s interval. Switching from counter to calendar mode clears the seconds, minutes, hours, day-of-week, and year counts and sets day-of-month and month counts to 1. In addition, RT0PS and RT1PS are cleared.

When RTCBCD = 1, BCD format is selected for the calendar registers. The format must be selected before the time is set. Changing the state of RTCBCD clears the seconds, minutes, hours, day-of-week, and year counts and sets day-of-month and month counts to 1. In addition, RT0PS and RT1PS are cleared.

Copyright © 2009–2013, Texas Instruments Incorporated

In calendar mode, the RT0SSEL, RT1SSEL, RT0PSDIV, RT1PSDIV, RT0PSHOLD, RT1PSHOLD, and RTCSSEL bits are don't care. Setting RTCHOLD halts the real-time counters and prescale counters, RT0PS and RT1PS.

15.2.2.2 Real-Time Clock Alarm Function

The RTC_A module provides for a flexible alarm system. There is a single user-programmable alarm that can be programmed based on the settings contained in the alarm registers for minutes, hours, day of week, and day of month. The user-programmable alarm function is only available in the calendar mode of operation.

Each alarm register contains an alarm enable (AE) bit that can be used to enable the respective alarm register. By setting AE bits of the various alarm registers, a variety of alarm events can be generated.

- Example 1: A user wishes to set an alarm every hour at 15 minutes past the hour; that is, at 00:15:00, 01:15:00, 02:15:00, and so on. This is possible by setting RTCAMIN to 15. By setting the AE bit of the RTCAMIN and clearing all other AE bits of the alarm registers, the alarm is enabled. When enabled, the AF is set when the count transitions from 00:14:59 to 00:15:00, 01:14:59 to 01:15:00, 02:14:59 to 02:15:00, etc.
- Example 2: A user wishes to set an alarm every day at 04:00:00. This is possible by setting RTCAHOUR to 4. By setting the AE bit of the RTCHOUR and clearing all other AE bits of the alarm registers, the alarm is enabled. When enabled, the AF is set when the count transitions from 03:59:59 to 04:00:00.
- Example 3: A user wishes to set an alarm for 06:30:00. RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCAHOUR and RTCAMIN, the alarm is enabled. Once enabled, the AF is set when the the time count transitions from 06:29:59 to 06:30:00. In this case, the alarm event occurs every day at 06:30:00.
- Example 4: A user wishes to set an alarm every Tuesday at 06:30:00. RTCADOW would be set to 2, RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCADOW, RTCAHOUR and RTCAMIN, the alarm is enabled. Once enabled, the AF is set when the the time count transitions from 06:29:59 to 06:30:00 and the RTCDOW transitions from 1 to 2.
- Example 5: A user wishes to set an alarm the fifth day of each month at 06:30:00. RTCADAY would be set to 5, RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCADAY, RTCAHOUR and RTCAMIN, the alarm is enabled. Once enabled, the AF is set when the the time count transitions from 06:29:59 to 06:30:00 and the RTCDAY equals 5.

NOTE: Invalid alarm settings

Invalid alarm settings are not checked via hardware. It is the user's responsibility to ensure that valid alarm settings are entered.

NOTE: Invalid time and date values

Writing of invalid date and/or time information or data values outside the legal ranges specified in the RTCSEC, RTCMIN, RTCHOUR, RTCDAY, RTCDOW, RTCYEARH, RTCYEARL, RTCAMIN, RTCAHOUR, RTCADAY, and RTCADOW registers can result in unpredictable behavior.

NOTE: Setting the alarm

To prevent potential erroneous alarm conditions from occurring, the alarms should be disabled by clearing the RTCAIE, RTCAIFG, and AE bits prior to writing new time values to the RTC time registers.

15.2.2.3 Reading or Writing Real-Time Clock Registers in Calendar Mode

Because the system clock may be asynchronous to the RTC_A clock source, special care must be taken when accessing the real-time clock registers.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

In calendar mode, the real-time clock registers are updated once per second. To prevent reading any real-time clock register at the time of an update, which could result in an invalid time being read, a keepout window is provided. The keepout window is centered approximately -128/32768 s around the update transition. The read-only RTCRDY bit is reset during the keepout window period and set outside the keepout the window period. Any read of the clock registers while RTCRDY is reset is considered to be potentially invalid, and the time read should be ignored.

An easy way to safely read the real-time clock registers is to use the RTCRDYIFG interrupt flag. Setting RTCRDYIE enables the RTCRDYIFG interrupt. Once enabled, an interrupt is generated based on the rising edge of the RTCRDY bit, causing the RTCRDYIFG to be set. At this point, the application has nearly a complete second to safely read any or all of the real-time clock registers. This synchronization process prevents reading the time value during transition. The RTCRDYIFG flag is reset automatically when the interrupt is serviced, or can be reset with software.

In counter mode, the RTCRDY bit remains reset. RTCRDYIE is a don't care and RTCRDYIFG remains reset.

NOTE: Reading or writing real-time clock registers

When the counter clock is asynchronous to the CPU clock, any read from any RTCSEC, RTCMIN, RTCHOUR, RTCDOW, RTCDAY, RTCMON, RTCYEARL, or RTCYEARH register while the RTCRDY is reset may result in invalid data being read. To safely read the counting registers, either polling of the RTCRDY bit or the synchronization procedure previously described can be used. Alternatively, the counter register can be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Reading the RT0PS and RT1PS can only be handled by reading the registers multiple times and a majority vote taken in software to determine the correct reading or by halting the counters.

Any write to any counting register takes effect immediately. However, the clock is stopped during the write. In addition, RT0PS and RT1PS registers are reset. This could result in losing up to 1 s during a write. Writing of data outside the legal ranges or invalid time stamp combinations results in unpredictable behavior.

15.2.3 Real-Time Clock Interrupts

The RTC_A module has five interrupt sources available, each with independent enables and flags.

15.2.3.1 Real-Time Clock Interrupts in Calendar Mode

In calendar mode, five sources for interrupts are available, namely RT0PSIFG, RT1PSIFG, RTCRDYIFG, RTCTEVIFG, and RTCAIFG. These flags are prioritized and combined to source a single interrupt vector. The interrupt vector register (RTCIV) is used to determine which flag requested an interrupt.

The highest-priority enabled interrupt generates a number in the RTCIV register (see register description). This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled RTC interrupts do not affect the RTCIV value.

Any access, read or write, of the RTCIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. In addition, all flags can be cleared via software.

The user-programmable alarm event sources the real-time clock interrupt, RTCAIFG. Setting RTCAIE enables the interrupt. In addition to the user-programmable alarm, the RTC_A module provides for an interval alarm that sources real-time clock interrupt, RTCTEVIFG. The interval alarm can be selected to cause an alarm event when RTCMIN changed or RTCHOUR changed, every day at midnight (00:00:00) or every day at noon (12:00:00). The event is selectable with the RTCTEV bits. Setting the RTCTEVIE bit enables the interrupt.

The RTCRDY bit sources the real-time clock interrupt, RTCRDYIFG, and is useful in synchronizing the read of time registers with the system clock. Setting the RTCRDYIE bit enables the interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

RT0PSIFG can be used to generate interrupt intervals selectable by the RT0IP bits. In calendar mode, RT0PS is sourced with ACLK at 32768 Hz, so intervals of 16384 Hz, 8192 Hz, 4096 Hz, 2048 Hz, 1024 Hz, 512 Hz, 256 Hz, or 128 Hz are possible. Setting the RT0PSIE bit enables the interrupt.

RT1PSIFG can generate interrupt intervals selectable by the RT1IP bits. In calendar mode, RT1PS is sourced with the output of RT0PS, which is 128 Hz (32768/256 Hz). Therefore, intervals of 64 Hz, 32 Hz, 16 Hz, 8 Hz, 4 Hz, 2 Hz, 1 Hz, or 0.5 Hz are possible. Setting the RT1PSIE bit enables the interrupt.

15.2.3.2 Real-Time Clock Interrupts in Counter Mode

In counter mode, three interrupt sources are available: RT0PSIFG, RT1PSIFG, and RTCTEVIFG. RTCAIFG and RTCRDYIFG are cleared. RTCRDYIE and RTCAIE are don't care.

RT0PSIFG can be used to generate interrupt intervals selectable by the RT0IP bits. In counter mode, RT0PS is sourced with ACLK or SMCLK, so divide ratios of /2, /4, /8, /16, /32, /64, /128, and /256 of the respective clock source are possible. Setting the RT0PSIE bit enables the interrupt.

RT1PSIFG can be used to generate interrupt intervals selectable by the RT1IP bits. In counter mode, RT1PS is sourced with ACLK, SMCLK, or the output of RT0PS, so divide ratios of /2, /4, /8, /16, /32, /64, /128, and /256 of the respective clock source are possible. Setting the RT1PSIE bit enables the interrupt.

The RTC_A module provides for an interval timer that sources real-time clock interrupt, RTCTEVIFG. The interval timer can be selected to cause an interrupt event when an 8-bit, 16-bit, 24-bit, or 32-bit overflow occurs within the 32-bit counter. The event is selectable with the RTCTEV bits. Setting the RTCTEVIE bit enables the interrupt.

15.2.3.2.1 RTCIV Software Example

The following software example shows the recommended use of RTCIV and the handling overhead. The RTCIV value is added to the PC to automatically jump to the appropriate routine.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself.

```
; Interrupt handler for RTC interrupt flags.              Cycles
RTC_HND                         ; Interrupt latency           6
         ADD   &RTCIV,PC        ; Add offset to Jump table    3
         RETI                   ; Vector 0: No interrupt      5
         JMP   RTCRDYIFG_HND    ; Vector 2: RTCRDYIFG         2
         JMP   RTCTEVIFG_HND    ; Vector 4: RTCTEVIFG         2
         JMP   RTCAIFG          ; Vector 6: RTCAIFG           5
         JMP   RT0PSIFG         ; Vector 8: RT0PSIFG          5
         JMP   RT1PSIFG         ; Vector A: RT1PSIFG          5
         RETI                   ; Vector C: Reserved          5
RTCRDYIFG_HND                   ; Vector 2: RTCRDYIFG Flag
          to                     ; Task starts here
         RETI                                                 5
RTCTEVIFG_HND                   ; Vector 4: RTCTEVIFG
          to                     ; Task starts here
         RETI                   ; Back to main program        5
RTCAIFG_HND                     ; Vector 6: RTCAIFG
          to                     ; Task starts here
RT0PSIFG_HND                    ; Vector 8: RT0PSIFG
          to                     ; Task starts here
RT1PSIFG_HND                    ; Vector A: RT1PSIFG
          to                     ; Task starts here
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.2.4 Real-Time Clock Calibration

The RTC_A module has calibration logic that allows for adjusting the crystal frequency in approximately +4-ppm or –2-ppm steps, allowing for higher time keeping accuracy from standard crystals.The RTCCAL bits are used to adjust the frequency. When RTCCALS is set, each RTCCAL LSB causes a ≈ +4-ppm adjustment. When RTCCALS is cleared, each RTCCAL LSB causes a ≈ –2-ppm adjustment. Calibration is available only in calendar mode. In counter mode (RTCMODE = 0), the calibration logic is disabled.

Calibration is accomplished by periodically adjusting the RT1PS counter based on the RTCCALS and RTCCALx settings. In calendar mode, the RT0PS divides the nominial 37268-Hz low-frequency (LF) crystal clock input by 256. A 64-minute period has 32768 cycles/sec × 60 sec/min × 64 min = 125829120 cycles. Therefore a –2-ppm reduction in frequency (down calibration) approximately equates to adding an additional 256 cycles every 125829120 cycles (256/125829120 = 2.035 ppm). This is accomplished by holding the RT1PS counter for one additional clock of the RT0PS output within a 64-minute period. Similary, a +4-ppm increase in frequency (up calibration) approximately equates to removing 512 cycles every 125829120 cycle (512/125829120 = 4.069 ppm). This is accomplished by incrementing the RT1PS counter for two additional clocks of the RT0PS output within a 64-minute period. Each RTCCALx calibration bit causes either 256 LF crystal clock cycles to be added every 64 minutes or 512 LF crystal clock cycles to be subtracted every 64 minutes, giving a frequency adjustment of approximately –2 ppm or +4 ppm, respectively.

To calibrate the frequency, the RTCCLK output signal is available at a pin. The RTCCALF bits can be used to select the frequency rate of the RTCCLK output signal, either no signal, 512 Hz, 256 Hz, or 1 Hz.

The basic flow to calibrate the frequency is as follows:

1. Configure the RTCCLK pin.
2. Measure the RTCCLK output signal with an appropriate resolution frequency counter; that is, within the resolution required.
3. Compute the absolute error in ppm: Absolute Error (ppm) = $|10^6 \times (f_{MEASURED} – f_{RTCCLK}) / f_{RTCCLK}|$, where f_{RTCCLK} is the expected frequency of 512 Hz, 256 Hz, or 1 Hz.
4. Adjust the frequency, by performing the following:
   (a) If the frequency is too low, set RTCALS = 1 and apply the appropriate RTCCALx bits, where RTCCALx = (Absolute Error) / 4.069, rounded to the nearest integer.
   (b) If the frequency is too high, clear RTCALS = 0 and apply the appropriate RTCCALx bits, where RTCCALx = (Absolute Error) / 2.035, rounded to the nearest integer.

For example, assume that RTCCLK is output at a frequency of 512 Hz. The measured RTCCLK is 511.9658 Hz. The frequency error is approximately 66.8 ppm low. To increase the frequency by 66.8 ppm, RTCCALS would be set, and RTCCAL would be set to 16 (66.8/4.069). Similarly, assume that the measured RTCCLK is 512.0125 Hz. The frequency error is approximately 24.4 ppm high. To decrease the frequency by 24.4 ppm, RTCCALS would be cleared, and RTCCAL would be set to 12 (24.4 / 2.035).

The calibration corrects only initial offsets and does not adjust for temperature and aging effects. This can be handled by periodically measuring temperature and using the crystal's charateristic curve to adjust the ppm based on temperature as required. In counter mode (RTCMODE = 0), the calibration logic is disabled.

NOTE: Minimum Possible Calibration

The minimial calibration possible is -4 ppm or +8 ppm. For example, setting RTCCALS = 0 and RTCCAL = 0h would result in a -4 ppm decrease in frequency. Similarly, setting RTCCALS = 1 and RTCCAL = 0h would result in a +8 ppm increase in frequency.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: Calibration output frequency

The 512-Hz and 256-Hz output frequencies observed at the RTCCLK pin are not affected by changes in the calibration settings since these output frequencies are generated prior to the calibration logic. The 1-Hz output frequency is affected by changes in the calibration settings. Because the frequency change is small and infrequent over a very long time interval, it can be difficult to observe.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3 RTC_A Registers

The RTC_A module registers are listed in and Table 15-1. The base register for the RTC_A module registers can be found in the device-specific data sheet. The address offsets are given in Table 15-1.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 15-1. RTC_A Registers

Offset	Acronym	Register Name	Type	Access	Reset
00h	RTCCTL01	Real-Time Clock Control 0, 1	Read/write	Word	4000h
00h	RTCCTL0	Real-Time Clock Control 0	Read/write	Byte	00h
	or RTCCTL01_L				
01h	RTCCTL1	Real-Time Clock Control 1	Read/write	Byte	40h
	or RTCCTL01_H				
02h	RTCCTL23	Real-Time Clock Control 2, 3	Read/write	Word	0000h
02h	RTCCTL2	Real-Time Clock Control 2	Read/write	Byte	00h
	or RTCCTL23_L				
03h	RTCCTL3	Real-Time Clock Control 3	Read/write	Byte	00h
	or RTCCTL23_H				
08h	RTCPS0CTL	Real-Time Prescale Timer 0 Control	Read/write	Word	0100h
08h	RTCPS0CTLL		Read/write	Byte	00h
	or RTCPS0CTL_L				
09h	RTCPS0CTLH		Read/write	Byte	01h
	or RTCPS0CTL_H				
0Ah	RTCPS1CTL	Real-Time Prescale Timer 1 Control	Read/write	Word	0100h
0Ah	RTCPS1CTLL		Read/write	Byte	00h
	or RTCPS1CTL_L				
0Bh	RTCPS0CTLH		Read/write	Byte	01h
	or RTCPS0CTL_H				
0Ch	RTCPS	Real-Time Prescale Timer 0, 1 Counter	Read/write	Word	undefined
0Ch	RT0PS	Real-Time Prescale Timer 0 Counter	Read/write	Byte	undefined
	or RTCPS_L				
0Dh	RT1PS	Real-Time Prescale Timer 1 Counter	Read/write	Byte	undefined
	or RTCPS_H				
0Eh	RTCIV	Real Time Clock Interrupt Vector	Read	Word	0000h
0Eh	RTCIV_L		Read	Byte	00h
0Fh	RTCIV_H		Read	Byte	00h
10h	RTCTIM0	Real-Time Clock Seconds, Minutes	Read/write	Word	undefined
	or RTCNT12	Real-Time Counter 1, 2			
10h	RTCSEC	Real-Time Clock Seconds	Read/write	Byte	undefined
	RTCNT1	Real-Time Counter 1			
	or RTCTIM0_L				
11h	RTCMIN	Real-Time Clock Minutes	Read/write	Byte	undefined
	RTCNT2	Real-Time Counter 2			
	or RTCTIM0_H				
12h	RTCTIM1	Real-Time Clock Hour, Day of Week	Read/write	Word	undefined
	or RTCNT34	Real-Time Counter 3, 4			
12h	RTCHOUR	Real-Time Clock Hour	Read/write	Byte	undefined

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 15-1. RTC_A Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset
	RTCNT3	Real-Time Counter 3			
	or RTCTIM1_L				
13h	RTCDOW	Real-Time Clock Day of Week	Read/write	Byte	undefined
	RTCNT4	Real-Time Counter 4			
	or RTCTIM1_H				
14h	RTCDATE	Real-Time Clock Date	Read/write	Word	undefined
14h	RTCDAY	Real-Time Clock Day of Month	Read/write	Byte	undefined
	or RTCDATE_L				
15h	RTCMON	Real-Time Clock Month	Read/write	Byte	undefined
	or RTCDATE_H				
16h	RTCYEAR	Real-Time Clock Year	Read/write	Word	undefined
16h	RTCYEARL		Read/write	Byte	undefined
	or RTCYEAR_L				
17h	RTCYEARH		Read/write	Byte	undefined
	or RTCYEAR_H				
18h	RTCAMINHR	Real-Time Clock Minutes, Hour Alarm	Read/write	Word	undefined
18h	RTCAMIN	Real-Time Clock Minutes Alarm	Read/write	Byte	undefined
	or RTCAMINHR_L				
19h	RTCAHOUR	Real-Time Clock Hours Alarm	Read/write	Byte	undefined
	or RTCAMINHR_H				
1Ah	RTCADOWDAY	Real-Time Clock Day of Week, Day of Month Alarm	Read/write	Word	undefined
1Ah	RTCADOW	Real-Time Clock Day of Week Alarm	Read/write	Byte	undefined
	or RTCADOWDAY_L				
1Bh	RTCADAY	Real-Time Clock Day of Month Alarm	Read/write	Byte	undefined
	or RTCADOWDAY_H				

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.1 RTCCTL0 Register

Real-Time Clock Control 0 Register

Figure 15-2. RTCCTL0 Register

7	6	5	4	3	2	1	0
Reserved	RTCTEVIE	RTCAIE	RTCRDYIE	Reserved	RTCTEVIFG	RTCAIFG	RTCRDYIFG
r0	rw-0	rw-0	rw-0	r0	rw-(0)	rw-(0)	rw-(0)

Table 15-2. RTCCTL0 Register Description

Bit	Field	Type	Reset	Description
7	Reserved	R	0h	Reserved. Always reads as 0.
6	RTCTEVIE	RW	0h	Real-time clock time event interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled
5	RTCAIE	RW	0h	Real-time clock alarm interrupt enable. This bit remains cleared when in counter mode (RTCMODE = 0). 0b = Interrupt not enabled 1b = Interrupt enabled
4	RTCRDYIE	RW	0h	Real-time clock read ready interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled
3	Reserved	R	0h	Reserved. Always reads as 0.
2	RTCTEVIFG	RW	0h	Real-time clock time event flag 0b = No time event occurred. 1b = Time event occurred.
1	RTCAIFG	RW	0h	Real-time clock alarm flag. This bit remains cleared when in counter mode (RTCMODE = 0). 0b = No time event occurred. 1b = Time event occurred.
0	RTCRDYIFG	RW	0h	Real-time clock read ready flag 0b = RTC cannot be read safely. 1b = RTC can be read safely.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.2 RTCCTL1 Register

Real-Time Clock Control Register 1

Figure 15-3. RTCCTL1 Register

7	6	5	4	3	2	1	0
RTCBCD	RTCHOLD	RTCMODE	RTCRDY	RTCSSEL		RTCTEV	
rw-(0)	rw-(1)	rw-(0)	r-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 15-3. RTCCTL1 Register Description

Bit	Field	Type	Reset	Description
7	RTCBCD	RW	0h	Real-time clock BCD select. Selects BCD counting for real-time clock. Applies to calendar mode (RTCMODE = 1) only; setting is ignored in counter mode. Changing this bit clears seconds, minutes, hours, day of week, and year to 0 and sets day of month and month to 1. The real-time clock registers must be set by software afterwards. 0b = Binary (hexadecimal) code selected 1b = Binary coded decimal (BCD) code selected
6	RTCHOLD	RW	1h	Real-time clock hold 0b = Real-time clock (32-bit counter or calendar mode) is operational. 1b = In counter mode (RTCMODE = 0), only the 32-bit counter is stopped. In calendar mode (RTCMODE = 1), the calendar is stopped as well as the prescale counters, RT0PS and RT1PS. RT0PSHOLD and RT1PSHOLD are don't care.
5	RTCMODE	RW	0h	Real-time clock mode 0b = 32-bit counter mode 1b = Calendar mode. Switching between counter and calendar mode resets the real-time clock counter registers. Switching to calendar mode clears seconds, minutes, hours, day of week, and year to 0 and sets day of month and month to 1. The real-time clock registers must be set by software afterwards. RT0PS and RT1PS are also cleared.
4	RTCRDY	RW	0h	Real-time clock ready 0b = RTC time values in transition (calendar mode only) 1b = RTC time values safe for reading (calendar mode only). This bit indicates when the real-time clock time values are safe for reading (calendar mode only). In counter mode, RTCRDY signal remains cleared.
3-2	RTCSSEL	RW	0h	Real-time clock source select. Selects clock input source to the RTC/32-bit counter. In calendar mode, these bits are don't care. The clock input is automatically set to the output of RT1PS. 00b = ACLK 01b = SMCLK 10b = Output from RT1PS 11b = Output from RT1PS
1-0	RTCTEV	RW	0h	Real-time clock time event Counter mode (RTCMODE = 0) 00b = 8-bit overflow 01b = 16-bit overflow 10b = 24-bit overflow 11b = 32-bit overflow Calendar mode (RTCMODE = 1) 00b = Minute changed 01b = Hour changed 10b = Every day at midnight (00:00) 11b = Every day at noon (12:00)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.3 RTCCTL2 Register

Real-Time Clock Control 2 Register

Figure 15-4. RTCCTL2 Register

7	6	5	4	3	2	1	0
RTCCALS	Reserved	RTCCAL					
rw-(0)	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 15-4. RTCCTL2 Register Description

Bit	Field	Type	Reset	Description
7	RTCCALS	RW	0h	Real-time clock calibration sign 0b = Frequency adjusted down 1b = Frequency adjusted up
6	Reserved	R	0h	Reserved. Always reads as 0.
5-0	RTCCAL	RW	0h	Real-time clock calibration. Each LSB represents approximately +4ppm (RTCCALS = 1) or a –2ppm (RTCCALS = 0) adjustment in frequency.

15.3.4 RTCCTL3 Register

Real-Time Clock Control 3 Register

Figure 15-5. RTCCTL3 Register

7	6	5	4	3	2	1	0
Reserved						RTCCALF	
r0	r0	r0	r0	r0	r0	rw-(0)	rw-(0)

Table 15-5. RTCCTL3 Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1-0	RTCCALF	RW	0h	Real-time clock calibration frequency. Selects frequency output to RTCCLK pin for calibration measurement. The corresponding port must be configured for the peripheral module function. The RTCCLK is not available in counter mode and remains low, and the RTCCALF bits are don't care. 00b = No frequency output to RTCCLK pin 01b = 512 Hz 10b = 256 Hz 11b = 1 Hz

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.5 RTCNT1 Register

Real-Time Clock Counter 1 Register – Counter Mode

Figure 15-6. RTCNT1 Register

7	6	5	4	3	2	1	0
RTCNT1							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-6. RTCNT1 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT1	RW	undefined	The RTCNT1 register is the count of RTCNT1

15.3.6 RTCNT2 Register

Real-Time Clock Counter 2 Register – Counter Mode

Figure 15-7. RTCNT2 Register

7	6	5	4	3	2	1	0
RTCNT2							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-7. RTCNT2 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT2	RW	undefined	The RTCNT2 register is the count of RTCNT2

15.3.7 RTCNT3 Register

Real-Time Clock Counter 3 Register – Counter Mode

Figure 15-8. RTCNT3 Register

7	6	5	4	3	2	1	0
RTCNT3							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-8. RTCNT3 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT3	RW	undefined	The RTCNT3 register is the count of RTCNT3

15.3.8 RTCNT4 Register

Real-Time Clock Counter 4 Register – Counter Mode

Figure 15-9. RTCNT4 Register

7	6	5	4	3	2	1	0
RTCNT4							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-9. RTCNT4 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT4	RW	undefined	The RTCNT4 register is the count of RTCNT4.

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.9 RTCSEC Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Seconds Register – Calendar Mode With Hexadecimal Format

Figure 15-10. RTCSEC Register

7	6	5	4	3	2	1	0
0		Seconds					
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 15-10. RTCSEC Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always 0
5-0	Seconds	RW	undefined	Seconds (0 to 59)

15.3.10 RTCSEC Register – Calendar Mode With BCD Format

Real-Time Clock Seconds Register – Calendar Mode With BCD Format

Figure 15-11. RTCSEC Register

7	6	5	4	3	2	1	0
0	Seconds – high digit			Seconds – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw

Table 15-11. RTCSEC Register Description

Bit	Field	Type	Reset	Description
7	0	R	0h	Always 0
6-4	Seconds – high digit	RW	undefined	Seconds – high digit (0 to 5)
3-0	Seconds – low digit	RW	undefined	Seconds – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.11 RTCMIN Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Minutes Register – Calendar Mode With Hexadecimal Format

Figure 15-12. RTCMIN Register

7	6	5	4	3	2	1	0
0		Minutes					
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 15-12. RTCMIN Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always 0
5-0	Minutes	RW	undefined	Minutes (0 to 59)

15.3.12 RTCMIN Register – Calendar Mode With BCD Format

Real-Time Clock Minutes Register – Calendar Mode With BCD Format

Figure 15-13. RTCMIN Register

7	6	5	4	3	2	1	0
0	Minutes – high digit			Minutes – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw

Table 15-13. RTCMIN Register Description

Bit	Field	Type	Reset	Description
7	0	R	0h	Always 0
6-4	Minutes – high digit	RW	undefined	Minutes – high digit (0 to 5)
3-0	Minutes – low digit	RW	undefined	Minutes – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.13 RTCHOUR Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Hours Register – Calendar Mode With Hexadecimal Format

Figure 15-14. RTCHOUR Register

7	6	5	4	3	2	1	0
0			Hours				
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 15-14. RTCHOUR Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always 0
4-0	Hours	RW	undefined	Hours (0 to 23)

15.3.14 RTCHOUR Register – Calendar Mode With BCD Format

Real-Time Clock Hours Register – Calendar Mode With BCD Format

Figure 15-15. RTCHOUR Register

7	6	5	4	3	2	1	0
0		Hours – high digit		Hours – low digit			
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 15-15. RTCHOUR Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always 0
5-4	Hours – high digit	RW	undefined	Hours – high digit (0 to 2)
3-0	Hours – low digit	RW	undefined	Hours – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.15 RTCDOW Register – Calendar Mode

Real-Time Clock Day of Week Register – Calendar Mode

Figure 15-16. RTCDOW Register

7	6	5	4	3	2	1	0
0					Day of week		
r-0	r-0	r-0	r-0	r-0	rw	rw	rw

Table 15-16. RTCDOW Register Description

Bit	Field	Type	Reset	Description
7-3	0	R	0h	Always 0
2-0	Day of week	RW	undefined	Day of week (0 to 6)

15.3.16 RTCDAY Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Day of Month Register – Calendar Mode With Hexadecimal Format

Figure 15-17. RTCDAY Register

7	6	5	4	3	2	1	0
0			Day of month				
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 15-17. RTCDAY Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always 0
4-0	Day of month	RW	undefined	Day of month (1 to 28, 29, 30, 31)

15.3.17 RTCDAY Register – Calendar Mode With BCD Format

Real-Time Clock Day of Month Register – Calendar Mode With BCD Format

Figure 15-18. RTCDAY Register

7	6	5	4	3	2	1	0
0		Day of month – high digit		Day of month – low digit			
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 15-18. RTCDAY Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	
5-4	Day of month – high digit	RW	undefined	Day of month – high digit (0 to 3)
3-0	Day of month – low digit	RW	undefined	Day of month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.18 RTCMON Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Month Register – Calendar Mode With Hexadecimal Format

Figure 15-19. RTCMON Register

7	6	5	4	3	2	1	0
0				Month			
r-0	r-0	r-0	r-0	rw	rw	rw	rw

Table 15-19. RTCMON Register Description

Bit	Field	Type	Reset	Description
7-4	0	R	0h	Always 0
3-0	Month	RW	undefined	Month (1 to 12)

15.3.19 RTCMON Register – Calendar Mode With BCD Format

Real-Time Clock Month Register – Calendar Mode With BCD Format

Figure 15-20. RTCMON Register

7	6	5	4	3	2	1	0
0			Month – high digit	Month – low digit			
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 15-20. RTCMON Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always 0
4	Month – high digit	RW	undefined	Month – high digit (0 or 1)
3-0	Month – low digit	RW	undefined	Month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.20 RTCYEARL Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Year Low-Byte Register – Calendar Mode With Hexadecimal Format

Figure 15-21. RTCYEARL Register

7	6	5	4	3	2	1	0
Year – low byte of 0 to 4095							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-21. RTCYEARL Register Description

Bit	Field	Type	Reset	Description
7-0	Year	RW	undefined	Year – low byte of 0 to 4095

15.3.21 RTCYEARL Register – Calendar Mode With BCD Format

Real-Time Clock Year Low-Byte Register – Calendar Mode With BCD Format

Figure 15-22. RTCYEARL Register

7	6	5	4	3	2	1	0
Decade				Year – lowest digit			
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-22. RTCYEARL Register Description

Bit	Field	Type	Reset	Description
7-4	Decade	RW	undefined	Decade (0 to 9)
3-0	Year – lowest digit	RW	undefined	Year – lowest digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.22 RTCYEARH Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Year High-Byte Register – Calendar Mode With Hexadecimal Format

Figure 15-23. RTCYEARH Register

7	6	5	4	3	2	1	0
0				Year – high byte of 0 to 4095			
r-0	r-0	r-0	r-0	rw	rw	rw	rw

Table 15-23. RTCYEARH Register Description

Bit	Field	Type	Reset	Description
7-4	0	R	0h	Always 0
3-0	Year	RW	undefined	Year – high byte of 0 to 4095

15.3.23 RTCYEARH Register – Calendar Mode With BCD Format

Real-Time Clock Year High-Byte Register – Calendar Mode With BCD Format

Figure 15-24. RTCYEARH Register

7	6	5	4	3	2	1	0
0	Century – high digit			Century – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw

Table 15-24. RTCYEARH Register Description

Bit	Field	Type	Reset	Description
7	0	R	0h	Always 0
6-4	Century – high digit	RW	undefined	Century – high digit (0 to 4)
3-0	Century – low digit	RW	undefined	Century – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.24 RTCAMIN Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Minutes Alarm Register – Calendar Mode With Hexadecimal Format

Figure 15-25. RTCAMIN Register

7	6	5	4	3	2	1	0
AE	0	Minutes					
rw	r-0	rw	rw	rw	rw	rw	rw

Table 15-25. RTCAMIN Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6	0	R	0h	Always 0
5-0	Minutes	RW	undefined	Minutes (0 to 59)

15.3.25 RTCAMIN Register – Calendar Mode With BCD Format

Real-Time Clock Minutes Alarm Register – Calendar Mode With BCD Format

Figure 15-26. RTCAMIN Register

7	6	5	4	3	2	1	0
AE	Minutes – high digit			Minutes – low digit			
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-26. RTCAMIN Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	0h	Alarm enable
6-4	Minutes – high digit	RW	undefined	Minutes – high digit (0 to 5)
3-0	Minutes – low digit	RW	undefined	Minutes – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.26 RTCAHOUR Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Hours Alarm Register – Calendar Mode With Hexadecimal Format

Figure 15-27. RTCAHOUR Register

7	6	5	4	3	2	1	0
AE	0		Hours				
rw	r-0	r-0	rw	rw	rw	rw	rw

Table 15-27. RTCAHOUR Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6-5	0	R	0h	Always 0
4-0	Hours	RW	undefined	Hours (0 to 23)

15.3.27 RTCAHOUR Register – Calendar Mode With BCD Format

Real-Time Clock Hours Alarm Register – Calendar Mode With BCD Format

Figure 15-28. RTCAHOUR Register

7	6	5	4	3	2	1	0
AE	0	Hours – high digit		Hours – low digit			
rw	r-0	rw	rw	rw	rw	rw	rw

Table 15-28. RTCAHOUR Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6	0	R	0h	Always 0
5-4	Hours – high digit	RW	undefined	Hours – high digit (0 to 2)
3-0	Hours – low digit	RW	undefined	Hours – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.28 RTCADOW Register

Real-Time Clock Day of Week Alarm Register – Calendar Mode

Figure 15-29. RTCADOW Register

7	6	5	4	3	2	1	0
AE	0				Day of week		
rw	r-0	r-0	r-0	r-0	rw	rw	rw

Table 15-29. RTCADOW Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6-3	0	R	0h	Always 0
2-0	Day of week	RW	undefined	Day of week (0 to 6)

15.3.29 RTCADAY Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Day of Month Alarm Register – Calendar Mode With Hexadecimal Format

Figure 15-30. RTCADAY Register

7	6	5	4	3	2	1	0
AE	0		Day of month				
rw	r-0	r-0	rw	rw	rw	rw	rw

Table 15-30. RTCADAY Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6-5	0	R	0h	Always 0
4-0	Day of month	RW	undefined	Day of month (1 to 28, 29, 30, 31)

15.3.30 RTCADAY Register – Calendar Mode With BCD Format

Real-Time Clock Day of Month Alarm Register – Calendar Mode With BCD Format

Figure 15-31. RTCADAY Register

7	6	5	4	3	2	1	0
AE	0	Day of month – high digit		Day of month – low digit			
rw	r-0	rw	rw	rw	rw	rw	rw

Table 15-31. RTCADAY Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	Alarm enable
6	0	R	0h	Always 0
5-4	Day of month – high digit	RW	undefined	Day of month – high digit (0 to 3)
3-0	Day of month – low digit	RW	undefined	Day of month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.31 RTCPS0CTL Register

Real-Time Clock Prescale Timer 0 Control Register

Figure 15-32. RTCPS0CTL Register

15	14	13	12	11	10	9	8
Reserved	RT0SSEL	RT0PSDIV			Reserved		RT0PSHOLD
rw-0	rw-0	rw-0	rw-0	rw-0	r0	r0	rw-1
7	6	5	4	3	2	1	0
Reserved			RT0IP			RT0PSIE	RT0PSIFG
r0	r0	r0	rw-0	rw-0	rw-0	rw-0	rw-(0)

Table 15-32. RTCPS0CTL Register Description

Bit	Field	Type	Reset	Description
15	Reserved	R	0h	Reserved. Always reads as 0.
14	RT0SSEL	RW	0h	Prescale timer 0 clock source select. Selects clock input source to the RT0PS counter. In real-time clock calendar mode, these bits are do not care. RT0PS clock input is automatically set to the output of RT0PS. 0b = ACLK 1b = SMCLK
13-11	RT0PSDIV	RW	0h	Prescale timer 0 clock divide. These bits control the divide ratio of the RT0PS counter. In real-time clock calendar mode, these bits are don't care for RT0PS and RT1PS. RT0PS clock output is automatically set to /256. RT1PS clock output is automatically set to /128. 00b = Divide by 2 01b = Divide by 4 10b = Divide by 8 11b = Divide by 16 00b = Divide by 32 01b = Divide by 64 10b = Divide by 128 11b = Divide by 256
10-9	Reserved	R	0h	Reserved. Always reads as 0.
8	RT0PSHOLD	RW	1h	Prescale timer 0 hold. In real-time clock calendar mode, this bit is don't care. RT0PS is stopped via the RTCHOLD bit. 0b = RT0PS operational 1b = RT0PS held
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-2	RT0IP	RW	0h	Prescale timer 0 interrupt interval 00b = Divide by 2 01b = Divide by 4 10b = Divide by 8 11b = Divide by 16 00b = Divide by 32 01b = Divide by 64 10b = Divide by 128 11b = Divide by 256
1	RT0PSIE	RW	0h	Prescale timer 0 interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled
0	RT0PSIFG	RW	0h	Prescale timer 0 interrupt flag 0b = No time event occurred 1b = Time event occurred

Copyright © 2009–2013, Texas Instruments Incorporated

15.3.32 RTCPS1CTL Register

Real-Time Clock Prescale Timer 1 Control Register

Figure 15-33. RTCPS1CTL Register

15	14	13	12	11	10	9	8
RT1SSEL			RT1PSDIV		Reserved		RT1PSHOLD
rw-0	rw-0	rw-0	rw-0	rw-0	r0	r0	rw-1
7	6	5	4	3	2	1	0
Reserved			RT1IP			RT1PSIE	RT1PSIFG
r0	r0	r0	rw-0	rw-0	rw-0	rw-0	rw-(0)

Table 15-33. RTCPS1CTL Register Description

Bit	Field	Type	Reset	Description
15-14	RT1SSEL	RW	0h	Prescale timer 1 clock source select. Selects clock input source to the RT1PS counter. In real-time clock calendar mode, these bits are do not care. RT1PS clock input is automatically set to the output of RT0PS. 00b = ACLK 01b = SMCLK 10b = Output from RT0PS 11b = Output from RT0PS
13-11	RT1PSDIV	RW	0h	Prescale timer 1 clock divide. These bits control the divide ratio of the RT0PS counter. In real-time clock calendar mode, these bits are don't care for RT0PS and RT1PS. RT0PS clock output is automatically set to /256. RT1PS clock output is automatically set to /128. 00b = Divide by 2 01b = Divide by 4 10b = Divide by 8 11b = Divide by 16 00b = Divide by 32 01b = Divide by 64 10b = Divide by 128 11b = Divide by 256
10-9	Reserved	R	0h	Reserved. Always reads as 0.
8	RT1PSHOLD	RW	1h	Prescale timer 1 hold. In real-time clock calendar mode, this bit is don't care. RT1PS is stopped via the RTCHOLD bit. 0b = RT1PS operational 1b = RT1PS held
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-2	RT1IP	RW	0h	Prescale timer 1 interrupt interval 00b = Divide by 2 01b = Divide by 4 10b = Divide by 8 11b = Divide by 16 00b = Divide by 32 01b = Divide by 64 10b = Divide by 128 11b = Divide by 256
1	RT1PSIE	RW	0h	Prescale timer 1 interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled
0	RT1PSIFG	RW	0h	Prescale timer 1 interrupt flag 0b = No time event occurred 1b = Time event occurred

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

15.3.33 RT0PS Register

Real-Time Clock Prescale Timer 0 Counter Register

Figure 15-34. RT0PS Register

7	6	5	4	3	2	1	0
RT0PS							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-34. RT0PS Register Description

Bit	Field	Type	Reset	Description
7-0	RT0PS	RW	Undefined	Prescale timer 0 counter value

15.3.34 RT1PS Register

Real-Time Clock Prescale Timer 1 Counter Register

Figure 15-35. RTPS1 Register

7	6	5	4	3	2	1	0
RT1PS							
rw	rw	rw	rw	rw	rw	rw	rw

Table 15-35. RT1PS Register Description

Bit	Field	Type	Reset	Description
7-0	RT1PS	RW	Undefined	Prescale timer 1 counter value

15.3.35 RTCIV Register

Real-Time Clock Interrupt Vector Register

Figure 15-36. RTCIV Register

15	14	13	12	11	10	9	8
RTCIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
RTCIV							
r0	r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 15-36. RTCIV Register Description

Bit	Field	Type	Reset	Description
15-0	RTCIV	R	0h	Real-time clock interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: RTC ready; Interrupt Flag: RTCRDYIFG 04h = Interrupt Source: RTC interval timer; Interrupt Flag: RTCTEVIFG 06h = Interrupt Source: RTC user alarm; Interrupt Flag: RTCAIFG 08h = Interrupt Source: RTC prescaler 0; Interrupt Flag: RT0PSIFG 0Ah = Interrupt Source: RTC prescaler 1; Interrupt Flag: RT1PSIFG 0Ch = Reserved 0Eh = Reserved 10h = Reserved ; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Real-Time Clock D (RTC_D)

The real-time clock RTC_D module provides clock counters with calendar mode, a flexible programmable alarm, and calibration. The RTC_D also supports operation in LPMx.5. This chapter describes the RTC_D module.

Topic **Page**

16.1 Real-Time Clock RTC_D Introduction 448
16.2 RTC_D Operation 450
16.3 RTC_D Registers 457

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.1 Real-Time Clock RTC_D Introduction

The RTC_D module provides configurable clock counters.

RTC_D features include:

- Configurable for real-time clock with calendar function ar general-purpose counter
- Real-time clock and calendar mode providing seconds, minutes, hours, day of week, day of month, month, and year (including leap year correction)
- Interrupt capability
- Selectable BCD or binary format in real-time clock and calendar mode
- Programmable alarms in real-time clock and calendar mode
- Calibration logic for time offset correction in real-time clock and calendar mode
- Operation in LPMx.5

The RTC_D block diagram is shown in Figure 16-1.

NOTE: Real-time clock initialization

Most RTC_D module registers have no initial condition. These registers must be configured by user software before use.

Copyright © 2009–2013, Texas Instruments Incorporated

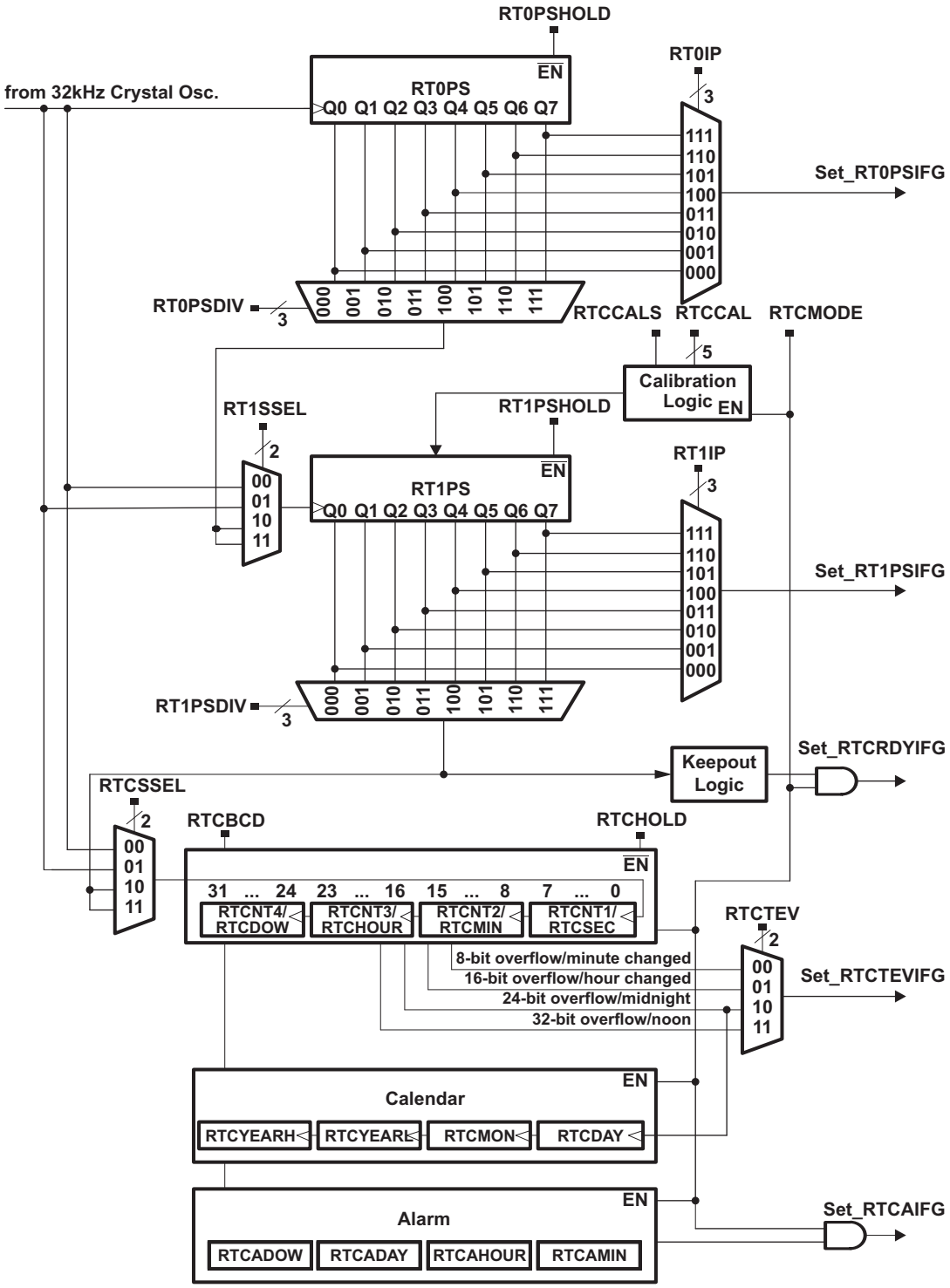


Figure 16-1. RTC_D Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.2 RTC_D Operation

The RTC_D module can be configured as a real-time clock with calendar function (calendar mode) or as a 32-bit general prupose counter (counter mode) with the RTCMODE bit.

16.2.1 Counter Mode

Counter mode is selected when RTCMODE is reset. In this mode, a 32-bit counter is provided that is directly accessible by software. Switching from calendar mode to counter mode **does not** reset the count value (RTCNT1, RTCNT2, RTCNT3, RTCNT4) nor the prescale counters (RT0PS, RT1PS). These registers must be configured by user software before use.

The clock to increment the counter can be sourced from the 32kHz cyrstal oscillator, or prescaled versions of the 32kHz cyrstal oscillator clock. Prescaled versions are sourced from the prescale dividers (RT0PS and RT1PS). RT0PS and RT1PS can output /2, /4, /8, 16, /32, /64, /128, and /256 versions of the 32kHz clock. The output of RT0PS can be cascaded with RT1PS. The cascaded output can also be used as a clock source input to the 32-bit counter.

Four individual 8-bit counters are cascaded to provide the 32-bit counter. This provides 8-bit, 16-bit, 24-bit, or 32-bit overflow intervals of the counter clock. The RTCTEV bits select the respective trigger event. An RTCTEV event can trigger an interrupt by setting the RTCTEVIE bit. Each counter, RTCNT1 through RTCNT4, is individually accessible and may be written to.

RT0PS and RT1PS can be configured as two 8-bit counters or cascaded into a single 16-bit counter. RT0PS and RT1PS can be halted on an individual basis by setting their respective RT0PSHOLD and RT1PSHOLD bits. When RT0PS is cascaded with RT1PS, setting RT0PSHOLD causes both RT0PS and RT1PS to be halted. The 32-bit counter can be halted several ways depending on the configuration. If the 32-bit counter is sourced directly by the 32kHz cyrstal clock, it can be halted by setting RTCHOLD. If it is sourced from the output of RT1PS, it can be halted by setting RT1PSHOLD or RTCHOLD. Finally, if it is sourced from the cascaded outputs of RT0PS and RT1PS, it can be halted by setting RT0PSHOLD, RT1PSHOLD, or RTCHOLD.

NOTE: Accessing the RTCNT1, RTCNT2, RTCNT3, RTCNT4, RT0PS, RT1PS registers

When the counter clock is asynchronous to the CPU clock, any read from any RTCNT1, RTCNT2, RTCNT3, RTCNT4, RT0PS, or RT1PS register should occur while the counter is not operating. Otherwise, the results may be unpredictable. Alternatively, the counter may be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Any write to these registers takes effect immediately.

16.2.2 Calendar Mode

Calendar mode is selected when RTCMODE is set. In calendar mode, the RTC_D module provides seconds, minutes, hours, day of week, day of month, month, and year in selectable BCD or hexadecimal format. The calendar includes a leap-year algorithm that considers all years evenly divisible by four as leap years. This algorithm is accurate from the year 1901 through 2099. Switching from counter mode to calendar mode **does not** reset the calendar registers (RTCSEC, RTCMIN, RTCHOUR, RTCDAY, RTCDOW, and RTCYEAR) nor the prescale counters (RT0PS, RT1PS). These registers must be configured by user software before use.

16.2.2.1 Real-Time Clock and Prescale Dividers

The prescale dividers, RT0PS and RT1PS, are automatically configured to provide a 1-s clock interval for the RTC_D. The low-frequency oscillator must be operated at 32768 Hz (nominal) for proper RTC_D operation. RT0PS is sourced directly from the low-frequency crystal oscillator. RT1PS is cascaded with the output ACLK/256 of RT0PS. The RTC_A is sourced with the /128 output of RT1PS, thereby providing the required 1-s interval. Switching from counter to calendar mode clears the seconds, minutes, hours, day-of-week, and year counts and sets day-of-month and month counts to 1. In addition, RT0PS and RT1PS are cleared. When RTCBCD = 1, BCD format is selected for the calendar registers. Setting RTCHOLD halts the real-time counters and prescale counters, RT0PS and RT1PS.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.2.2.2 Real-Time Clock Alarm Function

The RTC_D module provides for a flexible alarm system. There is a single user-programmable alarm that can be programmed based on the settings contained in the alarm registers for minutes, hours, day of week, and day of month. The user-programmable alarm function is only available in the calendar mode of operation.

Each alarm register contains an alarm enable (AE) bit that can be used to enable the respective alarm register. By setting AE bits of the various alarm registers, a variety of alarm events can be generated.

- Example 1: A user wishes to set an alarm every hour at 15 minutes past the hour, that is, at 00:15:00, 01:15:00, 02:15:00, and so on. This is possible by setting RTCAMIN to 15. By setting the AE bit of the RTCAMIN and clearing all other AE bits of the alarm registers, the alarm is enabled. When enabled, the RTCAIFG is set when the count transitions from 00:14:59 to 00:15:00, 01:14:59 to 01:15:00, 02:14:59 to 02:15:00, etc.
- Example 2: A user wishes to set an alarm every day at 04:00:00. This is possible by setting RTCAHOUR to 4. By setting the AE bit of the RTCHOUR and clearing all other AE bits of the alarm registers, the alarm is enabled. When enabled, the RTCAIFG is set when the count transitions from 03:59:59 to 04:00:00.
- Example 3: A user wishes to set an alarm for 06:30:00. RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCAHOUR and RTCAMIN, the alarm is enabled. Once enabled, the RTCAIFG is set when the time count transitions from 06:29:59 to 06:30:00. In this case, the alarm event occurs every day at 06:30:00.
- Example 4: A user wishes to set an alarm every Tuesday at 06:30:00. RTCADOW would be set to 2, RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCADOW, RTCAHOUR, and RTCAMIN, the alarm is enabled. Once enabled, the RTCAIFG is set when the time count transitions from 06:29:59 to 06:30:00 and the RTCDOW transitions from 1 to 2.
- Example 5: A user wishes to set an alarm the fifth day of each month at 06:30:00. RTCADAY would be set to 5, RTCAHOUR would be set to 6 and RTCAMIN would be set to 30. By setting the AE bits of RTCADAY, RTCAHOUR, and RTCAMIN, the alarm is enabled. Once enabled, the RTCAIFG is set when the time count transitions from 06:29:59 to 06:30:00 and the RTCDAY equals 5.

NOTE: Setting the alarm

Prior to setting an initial alarm, all alarm registers including the AE bits should be cleared.

To prevent potential erroneous alarm conditions from occurring, the alarms should be disabled by clearing the RTCAIE, RTCAIFG, and AE bits prior to writing initial or new time values to the RTC time registers.

NOTE: Invalid alarm settings

Invalid alarm settings are not checked via hardware. It is the user's responsibility that valid alarm settings are entered.

NOTE: Invalid time and date values

Writing of invalid date and/or time information or data values outside the legal ranges specified in the RTCSEC, RTCMIN, RTCHOUR, RTCDAY, RTCDOW, RTCYEAR, RTCAMIN, RTCAHOUR, RTCADAY, and RTCADOW registers can result in unpredictable behavior.

Also after switching from counter mode to calendar mode the registers must be correctly initialized to ensure values inside their legal ranges.

16.2.2.3 Reading or Writing Real-Time Clock Registers in Calendar Mode

Because the system clock may be asynchronous to the RTC_D clock source, special care must be taken when accessing the real-time clock registers.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The real-time clock registers are updated once per second. To prevent reading any real-time clock register at the time of an update that could result in an invalid time being read, a keep-out window is provided. The keep-out window is centered approximately 128/32768 seconds around the update transition. The read-only RTCRDY bit is reset during the keep-out window period and set outside the keep-out the window period. Any read of the clock registers while RTCRDY is reset is considered to be potentially invalid, and the time read should be ignored.

An easy way to safely read the real-time clock registers is to utilize the RTCRDYIFG interrupt flag. Setting RTCRDYIE enables the RTCRDYIFG interrupt. Once enabled, an interrupt is generated based on the rising edge of the RTCRDY bit, causing the RTCRDYIFG to be set. At this point, the application has nearly a complete second to safely read any or all of the real-time clock registers. This synchronization process prevents reading the time value during transition. The RTCRDYIFG flag is reset automatically when the interrupt is serviced, or it can be reset with software.

In counter mode, the RTCRDY bit remains reset. RTCRDYIE is a don't care and RTCRDYIFG remains reset.

NOTE: Reading or writing real-time clock registers

When the counter clock is asynchronous to the CPU clock, any read from any RTCSEC, RTCMIN, RTCHOUR, RTCDOW, RTCDAY, RTCMON, or RTCYEAR register while the RTCRDY is reset may result in invalid data being read. To safely read the counting registers, either polling of the RTCRDY bit or the synchronization procedure previously described can be used. Alternatively, the counter register can be read multiple times while operating, and a majority vote taken in software to determine the correct reading. Reading the RT0PS and RT1PS can only be handled by reading the registers multiple times and a majority vote taken in software to determine the correct reading or by halting the counters.

Any write to any counting register takes effect immediately. However, the clock is stopped during the write. In addition, RT0PS and RT1PS registers are reset. This could result in losing up to 1 second during a write. Writing of data outside the legal ranges or invalid time stamp combinations results in unpredictable behavior.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.2.3 Real-Time Clock Interrupts

The RTC_D module has six interrupt sources available, each with independent enables and flags.

16.2.3.1 Real-Time Clock Interrupts in Calendar Mode

Six sources for interrupts are available, namely RT0PSIFG, RT1PSIFG, RTCRDYIFG, RTCTEVIFG, RTCAIFG, and RTCOFIFG. These flags are prioritized and combined to source a single interrupt vector. The interrupt vector register (RTCIV) is used to determine which flag requested an interrupt.

The highest-priority enabled interrupt generates a number in the RTCIV register (see register description). This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled RTC interrupts do not affect the RTCIV value.

Any access, read or write, of the RTCIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. In addition, all flags can be cleared via software.

The user-programmable alarm event sources the real-time clock interrupt, RTCAIFG. Setting RTCAIE enables the interrupt. In addition to the user-programmable alarm, the RTC_D module provides for an interval alarm that sources real-time clock interrupt, RTCTEVIFG. The interval alarm can be selected to cause an alarm event when RTCMIN changed or RTCHOUR changed, every day at midnight (00:00:00) or every day at noon (12:00:00). The event is selectable with the RTCTEV bits. Setting the RTCTEVIE bit enables the interrupt.

The RTCRDY bit sources the real-time clock interrupt, RTCRDYIFG, and is useful in synchronizing the read of time registers with the system clock. Setting the RTCRDYIE bit enables the interrupt.

RT0PSIFG can be used to generate interrupt intervals selectable by the RT0IP bits. RT0PS is sourced with low-frequency oscillator clock at 32768 Hz, so intervals of 16384 Hz, 8192 Hz, 4096 Hz, 2048 Hz, 1024 Hz, 512 Hz, 256 Hz, or 128 Hz are possible. Setting the RT0PSIE bit enables the interrupt.

RT1PSIFG can be used to generate interrupt intervals selectable by the RT1IP bits. RT1PS is sourced with the output of RT0PS, which is 128 Hz (32768/256 Hz). Therefore, intervals of 64 Hz, 32 Hz, 16 Hz, 8 Hz, 4 Hz, 2 Hz, 1 Hz, or 0.5 Hz are possible. Setting the RT1PSIE bit enables the interrupt.

The RTCOFIFG bit flags a failure of the 32-kHz crystal oscillator. It's main purpose is to wake-up the CPU from LPM3.5 in case an oscillator failure occurred.

16.2.3.2 Real-Time Clock Interrupts in Counter Mode

In counter mode, four interrupt sources are available: RT0PSIFG, RT1PSIFG, RTCTEVIFG, and RTCOFIFG. RTCAIFG and RTCRDYIFG are cleared. RTCRDYIE and RTCAIE are don't care.

RT0PSIFG can be used to generate interrupt intervals selectable by the RT0IP bits. In counter mode, divide ratios of /2, /4, /8, /16, /32, /64, /128, and /256 of the clock source are possible. Setting the RT0PSIE bit enables the interrupt.

RT1PSIFG can be used to generate interrupt intervals selectable by the RT1IP bits. In counter mode, RT1PS is sourced with low-frequency oscillator clock, or the output of RT0PS, so divide ratios of /2, /4, /8, /16, /32, /64, /128, and /256 of the respective clock source are possible. Setting the RT1PSIE bit enables the interrupt.

In Counter Mode, the RTC_D module provides for an interval timer that sources real-time clock interrupt, RTCTEVIFG. The interval timer can be selected to cause an interrupt event when an 8-bit, 16-bit, 24-bit, or 32-bit overflow occurs within the 32-bit counter. The event is selectable with the RTCTEV bits. Setting the RTCTEVIE bit enables the interrupt.

The RTCOFIFG bit flags a failure of the 32-kHz crystal oscillator. It's main purpose is to wake-up the CPU from LPM3.5 in case an oscillator failure occurred.

16.2.3.2.1 RTCIV Software Example

The following software example shows the recommended use of RTCIV and the handling overhead. The RTCIV value is added to the PC to automatically jump to the appropriate routine.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself.

```
; Interrupt handler for RTC interrupt flags.

RTC_HND                       ; Interrupt latency         6
           ADD &RTCIV,PC      ; Add offset to Jump table  3
           RETI               ; Vector 0: No interrupt    5
           JMP RTCRDYIFG_HND  ; Vector 2: RTCRDYIFG       2
           JMP RTCTEVIFG_HND  ; Vector 4: RTCTEVIFG       2
           JMP RTCAIFG_HND    ; Vector 6: RTCAIFG         5
           JMP RT0PSIFG_HND   ; Vector 8: RT0PSIFG        5
           JMP RT1PSIFG_HND   ; Vector A: RT1PSIFG        5
           JMP RTCOFIFG_HND   ; Vector C: RTCOFIFG        5
           RETI               ; Vector E: Reserved        5

RTCRDYIFG_HND                 ; Vector 2: RTCRDYIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5

RTCTEVIFG_HND                 ; Vector 4: RTCTEVIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5

RTCAIFG_HND                   ; Vector 6: RTCAIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5

RT0PSIFG_HND                  ; Vector 8: RT0PSIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5

RT1PSIFG_HND                  ; Vector A: RT1PSIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5

RTCOFIFG_HND                  ; Vector C: RTCOFIFG Flag
           ...                ; Task starts here
           RETI               ; Back to main program       5
```

16.2.4 Real-Time Clock Calibration

The RTC_D module has calibration logic that allows for adjusting the crystal frequency in approximately +4-ppm or –2-ppm steps, allowing for higher time keeping accuracy from standard crystals.The RTCCALx bits are used to adjust the frequency. When RTCCALS is set, each RTCCALx LSB causes a ≈ +4-ppm adjustment. When RTCCALS is cleared, each RTCCALx LSB causes a ≈ –2-ppm adjustment. Calibration is available in calendar mode only. In counter mode (RTCMODE=0), the calibration logic is disabled.

Calibration is accomplished by periodically adjusting the RT1PS counter based on the RTCCALS and RTCCALx settings. In calendar mode, the RT0PS divides the nominial 37268-Hz low-frequency (LF) crystal clock input by 256. A 60-minute period has 32768 cycles/sec × 60 sec/min × 60 min = 117964800 cycles. Therefore, a –2-ppm reduction in frequency (down calibration) approximately equates to adding an additional 256 cycles every 117964800 cycles (256/117964800 = 2.17 ppm). This is accomplished by holding the RT1PS counter for one additional clock of the RT0PS output within a 60-minute period. Similary, a +4-ppm increase in frequency (up calibration) approximately equates to removing 512 cycles every 117964800 cycle (512/117964800 = 4.34 ppm). This is accomplished by incrementing the RT1PS counter for two additional clocks of the RT0PS output within a 60-minute period. Each RTCCALx calibration bit causes either 256 LF crystal clock cycles to be added every 60 minutes or 512 LF crystal clock cycles to be subtracted every 60 minutes, giving a frequency adjustment of approximately -2 ppm or +4 ppm, respectively.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

To calibrate the frequency, the RTCCLK output signal is available at a pin. RTCCALF bits can be used to select the frequency rate of the output signal, either no signal, 512 Hz, 256 Hz, or 1 Hz.

The basic flow to calibrate the frequency is as follows:

1. Configure the RTCCLK pin.
2. Measure the RTCCLK output signal with an appropriate resolution frequency counter ; that is, within the resolution required.
3. Compute the absolute error in ppm: Absolute error (ppm) = $|10^6 (f_{MEASURED} - f_{RTCCLK})/f_{RTCCLK}|$, where f_{RTCCLK} is the expected frequency of 512 Hz, 256 Hz, or 1 Hz.
4. Adjust the frequency by performing the following:
   (a) If the frequency is too low, set RTCCALS = 1 and apply the appropriate RTCCALx bits, where RTCCALx = (Absolute Error) / 4.34 rounded to the nearest integer
   (b) If the frequency is too high, clear RTCCALS = 0 and apply the appropriate RTCCALx bits, where RTCCALx = (Absolute Error) / 2.17 rounded to the nearest integer

For example, assume that RTCCLK is configured to output at a frequency of 512 Hz. The measured RTCCLK is 511.9658 Hz. This frequency error is approximately 66.8 ppm too low. To increase the frequency by 66.8 ppm, RTCCALS would be set, and RTCCALx would be set to 15 (66.8/4.34). Similarly, assume that the measured RTCCLK is 512.0125 Hz. The frequency error is approximately 24.4 ppm too high. To decrease the frequency by 24.4 ppm, RTCCALS would be cleared, and RTCCAL would be set to 11 (24.4/2.17).

The calibration corrects only initial offsets and does not adjust for temperature and aging effects. These effects can be handled by periodically measuring temperature and using the crystal's charateristic curve to adjust the ppm based on temperature, as required. In counter mode (RTCMODE = 0), the calibration logic is disabled.

NOTE: Calibration output frequency

The 512-Hz and 256-Hz output frequencies observed at the RTCCLK pin are not affected by changes in the calibration settings since these output frequencies are generated prior to the calibration logic. The 1-Hz output frequency is affected by changes in the calibration settings. Because the frequency change is small and infrequent over a very long time interval, it can be difficult to observe.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.2.5 Real-Time Clock Operation in LPMx.5 Low-Power Mode

The regulator of the Power Management Module (PMM) is disabled upon entering LPMx.5, which causes most of the RTC_D configuration registers to be lost; only the counters are retained. Table 16-1 lists the retained registers in LPMx.5. Also the configuration of the interrupts is stored so that the configured interrupts can cause a wakeup upon exit from LPMx.5.

The following interrupt flags can be used as RTC_D wake-up interrupt sources:

- RTCTEVIFG: Real-time clock time event interrupt flag
- RTCAIFG: Real-time clock alarm interrupt flag
- RT1PSIFG: Prescale timer 1 interrupt flag
- RTCOFIFG: 32-kHz crystal oscillator fault interrupt flag

After restoring the configuration registers and clearing LOCKLPM5, the interrupts can be serviced as usual. The detailed flow is as follows:

1. Set all I/Os to general purpose I/Os and configure as needed. Optionally configure input interrupt pins for wake-up. Configure RTC_D interrupts for wake-up (set RTCTEVIE, RTCAIE, RT1PSIE, or RTCOFIE. If the alarm interrupt is also used as wake-up event, the alarm registers must be configured as needed).
2. Ensure clock system settings allow LPMx.5 entry according to Table 3-1 in UCS chapter.
3. Enter LPMx.5 with LPMx.5 entry sequence.
   bis #PMMKEY + REGOFF, &PMMCTL0
   bis #LPM4, SR
4. LOCKLPM5 is automatically set by hardware upon entering LPMx.5, the core voltage regulator is disabled, and all clocks are disabled except for the 32-kHz crystal oscillator clock if the RTC is enabled with RTCHOLD = 0.
5. An LPMx.5 wake-up event, such as an edge on a wake-up input pin, are an RTC_D interrupt event and start the BOR entry sequence together with the core voltage regulator. All peripheral registers are set to their default conditions. The I/O pin state remains locked as well as the interrupt configuration for the RTC_D.
6. The device can be configured. The I/O configuration and the RTC_D control and interrupt configuration that was not retained during LPMx.5 should be restored to the values prior to entering LPMx.5. After restoring the configuration registers the LOCKLPM5 bit can be cleared, this releases the I/O pin conditions as well as the RTC_D interrupt configuration.
7. After enabling I/O and RTC_D interrupts, the interrupt that caused the wake-up can be serviced.

If the RTC is enabled (RTCHOLD = 0), the 32-kHz oscillator remains active during LPMx.5. The fault detection also remains functional. If a fault occurs during LPMx.5 and the RTCOFIE was set before entering LPMx.5, a wake-up event is issued.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3 RTC_D Registers

The RTC_D registers are listed in Table 16-1. This table also lists the retention during LPMx.5. Registers that are not retained during LPMx.5 must be restored after exit from LPMx.5. The base address for the RTC_D module registers can be found in the device-specific data sheet. The address offsets are given in Table 16-1.

NOTE: Most registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 16-1. RTC_D Registers

Offset	Acronym	Register Name	Type	Access	Reset	LPMx.5 or Backup
00h	RTCCTL01	Real-Time Clock Control 0, 1	Read/write	Word	4000h	not retained
00h	RTCCTL0	Real-Time Clock Control 0	Read/write	Byte	00h	not retained
	or RTCCTL01_L					
01h	RTCCTL1	Real-Time Clock Control 1	Read/write	Byte	40h	not retained
	or RTCCTL01_H					
02h	RTCCTL23	Real-Time Clock Control 2, 3	Read/write	Word	0000h	retained
02h	RTCCTL2	Real-Time Clock Control 2	Read/write	Byte	00h	retained
	or RTCCTL23_L					
03h	RTCCTL3	Real-Time Clock Control 3	Read/write	Byte	00h	retained
	or RTCCTL23_H					
08h	RTCPS0CTL	Real-Time Prescale Timer 0 Control	Read/write	Word	0000h	not retained
08h	RTCPS0CTLL		Read/write	Byte	00h	not retained
	or RTCPS0CTL_L					
09h	RTCPS0CTLH		Read/write	Byte	00h	not retained
	or RTCPS0CTL_H					
0Ah	RTCPS1CTL	Real-Time Prescale Timer 1 Control	Read/write	Word	0000h	not retained
0Ah	RTCPS1CTLL		Read/write	Byte	00h	not retained
	or RTCPS1CTL_L					
0Bh	RTCPS0CTLH		Read/write	Byte	00h	not retained
	or RTCPS0CTL_H					
0Ch	RTCPS	Real-Time Prescale Timer 0, 1 Counter	Read/write	Word	none	retained
0Ch	RT0PS	Real-Time Prescale Timer 0 Counter	Read/write	Byte	none	retained
	or RTCPS_L					
0Dh	RT1PS	Real-Time Prescale Timer 1 Counter	Read/write	Byte	none	retained
	or RTCPS_H					
0Eh	RTCIV	Real Time Clock Interrupt Vector	Read	Word	0000h	not retained
10h	RTCTIM0	Real-Time Clock Seconds, Minutes	Read/write	Word	undefined	retained
	or RTCNT12	Real-Time Counter 1, 2				
10h	RTCSEC	Real-Time Clock Seconds	Read/write	Byte	undefined	retained
	RTCNT1	Real-Time Counter 1				
	or RTCTIM0_L					
11h	RTCMIN	Real-Time Clock Minutes	Read/write	Byte	undefined	retained
	RTCNT2	Real-Time Counter 2				
	or RTCTIM0_H					

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 16-1. RTC_D Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	LPMx.5 or Backup
12h	RTCTIM1	Real-Time Clock Hour, Day of Week	Read/write	Word	undefined	retained
	or RTCNT34	Real-Time Counter 3, 4				
12h	RTCHOUR	Real-Time Clock Hour	Read/write	Byte	undefined	retained
	RTCNT3	Real-Time Counter 3				
	or RTCTIM1_L					
13h	RTCDOW	Real-Time Clock Day of Week	Read/write	Byte	undefined	retained
	RTCNT4	Real-Time Counter 4				
	or RTCTIM1_H					
14h	RTCDATE	Real-Time Clock Date	Read/write	Word	undefined	retained
14h	RTCDAY	Real-Time Clock Day of Month	Read/write	Byte	undefined	retained
	or RTCDATE_L					
15h	RTCMON	Real-Time Clock Month	Read/write	Byte	undefined	retained
	or RTCDATE_H					
16h	RTCYEAR	Real-Time Clock Year[(1)]	Read/write	Word	undefined	retained
18h	RTCAMINHR	Real-Time Clock Minutes, Hour Alarm	Read/write	Word	undefined	retained
18h	RTCAMIN	Real-Time Clock Minutes Alarm	Read/write	Byte	undefined	retained
	or RTCAMINHR_L					
19h	RTCAHOUR	Real-Time Clock Hours Alarm	Read/write	Byte	undefined	retained
	or RTCAMINHR_H					
1Ah	RTCADOWDAY	Real-Time Clock Day of Week, Day of Month Alarm	Read/write	Word	undefined	retained
1Ah	RTCADOW	Real-Time Clock Day of Week Alarm	Read/write	Byte	undefined	retained
	or RTCADOWDAY_L					
1Bh	RTCADAY	Real-Time Clock Day of Month Alarm	Read/write	Byte	undefined	retained
	or RTCADOWDAY_H					
1Ch	BIN2BCD	Binary-to-BCD conversion register	Read/write	Word	00h	not retained
1Eh	BCD2BIN	BCD-to-binary conversion register	Read/write	Word	00h	not retained

(1) Do not access the year register RTCYEAR in byte mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.1 RTCCTL0 Register

Real-Time Clock Control 0 Register

Figure 16-2. RTCCTL0 Register

7	6	5	4	3	2	1	0
RTCOFIE[1]	RTCTEVIE[1]	RTCAIE[1]	RTCRDYIE	RTCOFIFG	RTCTEVIFG	RTCAIFG	RTCRDYIFG
rw-0	rw-0	rw-0	rw-0	rw-(0)	rw-(0)	rw-(0)	rw-(0)

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

Table 16-2. RTCCTL0 Register Description

Bit	Field	Type	Reset	Description
7	RTCOFIE	RW	0h	32-kHz crystal oscillator fault interrupt enable. This interrupt can be used as LPMx.5 wake-up event. 0b = Interrupt not enabled 1b = Interrupt enabled (LPMx.5 wake-up enabled)
6	RTCTEVIE	RW	0h	Real-time clock time event interrupt enable. In modules supporting LPMx.5 this interrupt can be used as LPMx.5 wake-up event. 0b = Interrupt not enabled 1b = Interrupt enabled (LPMx.5 wake-up enabled)
5	RTCAIE	RW	0h	Real-time clock alarm interrupt enable. In modules supporting LPMx.5 this interrupt can be used as LPMx.5 wake-up event. 0b = Interrupt not enabled 1b = Interrupt enabled (LPMx.5 wake-up enabled)
4	RTCRDYIE	RW	0h	Real-time clock ready interrupt enable. 0b = Interrupt not enabled 1b = Interrupt enabled
3	RTCOFIFG	RW	0h	32-kHz crystal oscillator fault interrupt flag. This interrupt can be used as LPMx.5 wake-up event. It also indicates a clock failure during backup operation. 0b = No interrupt pending 1b = Interrupt pending. A 32-kHz crystal oscillator fault occurred after last reset.
2	RTCTEVIFG	RW	0h	Real-time clock time event interrupt flag. In modules supporting LPMx.5 this interrupt can be used as LPMx.5 wake-up event. 0b = No time event occurred. 1b = Time event occurred.
1	RTCAIFG	RW	0h	Real-time clock alarm interrupt flag. In modules supporting LPMx.5 this interrupt can be used as LPMx.5 wake-up event. 0b = No time event occurred. 1b = Time event occurred.
0	RTCRDYIFG	RW	0h	Real-time clock ready interrupt flag 0b = RTC cannot be read safely. 1b = RTC can be read safely.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.2 RTCCTL1 Register

Real-Time Clock Control Register 1

Figure 16-3. RTCCTL1 Register

7	6	5	4	3	2	1	0
RTCBCD	RTCHOLD[1]	RTCMODE[1]	RTCRDY	RTCSSELx[1]		RTCTEVx[1]	
rw-(0)	rw-(1)	rw-(1)	r-(1)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

Table 16-3. RTCCTL1 Register Description

Bit	Field	Type	Reset	Description
7	RTCBCD	RW	0h	Real-time clock BCD select. Selects BCD counting for real-time clock. Applies to calendar mode (RTCMODE = 1) only; setting is ignored in counter mode. 0b = Binary hexadecimal code selected 1b = BCD Binary coded decimal (BCD) code selected
6	RTCHOLD	RW	1h	Real-time clock hold 0b = Real-time clock (32-bit counter or calendar mode) is operational. 1b = In counter mode (RTCMODE = 0), only the 32-bit counter is stopped. In calendar mode (RTCMODE = 1), the calendar is stopped as well as the prescale counters, RT0PS and RT1PS. RT0PSHOLD and RT1PSHOLD are don't care.
5	RTCMODE	RW	1h	Real-time clock mode 0b = 32-bit counter mode 1b = Calendar mode. Switching between counter and calendar mode does not reset the real-time clock/counter registers. These registers must be configured by user software before use.
4	RTCRDY	RW	1h	Real-time clock ready 0b = RTC time values in transition (calendar mode only). 1b = RTC time values safe for reading (calendar mode only). This bit indicates when the real-time clock time values are safe for reading (calendar mode only). In counter mode, RTCRDY remains cleared.
3-2	RTCSSELx	RW	0h	Real-time clock source select. In counter mode, selects clock input source to the 32-bit counter. In calendar mode, these bits are don't care. The clock input is automatically set to the output of RT1PS. 00b = 32-kHz crystal oscillator clock 01b = 32-kHz crystal oscillator clock 10b = Output from RT1PS 11b = Output from RT1PS
1-0	RTCTEVx	RW	0h	Real-time clock time event. Specifies the interrupt interval. Calendar Mode (RTCMODE = 1) 00b = Minute changed 01b = Hour changed 10b = Every day at midnight (00:00) 11b = Every day at noon (12:00) Counter Mode (RTCMODE = 0) 00b = 8-bit overflow 01b = 16-bit overflow 10b = 24-bit overflow 11b = 32-bit overflow

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.3 RTCCTL2 Register

Real-Time Clock Control 2 Register

Figure 16-4. RTCCTL2 Register

7	6	5	4	3	2	1	0
RTCCALS	Reserved	RTCCALx					
rw-(0)	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 16-4. RTCCTL2 Register Description

Bit	Field	Type	Reset	Description
7	RTCCALS	RW	0h	Real-time clock calibration sign 0b = Frequency adjusted down 1b = Frequency adjusted up
6	Reserved	R	0h	Reserved. Always reads as 0.
5-0	RTCCALx	RW	0h	Real-time clock calibration. Each LSB represents approximately +4-ppm (RTCCALS = 1) or a –2-ppm (RTCCALS = 0) adjustment in frequency.

16.3.4 RTCCTL3 Register

Real-Time Clock Control 3 Register

Figure 16-5. RTCCTL3 Register

7	6	5	4	3	2	1	0
Reserved						RTCCALFx	
r0	r0	r0	r0	r0	r0	rw-(0)	rw-(0)

Table 16-5. RTCCTL3 Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1-0	RTCCALFx	RW	0h	Real-time clock calibration frequency. Selects frequency output to RTCCLK pin for calibration measurement. The corresponding port must be configured for the peripheral module function. The RTCCLK is not available in counter mode and remains low, and the RTCCALF bits are don't care. 00b = No frequency output to RTCCLK pin 01b = 512 Hz 10b = 256 Hz 11b = 1 Hz

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.5 RTCNT1 Register

Real-Time Clock Counter 1 Register – Counter Mode

Figure 16-6. RTCNT1 Register

7	6	5	4	3	2	1	0
RTCNT1							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-6. RTCNT1 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT1	RW	undefined	The RTCNT1 register is the count of RTCNT1

16.3.6 RTCNT2 Register

Real-Time Clock Counter 2 Register – Counter Mode

Figure 16-7. RTCNT2 Register

7	6	5	4	3	2	1	0
RTCNT2							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-7. RTCNT2 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT2	RW	undefined	The RTCNT2 register is the count of RTCNT2

16.3.7 RTCNT3 Register

Real-Time Clock Counter 3 Register – Counter Mode

Figure 16-8. RTCNT3 Register

7	6	5	4	3	2	1	0
RTCNT3							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-8. RTCNT3 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT3	RW	undefined	The RTCNT3 register is the count of RTCNT3

16.3.8 RTCNT4 Register

Real-Time Clock Counter 4 Register – Counter Mode

Figure 16-9. RTCNT4 Register

7	6	5	4	3	2	1	0
RTCNT4							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-9. RTCNT4 Register Description

Bit	Field	Type	Reset	Description
7-0	RTCNT4	RW	undefined	The RTCNT4 register is the count of RTCNT4.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.9 RTCSEC Register – Hexadecimal Format

Real-Time Clock Seconds Register – Hexadecimal Format

Figure 16-10. RTCSEC Register

7	6	5	4	3	2	1	0
0		Seconds					
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 16-10. RTCSEC Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always reads as 0.
5-0	Seconds	RW	undefined	Seconds (0 to 59)

16.3.10 RTCSEC Register – BCD Format

Real-Time Clock Seconds Register – BCD Format

Figure 16-11. RTCSEC Register

7	6	5	4	3	2	1	0
0	Seconds – high digit			Seconds – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw

Table 16-11. RTCSEC Register Description

Bit	Field	Type	Reset	Description
7	0	R	0h	Always reads as 0.
6-4	Seconds – high digit	RW	undefined	Seconds – high digit (0 to 5)
3-0	Seconds – low digit	RW	undefined	Seconds – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.11 RTCMIN Register – Hexadecimal Format

Real-Time Clock Minutes Register – Hexadecimal Format

Figure 16-12. RTCMIN Register

7	6	5	4	3	2	1	0
0		Minutes					
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 16-12. RTCMIN Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always reads as 0.
5-0	Minutes	RW	undefined	Minutes (0 to 59)

16.3.12 RTCMIN Register – BCD Format

Real-Time Clock Minutes Register – BCD Format

Figure 16-13. RTCMIN Register

7	6	5	4	3	2	1	0
0	Minutes – high digit			Minutes – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw

Table 16-13. RTCMIN Register Description

Bit	Field	Type	Reset	Description
7	0	R	0h	Always reads as 0.
6-4	Minutes – high digit	RW	undefined	Minutes – high digit (0 to 5)
3-0	Minutes – low digit	RW	undefined	Minutes – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.13 RTCHOUR Register – Hexadecimal Format

Real-Time Clock Hours Register – Hexadecimal Format

Figure 16-14. RTCHOUR Register

7	6	5	4	3	2	1	0
0			Hours				
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 16-14. RTCHOUR Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always reads as 0.
4-0	Hours	RW	undefined	Hours (0 to 23)

16.3.14 RTCHOUR Register – BCD Format

Real-Time Clock Hours Register – BCD Format

Figure 16-15. RTCHOUR Register

7	6	5	4	3	2	1	0
0		Hours – high digit		Hours – low digit			
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 16-15. RTCHOUR Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	Always reads as 0.
5-4	Hours – high digit	RW	undefined	Hours – high digit (0 to 2)
3-0	Hours – low digit	RW	undefined	Hours – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.15 RTCDOW Register – Calendar Mode

Real-Time Clock Day of Week Register – Calendar Mode

Figure 16-16. RTCDOW Register

7	6	5	4	3	2	1	0
0					Day of week		
r-0	r-0	r-0	r-0	r-0	rw	rw	rw

Table 16-16. RTCDOW Register Description

Bit	Field	Type	Reset	Description
7-3	0	R	0h	Always reads as 0.
2-0	Day of week	RW	undefined	Day of week (0 to 6)

16.3.16 RTCDAY Register – Hexadecimal Format

Real-Time Clock Day of Month Register – Hexadecimal Format

Figure 16-17. RTCDAY Register

7	6	5	4	3	2	1	0
0			Day of month				
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 16-17. RTCDAY Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always reads as 0.
4-0	Day of month	RW	undefined	Day of month (1 to 28, 29, 30, 31)

16.3.17 RTCDAY Register – BCD Format

Real-Time Clock Day of Month Register – BCD Format

Figure 16-18. RTCDAY Register

7	6	5	4	3	2	1	0
0		Day of month – high digit		Day of month – low digit			
r-0	r-0	rw	rw	rw	rw	rw	rw

Table 16-18. RTCDAY Register Description

Bit	Field	Type	Reset	Description
7-6	0	R	0h	
5-4	Day of month – high digit	RW	undefined	Day of month – high digit (0 to 3)
3-0	Day of month – low digit	RW	undefined	Day of month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

16.3.18 RTCMON Register – Hexadecimal Format

Real-Time Clock Month Register – Hexadecimal Format

Figure 16-19. RTCMON Register

7	6	5	4	3	2	1	0
0				Month			
r-0	r-0	r-0	r-0	rw	rw	rw	rw

Table 16-19. RTCMON Register Description

Bit	Field	Type	Reset	Description
7-4	0	R	0h	Always reads as 0.
3-0	Month	RW	undefined	Month (1 to 12)

16.3.19 RTCMON Register – BCD Format

Real-Time Clock Month Register – BCD Format

Figure 16-20. RTCMON Register

7	6	5	4	3	2	1	0
0			Month – high digit	Month – low digit			
r-0	r-0	r-0	rw	rw	rw	rw	rw

Table 16-20. RTCMON Register Description

Bit	Field	Type	Reset	Description
7-5	0	R	0h	Always reads as 0.
4	Month – high digit	RW	undefined	Month – high digit (0 or 1)
3-0	Month – low digit	RW	undefined	Month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.20 RTCYEAR Register – Calendar Mode With Hexadecimal Format

Real-Time Clock Year Register – Calendar Mode With Hexadecimal Format

Figure 16-21. RTCYEAR Register

15	14	13	12	11	10	9	8
0				Year – high byte of 0 to 4095			
r-0	r-0	r-0	r-0	rw	rw	rw	rw
7	6	5	4	3	2	1	0
Year – low byte of 0 to 4095							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-21. RTCYEAR Register Description

Bit	Field	Type	Reset	Description
15-12	0	R	0h	Always reads as 0.
11-0	Year	RW	undefined	Year (0 to 4095)

16.3.21 RTCYEAR Register – Calendar Mode With BCD Format

Real-Time Clock Year Register – Calendar Mode With BCD Format

Figure 16-22. RTCYEAR Register

15	14	13	12	11	10	9	8
0	Century – high digit			Century – low digit			
r-0	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
Decade				Year – lowest digit			
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-22. RTCYEAR Register Description

Bit	Field	Type	Reset	Description
15	0	R	0h	Always reads as 0.
14-12	Century – low digit	RW	undefined	Century – high digit (0 to 4)
11-8	Century_Low	RW	undefined	Century – low digit (0 to 9)
7-4	Decade	RW	undefined	Decade (0 to 9)
3-0	Year – lowest digit	RW	undefined	Year – lowest digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.22 RTCAMIN Register – Hexadecimal Format

Real-Time Clock Minutes Alarm Register – Hexadecimal Format

Figure 16-23. RTCAMIN Register

7	6	5	4	3	2	1	0
AE	0	Minutes					
rw	r-0	rw	rw	rw	rw	rw	rw

Table 16-23. RTCAMIN Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	AE
6	0	R	0h	Always reads as 0.
5-0	Minutes	RW	undefined	Minutes (0 to 59)

16.3.23 RTCAMIN Register – BCD Format

Real-Time Clock Minutes Alarm Register – BCD Format

Figure 16-24. RTCAMIN Register

7	6	5	4	3	2	1	0
AE	Minutes – high digit			Minutes – low digit			
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-24. RTCAMIN Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	0h	AE
6-4	Minutes – high digit	RW	undefined	Minutes – high digit (0 to 5)
3-0	Minutes – low digit	RW	undefined	Minutes – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

16.3.24 RTCAHOUR Register – Hexadecimal Format

Real-Time Clock Hours Alarm Register – Hexadecimal Format

Figure 16-25. RTCAHOUR Register

7	6	5	4	3	2	1	0
AE	0		Hours				
rw	r-0	r-0	rw	rw	rw	rw	rw

Table 16-25. RTCAHOUR Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	AE
6-5	0	R	0h	Always reads as 0.
4-0	Hours	RW	undefined	Hours (0 to 23)

16.3.25 RTCAHOUR Register – BCD Format

Real-Time Clock Hours Alarm Register – BCD Format

Figure 16-26. RTCAHOUR Register

7	6	5	4	3	2	1	0
AE	0	Hours – high digit		Hours – low digit			
rw	r-0	rw	rw	rw	rw	rw	rw

Table 16-26. RTCAHOUR Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	AE
6	0	R	0h	Always reads as 0.
5-4	Hours – high digit	RW	undefined	Hours – high digit (0 to 2)
3-0	Hours – low digit	RW	undefined	Hours – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.26 RTCADOW Register

Real-Time Clock Day of Week Alarm Register – Calendar Mode

Figure 16-27. RTCADOW Register

7	6	5	4	3	2	1	0
AE	0				Day of week		
rw	r-0	r-0	r-0	r-0	rw	rw	rw

Table 16-27. RTCADOW Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	AE
6-3	0	R	0h	Always reads as 0.
2-0	Day of week	RW	undefined	Day of week (0 to 6)

16.3.27 RTCADAY Register – Hexadecimal Format

Real-Time Clock Day of Month Alarm Register – Hexadecimal Format

Figure 16-28. RTCADAY Register

7	6	5	4	3	2	1	0
AE	0		Day of month				
rw	r-0	r-0	rw	rw	rw	rw	rw

Table 16-28. RTCADAY Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	AE
6-5	0	R	0h	Always reads as 0.
4-0	Day of month	RW	undefined	Day of month (1 to 28, 29, 30, 31)

16.3.28 RTCADAY Register – BCD Format

Real-Time Clock Day of Month Alarm Register – BCD Format

Figure 16-29. RTCADAY Register

7	6	5	4	3	2	1	0
AE	0	Day of month – high digit		Day of month – low digit			
rw	r-0	rw	rw	rw	rw	rw	rw

Table 16-29. RTCADAY Register Description

Bit	Field	Type	Reset	Description
7	AE	RW	undefined	
6	0	R	0h	
5-4	Day of month – high digit	RW	undefined	Day of month – high digit (0 to 3)
3-0	Day of month – low digit	RW	undefined	Day of month – low digit (0 to 9)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.29 RTCPS0CTL Register

Real-Time Clock Prescale Timer 0 Control Register

Figure 16-30. RTCPS0CTL Register

15	14	13	12	11	10	9	8
Reserved		RT0PSDIV[1]			Reserved		RT0PSHOLD[1]
r0	r0	rw-(0)	rw-(0)	rw-(0)	r0	r0	rw-(1)
7	6	5	4	3	2	1	0
Reserved			RT0IP[1]			RT0PSIE	RT0PSIFG
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-0	rw-(0)

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

Table 16-30. RTCPS0CTL Register Description

Bit	Field	Type	Reset	Description
15-14	Reserved	R	0h	Reserved. Always reads as 0.
13-11	RT0PSDIV	RW	0h	Prescale timer 0 clock divide. These bits control the divide ratio of the RT0PS counter. In real-time clock calendar mode, these bits are don't care for RT0PS and RT1PS. RT0PS clock output is automatically set to /256. RT1PS clock output is automatically set to /128. 000b = Divide by 2 001b = Divide by 4 010b = Divide by 8 011b = Divide by 16 100b = Divide by 32 101b = Divide by 64 110b = Divide by 128 111b = Divide by 256
10-9	Reserved	R	0h	Reserved. Always reads as 0.
8	RT0PSHOLD	RW	1h	Prescale timer 0 hold. In real-time clock calendar mode, this bit is don't care. RT0PS is stopped via the RTCHOLD bit. 0b = RT0PS is operational. 1b = RT0PS is held.
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-2	RT0IP	RW	0h	Prescale timer 0 interrupt interval 000b = Divide by 2 001b = Divide by 4 010b = Divide by 8 011b = Divide by 16 100b = Divide by 32 101b = Divide by 64 110b = Divide by 128 111b = Divide by 256
1	RT0PSIE	RW	0h	Prescale timer 0 interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled
0	RT0PSIFG	RW	0h	Prescale timer 0 interrupt flag 0b = No time event occurred. 1b = Time event occurred.

Copyright © 2009–2013, Texas Instruments Incorporated

16.3.30 RTCPS1CTL Register

Real-Time Clock Prescale Timer 1 Control Register

Figure 16-31. RTCPS1CTL Register

15	14	13	12	11	10	9	8
RT1SSELx(1)			RT1PSDIVx(1)		Reserved		RT1PSHOLD(1)
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r0	r0	rw-(1)
7	6	5	4	3	2	1	0
Reserved			RT1IPx(1)			RT1PSIE	RT1PSIFG
r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-0	rw-(0)

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

(1) The configuration of these bits is retained during LPMx.5 until LOCKLPM5 is cleared, but not the register bits itself; therefore, reconfiguration after wake-up from LPMx.5 before clearing LOCKLPM5 is required.

Table 16-31. RTCPS1CTL Register Description

Bit	Field	Type	Reset	Description
15-14	RT1SSELx	RW	0h	Prescale timer 1 clock source select. Selects clock input source to the RT1PS counter. In real-time clock calendar mode, these bits are do not care. RT1PS clock input is automatically set to the output of RT0PS. 00b = 32-kHz crystal oscillator clock 01b = 32-kHz crystal oscillator clock 10b = Output from RT0PS 11b = Output from RT0PS
13-11	RT1PSDIVx	RW	0h	Prescale timer 1 clock divide. These bits control the divide ratio of the RT0PS counter. In real-time clock calendar mode, these bits are don't care for RT0PS and RT1PS. RT0PS clock output is automatically set to /256. RT1PS clock output is automatically set to /128. 000b = Divide by 2 001b = Divide by 4 010b = Divide by 8 011b = Divide by 16 100b = Divide by 32 101b = Divide by 64 110b = Divide by 128 111b = Divide by 256
10-9	Reserved	R	0h	Reserved. Always reads as 0.
8	RT1PSHOLD	RW	1h	Prescale timer 1 hold. In real-time clock calendar mode, this bit is don't care. RT1PS is stopped via the RTCHOLD bit. 0b = RT1PS is operational. 1b = RT1PS is held.
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4-2	RT1IPx	RW	0h	Prescale timer 1 interrupt interval 000b = Divide by 2 001b = Divide by 4 010b = Divide by 8 011b = Divide by 16 100b = Divide by 32 101b = Divide by 64 110b = Divide by 128 111b = Divide by 256
1	RT1PSIE	RW	0h	Prescale timer 1 interrupt enable 0b = Interrupt not enabled 1b = Interrupt enabled (LPMx.5 wake-up enabled.)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 16-31. RTCPS1CTL Register Description (continued)

Bit	Field	Type	Reset	Description
0	RT1PSIFG	RW	0h	Prescale timer 1 interrupt flag. This interrupt can be used as LPMx.5 wake-up event. 0b = No time event occurred. 1b = Time event occurred.

16.3.31 RTCPS0 Register

Real-Time Clock Prescale Timer 0 Counter Register

Figure 16-32. RTCPS0 Register

7	6	5	4	3	2	1	0
RT0PS							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-32. RTCPS0 Register Description

Bit	Field	Type	Reset	Description
7-0	RT0PS	RW	undefined	Prescale timer 0 counter value

16.3.32 RTCPS1 Register

Real-Time Clock Prescale Timer 1 Counter Register

Figure 16-33. RTCPS1 Register

7	6	5	4	3	2	1	0
RT1PS							
rw	rw	rw	rw	rw	rw	rw	rw

Table 16-33. RTCPS1 Register Description

Bit	Field	Type	Reset	Description
7-0	RT1PS	RW	undefined	Prescale timer 1 counter value

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

16.3.33 RTCIV Register

Real-Time Clock Interrupt Vector Register

Figure 16-34. RTCIV Register

15	14	13	12	11	10	9	8
RTCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
RTCIVx							
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r0

Table 16-34. RTCIV Register Description

Bit	Field	Type	Reset	Description
15-0	RTCIVx	R	0h	Real-time clock interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: RTC ready; Interrupt Flag: RTCRDYIFG; Interrupt Priority: Highest 04h = Interrupt Source: RTC interval timer; Interrupt Flag: RTCTEVIFG 06h = Interrupt Source: RTC user alarm; Interrupt Flag: RTCAIFG 08h = Interrupt Source: RTC prescaler 0; Interrupt Flag: RT0PSIFG 0Ah = Interrupt Source: RTC prescaler 1; Interrupt Flag: RT1PSIFG 0Ch = Interrupt Source: RTC oscillator failure; Interrupt Flag: RTCOFIFG 0Eh = Reserved; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

16.3.34 BIN2BCD Register

Binary-to-BCD Conversion Register

Figure 16-35. BIN2BCD Register

15	14	13	12	11	10	9	8
BIN2BCDx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
BIN2BCDx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 16-35. BIN2BCD Register Description

Bit	Field	Type	Reset	Description
15-0	BIN2BCDx	RW	0h	Read: 16-bit BCD conversion of previously written 12-bit binary number Write: 12-bit binary number to be converted

16.3.35 BCD2BIN Register

BCD-to-Binary Conversion Register

Figure 16-36. BCD2BIN Register

15	14	13	12	11	10	9	8
BCD2BINx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
BCD2BINx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 16-36. BCD2BIN Register Description

Bit	Field	Type	Reset	Description
15-0	BCD2BINx	RW	0h	Read: 12-bit binary conversion of previously written 16-bit BCD number Write: 16-bit BCD number to be converted

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

32-Bit Hardware Multiplier (MPY32)

This chapter describes the 32-bit hardware multiplier (MPY32). The MPY32 module is implemented in all devices.

Topic **Page**

17.1 32-Bit Hardware Multiplier (MPY32) Introduction 478
17.2 MPY32 Operation 480
17.3 MPY32 Registers 492

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.1 32-Bit Hardware Multiplier (MPY32) Introduction

The MPY32 is a peripheral and is not part of the CPU. This means its activities do not interfere with the CPU activities. The multiplier registers are peripheral registers that are loaded and read with CPU instructions.

The MPY32 supports:

- Unsigned multiply
- Signed multiply
- Unsigned multiply accumulate
- Signed multiply accumulate
- 8-bit, 16-bit, 24-bit, and 32-bit operands
- Saturation
- Fractional numbers
- 8-bit and 16-bit operation compatible with 16-bit hardware multiplier
- 8-bit and 24-bit multiplications without requiring a "sign extend" instruction

The MPY32 block diagram is shown in Figure 17-1.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

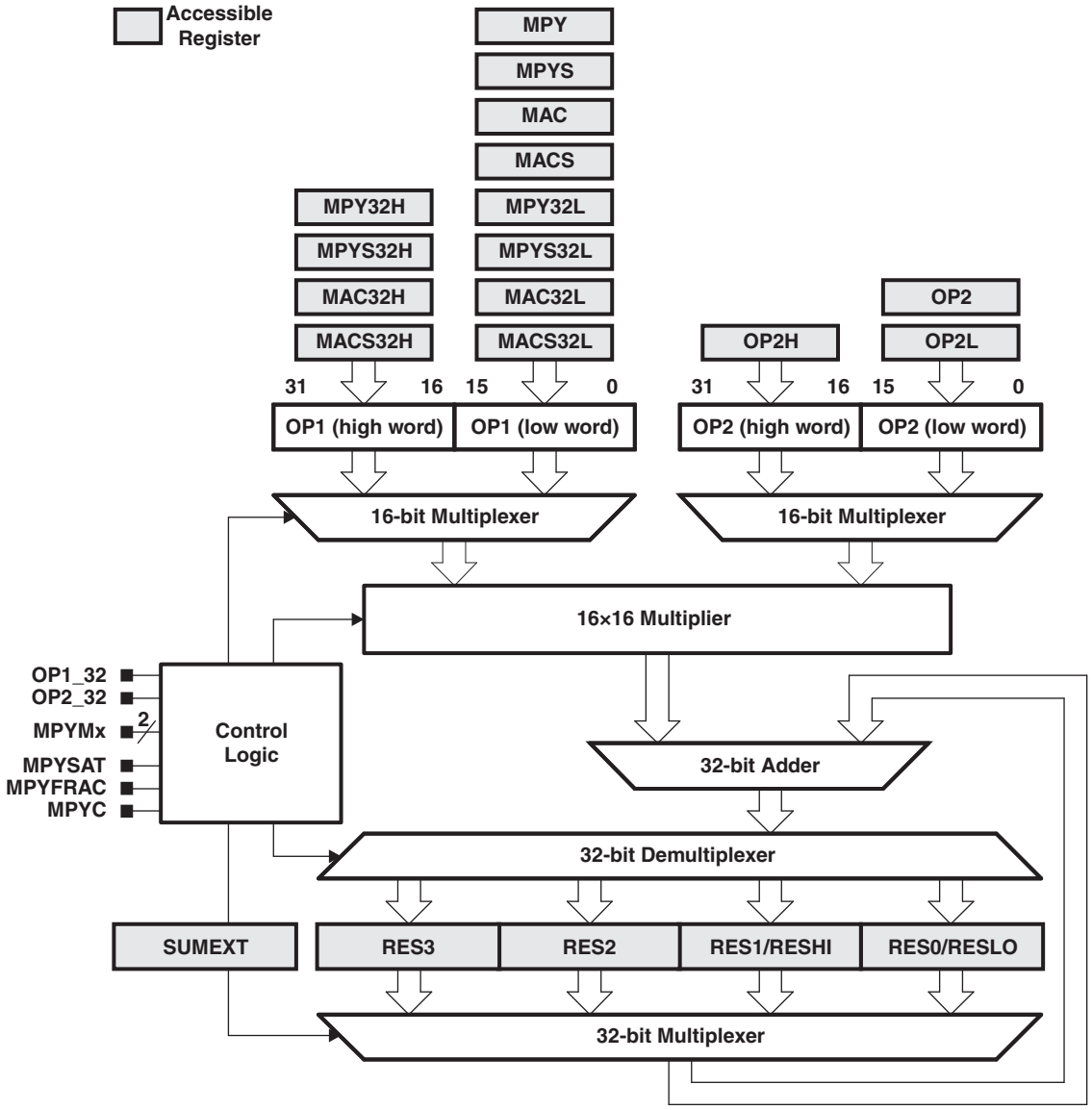


Figure 17-1. MPY32 Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2 MPY32 Operation

The MPY32 supports 8-bit, 16-bit, 24-bit, and 32-bit operands with unsigned multiply, signed multiply, unsigned multiply-accumulate, and signed multiply-accumulate operations. The size of the operands are defined by the address the operand is written to and if it is written as word or byte. The type of operation is selected by the address the first operand is written to.

The hardware multiplier has two 32-bit operand registers – operand one (OP1) and operand two (OP2), and a 64-bit result register accessible via registers RES0 to RES3. For compatibility with the 16×16 hardware multiplier, the result of a 8-bit or 16-bit operation is accessible via RESLO, RESHI, and SUMEXT, as well. RESLO stores the low word of the 16×16-bit result, RESHI stores the high word of the result, and SUMEXT stores information about the result.

The result of a 8-bit or 16-bit operation is ready in three MCLK cycles and can be read with the next instruction after writing to OP2, except when using an indirect addressing mode to access the result. When using indirect addressing for the result, a `NOP` is required before the result is ready.

The result of a 24-bit or 32-bit operation can be read with successive instructions after writing OP2 or OP2H starting with RES0, except when using an indirect addressing mode to access the result. When using indirect addressing for the result, a `NOP` is required before the result is ready.

Table 17-1 summarizes when each word of the 64-bit result is available for the various combinations of operand sizes. With a 32-bit-wide second operand, OP2L and OP2H must be written. Depending on when the two 16-bit parts are written, the result availability may vary; thus, the table shows two entries, one for OP2L written and one for OP2H written. The worst case defines the actual result availability.

Table 17-1. Result Availability (MPYFRAC = 0, MPYSAT = 0)

Operation (OP1 × OP2)	Result Ready in MCLK Cycles					After
	RES0	RES1	RES2	RES3	MPYC Bit	
8/16 × 8/16	3	3	4	4	3	OP2 written
24/32 × 8/16	3	5	6	7	7	OP2 written
8/16 × 24/32	3	5	6	7	7	OP2L written
	N/A	3	4	4	4	OP2H written
24/32 × 24/32	3	8	10	11	11	OP2L written
	N/A	3	5	6	6	OP2H written

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2.1 Operand Registers

Operand one (OP1) has 12 registers (see Table 17-2) used to load data into the multiplier and also select the multiply mode. Writing the low word of the first operand to a given address selects the type of multiply operation to be performed, but does not start any operation. When writing a second word to a high-word register with suffix 32H, the multiplier assumes a 32-bit-wide OP1, otherwise, 16 bits are assumed. The last address written prior to writing OP2 defines the width of the first operand. For example, if MPY32L is written first followed by MPY32H, all 32 bits are used and the data width of OP1 is set to 32 bits. If MPY32H is written first followed by MPY32L, the multiplication ignores MPY32H and assumes a 16-bit-wide OP1 using the data written into MPY32L.

Repeated multiply operations may be performed without reloading OP1 if the OP1 value is used for successive operations. It is not necessary to rewrite the OP1 value to perform the operations.

Table 17-2. OP1 Registers

OP1 Register	Operation
MPY	Unsigned multiply – operand bits 0 up to 15
MPYS	Signed multiply – operand bits 0 up to 15
MAC	Unsigned multiply accumulate –operand bits 0 up to 15
MACS	Signed multiply accumulate – operand bits 0 up to 15
MPY32L	Unsigned multiply – operand bits 0 up to 15
MPY32H	Unsigned multiply – operand bits 16 up to 31
MPYS32L	Signed multiply – operand bits 0 up to 15
MPYS32H	Signed multiply – operand bits 16 up to 31
MAC32L	Unsigned multiply accumulate – operand bits 0 up to 15
MAC32H	Unsigned multiply accumulate – operand bits 16 up to 31
MACS32L	Signed multiply accumulate – operand bits 0 up to 15
MACS32H	Signed multiply accumulate – operand bits 16 up to 31

Writing the second operand to the OP2 initiates the multiply operation. Writing OP2 starts the selected operation with a 16-bit-wide second operand together with the values stored in OP1. Writing OP2L starts the selected operation with a 32-bit-wide second operand and the multiplier expects a the high word to be written to OP2H. Writing to OP2H without a preceding write to OP2L is ignored.

Table 17-3. OP2 Registers

OP2 Register	Operation
OP2	Start multiplication with 16-bit-wide OP2 – operand bits 0 up to 15
OP2L	Start multiplication with 32-bit-wide OP2 – operand bits 0 up to 15
OP2H	Continue multiplication with 32-bit-wide OP2 – operand bits 16 up to 31

For 8-bit or 24-bit operands, the operand registers can be accessed with byte instructions. Accessing the multiplier with a byte instruction during a signed operation automatically causes a sign extension of the byte within the multiplier module. For 24-bit operands, only the high word should be written as byte. If the 24-bit operands are sign-extended as defined by the register, that is used to write the low word to, because this register defines if the operation is unsigned or signed.

The high-word of a 32-bit operand remains unchanged when changing the size of the operand to 16 bit, either by modifying the operand size bits or by writing to the respective operand register. During the execution of the 16-bit operation, the content of the high-word is ignored.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: Changing of first or second operand during multiplication

By default, changing OP1 or OP2 while the selected multiply operation is being calculated renders any results invalid that are not ready at the time the new operands are changed. Writing OP2 or OP2L aborts any ongoing calculation and starts a new operation. Results that are not ready at that time are also invalid for following MAC or MACS operations.

To avoid this behavior, the MPYDLYWRTEN bit can be set to 1. Then, all writes to any MPY32 registers are delayed with MPYDLY32 = 0 until the 64-bit result is ready or with MPYDLY32 = 1 until the 32-bit result is ready. For MAC and MACS operations, the complete 64-bit result should always be ready.

See Table 17-1 for how many CPU cycles are needed until a certain result register is ready and valid for each of the different modes.

17.2.2 Result Registers

The multiplication result is always 64 bits wide. It is accessible via registers RES0 to RES3. Used with a signed operation, MPYS or MACS, the results are appropriately sign extended. If the result registers are loaded with initial values before a MACS operation, the user software must take care that the written value is properly sign extended to 64 bits.

NOTE: Changing of result registers during multiplication

The result registers must not be modified by the user software after writing the second operand into OP2 or OP2L until the initiated operation is completed.

In addition to RES0 to RES3, for compatibility with the 16×16 hardware multiplier, the 32-bit result of a 8-bit or 16-bit operation is accessible via RESLO, RESHI, and SUMEXT. In this case, the result low register RESLO holds the lower 16 bits of the calculation result and the result high register RESHI holds the upper 16 bits. RES0 and RES1 are identical to RESLO and RESHI, respectively, in usage and access of calculated results.

The sum extension register SUMEXT contents depend on the multiply operation and are listed in Table 17-4. If all operands are 16 bits wide or less, the 32-bit result is used to determine sign and carry. If one of the operands is larger than 16 bits, the 64-bit result is used.

The MPYC bit reflects the multiplier's carry as listed in Table 17-4 and, thus, can be used as 33rd or 65th bit of the result, if fractional or saturation mode is not selected. With MAC or MACS operations, the MPYC bit reflects the carry of the 32-bit or 64-bit accumulation and is not taken into account for successive MAC and MACS operations as the 33rd or 65th bit.

Table 17-4. SUMEXT and MPYC Contents

Mode	SUMEXT		MPYC	
MPY	SUMEXT is always 0000h.		MPYC is always 0.	
MPYS	SUMEXT contains the extended sign of the result.		MPYC contains the sign of the result.	
	00000h	Result was positive or zero	0	Result was positive or zero
	0FFFFh	Result was negative	1	Result was negative
MAC	SUMEXT contains the carry of the result.		MPYC contains the carry of the result.	
	0000h	No carry for result	0	No carry for result
	0001h	Result has a carry	1	Result has a carry
MACS	SUMEXT contains the extended sign of the result.		MPYC contains the carry of the result.	
	00000h	Result was positive or zero	0	No carry for result
	0FFFFh	Result was negative	1	Result has a carry

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2.2.1 MACS Underflow and Overflow

The multiplier does not automatically detect underflow or overflow in MACS mode. For example, working with 16-bit input data and 32-bit results (that is, using only RESLO and RESHI), the available range for positive numbers is 0 to 07FFF FFFFh and for negative numbers is 0FFFF FFFFh to 08000 0000h. An underflow occurs when the sum of two negative numbers yields a result that is in the range for a positive number. An overflow occurs when the sum of two positive numbers yields a result that is in the range for a negative number.

The SUMEXT register contains the sign of the result in both cases described above, 0FFFFh for a 32-bit overflow and 0000h for a 32-bit underflow. The MPYC bit in MPY32CTL0 can be used to detect the overflow condition. If the carry is different from the sign reflected by the SUMEXT register, an overflow or underflow occurred. User software must handle these conditions appropriately.

17.2.3 Software Examples

Examples for all multiplier modes follow. All 8×8 modes use the absolute address for the registers, because the assembler does not allow .B access to word registers when using the labels from the standard definitions file.

There is no sign extension necessary in software. Accessing the multiplier with a byte instruction during a signed operation automatically causes a sign extension of the byte within the multiplier module.

```
; 32x32 Unsigned Multiply
    MOV     #01234h,&MPY32L   ; Load low  word of 1st operand
    MOV     #01234h,&MPY32H   ; Load high word of 1st operand
    MOV     #05678h,&OP2L     ; Load low  word of 2nd operand
    MOV     #05678h,&OP2H     ; Load high word of 2nd operand
;   ...                       ; Process results

; 16x16 Unsigned Multiply
    MOV     #01234h,&MPY      ; Load 1st operand
    MOV     #05678h,&OP2      ; Load 2nd operand
;   ...                       ; Process results

; 8x8 Unsigned Multiply. Absolute addressing.
    MOV.B   #012h,&MPY_B      ; Load 1st operand
    MOV.B   #034h,&OP2_B      ; Load 2nd operand
;   ...                       ; Process results

; 32x32 Signed Multiply
    MOV     #01234h,&MPYS32L  ; Load low  word of 1st operand
    MOV     #01234h,&MPYS32H  ; Load high word of 1st operand
    MOV     #05678h,&OP2L     ; Load low  word of 2nd operand
    MOV     #05678h,&OP2H     ; Load high word of 2nd operand
;   ...                       ; Process results

; 16x16 Signed Multiply
    MOV     #01234h,&MPYS     ; Load 1st operand
    MOV     #05678h,&OP2      ; Load 2nd operand
;   ...                       ; Process results

; 8x8 Signed Multiply. Absolute addressing.
    MOV.B   #012h,&MPYS_B     ; Load 1st operand
    MOV.B   #034h,&OP2_B      ; Load 2nd operand
;   ...                       ; Process results
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2.4 Fractional Numbers

The MPY32 provides support for fixed-point signal processing. In fixed-point signal processing, fractional number are numbers that have a fixed number of digits after (and sometimes also before) the radix point. To classify different ranges of binary fixed-point numbers, a Q-format is used. Different Q-formats represent different locations of the radix point. Figure 17-2 shows the format of a signed Q15 number using 16 bits. Every bit after the radix point has a resolution of 1/2, and the most significant bit (MSB) is used as the sign bit. The most negative number is 08000h and the maximum positive number is 07FFFh. This gives a range from –1.0 to 0.999969482 ≈ 1.0 for the signed Q15 format with 16 bits.

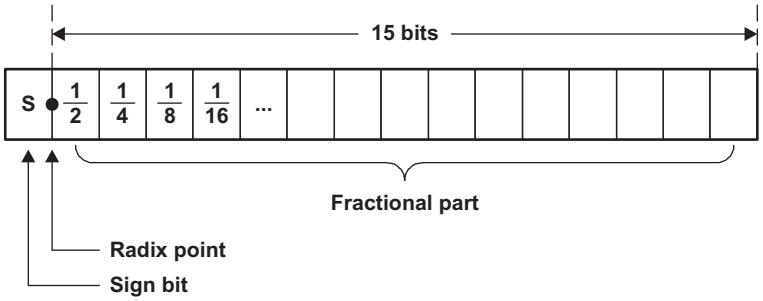


Figure 17-2. Q15 Format Representation

The range can be increased by shifting the radix point to the right as shown in Figure 17-3. The signed Q14 format with 16 bits gives a range from –2.0 to 1.999938965 ≈ 2.0.

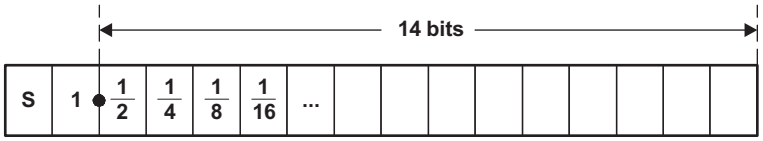


Figure 17-3. Q14 Format Representation

The benefit of using 16-bit signed Q15 or 32-bit signed Q31 numbers with multiplication is that the product of two number in the range from –1.0 to 1.0 is always in that same range.

17.2.4.1 Fractional Number Mode

Multiplying two fractional numbers using the default multiplication mode with MPYFRAC = 0 and MPYSAT = 0 gives a result with two sign bits. For example, if two 16-bit Q15 numbers are multiplied, a 32-bit result in Q30 format is obtained. To convert the result into Q15 format manually, the first 15 trailing bits and the extended sign bit must be removed. However, when the fractional mode of the multiplier is used, the redundant sign bit is automatically removed, yielding a result in Q31 format for the multiplication of two 16-bit Q15 numbers. Reading the result register RES1 gives the result as 16-bit Q15 number. The 32-bit Q31 result of a multiplication of two 32-bit Q31 numbers is accessed by reading registers RES2 and RES3.

The fractional mode is enabled with MPYFRAC = 1 in register MPY32CTL0. The actual content of the result registers is not modified when MPYFRAC = 1. When the result is accessed using software, the value is left shifted one bit, resulting in the final Q formatted result. This allows user software to switch between reading both the shifted (fractional) and the unshifted result. The fractional mode should only be enabled when required and disabled after use.

In fractional mode, the SUMEXT register contains the sign extended bits 32 and 33 of the shifted result for 16×16-bit operations and bits 64 and 65 for 32×32-bit operations – not only bits 32 or 64, respectively.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The MPYC bit is not affected by the fractional mode. It always reads the carry of the nonfractional result.

```
; Example using
; Fractional 16x16 multiplication
   BIS      #MPYFRAC,&MPY32CTL0   ; Turn on fractional mode
   MOV      &FRACT1,&MPYS         ; Load 1st operand as Q15
   MOV      &FRACT2,&OP2          ; Load 2nd operand as Q15
   MOV      &RES1,&PROD           ; Save result as Q15
   BIC      #MPYFRAC,&MPY32CTL0   ; Back to normal mode
```

Table 17-5. Result Availability in Fractional Mode (MPYFRAC = 1, MPYSAT = 0)

Operation (OP1 × OP2)	Result Ready in MCLK Cycles					After
	RES0	RES1	RES2	RES3	MPYC Bit	
8/16 × 8/16	3	3	4	4	3	OP2 written
24/32 × 8/16	3	5	6	7	7	OP2 written
8/16 × 24/32	3	5	6	7	7	OP2L written
	N/A	3	4	4	4	OP2H written
24/32 × 24/32	3	8	10	11	11	OP2L written
	N/A	3	5	6	6	OP2H written

17.2.4.2 Saturation Mode

The multiplier prevents overflow and underflow of signed operations in saturation mode. The saturation mode is enabled with MPYSAT = 1 in register MPY32CTL0. If an overflow occurs, the result is set to the most-positive value available. If an underflow occurs, the result is set to the most-negative value available. This is useful to reduce mathematical artifacts in control systems on overflow and underflow conditions. The saturation mode should only be enabled when required and disabled after use.

The actual content of the result registers is not modified when MPYSAT = 1. When the result is accessed using software, the value is automatically adjusted to provide the most-positive or most-negative result when an overflow or underflow has occurred. The adjusted result is also used for successive multiply-and-accumulate operations. This allows user software to switch between reading the saturated and the nonsaturated result.

With 16×16 operations, the saturation mode only applies to the least significant 32 bits; that is, the result registers RES0 and RES1. Using the saturation mode in MAC or MACS operations that mix 16×16 operations with 32×32, 16×32, or 32×16 operations leads to unpredictable results.

With 32×32, 16×32, and 32×16 operations, the saturated result can only be calculated when RES3 is ready.

Enabling the saturation mode does not affect the content of the SUMEXT register nor the content of the MPYC bit.

```
; Example using
; Fractional 16x16 multiply accumulate with Saturation
   ; Turn on fractional and saturation mode:
   BIS      #MPYSAT+MPYFRAC,&MPY32CTL0
   MOV      &A1,&MPYS                    ; Load A1 for 1st term
   MOV      &K1,&OP2                     ; Load K1 to get A1*K1
   MOV      &A2,&MACS                    ; Load A2 for 2nd term
   MOV      &K2,&OP2                     ; Load K2 to get A2*K2
   MOV      &RES1,&PROD                  ; Save A1*K1+A2*K2 as result
   BIC      #MPYSAT+MPYFRAC,&MPY32CTL0   ; turn back to normal
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 17-6. Result Availability in Saturation Mode (MPYSAT = 1)

Operation (OP1 × OP2)	Result Ready in MCLK Cycles					After
	RES0	RES1	RES2	RES3	MPYC Bit	
8/16 × 8/16	3	3	N/A	N/A	3	OP2 written
24/32 × 8/16	7	7	7	7	7	OP2 written
8/16 × 24/32	7	7	7	7	7	OP2L written
	4	4	4	4	4	OP2H written
24/32 × 24/32	11	11	11	11	11	OP2L written
	6	6	6	6	6	OP2H written

Figure 17-4 shows the flow for 32-bit saturation used for 16×16 bit multiplications and the flow for 64-bit saturation used in all other cases. Primarily, the saturated results depends on the carry bit MPYC and the MSB of the result. Secondly, if the fractional mode is enabled, it depends also on the two MSBs of the unshift result, that is, the result that is read with fractional mode disabled.

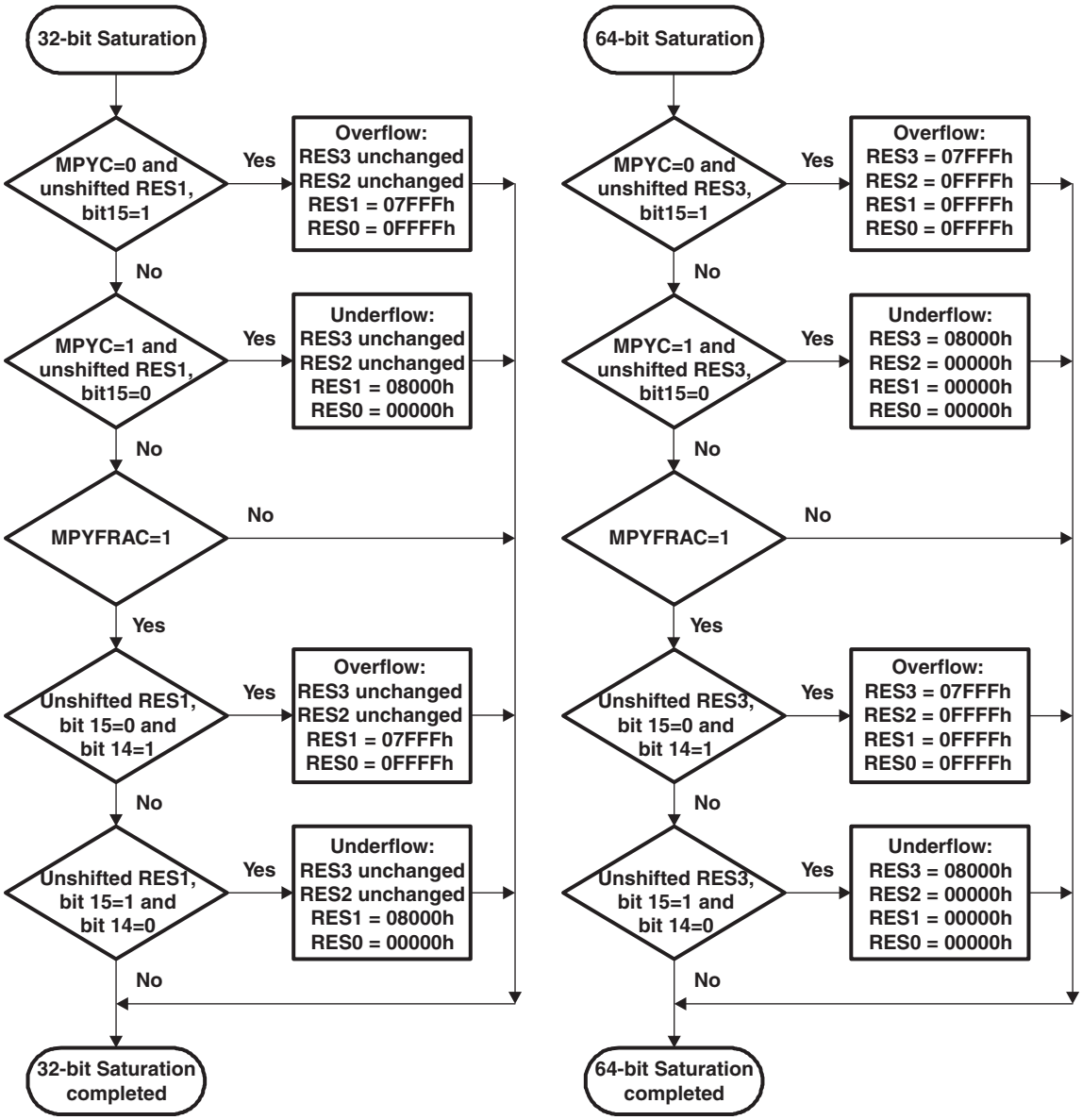


Figure 17-4. Saturation Flow Chart

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: Saturation in fractional mode

In case of multiplying –1.0 × –1.0 in fractional mode, the result of +1.0 is out of range, thus, the saturated result gives the most positive result.

When using multiply-and-accumulate operations, the accumulated values are saturated as if MPYFRAC = 0; only during read accesses to the result registers the values are saturated taking the fractional mode into account. This provides additional dynamic range during the calculation and only the end result is then saturated if needed.

The following example illustrates a special case showing the saturation function in fractional mode. It also uses the 8-bit functionality of the MPY32 module.

```
; Turn on fractional and saturation mode,
; clear all other bits in MPY32CTL0:
MOV      #MPYSAT+MPYFRAC,&MPY32CTL0
;Pre-load result registers to demonstrate overflow
MOV      #0,&RES3          ;
MOV      #0,&RES2          ;
MOV      #07FFFh,&RES1     ;
MOV      #0FA60h,&RES0     ;
MOV.B    #050h,&MACS_B     ; 8-bit signed MAC operation
MOV.B    #012h,&OP2_B      ; Start 16x16 bit operation
MOV      &RES0,R6          ; R6 = 0FFFFh
MOV      &RES1,R7          ; R7 = 07FFFh
```

The result is saturated because already the result not converted into a fractional number shows an overflow. The multiplication of the two positive numbers 00050h and 00012h gives 005A0h. 005A0h added to 07FFF FA60h results in 8000 059Fh, without MPYC being set. Because the MSB of the unmodified result RES1 is 1 and MPYC = 0, the result is saturated according Figure 17-4.

NOTE: Validity of saturated result

The saturated result is valid only if the registers RES0 to RES3, the size of OP1 and OP2, and MPYC are not modified.

If the saturation mode is used with a preloaded result, user software must ensure that MPYC in the MPY32CTL0 register is loaded with the sign bit of the written result; otherwise, the saturation mode erroneously saturates the result.

17.2.5 Putting It All Together

Figure 17-5 shows the complete multiplication flow, depending on the various selectable modes for the MPY32 module.

Copyright © 2009–2013, Texas Instruments Incorporated

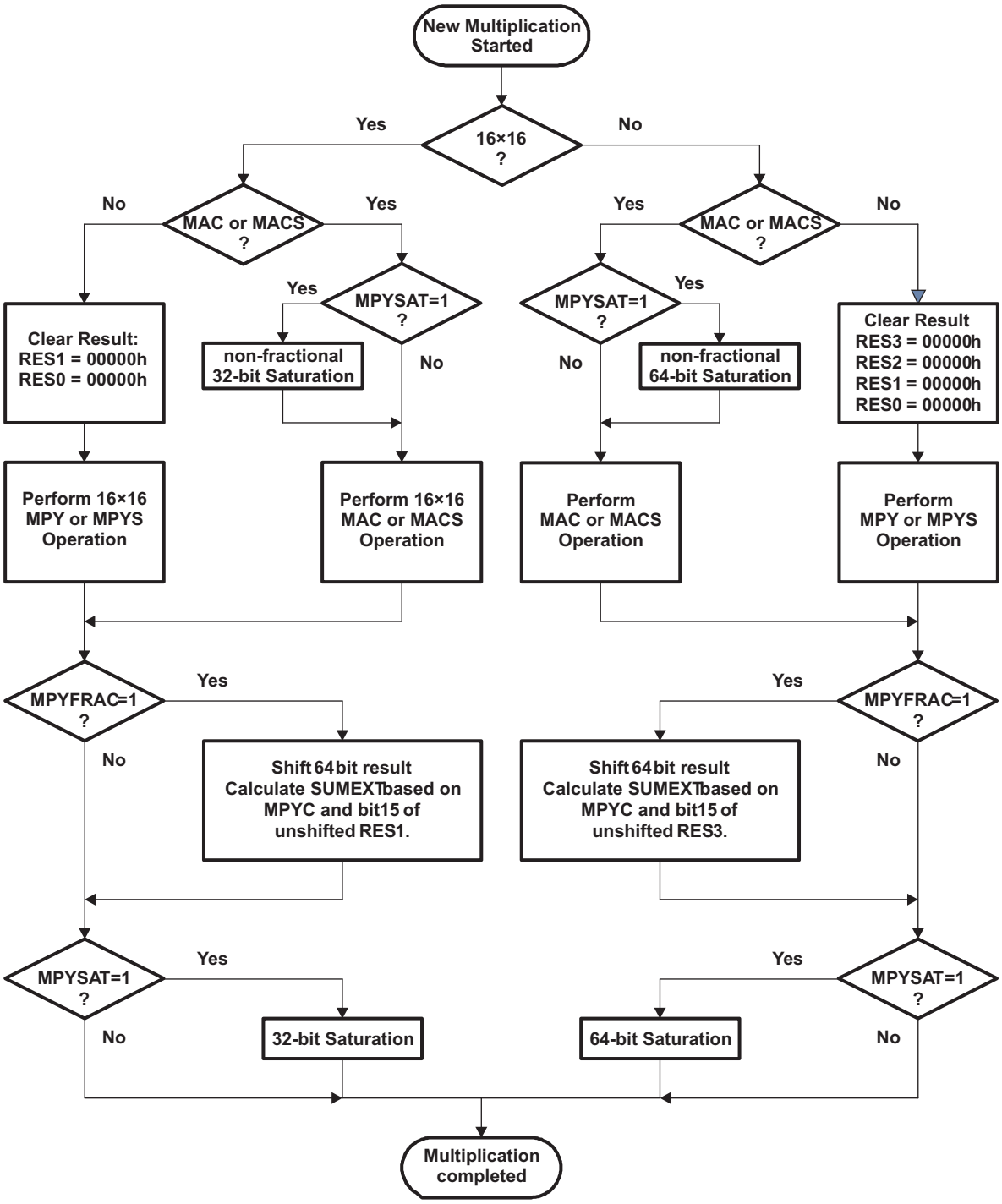


Figure 17-5. Multiplication Flow Chart

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Given the separation in processing of 16-bit operations (32-bit results) and 32-bit operations (64-bit results) by the module, it is important to understand the implications when using MAC/MACS operations and mixing 16-bit operands or results with 32-bit operands or results. User software must address these points during use when mixing these operations. The following code illustrates the issue.

```
; Mixing 32x24 multiplication with 16x16 MACS operation
   MOV      #MPYSAT,&MPY32CTL0   ; Saturation mode
   MOV      #052C5h,&MPY32L      ; Load low word of 1st operand
   MOV      #06153h,&MPY32H      ; Load high word of 1st operand
   MOV      #001ABh,&OP2L        ; Load low word of 2nd operand
   MOV.B    #023h,&OP2H_B        ; Load high word of 2nd operand
                                 ;... 5 NOPs required
   MOV      &RES0,R6             ; R6 = 00E97h
   MOV      &RES1,R7             ; R7 = 0A6EAh
   MOV      &RES2,R8             ; R8 = 04F06h
   MOV      &RES3,R9             ; R9 = 0000Dh
                                 ; Note that MPYC = 0!
   MOV      #0CCC3h,&MACS        ; Signed MAC operation
   MOV      #0FFB6h,&OP2         ; 16x16 bit operation
   MOV      &RESLO,R6            ; R6 = 0FFFFh
   MOV      &RESHI,R7            ; R7 = 07FFFh
```

The second operation gives a saturated result because the 32-bit value used for the 16×16-bit MACS operation was already saturated when the operation was started; the carry bit MPYC was 0 from the previous operation, but the MSB in result register RES1 is set. As one can see in the flow chart, the content of the result registers are saturated for multiply-and-accumulate operations after starting a new operation based on the previous results, but depending on the size of the result (32 bit or 64 bit) of the newly initiated operation.

The saturation before the multiplication can cause issues if the MPYC bit is not properly set as the following code shows.

```
   ;Pre-load result registers to demonstrate overflow
   MOV      #0,&RES3          ;
   MOV      #0,&RES2          ;
   MOV      #0,&RES1          ;
   MOV      #0,&RES0          ;
   ; Saturation mode and set MPYC:
   MOV      #MPYSAT+MPYC,&MPY32CTL0
   MOV.B    #082h,&MACS_B     ; 8-bit signed MAC operation
   MOV.B    #04Fh,&OP2_B      ; Start 16x16 bit operation
   MOV      &RES0,R6          ; R6 = 00000h
   MOV      &RES1,R7          ; R7 = 08000h
```

Even though the result registers were loaded with all zeros, the final result is saturated. This is because the MPYC bit was set, causing the result used for the multiply-and-accumulate to be saturated to 08000 0000h. Adding a negative number to it would again cause an underflow, thus, the final result is also saturated to 08000 0000h.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2.6 Indirect Addressing of Result Registers

When using indirect or indirect autoincrement addressing mode to access the result registers and the multiplier requires three cycles until result availability according to Table 17-1, at least one instruction is needed between loading the second operand and accessing the result registers:

```
; Access multiplier 16x16 results with indirect addressing
   MOV      #RES0,R5         ; RES0 address in R5 for indirect
   MOV      &OPER1,&MPY      ; Load 1st operand
   MOV      &OPER2,&OP2      ; Load 2nd operand
   NOP                       ; Need one cycle
   MOV      @R5+,&xxx        ; Move RES0
   MOV      @R5,&xxx         ; Move RES1
```

In case of a 32×16 multiplication, there is also one instruction required between reading the first result register RES0 and the second result register RES1:

```
; Access multiplier 32x16 results with indirect addressing
   MOV   #RES0,R5          ; RES0 address in R5 for indirect
   MOV   &OPER1L,&MPY32L   ; Load low word of 1st operand
   MOV   &OPER1H,&MPY32H   ; Load high word of 1st operand
   MOV   &OPER2,&OP2       ; Load 2nd operand (16 bits)
   NOP                     ; Need one cycle
   MOV   @R5+,&xxx         ; Move RES0
   NOP                     ; Need one additional cycle
   MOV   @R5,&xxx          ; Move RES1
                           ; No additional cycles required!
   MOV   @R5,&xxx          ; Move RES2
```

17.2.7 Using Interrupts

If an interrupt occurs after writing OP, but before writing OP2, and the multiplier is used in servicing that interrupt, the original multiplier mode selection is lost and the results are unpredictable. To avoid this, disable interrupts before using the MPY32, do not use the MPY32 in interrupt service routines, or use the save and restore functionality of the MPY32.

```
; Disable interrupts before using the hardware multiplier
   DINT                ; Disable interrupts
   NOP                 ; Required for DINT
   MOV    #xxh,&MPY    ; Load 1st operand
   MOV    #xxh,&OP2    ; Load 2nd operand
   EINT                ; Interrupts may be enabled before
                       ; processing results if result
                       ; registers are stored and restored in
                       ; interrupt service routines
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.2.7.1 Save and Restore

If the multiplier is used in interrupt service routines, its state can be saved and restored using the MPY32CTL0 register. The following code example shows how the complete multiplier status can be saved and restored to allow interruptible multiplications together with the usage of the multiplier in interrupt service routines. Because the state of the MPYSAT and MPYFRAC bits are unknown, they should be cleared before the registers are saved as shown in the code example.

```
; Interrupt service routine using multiplier
MPY_USING_ISR
   PUSH   &MPY32CTL0     ; Save multiplier mode, etc.
   BIC    #MPYSAT+MPYFRAC,&MPY32CTL0
                         ; Clear MPYSAT+MPYFRAC
   PUSH   &RES3          ; Save result 3
   PUSH   &RES2          ; Save result 2
   PUSH   &RES1          ; Save result 1
   PUSH   &RES0          ; Save result 0
   PUSH   &MPY32H        ; Save operand 1, high word
   PUSH   &MPY32L        ; Save operand 1, low word
   PUSH   &OP2H          ; Save operand 2, high word
   PUSH   &OP2L          ; Save operand 2, low word
                         ;
   ...                   ; Main part of ISR
                         ; Using standard MPY routines
                         ;
   POP    &OP2L          ; Restore operand 2, low word
   POP    &OP2H          ; Restore operand 2, high word
                         ; Starts dummy multiplication but
                         ; result is overwritten by
                         ; following restore operations:
   POP    &MPY32L        ; Restore operand 1, low word
   POP    &MPY32H        ; Restore operand 1, high word
   POP    &RES0          ; Restore result 0
   POP    &RES1          ; Restore result 1
   POP    &RES2          ; Restore result 2
   POP    &RES3          ; Restore result 3
   POP    &MPY32CTL0     ; Restore multiplier mode, etc.
   reti                  ; End of interrupt service routine
```

17.2.8 Using DMA

In devices with a DMA controller, the multiplier can trigger a transfer when the complete result is available. The DMA controller needs to start reading the result with MPY32RES0 successively up to MPY32RES3. Not all registers need to be read. The trigger timing is such that the DMA controller starts reading MPY32RES0 when its ready, and that the MPY32RES3 can be read exactly in the clock cycle when it is available to allow fastest access via DMA. The signal into the DMA controller is 'Multiplier ready' (see the DMA Controller chapter for details).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.3 MPY32 Registers

MPY32 registers are listed in Table 17-7. The base address can be found in the device-specific data sheet. The address offsets are listed in Table 17-7.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 17-7. MPY32 Registers

Offset	Acronym	Register Name	Type	Access	Reset
00h	MPY	16-bit operand one – multiply	Read/write	Word	Undefined
00h	MPY_L		Read/write	Byte	Undefined
01h	MPY_H		Read/write	Byte	Undefined
00h	MPY_B	8-bit operand one – multiply	Read/write	Byte	Undefined
02h	MPYS	16-bit operand one – signed multiply	Read/write	Word	Undefined
02h	MPYS_L		Read/write	Byte	Undefined
03h	MPYS_H		Read/write	Byte	Undefined
02h	MPYS_B	8-bit operand one – signed multiply	Read/write	Byte	Undefined
04h	MAC	16-bit operand one – multiply accumulate	Read/write	Word	Undefined
04h	MAC_L		Read/write	Byte	Undefined
05h	MAC_H		Read/write	Byte	Undefined
04h	MAC_B	8-bit operand one – multiply accumulate	Read/write	Byte	Undefined
06h	MACS	16-bit operand one – signed multiply accumulate	Read/write	Word	Undefined
06h	MACS_L		Read/write	Byte	Undefined
07h	MACS_H		Read/write	Byte	Undefined
06h	MACS_B	8-bit operand one – signed multiply accumulate	Read/write	Byte	Undefined
08h	OP2	16-bit operand two	Read/write	Word	Undefined
08h	OP2_L		Read/write	Byte	Undefined
09h	OP2_H		Read/write	Byte	Undefined
08h	OP2_B	8-bit operand two	Read/write	Byte	Undefined
0Ah	RESLO	16x16-bit result low word	Read/write	Word	Undefined
0Ah	RESLO_L		Read/write	Byte	Undefined
0Ch	RESHI	16x16-bit result high word	Read/write	Word	Undefined
0Eh	SUMEXT	16x16-bit sum extension register	Read	Word	Undefined
10h	MPY32L	32-bit operand 1 – multiply – low word	Read/write	Word	Undefined
10h	MPY32L_L		Read/write	Byte	Undefined
11h	MPY32L_H		Read/write	Byte	Undefined
12h	MPY32H	32-bit operand 1 – multiply – high word	Read/write	Word	Undefined
12h	MPY32H_L		Read/write	Byte	Undefined
13h	MPY32H_H		Read/write	Byte	Undefined
12h	MPY32H_B	24-bit operand 1 – multiply – high byte	Read/write	Byte	Undefined
14h	MPYS32L	32-bit operand 1 – signed multiply – low word	Read/write	Word	Undefined
14h	MPYS32L_L		Read/write	Byte	Undefined
15h	MPYS32L_H		Read/write	Byte	Undefined
16h	MPYS32H	32-bit operand 1 – signed multiply – high word	Read/write	Word	Undefined
16h	MPYS32H_L		Read/write	Byte	Undefined
17h	MPYS32H_H		Read/write	Byte	Undefined
16h	MPYS32H_B	24-bit operand 1 – signed multiply – high byte	Read/write	Byte	Undefined
18h	MAC32L	32-bit operand 1 – multiply accumulate – low word	Read/write	Word	Undefined

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 17-7. MPY32 Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset
18h	MAC32L_L		Read/write	Byte	Undefined
19h	MAC32L_H		Read/write	Byte	Undefined
1Ah	MAC32H	32-bit operand 1 – multiply accumulate – high word	Read/write	Word	Undefined
1Ah	MAC32H_L		Read/write	Byte	Undefined
1Bh	MAC32H_H		Read/write	Byte	Undefined
1Ah	MAC32H_B	24-bit operand 1 – multiply accumulate – high byte	Read/write	Byte	Undefined
1Ch	MACS32L	32-bit operand 1 – signed multiply accumulate – low word	Read/write	Word	Undefined
1Ch	MACS32L_L		Read/write	Byte	Undefined
1Dh	MACS32L_H		Read/write	Byte	Undefined
1Eh	MACS32H	32-bit operand 1 – signed multiply accumulate – high word	Read/write	Word	Undefined
1Eh	MACS32H_L		Read/write	Byte	Undefined
1Fh	MACS32H_H		Read/write	Byte	Undefined
1Eh	MACS32H_B	24-bit operand 1 – signed multiply accumulate – high byte	Read/write	Byte	Undefined
20h	OP2L	32-bit operand 2 – low word	Read/write	Word	Undefined
20h	OP2L_L		Read/write	Byte	Undefined
21h	OP2L_H		Read/write	Byte	Undefined
22h	OP2H	32-bit operand 2 – high word	Read/write	Word	Undefined
22h	OP2H_L		Read/write	Byte	Undefined
23h	OP2H_H		Read/write	Byte	Undefined
22h	OP2H_B	24-bit operand 2 – high byte	Read/write	Byte	Undefined
24h	RES0	32x32-bit result 0 – least significant word	Read/write	Word	Undefined
24h	RES0_L		Read/write	Byte	Undefined
26h	RES1	32x32-bit result 1	Read/write	Word	Undefined
28h	RES2	32x32-bit result 2	Read/write	Word	Undefined
2Ah	RES3	32x32-bit result 3 – most significant word	Read/write	Word	Undefined
2Ch	MPY32CTL0	MPY32 control register 0	Read/write	Word	Undefined
2Ch	MPY32CTL0_L		Read/write	Byte	Undefined
2Dh	MPY32CTL0_H		Read/write	Byte	00h

The registers listed in Table 17-8 are treated equally.

Table 17-8. Alternative Registers

Register	Alternative 1	Alternative 2
16-bit operand one – multiply	MPY	MPY32L
8-bit operand one – multiply	MPY_B or MPY_L	MPY32L_B or MPY32L_L
16-bit operand one – signed multiply	MPYS	MPYS32L
8-bit operand one – signed multiply	MPYS_B or MPYS_L	MPYS32L_B or MPYS32L_L
16-bit operand one – multiply accumulate	MAC	MAC32L
8-bit operand one – multiply accumulate	MAC_B or MAC_L	MAC32L_B or MAC32L_L
16-bit operand one – signed multiply accumulate	MACS	MACS32L
8-bit operand one – signed multiply accumulate	MACS_B or MACS_L	MACS32L_B or MACS32L_L
16x16-bit result low word	RESLO	RES0
16x16-bit result high word	RESHI	RES1

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

17.3.1 MPY32CTL0 Register

32-Bit Hardware Multiplier Control 0 Register

Figure 17-6. MPY32CTL0 Register

15	14	13	12	11	10	9	8
Reserved						MPYDLY32	MPYDLYWRTEN
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0
7	6	5	4	3	2	1	0
MPYOP2_32	MPYOP1_32	MPYMx		MPYSAT	MPYFRAC	Reserved	MPYC
rw	rw	rw	rw	rw-0	rw-0	rw-0	rw

Table 17-9. MPY32CTL0 Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9	MPYDLY32	RW	0h	Delayed write mode 0b = Writes are delayed until 64-bit result (RES0 to RES3) is available. 1b = Writes are delayed until 32-bit result (RES0 to RES1) is available.
8	MPYDLYWRTEN	RW	0h	Delayed write enable All writes to any MPY32 register are delayed until the 64-bit (MPYDLY32 = 0) or 32-bit (MPYDLY32 = 1) result is ready. 0b = Writes are not delayed. 1b = Writes are delayed.
7	MPYOP2_32	RW	0h	Multiplier bit width of operand 2 0b = 16 bits 1b = 32 bits
6	MPYOP1_32	RW	0h	Multiplier bit width of operand 1 0b = 16 bits 1b = 32 bits
5-4	MPYMx	RW	0h	Multiplier mode 00b = MPY – Multiply 01b = MPYS – Signed multiply 10b = MAC – Multiply accumulate 11b = MACS – Signed multiply accumulate
3	MPYSAT	RW	0h	Saturation mode 0b = Saturation mode disabled 1b = Saturation mode enabled
2	MPYFRAC	RW	0h	Fractional mode 0b = Fractional mode disabled 1b = Fractional mode enabled
1	Reserved	RW	0h	Reserved. Always reads as 0.
0	MPYC	RW	0h	Carry of the multiplier. It can be considered as 33rd or 65th bit of the result if fractional or saturation mode is not selected, because the MPYC bit does not change when switching to saturation or fractional mode. It is used to restore the SUMEXT content in MAC mode. 0b = No carry for result 1b = Result has a carry

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

REF

The REF module is a general purpose reference system that is used to generate voltage references required for other subsystems available on a given device such as digital-to-analog converters, analog-to-digital converters, comparators, etc. This chapter describes the REF module.

Topic **Page**

18.1 REF Introduction **496**
18.2 Principle of Operation **498**
18.3 REF Registers **502**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

18.1 REF Introduction

The reference module (REF) is responsible for generation of all critical reference voltages that can be used by various analog peripherals in a given device. These include, but are not necessarily limited to, the ADC10_A, ADC12_A, DAC12_A, LCD_B, and COMP_B modules dependent upon the particular device. The heart of the reference system is the bandgap from which all other references are derived by unity or non-inverting gain stages. The REFGEN subsystem consists of the bandgap, the bandgap bias, and the non-inverting buffer stage which generates the three primary voltage reference available in the system, namely 1.5 V, 2.0 V, and 2.5 V. In addition, when enabled, a buffered bandgap voltage is also available.

Features of the REF include:

- Centralized, factory trimmed bandgap with excellent PSRR, temperature coefficient, and accuracy
- 1.5-V, 2.0-V, or 2.5-V user selectable internal references
- Buffered bandgap voltage available to rest of system
- Power saving features
- Backward compatibility to existing reference system

The block diagram of the REF module (example of a device with ADC12_A) is shown in Figure 18-1.

Devices with ADC10_A might not include the reference voltage output to the external pad. Please refer to the device-specific data sheet.

Copyright © 2009–2013, Texas Instruments Incorporated

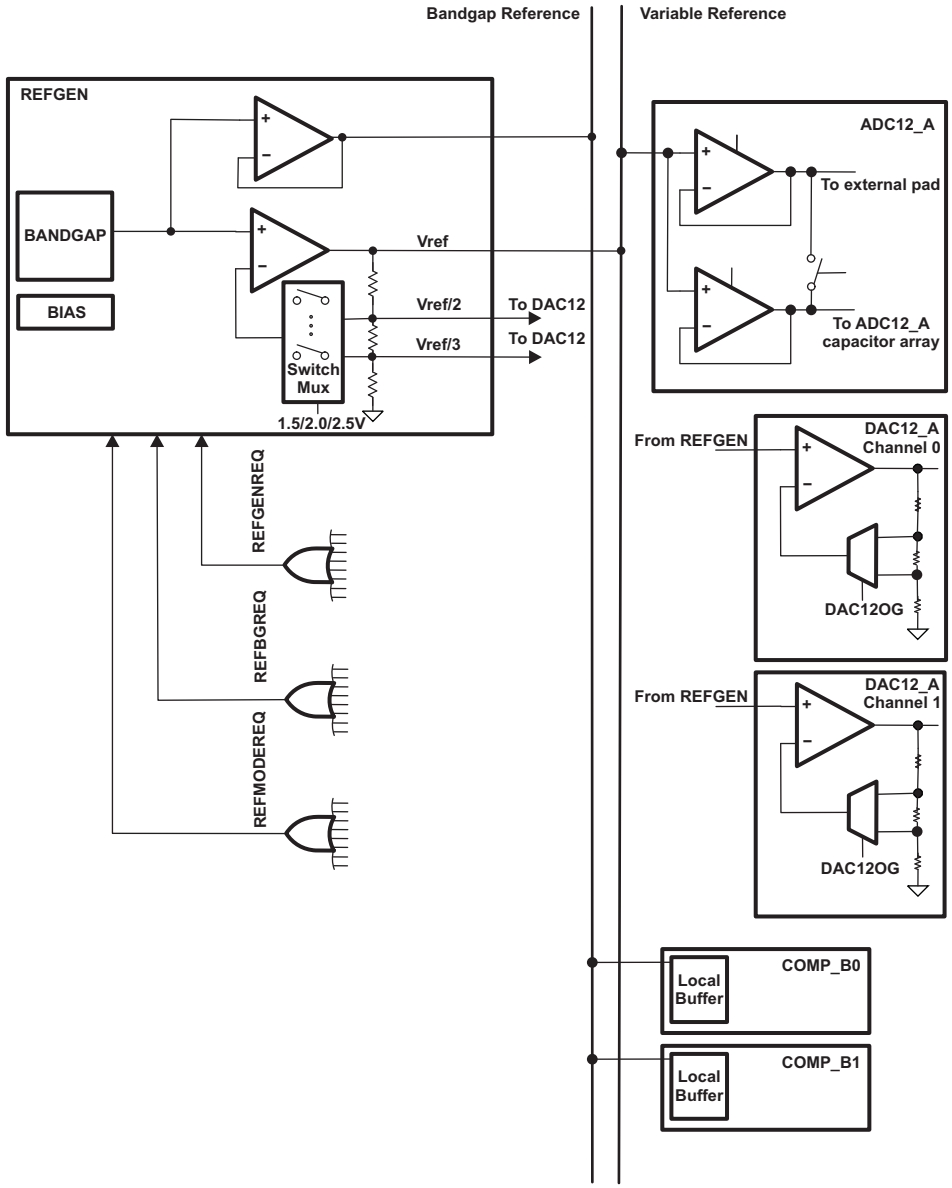


Figure 18-1. REF Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

18.2 Principle of Operation

The REF module provides all the necessary voltage references to be used by various peripheral modules throughout the system. These may include, but are not limited to, devices that contain an ADC10_A, ADC12_A, DAC12_A, LCD_B, or COMP_B.

The REFGEN subsystem contains a high-performance bandgap. This bandgap has very good accuracy (factory trimmed), low temperature coefficient, and high PSRR while operating at low power. The bandgap voltage is used to generate three voltages via a non-inverting amplifier stage, namely 1.5 V, 2.0 V, and 2.5 V. One voltage can be selected at a time. One output of the REFGEN subsystem is the variable reference line. The variable reference line provides either 1.5 V, 2.0 V, or 2.5 V to the rest of the system. A second output of the REFGEN subsystem provides a buffered bandgap reference line that can also be used by modules throughout the system. Additionally, the REFGEN supports voltage references required for the DAC12_A module, when available. Lastly, the REFGEN subsystem also includes the temperature sensor circuitry, because it is derived from the bandgap. The temperature sensor is used by an ADC to measure a voltage proportional to temperature.

18.2.1 Low-Power Operation

The REF module is capable of supporting low-power applications such as LCD generation. Many of these applications do not require a very accurate reference, compared to data conversion, yet power is of prime concern. To support these kinds of applications, the bandgap is capable of being used in a sampled mode. In sampled mode, the bandgap circuitry is clocked via the VLO at an appropriate duty cycle. This reduces the average power of the bandgap circuitry significantly, at the cost of accuracy. When not in sampled mode, the bandgap is in static mode. Its power is at its highest, but so is its accuracy.

Modules automatically can request static mode or sampled mode via their own individual request lines. In this way, the particular module determines what mode is appropriate for its proper operation and performance. Any one active module that requests static mode causes all other modules to use static mode, even if another module is requesting sampled mode. In other words, static mode always has higher priority than sampled mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

18.2.2 REFCTL

The REFCTL registers provide a way to control the reference system from one centralized set of registers. By default, REFCTL is used as the primary control of the reference system.

18.2.2.1 REFMSTR = 1

This mode is implemented in all devices with ADC10_A. Also all ADC12_A devices except for MSP430F5438 and MSP430F5438A support this mode.

Setting the reference master bit (REFMSTR = 1), allows the reference system to be controlled via the REFCTL register. This is the default setting.
Devices with ADC12_A: In this mode (REFMSTR = 1), the legacy control bits inside the ADC register set (ADC12REFON, ADC12REF2_5, ADC12TCOFF, and ADC12REFOUT) are do not care. The ADC12SR and ADC12REFBURST are still controlled via the ADC12_A, because these are very specific to the ADC12_A module. If REFMSTR is cleared, all settings in the REFCTL are do not care and the reference system is controlled completely by the legacy control bits inside the ADC12_A module.
Devices with ADC10_A: This is the only mode supported. REFMSTR must be set at all times. ADC10SR is controlled via the ADC10_A, because these are very specific to the ADC10_A module.
Table 18-1 summarizes the REFCTL bits and their effect on the REF module.

Table 18-1. REF Control of Reference System (REFMSTR = 1) (Default)

REF Register Setting	Function
REFON	Setting this bit enables the REFGEN subsystem which includes the bandgap, the bandgap bias circuitry, and the 1.5-V, 2.0-V, 2.5-V buffer. Setting this bit causes the REFGEN subsystem to remain enabled regardless of whether or not any module has requested it. Clearing this bit disables the REFGEN subsystem only when there are no pending requests for REFGEN from any module. REFON must also be set to enable the temperature sensor when required.
REFVSEL	Selects 1.5 V, 2.0 V, or 2.5 V to be present on the variable reference line when REFON = 1 or REFGEN is requested by any module.
REFOUT	Setting this bits enables the variable reference line voltage to be present external to the device via a buffer (external reference buffer).
REFTCOFF	Setting this bit disables the temperature sensor (when available) to conserve power.

18.2.2.2 REFMSTR = 0

This setting is applicable to devices with ADC12_A.
On legacy devices, the ADC12_A provided the control bits necessary to configure the reference system, namely ADC12REFON, ADC12REF2_5, ADC12TCOFF, ADC12REFOUT, ADC12SR, and ADC12REFBURST. The ADC12SR and ADC12REFBURST bits are very specific to the ADC12 operation and therefore are not included in REFCTL. All legacy control bits can still be used to configure the reference system allowing for backward compatibility by clearing REFMSTR. In this case, the REFCTL register bits are a 'do not care'.
Devices with ADC10_A do not support this mode. REFMSTR bit must not be cleared.

Table 18-2 summarizes the ADC12_A control bits and their effect on the REF module. Please see the ADC12_A module description for further details.

NOTE: Although the REF module supports using the ADC12_A bits as control for the reference system, it is recommended that the use of the new REFCTL register be used and older code migrated to this methodology. This allows the logical partitioning of the reference system to be separate from the ADC12_A system and forms a more natural partitioning for future products.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 18-2. Control of Reference System (REFMSTR = 0, ADC12_A only)

ADC12_A Register Setting	Function
ADC12REFON	Setting this bit enables the REFGEN subsystem which includes the bandgap, the bandgap bias circuitry, and the 1.5-V, 2.0-V, 2.5-V buffer. Setting this bit causes the REFGEN subsystem to remain enabled regardless if any module has requested it. Clearing this bit disables the REFGEN subsystem only when there are no pending requests for REFGEN from all modules.
ADC12REF2_5	Setting this bits causes 2.5 V to be present on the variable reference line when ADC12REFON = 1. Clearing this bit causes 1.5 V to be present on the variable reference line when ADC12REFON = 1.
ADC12REFOUT	Setting this bits enables the variable reference line voltage to be present external to the device via a buffer (external reference buffer).
ADC12TCOFF	Setting this bit disables the temperature sensor to conserve power.

As stated previously, the ADC12REFBURST does have an effect on the reference system and can be controlled via the ADC12_A. This bit is in effect regardless if REFCTL or the ADC12_A is controlling the reference system. Setting ADC12REFBURST = 1 enables burst mode when REFON = 1 and REFMSTR = 1 or when ADC12REFON = 1 and REFMSTR = 0. In burst mode, the internal buffer (ADC12REFOUT = 0) or the external buffer (ADC12REFOUT = 1) is enabled only during a conversion and disabled automatically to conserve power.

NOTE: The legacy ADC12_A bit ADC12REF2_5 only allows for selecting either 1.5 V or 2.5 V. To select 2.0 V, the REFVSEL control bits must be used (REFMSTR = 1).

18.2.3 Reference System Requests

There are three basic reference system requests that are used by the reference system. Each module can utilize these requests to obtain the proper response from the reference system. The three basic requests are REFGENREQ, REFBGREQ, and REFMODEREQ. No interaction is required by the user code. The modules select the proper requests automatically.

A reference request signal, REFGENREQ, is available as an input into the REFGEN subsystem. This signal represents a logical OR of individual requests coming from the various modules in the system that require a voltage reference to be available on the variable reference line. When a module requires a voltage reference, it asserts its corresponding REFGENREQ signal. When the REFGENREQ is asserted, the REFGEN subsystem is enabled. After the specified settling time, the variable reference line voltage is stable and ready for use. The REFVSEL settings determine which voltage is generated on the variable reference line.

In addition to the REFGENREQ, a second reference request signal, REFBGREQ is available. The REFBGREQ signal represents a logical OR of requests coming from the various modules that require the bandgap reference line. When the REFBGREQ is asserted, the bandgap, along with its bias circuitry and local buffer, is enabled if it is not already enabled by a prior request.

The REFMODEREQ request signal is available that configures the bandgap and its bias circuitry to operate in a sampled or static mode of operation. The REFMODEREQ signal basically represents a logical AND of individual requests coming from the various analog modules. In reality, a REFMODEREQ occurs only if a module's REFGENREQ or REFBGQ is also asserted, otherwise it is a do not care. When REFMODEREQ = 1, the bandgap operates in sampled mode. When a module asserts its corresponding REFMODEREQ signal, it is requesting that the bandgap operate in sampled mode. Because REMODEREQ is a logical AND of all individual requests, any modules requesting static mode cause the bandgap to operate in static mode. The BGMODE bit can be used as an indicator of static or sampled mode of operation.

18.2.3.1 REFBGACT, REFGENACT, REFGENBUSY

Any module that is using the variable reference line causes REFGENACT to be set inside the REFCTL register. This bit is read only and indicates to the user that the REFGEN is active or off. Similarly, the REFBGACT is active any time one or more modules is actively utilizing the bandgap reference line and indicates to the user that the REFBG is active or off.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The REFGENBUSY signal, when asserted, indicates that a module is using the reference and cannot have any of it settings changed. For example, during an active ADC12_A conversion, the reference voltage level should not be changed. REFGENBUSY is asserted when there is an active ADC12_A conversion (ENC = 1) or when the DAC12_A is actively converting (DAC12AMPx > 1 and DAC12SREFx = 0). REFGENBUSY when asserted, write protects the REFCTL register. This prevents the reference from being disabled or its level changed during any active conversion. Please note that there is no such protection for the DAC12_A if the ADC12_A legacy control bits are used for the reference control. If the user changes the ADC12_A settings and the DAC12_A is using the reference, the DAC12_A conversion is affected.

18.2.3.2 ADC10_A

For devices that contain an ADC10_A module, the ADC10_A module contains only one local buffer. This buffer is required when using the internal reference voltage and must be enabled and stable prior to a conversion.

In devices without a reference output buffer REFOUT must be written 0. Please refer to the device-specific data sheet.

In devices with ADC10_A the REFMSTR bit must be set at all times.

In devices with ADC10_A the REFON bit must be set if the internal reference voltage is used.

18.2.3.3 ADC12_A

For devices that contain an ADC12_A module, the ADC12_A module contains two local buffers. The larger buffer can be used to drive the reference voltage, present on the variable reference line, external to the device. This buffer has larger power consumption due to a selectable burst mode, as well as, its need to drive larger DC loads that may be present outside the device. The large buffer is enabled continuously when REFON = 1, REFOUT =1, and ADC12REFBURST = 0. When ADC12REFBURST = 1, the buffer is enabled only during an ADC conversion, shutting down automatically upon completion of a conversion to save power. In addition, when REFON = 1 and REFOUT = 1, the second smaller buffer is automatically disabled. In this case, the output of the large buffer is connected to the capacitor array via an internal analog switch. This ensures the same reference is used throughout the system. If REFON = 1 and REFOUT = 0, the internal buffer is used for ADC conversion and the large buffer remains disabled. The small internal buffer can operate in burst mode as well by setting ADC12REFBURST = 1

18.2.3.4 DAC12_A

Some devices may contain a DAC12_A module. The DAC12_A can use the 1.5 V, 2.0 V, or 2.5 V from the variable reference line for its reference. The DAC12_A can request its reference directly by the settings within the DAC12_A module itself. Therefore, if the DAC is enabled and the internal reference is selected, the DAC requests the reference voltage from the REF module. In addition, as before, setting REFON = 1 (REFMSTR = 1) or ADC12REFON = 1 (REFMSTR = 0) can enable the variable reference line independent of the DAC12_A control bits.

The REGEN subsystem provides divided versions of the variable reference line for use in the DAC12_A module. The DAC12_A module requires either /2 or /3 of the variable reference. The selection of these depends on the control bits inside the DAC12_A module (DAC12IR, DAC12OG) and is handled automatically by the REF module.

When the DAC12_A selects AVcc or VeREF+ as its reference, the DAC12_A has its own /2 and /3 resistor string available that scales the input reference appropriately based on the DAC12IR and DAC12OG settings.

18.2.3.5 LCD_B

Devices that contain an LCD use the LCD_B module. The LCD_B module requires a reference to generate the proper LCD voltages. The bandgap reference line from the REFGEN subsystem is used for this purpose. The LCD is enabled when LCDON = 1 of the LCD_B module. This causes a REFBGREQ from the LCD module to be asserted. The buffered bandgap is made available on the bandgap reference line for use inside the LCD_B module.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

18.3 REF Registers

The REF registers are listed in Table 18-3. The base address can be found in the device specific datasheet. The address offset is listed in Table 18-3.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 18-3. REF Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	REFCTL0	REFCTL0	Read/write	Word	0080h	Section 18.3.1
00h	REFCTL0_L		Read/write	Byte	80h	
01h	REFCTL0_H		Read/write	Byte	00h	

Copyright © 2009–2013, Texas Instruments Incorporated

18.3.1 REFCTL0 Register (offset = 00h) [reset = 0080h]

REF Control Register 0

Figure 18-2. REFCTL0 Register

15	14	13	12	11	10	9	8
Reserved				BGMODE	REFGENBUSY	REFBGACT	REFGENACT
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r-(0)
7	6	5	4	3	2	1	0
REFMSTR	Reserved	REFVSEL		REFTCOFF	Reserved	REFOUT	REFON
rw-(1)	r0	rw-(0)	rw-(0)	rw-(0)	r0	rw-(0)	rw-(0)

Can be modified only when REFGENBUSY = 0.

Table 18-4. REFCTL0 Register Description

Bit	Field	Type	Reset	Description
15-12	Reserved	R	0h	Reserved. Always reads as 0.
11	BGMODE	R	0h	Bandgap mode. Read only. 0b = Static mode 1b = Sampled mode
10	REFGENBUSY	R	0h	Reference generator busy. Read only. 0b = Reference generator not busy 1b = Reference generator busy
9	REFBGACT	R	0h	Reference bandgap active. Read only. 0b = Reference bandgap buffer not active 1b = Reference bandgap buffer active
8	REFGENACT	R	0h	Reference generator active. Read only. 0b = Reference generator not active 1b = Reference generator active
7	REFMSTR	RW	0h	REF master control. ADC10_A devices: Must be written 1. 0b = Reference system controlled by legacy control bits inside the ADC12_A module when available. 1b = Reference system controlled by REFCTL register. Common settings inside the ADC12_A module (if exists) are do not care.
6	Reserved	R	0h	Reserved. Always reads as 0.
5-4	REFVSEL	RW	0h	Reference voltage level select 00b = 1.5 V available when reference requested or REFON = 1 01b = 2.0 V available when reference requested or REFON = 1 10b = 2.5 V available when reference requested or REFON = 1 11b = 2.5 V available when reference requested or REFON = 1
3	REFTCOFF	RW	0h	Temperature sensor disabled 0b = Temperature sensor enabled 1b = Temperature sensor disabled to save power
2	Reserved	R	0h	Reserved. Always reads as 0.
1	REFOUT	RW	0h	Reference output buffer. ADC10_A devices without reference output buffer: Must be written 0. 0b = Reference output not available externally. 1b = Reference output available externally. If ADC12REFBURST = 0, or DAC12_A is enabled, output is available continuously. If ADC12REFBURST = 1, output is available only during an ADC12_A conversion.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 18-4. REFCTL0 Register Description (continued)

Bit	Field	Type	Reset	Description
0	REFON	RW	0h	Reference enable. ADC10_A: The ADC10_A does not support the reference request. REFON must be set if the internal reference voltage is used. 0b = Disables reference if no other reference requests are pending. 1b = Enables reference.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

ADC10_A

The ADC10_A module is a high-performance 10-bit analog-to-digital converter (ADC). This chapter describes the operation of the ADC10_A module.

Topic **Page**

19.1 ADC10_A Introduction **506**
19.2 ADC10_A Operation **508**
19.3 ADC10_A Registers **520**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.1 ADC10_A Introduction

The ADC10_A module supports fast 10-bit analog-to-digital conversions. The module implements a 10-bit SAR core with sample select control and a window comparator.

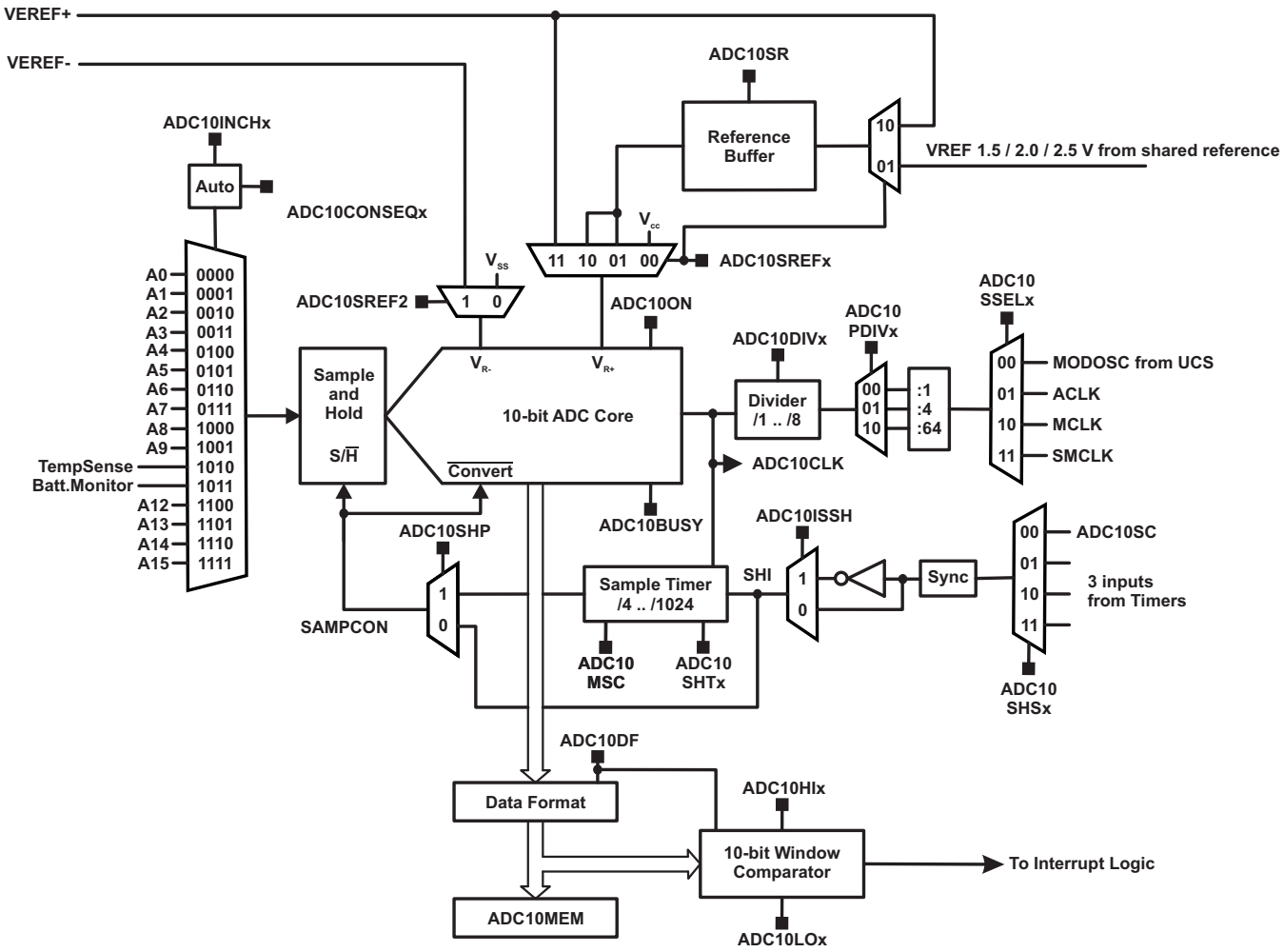
ADC10_A features include:

- Greater than 200-ksps maximum conversion rate
- Monotonic 10-bit converter with no missing codes
- Sample-and-hold with programmable sampling periods controlled by software or timers.
- Conversion initiation by software or different Timers
- Software-selectable on-chip reference using the REF module or external reference
- 12 individually configurable external input channels
- Conversion channel for temperature sensor of the REF module
- Selectable conversion clock source
- Single-channel, repeat-single-channel, sequence (autoscan), and repeat-sequence (repeated autoscan) conversion modes
- Window comparator for low power monitoring of input signals
- Interrupt vector register for fast decoding of six ADC Interrupts (ADC10IFG0, ADC10TOVIFG, ADC10OVIFG, ADC10LOIFG, ADC10INIFG, ADC10HIIFG)

The block diagram of ADC10_A is shown in Figure 19-1. The on-chip reference voltage generation is located in the reference module (see the device-specific data sheet).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.



A The MODOSC is part of the UCS. See the UCS chapter for more information.

B When using ADC10SHP = 0 no synchronisation of the trigger input is done.

Figure 19-1. ADC10_A Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2 ADC10_A Operation

The ADC10_A module is configured with user software. The setup and operation of the ADC10_A is discussed in the following sections.

19.2.1 10-Bit ADC Core

The ADC core converts an analog input to its 10-bit digital representation and stores the result in the conversion register ADC10MEM0. The core uses two programmable/selectable voltage levels (V_{R+} and V_{R-}) to define the upper and lower limits of the conversion. The digital output (N_{ADC}) is full scale (03FFh) when the input signal is equal to or higher than V_{R+}, and zero when the input signal is equal to or lower than V_{R-}. The input channel and the reference voltage levels (V_{R+} and V_{R-}) are defined in the conversion-control memory. The conversion formula for the ADC result N_{ADC} is shown in Equation 11:

$$N_{ADC} = 1023 \times \frac{Vin - V_{R-}}{V_{R+} - V_{R-}} \quad (11)$$

The ADC10_A core is configured by the control registers ADC10CTL0, ADC10CTL1 and ADC10CTL2. The core is enabled with the ADC10ON bit. The ADC10_A can be turned off when not in use to save power. With few exceptions, the ADC10_A control bits can only be modified when ADC10ENC = 0. ADC10ENC must be set to 1 before any conversion can take place.

19.2.1.1 Conversion Clock Selection

The ADC10CLK is used both as the conversion clock and to generate the sampling period when the pulse sampling mode is selected. The ADC10_A source clock is selected using the ADC10SSELx bits. Possible ADC10CLK sources are SMCLK, MCLK, ACLK, and the MODOSC. The input clock can be divided from 1–512 using both the ADC10DIVx bits and the ADC10PDIVx bits.

MODOSC, generated internally in the UCS, is in the 5-MHz range, but varies with individual devices, supply voltage, and temperature. See the device-specific data sheet for the MODOSC specification.

The user must ensure that the clock chosen for ADC10CLK remains active until the end of a conversion. If the clock is removed during a conversion, the operation does not complete and any result is invalid.

19.2.2 ADC10_A Inputs and Multiplexer

The 14 external and 2 internal analog signals are selected as the channel for conversion by the analog input multiplexer. The input multiplexer is a break-before-make type to reduce input-to-input noise injection resulting from channel switching (see Figure 19-2). The input multiplexer is also a T-switch to minimize the coupling between channels. Channels that are not selected are isolated from the A/D and the intermediate node is connected to analog ground (AV_{SS}), so that the stray capacitance is grounded to eliminate crosstalk.

The ADC10_A uses the charge redistribution method. When the inputs are internally switched, the switching action may cause transients on the input signal. These transients decay and settle before causing errant conversion.

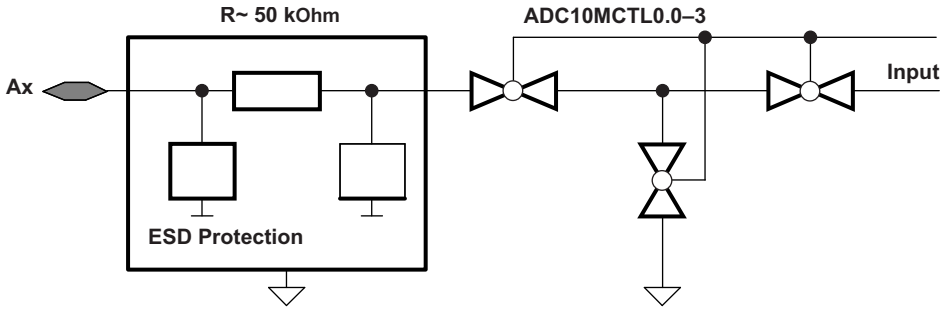


Figure 19-2. Analog Multiplexer

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.2.1 Analog Port Selection

The ADC10_A inputs are multiplexed with digital port pins. When analog signals are applied to digital gates, parasitic current can flow from V_{CC} to GND. This parasitic current occurs if the input voltage is near the transition level of the gate. Disabling the digital part of the port pin eliminates the parasitic current flow and, therefore, reduces overall current consumption. The PySELx bits provide the ability to disable the port pin input and output buffers.

```
                    ; Py.0 and Py.1 configured for analog input
BIS.B #3h,&PySEL    ; Py.1 and Py.0 ADC10_A function
```

19.2.3 Voltage Reference Generator

The ADC10_A module is designed to be used either with the on-chip reference supplied by the REF module or an externally reference voltage supplied on external pins.

The on-chip reference is capable of supplying 1.5 V, 2.0 V, and 2.5 V. The reference voltages are controlled by the control registers of the REF module (see the REF chapter for details). The internal V_{CC} can also be used as the voltage reference.

External reference voltages may be supplied for V_{R+} and V_{R-} through pins VEREF+ and VEREF-, respectively.

19.2.3.1 Internal Reference Low-Power Features

The on-chip reference is designed for low-power applications. This reference includes a band-gap voltage source and a separate reference buffer both located in the REF module. The current consumption of each is specified separately in the device-specific data sheet. The ADC10_A also contains an internal buffer for reference voltages. This buffer is automatically enabled when the internal reference is selected for VREF+, but it is also optionally available for VEREF+. The on-chip reference from the REF module must be enabled by software. Its settling time is typical 25 µs. See the device-specific data sheet and the REF chapter for further information on the on-chip reference.

The reference buffer of the ADC10_A also has selectable speed versus power settings. When the maximum conversion rate is below 50 ksps, setting ADC10SR = 1 reduces the current consumption of the buffer approximately 50%.

19.2.4 Auto Power Down

The ADC10_A is designed for low-power applications. When the ADC10_A is not actively converting, the core is automatically disabled and automatically reenabled when needed. The MODOSC is also automatically enabled when needed and disabled when not needed.

19.2.5 Sample and Conversion Timing

An analog-to-digital conversion is initiated with a rising edge of the sample input signal SHI. The source for SHI is selected with the ADC10SHSx bits and includes the following:

- ADC10SC bit and
- Three timer outputs

The polarity of the SHI signal source can be inverted with the ADC10ISSH bit. The SAMPCON signal controls the sample period and start of conversion. When SAMPCON is high, sampling is active. The high-to-low SAMPCON transition starts the analog-to-digital conversion, which requires 12 ADC10CLK cycles in 10-bit resolution mode. One additional ADC10CLK is used for the window comparator. Two different sample-timing methods are defined by control bit ADC10SHP, extended sample mode, and pulse mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.5.1 Extended Sample Mode

The extended sample mode is selected when ADC10SHP = 0. The SHI signal directly controls SAMPCON and defines the length of the sample period $t_{sample.}$ When SAMPCON is high, sampling is active. The high-to-low SAMPCON transition starts the conversion after synchronization with ADC10CLK (see Figure 19-3).

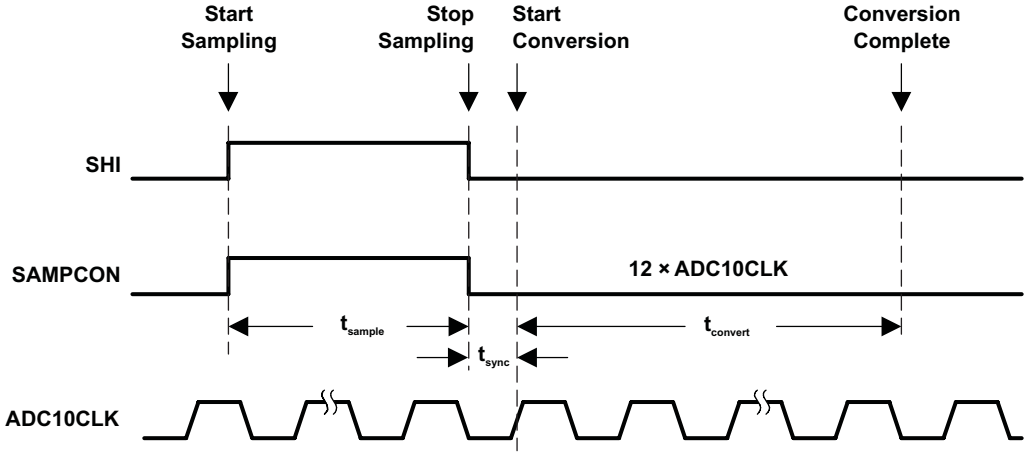


Figure 19-3. Extended Sample Mode

19.2.5.2 Pulse Sample Mode

The pulse sample mode is selected when ADC10SHP = 1. The SHI signal is used to trigger the sampling timer. The ADC10SHTx bits in ADC10CTL0 control the interval of the sampling timer that defines the SAMPCON sample period $t_{sample.}$ The sampling timer keeps SAMPCON high after synchronization with AD10CLK for a programmed interval t_{sample}. The total sampling time is t_{sample} plus t_{sync} (see Figure 19-4).

The ADC10SHTx bits select the sampling time in 4× multiples of ADC10CLK.

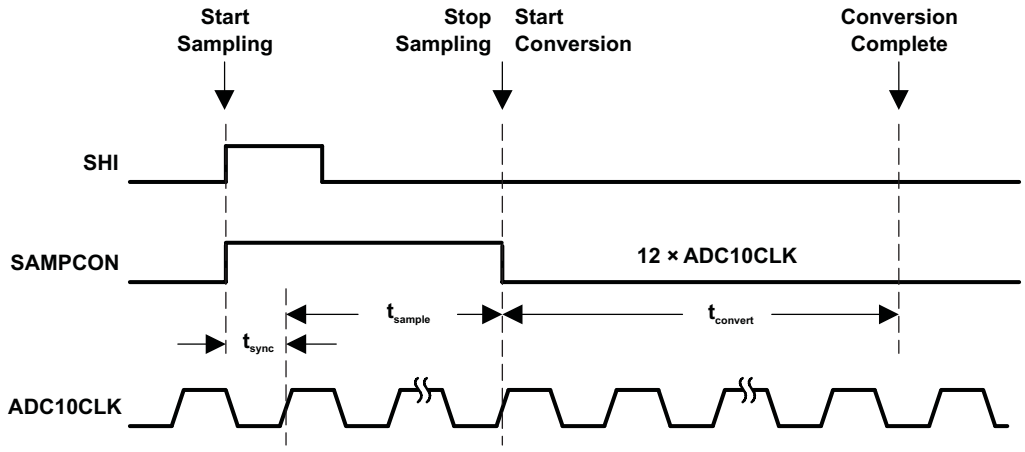


Figure 19-4. Pulse Sample Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.5.3 Sample Timing Considerations

When SAMPCON = 0, all Ax inputs are high impedance. When SAMPCON = 1, the selected Ax input can be modeled as an RC low-pass filter during the sampling time t_{sample} (see Figure 19-5). An internal MUX-on input resistance R_I (see device specific datasheet) in series with capacitor C_I (see device specific datasheet) is seen by the source. The capacitor C_I voltage V_C must be charged to within one-half LSB of the source voltage V_S for an accurate 10-bit conversion.

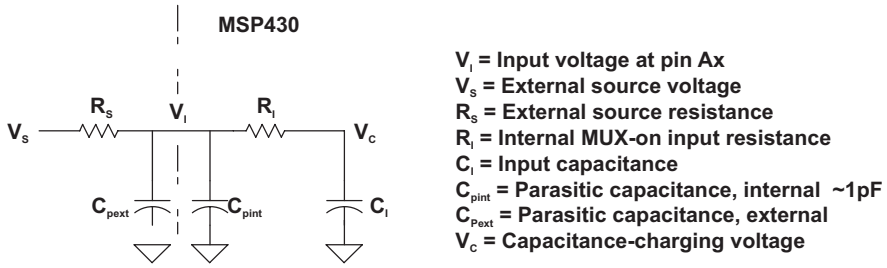


Figure 19-5. Analog Input Equivalent Circuit

The resistance of the source R_S and R_I affect t_{sample}. The minimum sample time must not be violated. Violation of the minimum sample time may cause a conversion not to take place. See the device specific datasheet for the t_{sample} limits.

19.2.6 Conversion Result

The conversion result is accessible using the ADC10MEM0 register independently of the conversion mode selected by the user. When a conversion result is written to ADC10MEM0, the ADC10IFG0 is set.

19.2.7 ADC10_A Conversion Modes

The ADC10_A has four operating modes selected by the CONSEQx bits as listed in Table 19-1.

Table 19-1. Conversion Mode Summary

ADC10CONSEQx	Mode	Operation
00	Single-channel single-conversion	A single channel is converted once.
01	Sequence-of-channels (autoscan)	A sequence of channels is converted once.
10	Repeat-single-channel	A single channel is converted repeatedly.
11	Repeat-sequence-of-channels (repeated autoscan)	A sequence of channels is converted repeatedly.

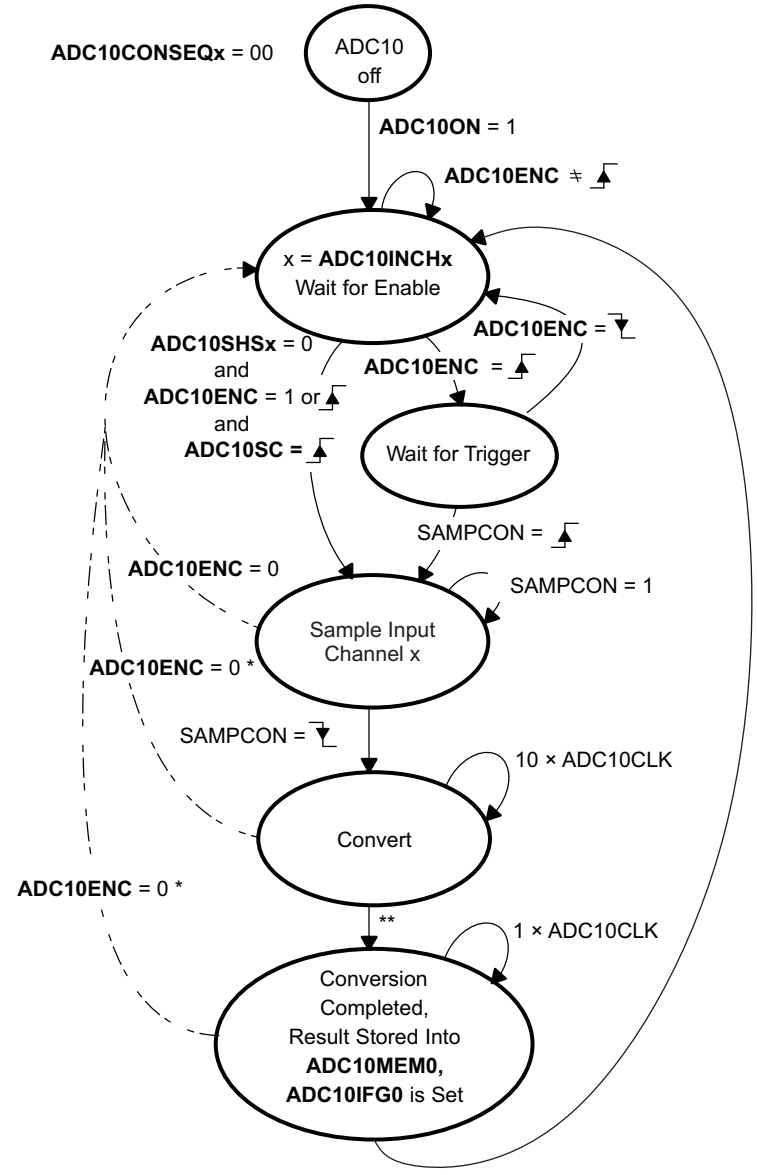
Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.7.1 Single-Channel Single-Conversion Mode

A single channel selected by ADC10INCHx is sampled and converted once. The ADC result is written to ADC10MEM0. Figure 19-6 shows the flow of the single-channel single-conversion mode. When ADC10SC triggers a conversion, successive conversions can be triggered by the ADC10SC bit. When any other trigger source is used, ADC10ENC must be toggled between each conversion.

Resetting ADC10ON bit within a conversion causes the ADC10_A to go back into "ADC10 off" state. In this case, the value of the conversion register and the value of the interrupt flags are unpredictable.



* Conversion result is unpredictable
** Two ADC10CLK cycles needed
x - pointer to the selected ADC10_A channel defined by **ADC10INCHx**
All bit- or registernames are marked with bold font, signals are noted in normal font

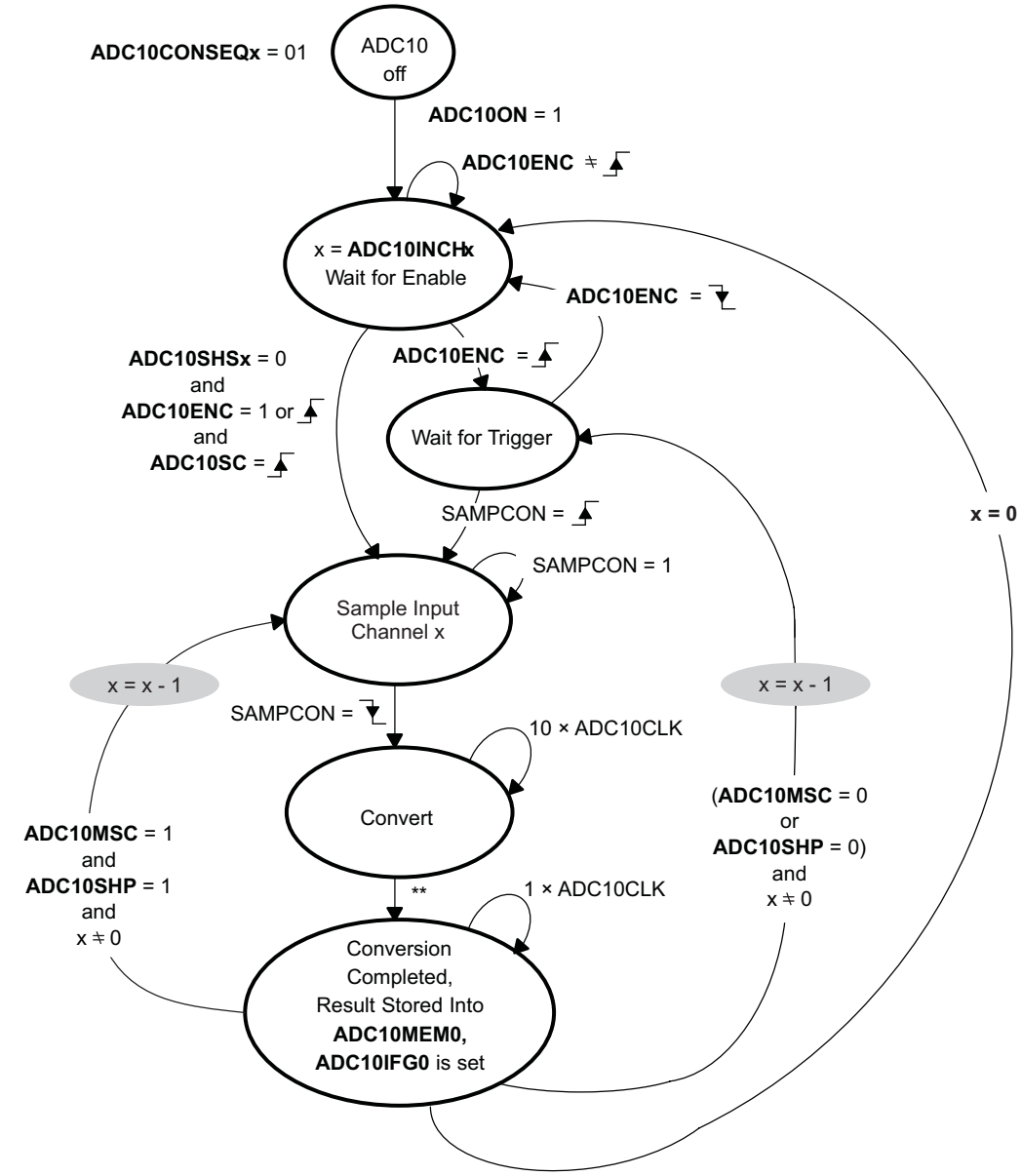
Figure 19-6. Single-Channel Single-Conversion Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.7.2 Sequence-of-Channels Mode (Autoscan Mode)

In sequence-of-channels mode, also referred to as autoscan mode, a sequence of channels is sampled and converted once. The sequence begins with the channel selected by the ADC10INCHx bits and decrements to channel A0. Each ADC result is written to ADC10MEM0. The sequence stops after conversion of channel A0. Figure 19-7 shows the sequence-of-channels mode. When ADC10SC triggers a sequence, successive sequences can be triggered by the ADC10SC bit. When any other trigger source is used, ADC10ENC must be toggled between each sequence. As in all conversion modes resetting ADC10ON bit within a conversion causes the ADC10_A to go back into "ADC10 off" state.



x - input channel Ax
** Two ADC10CLK cycles needed
All bit- or registernames are marked with bold font, signals are noted in normal font

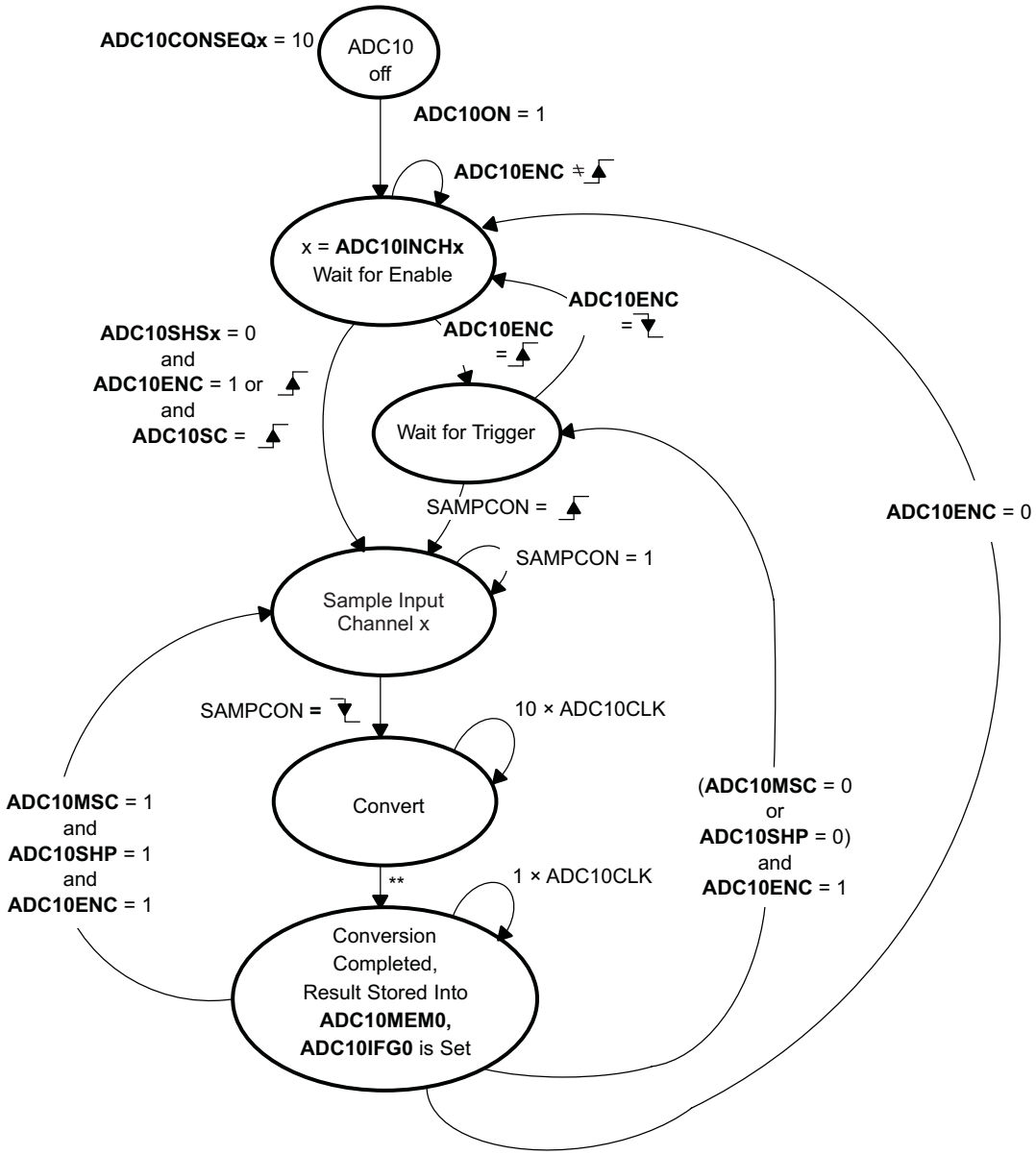
Figure 19-7. Sequence-of-Channels Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.7.3 Repeat-Single-Channel Mode

A single channel selected by ADC10INCHx is sampled and converted continuously. Each ADC result is written to ADC10MEM0. Figure 19-8 shows the repeat-single-channel mode.



x - pointer to the selected ADC10_A channel defined by **ADC10INCHx**
** Two ADC10CLK cycles needed
All bit- or registernames are marked with bold font, signals are noted in normal font

Figure 19-8. Repeat-Single-Channel Mode

Copyright © 2009–2013, Texas Instruments Incorporated

19.2.7.4 Repeat-Sequence-of-Channels Mode (Repeated Autoscan Mode)

In this mode, a sequence of channels is sampled and converted repeatedly. This mode is also referred to as repeated autoscan mode. The sequence begins with the channel selected by ADC10INCHx and decrements to channel A0. Each ADC result is written to ADC10MEM0. The sequence ends after conversion of channel A0, and the next trigger signal re-starts the sequence. Figure 19-9 shows the repeat-sequence-of-channels mode.

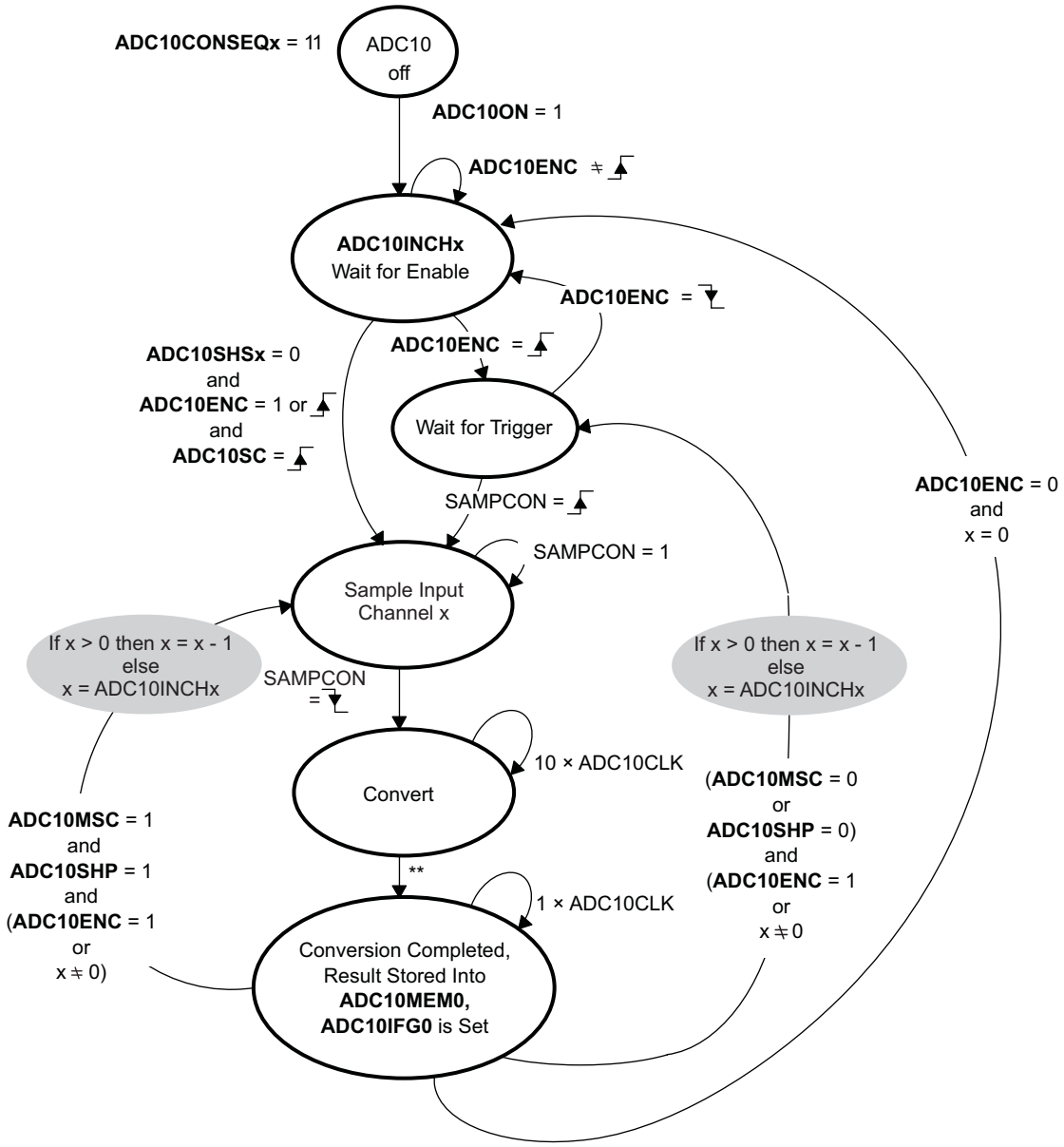


Figure 19-9. Repeat-Sequence-of-Channels Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.7.5 Using the Multiple Sample and Convert (ADC10MSC) Bit

To configure the converter to perform successive conversions automatically and as quickly as possible, a multiple sample and convert function is available. When ADC10MSC = 1, CONSEQx > 0, and the sample timer is used, the first rising edge of the SHI signal triggers the first conversion. Successive conversions are triggered automatically as soon as the prior conversion is completed. Additional rising edges on SHI are ignored until the sequence is completed in the single-sequence mode, or until the ADC10ENC bit is toggled in repeat-single-channel or repeated-sequence modes. The function of the ADC10ENC bit is unchanged when using the ADC10MSC bit.

19.2.7.6 Stopping Conversions

Stopping ADC10_A activity depends on the mode of operation. The recommended ways to stop an active conversion or conversion sequence are:

- Resetting ADC10ENC in single-channel single-conversion mode stops a conversion immediately and the results are unpredictable. For correct results, poll the busy bit until reset before clearing ADC10ENC.
- Resetting ADC10ENC during repeat-single-channel operation stops the converter at the end of the current conversion.
- Resetting ADC10ENC during a sequence or repeat-sequence mode stops the converter at the end of the sequence.
- Any conversion mode may be stopped immediately by setting the CONSEQx = 0 and resetting the ADC10ENC bit. Conversion data are unreliable.

19.2.8 Window Comparator

The window comparator allows to monitor analog signals without any CPU interaction. In the following list one can find the available Interrupt flags and the conditions, when they are asserted:

- The ADC10LO-Interrupt flag (ADC10LOIFG) gets set if the current result of the ADC10_A conversion is below the low threshold defined in register ADC10LO
- The ADC10HI-Interrupt flag (ADC10HIIFG) gets set if the current result of the ADC10_A conversion is greater than the high threshold defined in register ADC10HI
- The ADC10IN-Interrupt flag (ADC10INIFG) gets set if the current result of the ADC10_A conversion is greater than the low threshold defined in register ADC10LO and less than the high threshold defined in ADC10HI

These Interrupts are generated independently of the conversion mode selected by the user. The update of the window comparator interrupt-flags happens in parallel to the ADC10IFG0.

The user always needs to ensure, that the values in the ADC10HI and ADC10LO registers are in the correct data format. If for example the binary data format is selected (ADC10DF = 0), then the thresholds in the threshold registers ADC10HI and ADC10LO also need to be entered binary coded. Changing the ADC10DF or the ADC10RES resets the threshold registers.

The interrupt flags need to be reset by the user software. The ADC10_A only updates the flags each time a new value is available in the ADC10MEM0. This update is only a set of the corresponding interrupt flag. When the user uses the window comparator flags, it must be ensured that they are reset by software according to the application needs.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.9 Using the Integrated Temperature Sensor

To use the on-chip temperature sensor, the user selects the analog input channel ADC10INCHx = 1010. Any other configuration is done as if an external channel was selected, including reference selection, conversion-mode selection, etc. The temperature sensor is located in the REF module of the device is configured by using the REF module's control registers.

The typical temperature sensor transfer function is shown in Figure 19-10. When using the temperature sensor, the sample period must be greater than 30 µs. The temperature sensor offset error can be large and may need to be calibrated for most applications (see the device-specific data sheet for parameters).

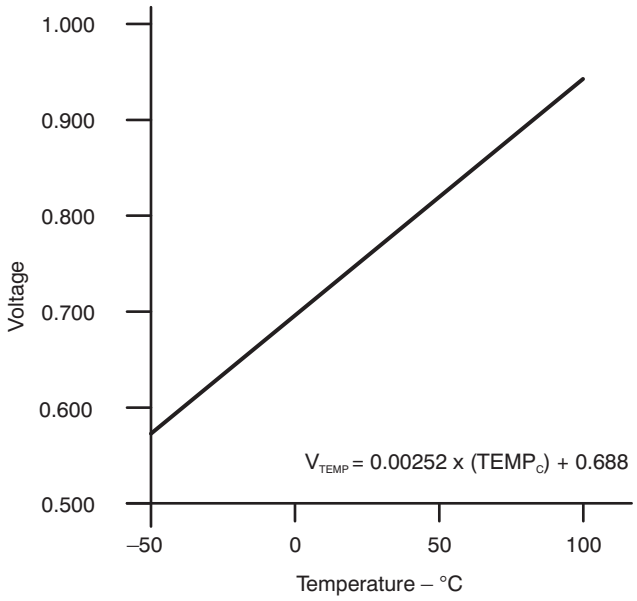


Figure 19-10. Typical Temperature Sensor Transfer Function

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.2.10 ADC10_A Grounding and Noise Considerations

As with any high-resolution ADC, appropriate printed-circuit-board layout and grounding techniques should be followed to eliminate ground loops, unwanted parasitic effects, and noise.

Ground loops are formed when return current from the A/D flows through paths that are common with other analog or digital circuitry. If care is not taken, this current can generate small, unwanted offset voltages that can add to or subtract from the reference or input voltages of the ADC. The connections shown in Figure 19-11 prevent this.

In addition to grounding, ripple and noise spikes on the power-supply lines due to digital switching or switching power supplies can corrupt the conversion result. A noise-free design using separate analog and digital ground planes with a single-point connection is recommended to achieve high accuracy.

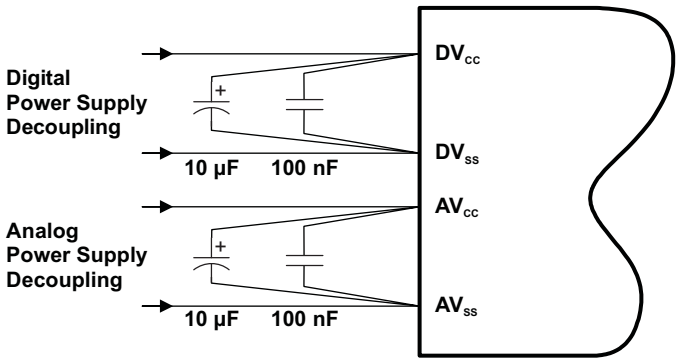


Figure 19-11. ADC10_A Grounding and Noise Considerations

19.2.11 ADC10_A Interrupts

The ADC10_A has 6 Interrupt sources:

- ADC10IFG0 : conversion ready Interrupt
- ADC10OVIFG : ADC10MEM0 overflow
- ADC10TOVIFG : ADC10_A conversion time overflow
- ADC10LOIFG, ADC10INIFG, ADC10HIIFG : window comparator Interrupt flags

The ADC10IFG0 bit is set when the ADC10MEM0 memory register is loaded with the conversion result. An Interrupt request is generated if ADC10IE0 bit and the GIE bit are set. The ADC10OV condition occurs when a conversion result is written to the ADC10MEM0 before its previous conversion result was read. The ADC10TOV condition is generated when another sample-and-conversion is requested before the current conversion is completed. The DMA is triggered after each conversion.

The window comparator Interrupt flags are set corresponding to the description in the Window Comparator section (see Section 19.2.8).

19.2.11.1 ADC10IV, Interrupt Vector Generator

All ADC10_A Interrupt sources are prioritized and combined to source a single Interrupt vector. The Interrupt vector register ADC10IV is used to determine which enabled ADC10_A Interrupt source requested an Interrupt.

The highest-priority enabled ADC10_A Interrupt generates a number in the ADC10IV register (see register description). This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled ADC10_A Interrupts do not affect the ADC10IV value.

Read access of the ADC10IV register automatically resets the highest-pending Interrupt condition and flag. Only the ADC10IFG0 is not reset by this ADC10IV read access. ADC10IFG0 is automatically reset by reading the ADC10MEM0 register or may be reset with software.

Write access of the ADC10IV register clears all pending Interrupt conditions and flags.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

If another Interrupt is pending after servicing of an Interrupt, another Interrupt is generated. For example, if the ADC10OV, ADC10HIIFG and ADC10IFG0 Interrupts are pending when the Interrupt service routine accesses the ADC10IV register, the highest priority interrupt (ADC10OV Interrupt condition) is reset automatically. After the RETI instruction of the Interrupt service routine is executed, the ADC10HIIFG generates another Interrupt.

19.2.11.2 ADC10_A Interrupt Handling Software Example

The following software example shows the recommended use of the ADC10IV. The ADC10IV value is added to the PC to automatically jump to the appropriate routine.

- ADC10IFG0, ADC10TOV, and ADC10OV: 16 cycles

```
; Interrupt handler for ADC10_A.
INT_ADC10_A                          ; Enter Interrupt Service Routine
 ADD   &ADC10IV,PC                   ; Add offset to PC
 RETI                                ; Vector  0: No Interrupt
 JMP   ADOV                          ; Vector  2: ADC10_A overflow
 JMP   ADTOV                         ; Vector  4: ADC10_A timing overflow
 JMP   ADHI                          ; Vector  6: ADC10_A window comparator high
Interrupt
 JMP   ADLO                          ; Vector  8: ADC10_A window comparator low
Interrupt
 JMP   ADIN                          ; Vector 10: ADC10_A window comparator in
Interrupt
;
; Handler for ADC10IFG0 starts here. No JMP required.
;
ADMEM  MOV &ADC10MEM0,xxx            ; Move result, flag is reset
       ...                           ; Other instruction needed?
       RETI                          ; Return ;
ADOV   ...                           ; Handle ADCMEM0 overflow
       RETI                          ; Return ;
ADTOV  ...                           ; Handle Conv. time overflow
       RETI                          ; Return ;
ADHI   ...                           ; Handle window comparator high Interrupt
       RETI                          ; Return ;
ADLO   ...                           ; Handle window comparator low Interrupt
       RETI                          ; Return ;
ADIN   ...                           ; Handle window comparator in window Interrupt
       RETI                          ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3 ADC10_A Registers

The ADC10_A registers are listed in Table 19-2. The base address of the ADC10_A can be found in the device-specific data sheet. The address offset of each ADC10_A register is given in Table 19-2.

Table 19-2. ADC10_A Registers

Offset	Acronym	Register Name	Type	Reset	Section
00h	ADC10CTL0	ADC10_A Control 0 register	Read/write	0000h	Section 19.3.1
02h	ADC10CTL1	ADC10_A Control 1 register	Read/write	0000h	Section 19.3.2
04h	ADC10CTL2	ADC10_A Control 2 register	Read/write	1000h	Section 19.3.3
06h	ADC10LO	ADC10_A Window Comparator Low Threshold register	Read/write	0000h	Section 19.3.9
08h	ADC10HI	ADC10_A Window Comparator High Threshold register	Read/write	FF03h	Section 19.3.7
0Ah	ADC10MCTL0	ADC10_A Memory Control register	Read/write	00h	Section 19.3.6
12h	ADC10MEM0	ADC10_A Conversion Memory register	Read/write	undefined	Section 19.3.4
1Ah	ADC10IE	ADC10_A Interrupt Enable register	Read/write	0000h	Section 19.3.11
1Ch	ADC10IFG	ADC10_A Interrupt Flag register	Read/write	0000h	Section 19.3.12
1Eh	ADC10IV	ADC10_A Interrupt Vector register	Read/write	0000h	Section 19.3.13

Copyright © 2009–2013, Texas Instruments Incorporated

19.3.1 ADC10CTL0 Register

ADC10_A Control Register 0

Figure 19-12. ADC10CTL0 Register

15	14	13	12	11	10	9	8
Reserved				ADC10SHTx			
r0	r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC10MSC	Reserved		ADC10ON	Reserved		ADC10ENC	ADC10SC
rw-(0)	r0	r0	rw-(0)	r0	r0	rw-(0)	rw-(0)

Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active.

Table 19-3. ADC10CTL0 Register Description

Bit	Field	Type	Reset	Description
15-12	Reserved	R	0h	Reserved. Always reads as 0.
11-8	ADC10SHTx	RW	0h	ADC10_A sample-and-hold time. These bits define the number of ADC10CLK cycles in the sampling period for the ADC10. 0000b = 4 ADC10CLK cycles 0001b = 8 ADC10CLK cycles 0010b = 16 ADC10CLK cycles 0011b = 32 ADC10CLK cycles 0100b = 64 ADC10CLK cycles 0101b = 96 ADC10CLK cycles 0110b = 128 ADC10CLK cycles 0111b = 192 ADC10CLK cycles 1000b = 256 ADC10CLK cycles 1001b = 384 ADC10CLK cycles 1010b = 512 ADC10CLK cycles 1011b = 768 ADC10CLK cycles 1100b = 1024 ADC10CLK cycles 1101b = 1024 ADC10CLK cycles 1110b = 1024 ADC10CLK cycles 1111b = 1024 ADC10CLK cycles
7	ADC10MSC	RW	0h	ADC10_A multiple sample and conversion. Valid only for sequence or repeated modes. 0b = The sampling timer requires a rising edge of the SHI signal to trigger each sample-and-convert. 1b = The first rising edge of the SHI signal triggers the sampling timer, but further sample-and-conversions are performed automatically as soon as the prior conversion is completed.
6-5	Reserved	R	0h	Reserved. Always reads as 0.
4	ADC10ON	RW	0h	ADC10_A on 0b = ADC10_A off 1b = ADC10_A on
3-2	Reserved	R	0h	Reserved. Always reads as 0.
1	ADC10ENC	RW	0h	ADC10_A enable conversion 0b = ADC10_A disabled 1b = ADC10_A enabled
0	ADC10SC	RW	0h	ADC10_A start conversion. Software-controlled sample-and-conversion start. ADC10SC and ADC10ENC may be set together with one instruction. ADC10SC is reset automatically. 0b = No sample-and-conversion-start 1b = Start sample-and-conversion

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.2 ADC10CTL1 Register

ADC10_A Control Register 1

Figure 19-13. ADC10CTL1 Register

15	14	13	12	11	10	9	8
Reserved				ADC10SHSx		ADC10SHP	ADC10ISSH
r0	r0	r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC10DIVx			ADC10SSELx		ADC10CONSEQx		ADC10BUSY
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r-(0)

Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active.

Table 19-4. ADC10CTL1 Register Description

Bit	Field	Type	Reset	Description
15-12	Reserved	R	0h	Reserved. Always reads as 0.
11-10	ADC10SHSx	RW	0h	ADC10_A sample-and-hold source select Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 00b = ADC10SC bit 01b = Timer trigger 0 - see device specific datasheet 10b = Timer trigger 1 - see device specific datasheet 11b = Timer trigger 2 - see device specific datasheet
9	ADC10SHP	RW	0h	ADC10_A sample-and-hold pulse-mode select. This bit selects the source of the sampling signal (SAMPCON) to be either the output of the sampling timer or the sample-input signal directly. Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 0b = SAMPCON signal is sourced from the sample-input signal. 1b = SAMPCON signal is sourced from the sampling timer.
8	ADC10ISSH	RW	0h	ADC10_A invert signal sample-and-hold Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 0b = The sample-input signal is not inverted. 1b = The sample-input signal is inverted.
7-5	ADC10DIVx	RW	0h	ADC10_A clock divider Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 000b = Divide by 1 001b = Divide by 2 010b = Divide by 3 011b = Divide by 4 100b = Divide by 5 101b = Divide by 6 110b = Divide by 7 111b = Divide by 8

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 19-4. ADC10CTL1 Register Description (continued)

Bit	Field	Type	Reset	Description
4-3	ADC10SSELx	RW	0h	ADC10_A clock source select Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 00b = MODCLK 01b = ACLK 10b = MCLK 11b = SMCLK
2-1	ADC10CONSEQx	RW	0h	ADC10_A conversion sequence mode select Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 00b = Single-channel, single-conversion 01b = Sequence-of-channels 10b = Repeat-single-channel 11b = Repeat-sequence-of-channels
0	ADC10BUSY	R	0h	ADC10_A busy. This bit indicates an active sample or conversion operation. 0b = No operation is active. 1b = A sequence, sample, or conversion is active.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.3 ADC10CTL2 Register

ADC10_A Control Register 2

Figure 19-14. ADC10CTL2 Register

15	14	13	12	11	10	9	8
Reserved						ADC10PDIVx	
r0	r0	r0	r0	r0	r0	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Reserved			ADC10RES	ADC10DF	ADC10SR	Reserved	
r0	r0	r0	rw-(1)	rw-(0)	rw-(0)	r0	r0

Can be modifieed only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active.

Table 19-5. ADC10CTL2 Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9-8	ADC10PDIVx	RW	0h	ADC10_A predivider. This bit predivides the selected ADC10_A clock source before it gets divided again using ADC10DIVx. 00b = Predivide by 1 01b = Predivide by 4 10b = Predivide by 64 11b = Reserved
7-5	Reserved	R	0h	Reserved. Always reads as 0.
4	ADC10RES	RW	1h	ADC10_A resolution. This bit defines the conversion result resolution. 0b = 8 bit (10 clock cycle conversion time) 1b = 10 bit (12 clock cycle conversion time)
3	ADC10DF	RW	0h	ADC10_A data read-back format. Data is always stored in the binary unsigned format. 0b = Binary unsigned. Theoretically the analog input voltage –V(REF) results in 0000h, the analog input voltage +V(REF) results in 03FFh. 1b = Signed binary (2s complement), left aligned. Theoretically the analog input voltage –V(REF) results in 8000h, the analog input voltage +V(REF) results in 7FC0h.
2	ADC10SR	RW	0h	ADC10_A sampling rate. This bit selects drive capability of the ADC10_A reference buffer for the maximum sampling rate. Setting ADC10SR reduces the current consumption of this buffer. 0b = ADC10_A buffer supports up to approximately 200 ksps. 1b = ADC10_A buffer supports up to approximately 50 ksps.
1-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.4 ADC10MEM0 Register

ADC10_A Conversion Memory Register

Figure 19-15. ADC10MEM0 Register

15	14	13	12	11	10	9	8
Reserved						Conversion_Results	
r0	r0	r0	r0	r0	r0	rw	rw
7	6	5	4	3	2	1	0
Conversion_Results							
rw	rw	rw	rw	rw	rw	rw	rw

Table 19-6. ADC10MEM0 Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	Conversion_Results	RW	undefined	The 10-bit conversion results are right justified. Bit 9 is the MSB. Bits 15–10 are 0 in 10-bit mode, and bits 15–8 are 0 in 8-bit mode. Writing to the conversion memory register corrupts the results. This data format is used if ADC10DF = 0.

19.3.5 ADC10MEM0 Register, 2s-Complement Format

ADC10_A Conversion Memory Register, 2s-Complement Format

Figure 19-16. ADC10MEM0 Register

15	14	13	12	11	10	9	8
Conversion_Results							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
Conversion_Results		Reserved					
rw	rw	r0	r0	r0	r0	r0	r0

Table 19-7. ADC10MEM0 Register Description

Bit	Field	Type	Reset	Description
15-6	Conversion_Results	RW	undefined	The 10-bit conversion results are left justified, 2s-complement format. Bit 15 is the MSB. Bits 5–0 are 0 in 10-bit mode, and bits 7–0 are 0 in 8-bit mode. This data format is used if ADC10DF = 1. The data is stored in the right-justified format and is converted to the left-justified 2s-complement format during read back. Writing to the conversion memory register corrupts the results.
5-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.6 ADC10MCTL0 Register

ADC10_A Conversion Memory Control Register

Figure 19-17. ADC10MCTL0 Register

7	6	5	4	3	2	1	0
Reserved	ADC10SREFx			ADC10INCHx			
r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active.

Table 19-8. ADC10MCTL0 Register Description

Bit	Field	Type	Reset	Description
7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	ADC10SREFx	RW	0h	Select reference. It is not recommended to change this setting while a conversion is ongoing. Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 000b = V(R+) = AVCC and V(R-) = AVSS 001b = V(R+) = VREF and V(R-) = AVSS 010b = V(R+) = VEREF+ buffered and V(R-) = AVSS 011b =V(R+) = VEREF+ and V(R-) = AVSS 100b = V(R+) = AVCC and V(R-) = VEREF- 101b = V(R+) = VREF and V(R-) = VEREF- 110b = V(R+) = VEREF+ buffered and V(R-) = VEREF- 111b = V(R+) = VEREF+ and V(R-) = VEREF-
3-0	ADC10INCHx	RW	0h	Input channel select. Writing these bits select the channel for a single-conversion or the highest channel for a sequence of conversions. Reading these bits in ADC10CONSEQ = 01,11 returns the channel currently converted. ADC10INCHx is not synchronized, so a read while the state machine is not in "wait for enable" or "wait for trigger" could lead to a wrong result. Can be modified only when ADC10ENC = 0. Resetting ADC10ENC = 0 by software and changing these fields immediately shows effect also when a conversion is active. 0000b = A0 0001b = A1 0010b = A2 0011b = A3 0100b = A4 0101b = A5 0110b = A6 0111b = A7 1000b = A8 1001b = A9 1010b = A10 1011b = A11 1100b = A12 1101b = A13 1110b = A14 1111b = A15

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.7 ADC10HI Register

ADC10_A Window Comparator High Threshold Register

Figure 19-18. ADC10HI Register

15	14	13	12	11	10	9	8
Reserved						High_Threshold	
r0	r0	r0	r0	r0	r0	rw-(1)	rw-(1)
7	6	5	4	3	2	1	0
High_Threshold							
rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)

Table 19-9. ADC10HI Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	High_Threshold	RW	3FFh	The 10-bit threshold value needs to be right justified. Bit 9 is the MSB. Bits 15–10 are 0 in 10-bit mode, and bits 15–8 are 0 in 8-bit mode. This data format is used if ADC10DF = 0.

19.3.8 ADC10HI Register, 2s-Complement Format

ADC10_A Window Comparator High Threshold Register, 2s-Complement Format

Figure 19-19. ADC10HI Register

15	14	13	12	11	10	9	8
High_Threshold							
rw-(0)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)	rw-(1)
7	6	5	4	3	2	1	0
High_Threshold		Reserved					
rw-(1)	rw-(1)	r0	r0	r0	r0	r0	r0

Table 19-10. ADC10HI Register Description

Bit	Field	Type	Reset	Description
15-6	High_Threshold	RW	1FFh	The 10-bit threshold value needs to be left justified if 2s-complement format is chosen. Bit 15 is the MSB. Bits 5–0 are 0 in 10-bit mode, and bits 7–0 are 0 in 8-bit mode. This data format is used if ADC10DF = 1.
5-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.9 ADC10LO Register

ADC10_A Window Comparator Low Threshold Register

Figure 19-20. ADC10LO Register

15	14	13	12	11	10	9	8
Reserved						Low_Threshold	
r0	r0	r0	r0	r0	r0	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Low_Threshold							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 19-11. ADC10LO Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	Low_Threshold	RW	0h	The 10-bit threshold value needs to be right justified. Bit 9 is the MSB. Bits 15–10 are 0 in 10-bit mode, and bits 15–8 are 0 in 8-bit mode. This data format is used if ADC10DF = 0.

19.3.10 ADC10LO Register, 2s-Complement Format

ADC10_A Window Comparator Low Threshold Register, 2s-Complement Format

Figure 19-21. ADC10LO Register

15	14	13	12	11	10	9	8
Low_Threshold							
rw-(1)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
Low_Threshold		Reserved					
rw-(0)	rw-(0)	r0	r0	r0	r0	r0	r0

Table 19-12. ADC10LO Register Description

Bit	Field	Type	Reset	Description
15-6	Low_Threshold	RW	200h	The 10-bit threshold value needs to be left justified if 2s-complement format is chosen. Bit 15 is the MSB. Bits 5–0 are 0 in 10-bit mode, and bits 7–0 are 0 in 8-bit mode. This data format is used if ADC10DF = 1.
5-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.11 ADC10IE Register

ADC10_A Interrupt Enable Register

Figure 19-22. ADC10IE Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved		ADC10TOVIE	ADC10OVIE	ADC10HIIE	ADC10LOIE	ADC10INIE	ADC10IE0
r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 19-13. ADC10IE Register Description

Bit	Field	Type	Reset	Description
15-6	Reserved	R	0h	Reserved. Always reads as 0.
5	ADC10TOVIE	RW	0h	ADC10_A conversion-time-overflow Interrupt enable. 0b = Conversion time overflow Interrupt disabled 1b = Conversion time overflow Interrupt enabled
4	ADC10OVIE	RW	0h	ADC10MEM0 overflow Interrupt enable. 0b = Overflow Interrupt disabled 1b = Overflow Interrupt enabled
3	ADC10HIIE	RW	0h	Interrupt enable for the above upper threshold Interrupt of the Window comparator. 0b = Above upper threshold Interrupt disabled 1b = Above upper threshold Interrupt enabled
2	ADC10LOIE	RW	0h	Interrupt enable for the below lower threshold Interrupt of the Window comparator. 0b = Below lower threshold Interrupt disabled 1b = Below lower threshold Interrupt enabled
1	ADC10INIE	RW	0h	Interrupt enable for the inside of window Interrupt of the Window comparator. 0b = Inside of window Interrupt disabled 1b = Inside of window Interrupt enabled
0	ADC10IE0	RW	0h	Interrupt enable. This bits enable or disable the Interrupt request for a completed ADC10_A conversion. 0b = Interrupt disabled 1b = Interrupt enabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.12 ADC10IFG Register

ADC10_A Interrupt Flag Register

Figure 19-23. ADC10IFG Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved		ADC10TOVIFG	ADC10OVIFG	ADC10HIIFG	ADC10LOIFG	ADC10INIFG	ADC10IFG0
r0	r0	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 19-14. ADC10IFG Register Description

Bit	Field	Type	Reset	Description
15-6	Reserved	R	0h	Reserved. Always reads as 0.
5	ADC10TOVIFG	RW	0h	The ADC10TOVIFG is set when an ADC10_A conversion is triggered before the actual conversion has completed. 0b = No Interrupt pending 1b = Interrupt pending
4	ADC10OVIFG	RW	0h	The ADC10OVIFG is set when the ADC10MEM0 register is written before the last conversion result has been read. 0b = No Interrupt pending 1b = Interrupt pending
3	ADC10HIIFG	RW	0h	The ADC10HIIFG is set when the result of the current ADC10_A conversion is greater than the upper threshold defined by the Window Comparators upper threshold register. 0b = No Interrupt pending 1b = Interrupt pending
2	ADC10LOIFG	RW	0h	The ADC10LOIFG is set when the result of the current ADC10_A conversion is below the lower threshold defined by the Window Comparators lower threshold register. 0b = No Interrupt pending 1b = Interrupt pending
1	ADC10INIFG	RW	0h	The ADC10INIFG is set when the result of the current ADC10_A conversion is within the thresholds defined by the Window Comparators threshold registers. 0b = No Interrupt pending 1b = Interrupt pending
0	ADC10IFG0	RW	0h	The ADC10IFG0 is set when an ADC10_A conversion is completed. This bit gets reset, when the ADC10MEM0 get read, or may be reset by software. 0b = No Interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

19.3.13 ADC10IV Register

ADC10_A Interrupt Vector Register

Figure 19-24. ADC10IV Register

15	14	13	12	11	10	9	8
ADC10IVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
ADC10IVx							
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r0

Table 19-15. ADC10IV Register Description

Bit	Field	Type	Reset	Description
15-0	ADC10IVx	R	0h	ADC10_A Interrupt vector value. It generates an value that can be used as address offset for fast interrupt service routine handling. Writing to this register clears all pending interrupt flags. 00h = No interrupt pending 02h = Interrupt Source: ADC10MEM0 overflow; Interrupt Flag: ADC10OVIFG; Interrupt Priority: Highest 04h = Interrupt Source: Conversion time overflow; Interrupt Flag: ADC10TOVIFG 06h = Interrupt Source: ADC10HI Interrupt flag; Interrupt Flag: ADC10HIIFG 08h = Interrupt Source: ADC10LO Interrupt flag; Interrupt Flag: ADC10LOIFG 0Ah = Interrupt Source: ADC10IN Interrupt flag; Interrupt Flag: ADC10INIFG 0Ch = Interrupt Source: ADC10_A memory Interrupt flag; Interrupt Flag: ADC10IFG0; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

ADC12_A

The ADC12_A module is a high-performance 12-bit analog-to-digital converter (ADC). This chapter describes the operation of the ADC12_A module.

Topic **Page**

20.1 ADC12_A Introduction 533
20.2 ADC12_A Operation 536
20.3 ADC12_A Registers 550

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.1 ADC12_A Introduction

The ADC12_A module supports fast 12-bit analog-to-digital conversions. The module implements a 12-bit SAR core, sample select control, reference generator (MSP430F54xx (non-A only) – in other devices, separate REF module), and a 16-word conversion-and-control buffer. The conversion-and-control buffer allows up to 16 independent analog-to-digital converter (ADC) samples to be converted and stored without any CPU intervention.

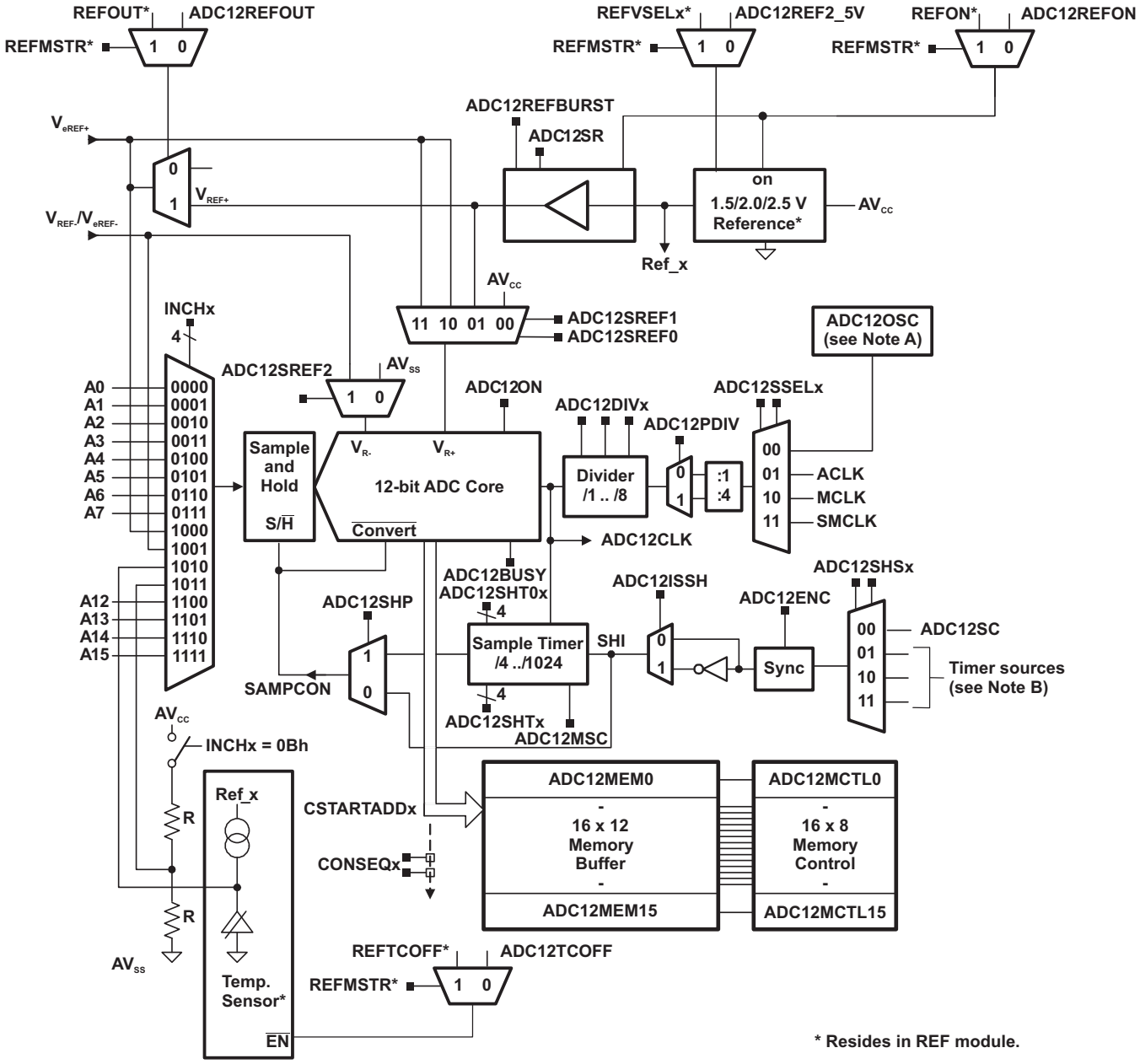
ADC12_A features include:

- Greater than 200-ksps maximum conversion rate
- Monotonic 12-bit converter with no missing codes
- Sample-and-hold with programmable sampling periods controlled by software or timers
- Conversion initiation by software or timers
- Software-selectable on-chip reference voltage generation (MSP430F54xx (non-A only): 1.5 V or 2.5 V, all other devices: 1.5 V, 2.0 V, or 2.5 V)
- Software-selectable internal or external reference
- Up to 12 individually configurable external input channels
- Conversion channels for internal temperature sensor, AV_{CC}, and external references
- Independent channel-selectable reference sources for both positive and negative references
- Selectable conversion clock source
- Single-channel, repeat-single-channel, sequence (autoscan), and repeat-sequence (repeated autoscan) conversion modes
- ADC core and reference voltage can be powered down separately
- Interrupt vector register for fast decoding of 18 ADC interrupts
- 16 conversion-result storage registers

The block diagram of ADC12_A is shown in Figure 20-1. In MSP430F54xx (non-A only), the reference generator is located in the ADC12_A module itself. In other devices, the reference generator is located in the reference module, REF. See the REF module chapter and the device-specific data sheet for further details. Figure 20-1 shows the block diagram for devices that have the REF module available. Figure 20-2 shows the block diagram for the MSP430F54xx (non-A only) which does not incorporate the REF module.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

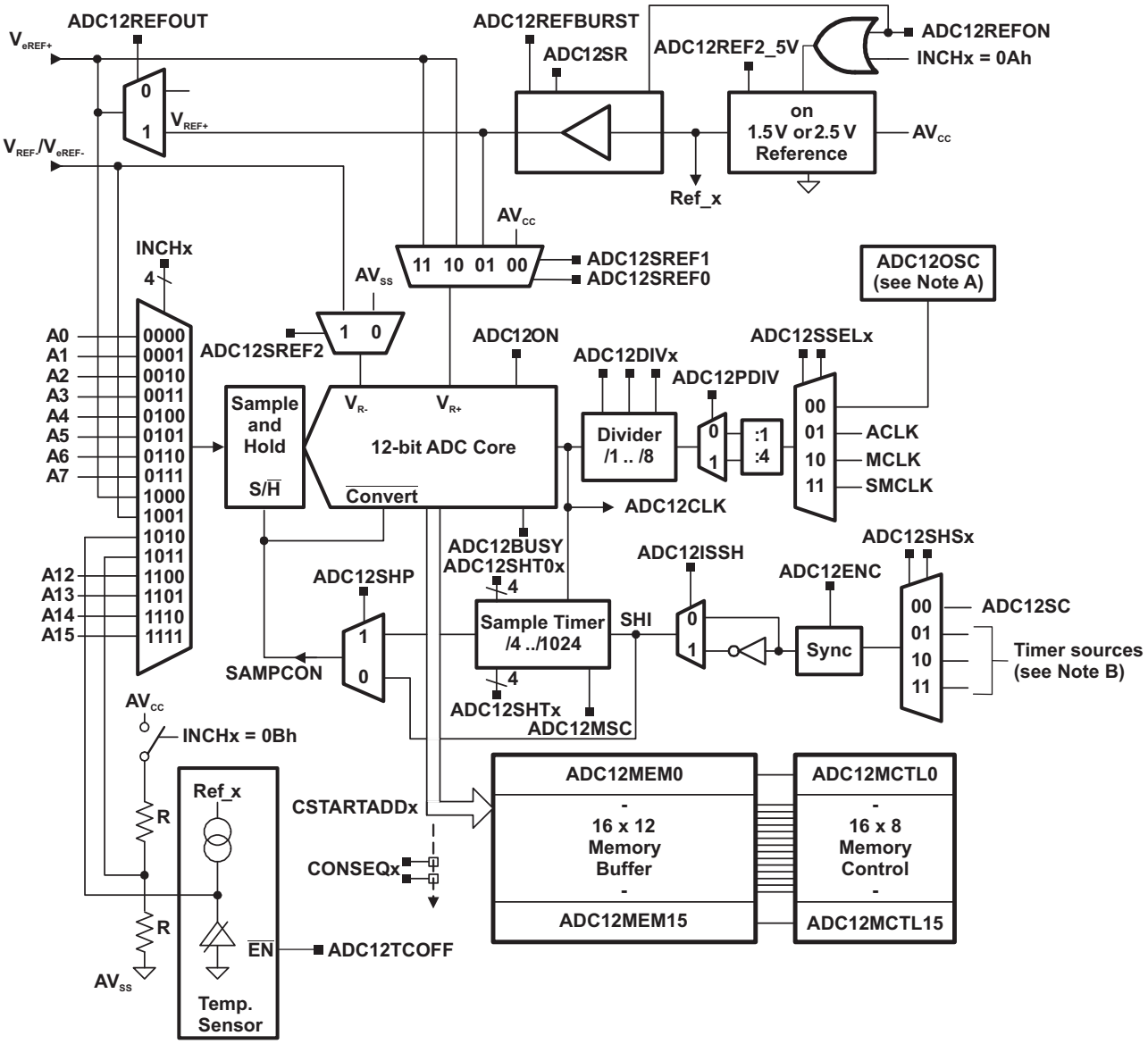


A The MODOSC is part of the UCS. See the UCS chapter for more information.

B See the device-specific data sheet for timer sources available.

Figure 20-1. ADC12_A Block Diagram (Devices With REF Module)

Copyright © 2009–2013, Texas Instruments Incorporated



A The MODOSC is part of the UCS. See the UCS chapter for more information.

B See the device-specific data sheet for timer sources available.

Figure 20-2. ADC12_A MSP430F54xx (non-A) Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

20.2 ADC12_A Operation

The ADC12_A module is configured with user software. The setup and operation of the ADC12_A is discussed in the following sections.

20.2.1 12-Bit ADC Core

The ADC core converts an analog input to its 12-bit digital representation and stores the result in conversion memory. The core uses two programmable and selectable voltage levels (V_{R+} and V_{R-}) to define the upper and lower limits of the conversion. The digital output (N_{ADC}) is full scale (0FFFh) when the input signal is equal to or higher than V_{R+}. The digital output (N_{ADC}) is zero when the input signal is equal to or lower than V_{R-}. The input channel and the reference voltage levels (V_{R+} and V_{R-}) are defined in the conversion-control memory. The conversion formula for the ADC result N_{ADC} is:

$$N_{ADC} = 4095 \times \frac{Vin - V_{R-}}{V_{R+} - V_{R-}}$$

The ADC12_A core is configured by two control registers, ADC12CTL0 and ADC12CTL1. The core is enabled with the ADC12ON bit. The ADC12_A can be turned off when it is not in use to save power. With few exceptions, the ADC12_A control bits can be modified only when ADC12ENC = 0. ADC12ENC must be set to 1 before any conversion can take place.

20.2.1.1 Conversion Clock Selection

The ADC12CLK is used both as the conversion clock and to generate the sampling period when the pulse sampling mode is selected. The ADC12_A source clock is selected using the predivider controlled by the ADC12PDIV bit and the divider using the ADC12SSELx bits. The input clock can be divided from 1 to 32 using both the ADC12DIVx bits and the ADC12PDIV bit. Possible ADC12CLK sources are SMCLK, MCLK, ACLK, and the ADC12OSC.

The ADC12OSC in the block diagram (see Figure 20-1) refers to the MODOSC 5-MHz oscillator from the UCS (see the UCS module for more information) which can vary with individual devices, supply voltage, and temperature. See the device-specific data sheet for the ADC12OSC specification.

The user must ensure that the clock chosen for ADC12CLK remains active until the end of a conversion. If the clock is removed during a conversion, the operation does not complete and the results are invalid.

20.2.2 ADC12_A Inputs and Multiplexer

The 12 external and 4 internal analog signals are selected as the channel for conversion by the analog input multiplexer. The input multiplexer is a break-before-make type to reduce input-to-input noise injection resulting from channel switching (see Figure 20-3). The input multiplexer is also a T-switch to minimize the coupling between channels. Channels that are not selected are isolated from the A/D and the intermediate node is connected to analog ground (AV_{SS}), so that the stray capacitance is grounded to eliminate crosstalk.

The ADC12_A uses the charge redistribution method. When the inputs are internally switched, the switching action may cause transients on the input signal. These transients decay and settle before causing errant conversion.

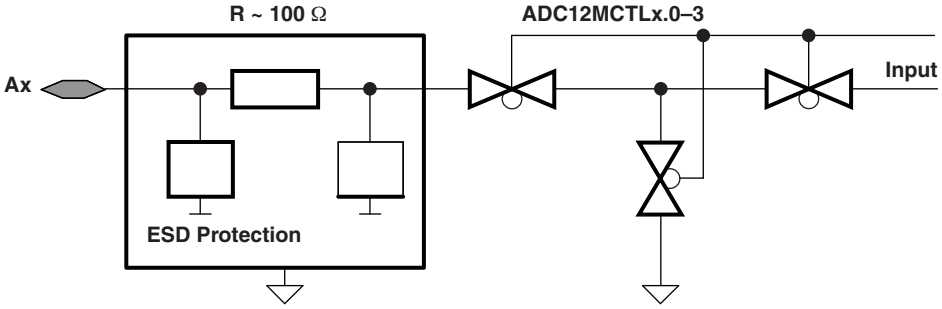


Figure 20-3. Analog Multiplexer

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.2.1 Analog Port Selection

The ADC12_A inputs are multiplexed with digital port pins. When analog signals are applied to digital gates, parasitic current can flow from V_{CC} to GND. This parasitic current occurs if the input voltage is near the transition level of the gate. Disabling the digital part of the port pin eliminates the parasitic current flow and, therefore, reduces overall current consumption. The PySELx bits provide the ability to disable the port pin input and output buffers.

```
; Py.0 and Py.1 configured for analog input
      BIS.B #3h,&PySEL   ; Py.1 and Py.0 ADC12_A function
```

20.2.3 Voltage Reference Generator

The ADC12_A modules have a separate reference module (REF) that supplies three selectable voltage levels, 1.5 V, 2.0 V, and 2.5 V to the ADC12_A. Any of these voltages may be used internally and externally on pin VREF+. The internal AV_{CC} can also be used as the reference.

The ADC12_A module of the MSP430F54xx devices (non-A only) does not use the REF module and only has two selectable voltage levels, 1.5 V and 2.5 V. The internal AV_{CC} can also be used as the reference.

On devices with the REF module, the voltage reference settings can be controlled either by the REF module or by the ADC12_A module. This is to allow for backward compatability with older families. This is handled by the REFMSTR bit in the REF module. If REFMSTR = 1 (default), the REF module registers control the reference settings. If REFMSTR = 0, the ADC12_A reference setting define the reference voltage of the ADC12_A module. Four control settings that reside in the ADC12_A can be controlled also by four corresponding settings in the REF module: ADC12REF2_5V (REFVSEL), ADC12REFON (REFON), ADC12REFOUT (REFOUT), and ADC12TCOFF (REFTCOFF), respectively. When REFMSTR = 1, ADC12REF2_5V, ADC12REFON, ADC12REFOUT, and ADC12TCOFF are do not care. Similarly, when REFMSTR = 0, REFVSEL, REFON, REFOUT, and REFTCOFF are do not care. See the REF module chapter for further details.

On devices with the REF module, to use the ADC12_A reference control bits, set REFMSTR = 0. In this case, setting ADC12REFON = 1 enables the reference voltage of the ADC12_A module. When ADC12REF2_5V = 1, the internal reference is 2.5 V; when ADC12REF2_5V = 0, the reference is 1.5 V. Similarly, on devices with the REF module, to use the REF module reference control bits, set REFMSTR = 1. In this case, setting REFON = 1 of the REF module enables the reference voltage. The REFVSEL bits of the REF module can be used to select either 1.5 V, 2.0 V, or 2.5 V.The reference can be turned off to save power when not in use. On the MSP430F54xx devices (non-A only), as stated previously, the REF module is not present and behaves the same as devices the REF module with REFMSTR = 0.

External references may be supplied for V_{R+} and V_{R-} through pins VREF+/VeREF+ and VREF-/VeREF-, respectively.

External storage capacitors are required only if ADC12REFOUT = 1 (REFOUT = 1 when using REF module) and the reference voltage is made available at the pins.

20.2.3.1 Internal Reference Low-Power Features

The ADC12_A internal reference generator is designed for low-power applications. The reference generator includes a bandgap voltage source and a separate buffer. The current consumption and settling time of each is specified separately in the device-specific data sheet. When ADC12REFON = 1 (REFON = 1 when using REF module), both are enabled; when ADC12REFON = 0 (REFON = 0 when using REF module), both are disabled.

When ADC12REFON = 1 (REFON = 1 when using REF module) and ADC12REFBURST = 1 but no conversion is active, the buffer is automatically disabled and automatically reenabled when needed. When the buffer is disabled, it consumes no current. In this case, the bandgap voltage source remains enabled.

The ADC12REFBURST bit controls the operation of the reference buffer. When ADC12REFBURST = 1, the buffer is automatically disabled when the ADC12_A is not actively converting, and is automatically reenabled when needed. When ADC12REFBURST = 0, the buffer is on continuously. This allows the reference voltage to be present outside the device continuously if ADC12REFOUT = 1 (REFOUT = 1 when using REF module).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The internal reference buffer also has selectable speed versus power settings. When the maximum conversion rate is below 50 ksps, setting ADC12SR = 1 reduces the current consumption of the buffer by approximately 50%.

20.2.4 Auto Power Down

The ADC12_A is designed for low-power applications. When the ADC12_A is not actively converting, the core is automatically disabled, and it is automatically reenabled when needed. The MODOSC is also automatically enabled when needed and disabled when not needed.

20.2.5 Sample and Conversion Timing

An analog-to-digital conversion is initiated with a rising edge of the sample input signal SHI. The source for SHI is selected with the SHSx bits and includes the following:

- ADC12SC bit
- Up to three timer outputs (see the device-specific data sheet for available timer sources)

The ADC12_A supports 8-bit, 10-bit, and 12-bit resolution modes selectable by the ADC12RES bits. The analog-to-digital conversion requires 9, 11, and 13 ADC12CLK cycles, respectively. The polarity of the SHI signal source can be inverted with the ADC12ISSH bit. The SAMPCON signal controls the sample period and start of conversion. When SAMPCON is high, sampling is active. The high-to-low SAMPCON transition starts the analog-to-digital conversion. Two different sample-timing methods are defined by control bit ADC12SHP, extended sample mode, and pulse mode. See the device-specific data sheet for available timers for SHI sources.

20.2.5.1 Extended Sample Mode

The extended sample mode is selected when ADC12SHP = 0. The SHI signal directly controls SAMPCON and defines the length of the sample period $t_{sample.}$ When SAMPCON is high, sampling is active. The high-to-low SAMPCON transition starts the conversion after synchronization with ADC12CLK (see Figure 20-4).

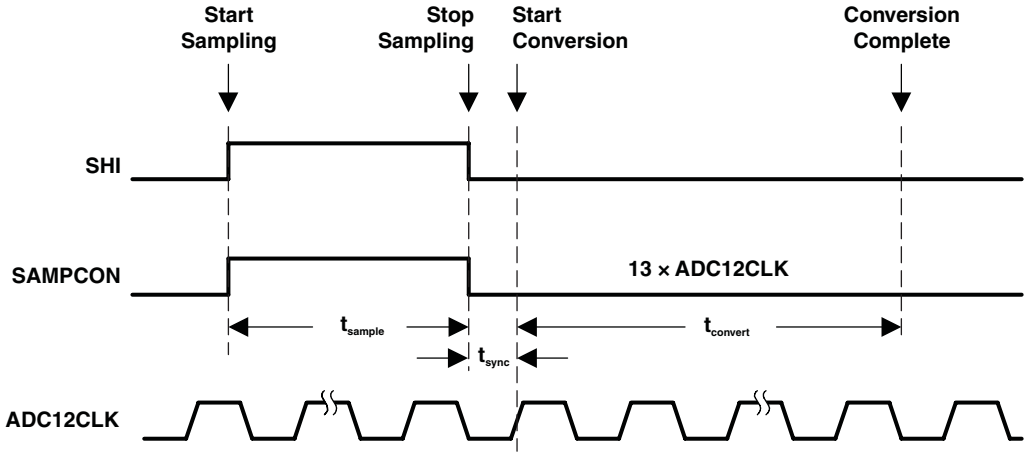


Figure 20-4. Extended Sample Mode

20.2.5.2 Pulse Sample Mode

Set ADC12SHP = 1 to select the pulse sample mode. The SHI signal is used to trigger the sampling timer. The ADC12SHT0x and ADC12SHT1x bits in ADC12CTL0 control the interval of the sampling timer that defines the SAMPCON sample period $t_{sample.}$ The sampling timer keeps SAMPCON high after synchronization with AD12CLK for a programmed interval t_{sample}. The total sampling time is t_{sample} plus t_{sync} (see Figure 20-5).

The ADC12SHTx bits select the sampling time in 4× multiples of ADC12CLK. ADC12SHT0x selects the sampling time for ADC12MCTL0 to ADC12MCTL7. ADC12SHT1x selects the sampling time for ADC12MCTL8 to ADC12MCTL15.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

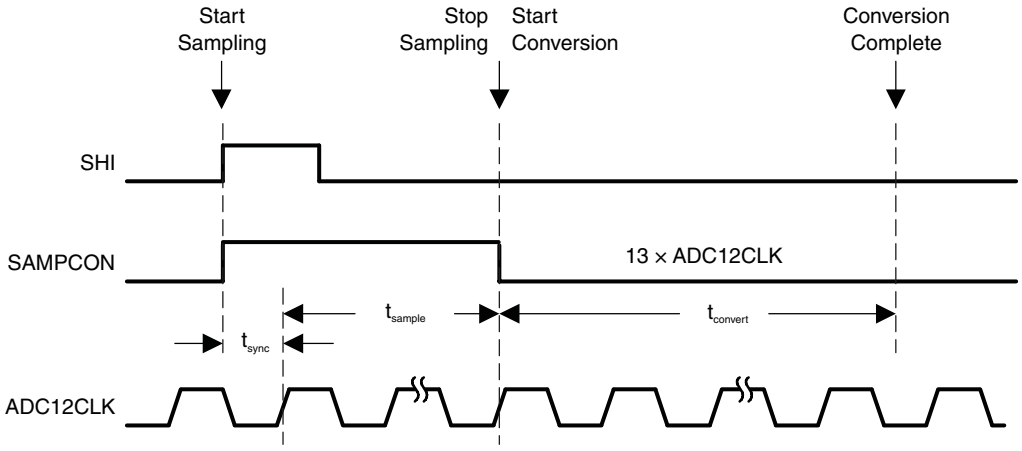


Figure 20-5. Pulse Sample Mode

20.2.5.3 Sample Timing Considerations

When SAMPCON = 0, all Ax inputs are high impedance. When SAMPCON = 1, the selected Ax input can be modeled as an RC low-pass filter during the sampling time t_{sample} (see Figure 20-6). An internal MUX-on input resistance R_I (maximum 1.8 kΩ) in series with capacitor C_I (25 pF maximum) is seen by the source. The capacitor C_I voltage V_C must be charged to within one-half LSB of the source voltage V_S for an accurate n-bit conversion, where n is the bits of resolution required.

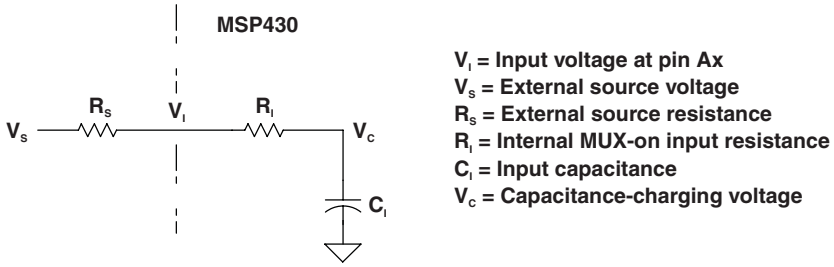


Figure 20-6. Analog Input Equivalent Circuit

The resistance of the source R_S and R_I affect t_{sample}. The following equation can be used to calculate the minimum sampling time t_{sample} for a n-bit conversion, where n equals the bits of resolution:

$$t_{sample} > (R_S + R_I) \times \ln(2^{n+1}) \times C_I + 800\ ns$$

Substituting the values for R_I and C_I given above, the equation becomes:

$$t_{sample} > (R_S + 1.8\ k\Omega) \times \ln(2^{n+1}) \times 25\ pF + 800\ ns$$

For example, for 12-bit resolution, if R_S is 10 kΩ, t_{sample} must be greater than 3.46 µs.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.6 Conversion Memory

There are 16 ADC12MEMx conversion memory registers to store conversion results. Each ADC12MEMx is configured with an associated ADC12MCTLx control register. The SREFx bits define the voltage reference and the INCHx bits select the input channel. The ADC12EOS bit defines the end of sequence when a sequential conversion mode is used. A sequence rolls over from ADC12MEM15 to ADC12MEM0 when the ADC12EOS bit in ADC12MCTL15 is not set.

The CSTARTADDx bits define the first ADC12MCTLx used for any conversion. If the conversion mode is single-channel or repeat-single-channel, the CSTARTADDx points to the single ADC12MCTLx to be used.

If the conversion mode selected is either sequence-of-channels or repeat-sequence-of-channels, CSTARTADDx points to the first ADC12MCTLx location to be used in a sequence. A pointer, not visible to software, is incremented automatically to the next ADC12MCTLx in a sequence when each conversion completes. The sequence continues until an ADC12EOS bit in ADC12MCTLx is processed; this is the last control byte processed.

When conversion results are written to a selected ADC12MEMx, the corresponding flag in the ADC12IFGx register is set.

There are two formats available to store the conversion result, ADC12MEMx. When ADC12DF = 0, the conversion is right justified, unsigned. For 8-bit, 10-bit, and 12-bit resolutions, the upper 8, 6, and 4 bits of ADC12MEMx are always zeros, respectively. When ADC12DF = 1, the conversion result is left justified, two's complement. For 8-bit, 10-bit, and 12-bit resolutions, the lower 8, 6, and 4 bits of ADC12MEMx are always zeros, respectively. This is summarized in Table 20-1.

Table 20-1. ADC12_A Conversion Result Formats

Analog Input Voltage	ADC12DF	ADC12RES	Ideal Conversion Results	ADC12MEMx
$-V_{REF}$ to $+V_{REF}$	0	00	0 to 255	0000h - 00FFh
	0	01	0 to 1023	0000h - 03FFh
	0	10	0 to 4095	0000h - 0FFFh
	1	00	-128 to 127	8000h - 7F00h
	1	01	-512 to 511	8000h - 7FC0h
	1	10	-2048 to 2047	8000h - 7FF0h

20.2.7 ADC12_A Conversion Modes

The ADC12_A has four operating modes selected by the CONSEQx bits as listed in Table 20-2. All state diagrams assume a 12-bit resolution setting.

Table 20-2. Conversion Mode Summary

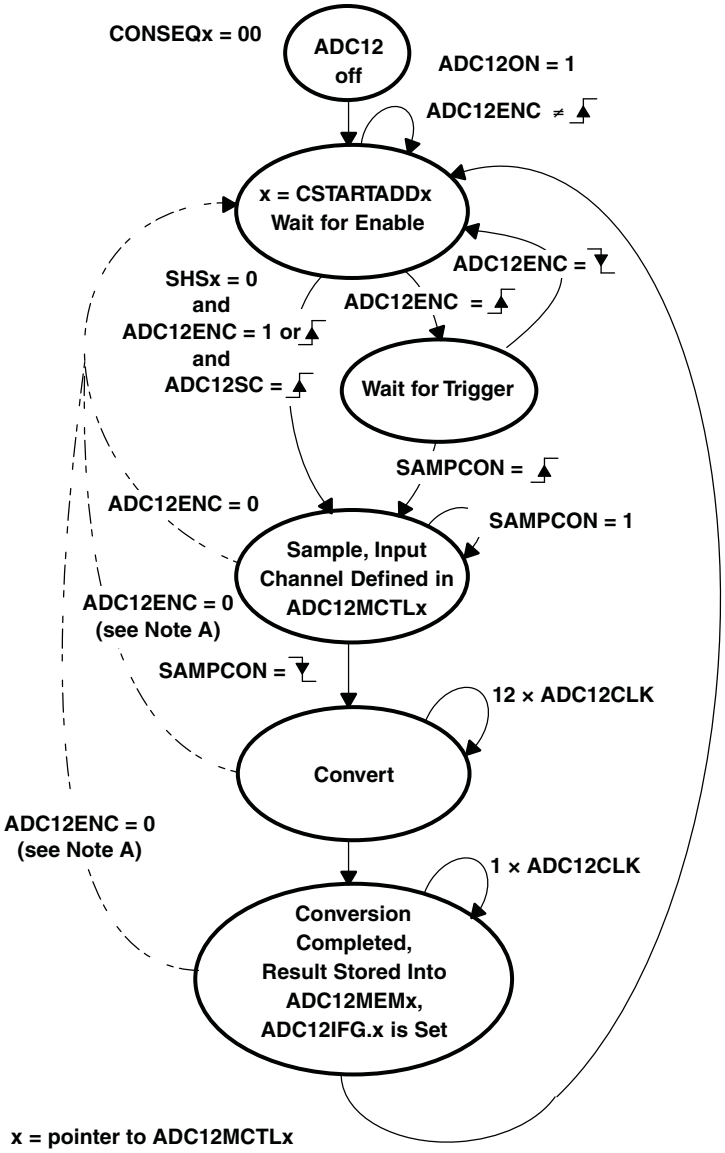
ADC12CONSEQx	Mode	Operation
00	Single-channel single-conversion	A single channel is converted once.
01	Sequence-of-channels (autoscan)	A sequence of channels is converted once.
10	Repeat-single-channel	A single channel is converted repeatedly.
11	Repeat-sequence-of-channels (repeated autoscan)	A sequence of channels is converted repeatedly.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.7.1 Single-Channel Single-Conversion Mode

A single channel is sampled and converted once. The ADC result is written to the ADC12MEMx defined by the CSTARTADDx bits. Figure 20-7 shows the flow of the single-channel single-conversion mode. When ADC12SC triggers a conversion, successive conversions can be triggered by the ADC12SC bit. When any other trigger source is used, ADC12ENC must be toggled between each conversion.



A Conversion result is unpredictable.

Figure 20-7. Single-Channel Single-Conversion Mode

Copyright © 2009–2013, Texas Instruments Incorporated

20.2.7.2 Sequence-of-Channels Mode (Autoscan Mode)

In sequence-of-channels mode, also referred to as autoscan mode, a sequence of channels is sampled and converted once. The ADC results are written to the conversion memories starting with the ADCMEMx defined by the CSTARTADDx bits. The sequence stops after the measurement of the channel with a set ADC12EOS bit. Figure 20-8 shows the sequence-of-channels mode. When ADC12SC triggers a sequence, successive sequences can be triggered by the ADC12SC bit. The ADC12SC must be cleared by software after each sequence to trigger another sequence. When any other trigger source is used, ADC12ENC must be toggled between each sequence.

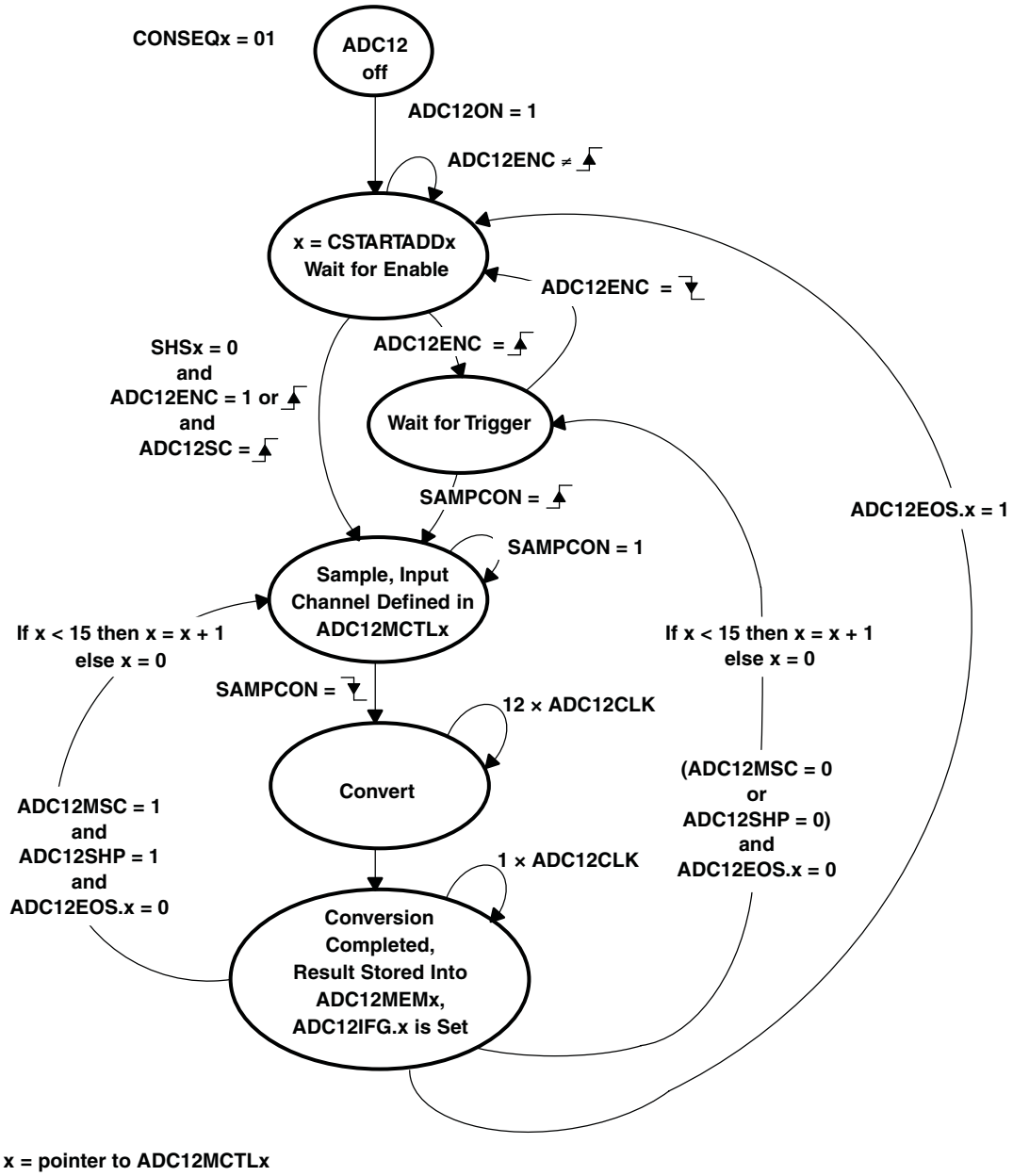


Figure 20-8. Sequence-of-Channels Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.7.3 Repeat-Single-Channel Mode

A single channel is sampled and converted continuously. The ADC results are written to the ADC12MEMx defined by the CSTARTADDx bits. It is necessary to read the result after the completed conversion because only one ADC12MEMx memory is used and is overwritten by the next conversion. Figure 20-9 shows the repeat-single-channel mode.

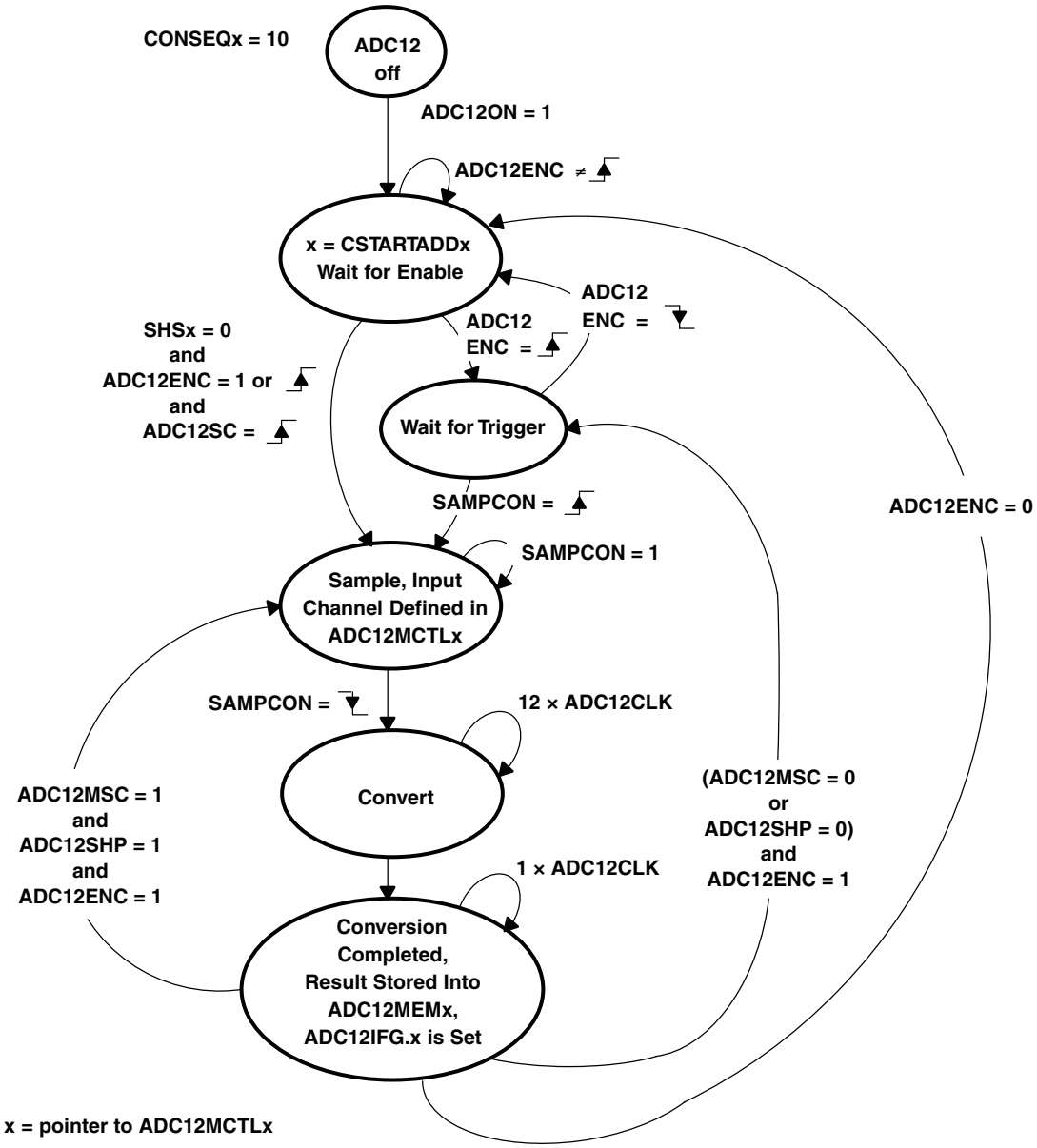


Figure 20-9. Repeat-Single-Channel Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.7.4 Repeat-Sequence-of-Channels Mode (Repeated Autoscan Mode)

In this mode, a sequence of channels is sampled and converted repeatedly. This mode is also referred to as repeated autoscan mode. The ADC results are written to the conversion memories starting with the ADC12MEMx defined by the CSTARTADDx bits. The sequence ends after the measurement of the channel with a set ADC12EOS bit and the next trigger signal restarts the sequence. Figure 20-10 shows the repeat-sequence-of-channels mode.

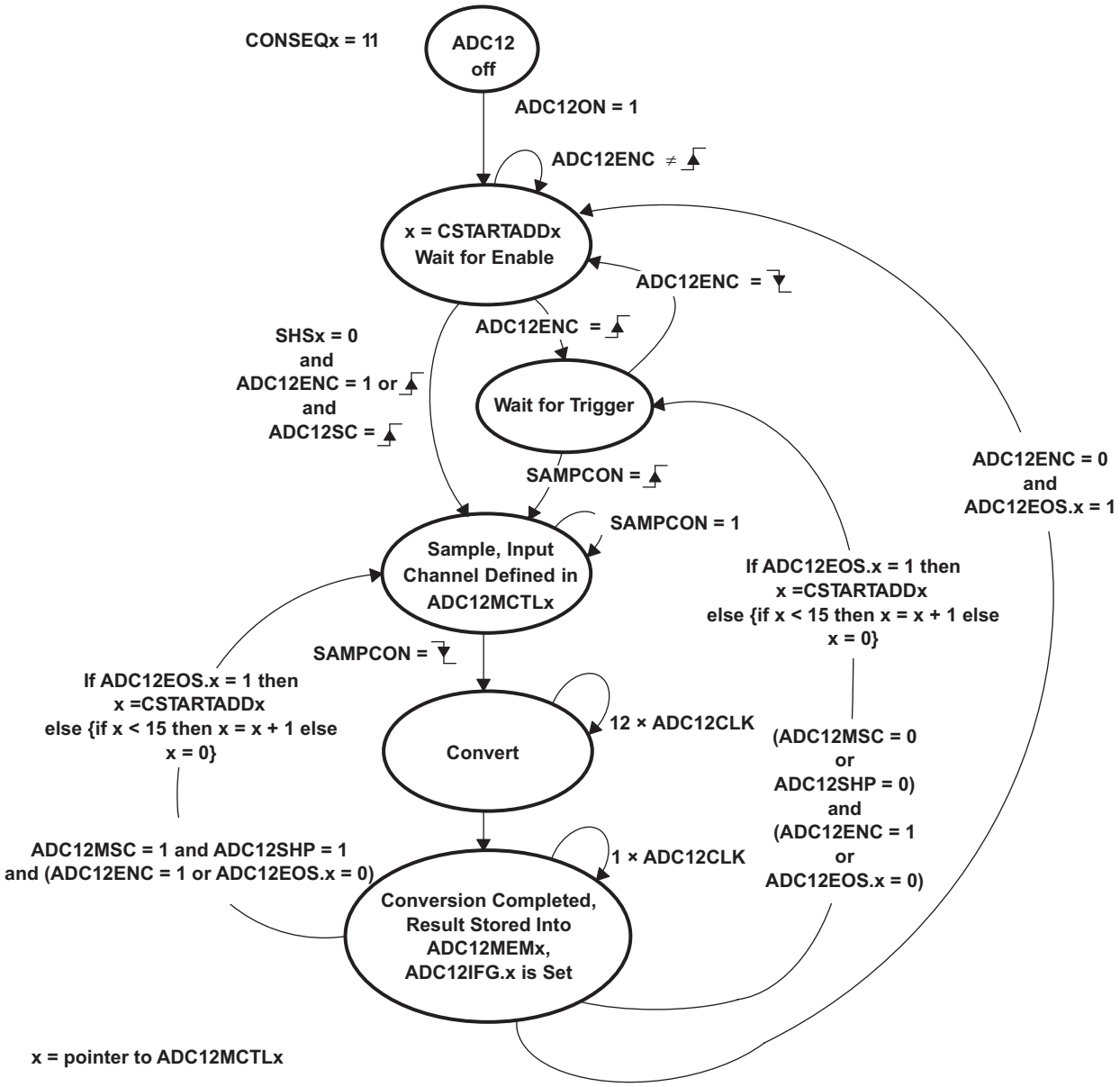


Figure 20-10. Repeat-Sequence-of-Channels Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.7.5 Using the Multiple Sample and Convert (ADC12MSC) Bit

To configure the converter to perform successive conversions automatically and as quickly as possible, a multiple sample and convert function is available. When ADC12MSC = 1, CONSEQx > 0, and the sample timer is used, the first rising edge of the SHI signal triggers the first conversion. Successive conversions are triggered automatically as soon as the prior conversion is completed. Additional rising edges on SHI are ignored until the sequence is completed in the single-sequence mode, or until the ADC12ENC bit is toggled in repeat-single-channel or repeated-sequence modes. The function of the ADC12ENC bit is unchanged when using the ADC12MSC bit.

20.2.7.6 Stopping Conversions

Stopping ADC12_A activity depends on the mode of operation. The recommended ways to stop an active conversion or conversion sequence are:

- Resetting ADC12ENC in single-channel single-conversion mode stops a conversion immediately and the results are unpredictable. For correct results, poll the busy bit until reset before clearing ADC12ENC.
- Resetting ADC12ENC during repeat-single-channel operation stops the converter at the end of the current conversion.
- Resetting ADC12ENC during a sequence or repeat-sequence mode stops the converter at the end of the sequence.
- Any conversion mode may be stopped immediately by setting the CONSEQx = 0 and resetting the ADC12ENC bit. Conversion data are unreliable.

NOTE: No ADC12EOS bit set for sequence

If no ADC12EOS bit is set and a sequence mode is selected, resetting the ADC12ENC bit does not stop the sequence. To stop the sequence, first select a single-channel mode and then reset ADC12ENC.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.8 Using the Integrated Temperature Sensor

To use the on-chip temperature sensor, the user selects the analog input channel INCHx = 1010. Any other configuration is done as if an external channel were selected, including reference selection, conversion-memory selection, etc. The temperature sensor is part of the reference. Therefore, for devices with the REF module, in addition to the input channels selection INCHx = 1010, configuring ADC12REFON = 1 (for REFMSTR = 0) or REFON = 1 (for REFMSTR = 1) is required to enable the temperature sensor.

For the MSP430F54xx (non-A) devices, which do not include the REF module, selecting the temperature sensor by configuring INCHx = 1010 automatically enables the reference generator required for the temperature sensor. Any other configuration is done as if an external channel were selected, included reference selection, conversion-memory selection, etc.

A typical temperature sensor transfer function is shown in Figure 20-11 . The transfer function shown in Figure 20-11 is only an example—the device-specific data sheet contains the actual parameters for a given device. When using the temperature sensor, the sample period must be greater than 30 µs. The temperature sensor offset error can be large and may need to be calibrated for most applications. Temperature calibration values are available for use in the TLV descriptors (see the device-specific data sheet for locations).

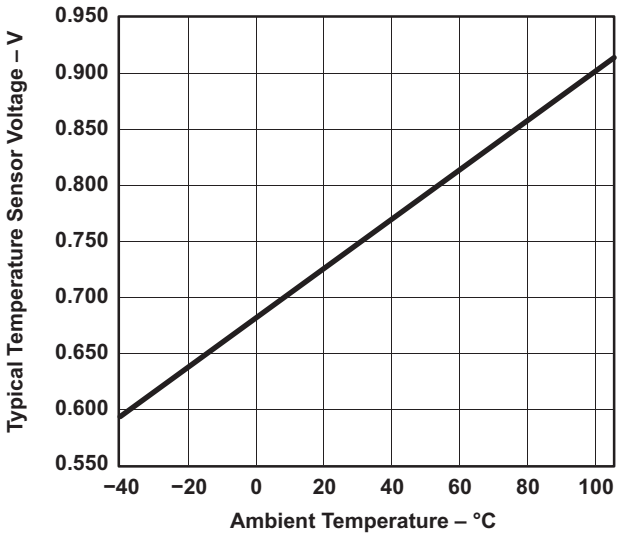


Figure 20-11. Typical Temperature Sensor Transfer Function

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.9 ADC12_A Grounding and Noise Considerations

As with any high-resolution ADC, appropriate printed circuit board layout and grounding techniques should be followed to eliminate ground loops, unwanted parasitic effects, and noise.

Ground loops are formed when return current from the ADC flows through paths that are common with other analog or digital circuitry. If care is not taken, this current can generate small unwanted offset voltages that can add to or subtract from the reference or input voltages of the ADC. The connections shown in Figure 20-12 prevent this.

In addition to grounding, ripple and noise spikes on the power-supply lines due to digital switching or switching power supplies can corrupt the conversion result. A noise-free design using separate analog and digital ground planes with a single-point connection is recommend to achieve high accuracy.

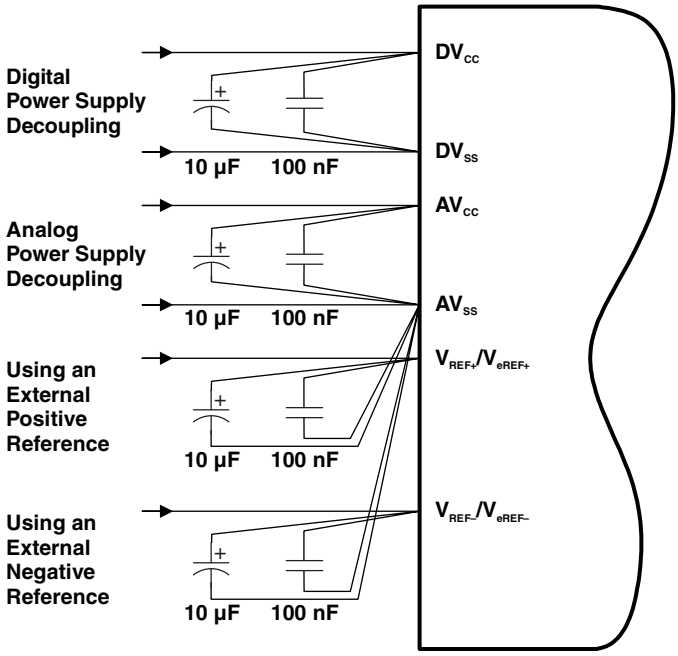


Figure 20-12. ADC12_A Grounding and Noise Considerations

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.10 ADC12_A Interrupts

The ADC12_A has 18 interrupt sources:

- ADC12IFG0 to ADC12IFG15
- ADC12OV, ADC12MEMx overflow
- ADC12TOV, ADC12_A conversion time overflow

The ADC12IFGx bits are set when their corresponding ADC12MEMx memory register is loaded with a conversion result. An interrupt request is generated if the corresponding ADC12IEx bit and the GIE bit are set. The ADC12OV condition occurs when a conversion result is written to any ADC12MEMx before its previous conversion result was read. The ADC12TOV condition is generated when another sample-and-conversion is requested before the current conversion is completed. The DMA is triggered after the conversion in single-channel conversion mode or after the completion of a sequence of channel conversions in sequence-of-channels conversion mode.

20.2.10.1 ADC12IV, Interrupt Vector Generator

All ADC12_A interrupt sources are prioritized and combined to source a single interrupt vector. The interrupt vector register ADC12IV is used to determine which enabled ADC12_A interrupt source requested an interrupt.

The highest-priority enabled ADC12_A interrupt generates a number in the ADC12IV register (see register description). This number can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled ADC12_A interrupts do not affect the ADC12IV value.

Any access, read or write, of the ADC12IV register automatically resets the ADC12OV condition or the ADC12TOV condition, if either was the highest-pending interrupt. Neither interrupt condition has an accessible interrupt flag. The ADC12IFGx flags are not reset by an ADC12IV access. ADC12IFGx bits are reset automatically by accessing their associated ADC12MEMx register or may be reset with software.

If another interrupt is pending after servicing of an interrupt, another interrupt is generated. For example, if the ADC12OV and ADC12IFG3 interrupts are pending when the interrupt service routine accesses the ADC12IV register, the ADC12OV interrupt condition is reset automatically. After the RETI instruction of the interrupt service routine is executed, the ADC12IFG3 generates another interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.2.10.2 ADC12_A Interrupt Handling Software Example

The following software example shows the recommended use of the ADC12IV and handling overhead. The ADC12IV value is added to the PC to automatically jump to the appropriate routine.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself. The latencies are:

- ADC12IFG0 to ADC12IFG14, ADC12TOV, and ADC12OV: 16 cycles
- ADC12IFG15: 14 cycles

The interrupt handler for ADC12IFG15 shows a way to check immediately if a higher-prioritized interrupt occurred during the processing of ADC12IFG15. This saves nine cycles if another ADC12_A interrupt is pending.

```
; Interrupt handler for ADC12.
INT_ADC12               ; Enter Interrupt Service Routine
   ADD      &ADC12IV,PC  ; Add offset to PC
   RETI                  ; Vector 0: No interrupt
   JMP      ADOV         ; Vector 2: ADC overflow
   JMP      ADTOV        ; Vector 4: ADC timing overflow
   JMP      ADM0         ; Vector 6: ADC12IFG0
      ...               ; Vectors 8-32
   JMP      ADM14        ; Vector 34: ADC12IFG14
;
; Handler for ADC12IFG15 starts here. No JMP required.
;
ADM15     MOV   &ADC12MEM15,xxx     ; Move result, flag is reset
          ...                       ; Other instruction needed?
          JMP   INT_ADC12           ; Check other int pending
;
; ADC12IFG14-ADC12IFG1 handlers go here
;
ADM0      MOV   &ADC12MEM0,xxx      ; Move result, flag is reset
          ...                       ; Other instruction needed?
RETI                                ; Return
;
ADTOV     ...                       ; Handle Conv. time overflow
          RETI                      ; Return
;
ADOV      ...                       ; Handle ADCMEMx overflow
          RETI                      ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3 ADC12_A Registers

The ADC12_A registers are listed in Table 20-3. The base address of the ADC12_A can be found in the device-specific data sheet. The address offset of each ADC12_A register is given in Table 20-3.

NOTE: All registers have word or byte register access. For a generic register *ANYREG*, the suffix "_L" (*ANYREG_L*) refers to the lower byte of the register (bits 0 through 7). The suffix "_H" (*ANYREG_H*) refers to the upper byte of the register (bits 8 through 15).

Table 20-3. ADC12_A Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	ADC12CTL0	ADC12_A Control 0	Read/write	Word	0000h	Section 20.3.1
00h	ADC12CTL0_L		Read/write	Byte	00h	
01h	ADC12CTL0_H		Read/write	Byte	00h	
02h	ADC12CTL1	ADC12_A Control 1	Read/write	Word	0000h	Section 20.3.2
02h	ADC12CTL1_L		Read/write	Byte	00h	
03h	ADC12CTL1_H		Read/write	Byte	00h	
04h	ADC12CTL2	ADC12_A Control 2	Read/write	Word	0020h	Section 20.3.3
04h	ADC12CTL2_L		Read/write	Byte	20h	
05h	ADC12CTL2_H		Read/write	Byte	00h	
0Ah	ADC12IFG	ADC12_A Interrupt Flag	Read/write	Word	0000h	Section 20.3.7
0Ah	ADC12IFG_L		Read/write	Byte	00h	
0Bh	ADC12IFG_H		Read/write	Byte	00h	
0Ch	ADC12IE	ADC12_A Interrupt Enable	Read/write	Word	0000h	Section 20.3.6
0Ch	ADC12IE_L		Read/write	Byte	00h	
0Dh	ADC12IE_H		Read/write	Byte	00h	
0Eh	ADC12IV	ADC12_A Interrupt Vector	Read	Word	0000h	Section 20.3.8
0Eh	ADC12IV_L		Read	Byte	00h	
0Fh	ADC12IV_H		Read	Byte	00h	
20h	ADC12MEM0	ADC12_A Memory 0	Read/write	Word	undefined	Section 20.3.4
20h	ADC12MEM0_L		Read/write	Byte	undefined	
21h	ADC12MEM0_H		Read/write	Byte	undefined	
22h	ADC12MEM1	ADC12_A Memory 1	Read/write	Word	undefined	Section 20.3.4
22h	ADC12MEM1_L		Read/write	Byte	undefined	
23h	ADC12MEM1_H		Read/write	Byte	undefined	
24h	ADC12MEM2	ADC12_A Memory 2	Read/write	Word	undefined	Section 20.3.4
24h	ADC12MEM2_L		Read/write	Byte	undefined	
25h	ADC12MEM2_H		Read/write	Byte	undefined	
26h	ADC12MEM3	ADC12_A Memory 3	Read/write	Word	undefined	Section 20.3.4
26h	ADC12MEM3_L		Read/write	Byte	undefined	
27h	ADC12MEM3_H		Read/write	Byte	undefined	
28h	ADC12MEM4	ADC12_A Memory 4	Read/write	Word	undefined	Section 20.3.4
28h	ADC12MEM4_L		Read/write	Byte	undefined	
29h	ADC12MEM4_H		Read/write	Byte	undefined	
2Ah	ADC12MEM5	ADC12_A Memory 5	Read/write	Word	undefined	Section 20.3.4
2Ah	ADC12MEM5_L		Read/write	Byte	undefined	
2Bh	ADC12MEM5_H		Read/write	Byte	undefined	
2Ch	ADC12MEM6	ADC12_A Memory 6	Read/write	Word	undefined	Section 20.3.4
2Ch	ADC12MEM6_L		Read/write	Byte	undefined	
2Dh	ADC12MEM6_H		Read/write	Byte	undefined	

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 20-3. ADC12_A Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset	Section
2Eh	ADC12MEM7	ADC12_A Memory 7	Read/write	Word	undefined	Section 20.3.4
2Eh	ADC12MEM7_L		Read/write	Byte	undefined	
2Fh	ADC12MEM7_H		Read/write	Byte	undefined	
30h	ADC12MEM8	ADC12_A Memory 8	Read/write	Word	undefined	Section 20.3.4
30h	ADC12MEM8_L		Read/write	Byte	undefined	
31h	ADC12MEM8_H		Read/write	Byte	undefined	
32h	ADC12MEM9	ADC12_A Memory 9	Read/write	Word	undefined	Section 20.3.4
32h	ADC12MEM9_L		Read/write	Byte	undefined	
33h	ADC12MEM9_H		Read/write	Byte	undefined	
34h	ADC12MEM10	ADC12_A Memory 10	Read/write	Word	undefined	Section 20.3.4
34h	ADC12MEM10_L		Read/write	Byte	undefined	
35h	ADC12MEM10_H		Read/write	Byte	undefined	
36h	ADC12MEM11	ADC12_A Memory 11	Read/write	Word	undefined	Section 20.3.4
36h	ADC12MEM11_L		Read/write	Byte	undefined	
37h	ADC12MEM11_H		Read/write	Byte	undefined	
38h	ADC12MEM12	ADC12_A Memory 12	Read/write	Word	undefined	Section 20.3.4
38h	ADC12MEM12_L		Read/write	Byte	undefined	
39h	ADC12MEM12_H		Read/write	Byte	undefined	
3Ah	ADC12MEM13	ADC12_A Memory 13	Read/write	Word	undefined	Section 20.3.4
3Ah	ADC12MEM13_L		Read/write	Byte	undefined	
3Bh	ADC12MEM13_H		Read/write	Byte	undefined	
3Ch	ADC12MEM14	ADC12_A Memory 14	Read/write	Word	undefined	Section 20.3.4
3Ch	ADC12MEM14_L		Read/write	Byte	undefined	
3Dh	ADC12MEM14_H		Read/write	Byte	undefined	
3Dh	ADC12MEM15	ADC12_A Memory 15	Read/write	Word	undefined	Section 20.3.4
3Dh	ADC12MEM15_L		Read/write	Byte	undefined	
3Eh	ADC12MEM15_H		Read/write	Byte	undefined	
10h	ADC12MCTL0	ADC12_A Memory Control 0	Read/write	Byte	undefined	Section 20.3.5
11h	ADC12MCTL1	ADC12_A Memory Control 1	Read/write	Byte	undefined	Section 20.3.5
12h	ADC12MCTL2	ADC12_A Memory Control 2	Read/write	Byte	undefined	Section 20.3.5
13h	ADC12MCTL3	ADC12_A Memory Control 3	Read/write	Byte	undefined	Section 20.3.5
14h	ADC12MCTL4	ADC12_A Memory Control 4	Read/write	Byte	undefined	Section 20.3.5
15h	ADC12MCTL5	ADC12_A Memory Control 5	Read/write	Byte	undefined	Section 20.3.5
16h	ADC12MCTL6	ADC12_A Memory Control 6	Read/write	Byte	undefined	Section 20.3.5
17h	ADC12MCTL7	ADC12_A Memory Control 7	Read/write	Byte	undefined	Section 20.3.5
18h	ADC12MCTL8	ADC12_A Memory Control 8	Read/write	Byte	undefined	Section 20.3.5
19h	ADC12MCTL9	ADC12_A Memory Control 9	Read/write	Byte	undefined	Section 20.3.5
1Ah	ADC12MCTL10	ADC12_A Memory Control 10	Read/write	Byte	undefined	Section 20.3.5
1Bh	ADC12MCTL11	ADC12_A Memory Control 11	Read/write	Byte	undefined	Section 20.3.5
1Ch	ADC12MCTL12	ADC12_A Memory Control 12	Read/write	Byte	undefined	Section 20.3.5
1Dh	ADC12MCTL13	ADC12_A Memory Control 13	Read/write	Byte	undefined	Section 20.3.5
1Eh	ADC12MCTL14	ADC12_A Memory Control 14	Read/write	Byte	undefined	Section 20.3.5
1Fh	ADC12MCTL15	ADC12_A Memory Control 15	Read/write	Byte	undefined	Section 20.3.5

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3.1 ADC12CTL0 Register

ADC12_A Control Register 0

Figure 20-13. ADC12CTL0 Register

15	14	13	12	11	10	9	8
ADC12SHT1x				ADC12SHT0x			
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC12MSC	ADC12REF2_5V	ADC12REFON	ADC12ON	ADC12OVIE	ADC12TOVIE	ADC12ENC	ADC12SC
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Can be modified only when ADC12ENC = 0

Table 20-4. ADC12CTL0 Register Description

Bit	Field	Type	Reset	Description
15-12	ADC12SHT1x	RW	0h	ADC12_A sample-and-hold time. These bits define the number of ADC12CLK cycles in the sampling period for registers ADC12MEM8 to ADC12MEM15.
11-8	ADC12SHT0x	RW	0h	ADC12_A sample-and-hold time. These bits define the number of ADC12CLK cycles in the sampling period for registers ADC12MEM0 to ADC12MEM7. 0000b = 4 ADC12CLK cycles 0001b = 8 ADC12CLK cycles 0010b = 16 ADC12CLK cycles 0011b = 32 ADC12CLK cycles 0100b = 64 ADC12CLK cycles 0101b = 96 ADC12CLK cycles 0110b = 128 ADC12CLK cycles 0111b = 192 ADC12CLK cycles 1000b = 256 ADC12CLK cycles 1001b = 384 ADC12CLK cycles 1010b = 512 ADC12CLK cycles 1011b = 768 ADC12CLK cycles 1100b = 1024 ADC12CLK cycles 1101b = 1024 ADC12CLK cycles 1110b = 1024 ADC12CLK cycles 1111b = 1024 ADC12CLK cycles
7	ADC12MSC	RW	0h	ADC12_A multiple sample and conversion. Valid only for sequence or repeated modes. 0b = The sampling timer requires a rising edge of the SHI signal to trigger each sample-and-convert. 1b = The first rising edge of the SHI signal triggers the sampling timer, but further sample-and-conversions are performed automatically as soon as the prior conversion is completed.
6	ADC12REF2_5V	RW	0h	ADC12_A reference generator voltage. ADC12REFON must also be set. In devices with the REF module, this bit is only valid if the REFMSTR bit of the REF module is set to 0. In the F54xx devices (non-A), the REF module is not available. 0b = 1.5 V 1b = 2.5 V
5	ADC12REFON	RW	0h	ADC12_A reference generator on. In devices with the REF module, this bit is only valid if the REFMSTR bit of the REF module is set to 0. In the F54xx devices (non-A), the REF module is not available. 0b = Reference off 1b = Reference on

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 20-4. ADC12CTL0 Register Description (continued)

Bit	Field	Type	Reset	Description
4	ADC12ON	RW	0h	ADC12_A on 0b = ADC12_A off 1b = ADC12_A on
3	ADC12OVIE	RW	0h	ADC12MEMx overflow-interrupt enable. The GIE bit must also be set to enable the interrupt. 0b = Overflow interrupt disabled 1b = Overflow interrupt enabled
2	ADC12TOVIE	RW	0h	ADC12_A conversion-time-overflow interrupt enable. The GIE bit must also be set to enable the interrupt. 0b = Conversion time overflow interrupt disabled 1b = Conversion time overflow interrupt enabled
1	ADC12ENC	RW	0h	ADC12_A enable conversion 0b = ADC12_A disabled 1b = ADC12_A enabled
0	ADC12SC	RW	0h	ADC12_A start conversion. Software-controlled sample-and-conversion start. ADC12SC and ADC12ENC may be set together with one instruction. ADC12SC is reset automatically. 0b = No sample-and-conversion-start 1b = Start sample-and-conversion

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3.2 ADC12CTL1 Register

ADC12_A Control Register 1

Figure 20-14. ADC12CTL1 Register

15	14	13	12	11	10	9	8
ADC12CSTARTADDx				ADC12SHSx		ADC12SHP	ADC12ISSH
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC12DIVx			ADC12SSELx		ADC12CONSEQx		ADC12BUSY
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r-(0)

Can be modified only when ADC12ENC = 0

Table 20-5. ADC12CTL1 Register Description

Bit	Field	Type	Reset	Description
15-12	ADC12CSTARTADDx	RW	0h	ADC12_A conversion start address. These bits select which ADC12_A conversion-memory register is used for a single conversion or for the first conversion in a sequence. The value of CSTARTADDx is 0 to 0Fh, corresponding to ADC12MEM0 to ADC12MEM15.
11-10	ADC12SHSx	RW	0h	ADC12_A sample-and-hold source select 00b = ADC12SC bit 01b = Timer source (see device-specific data sheet for exact timer and locations) 10b = Timer source (see device-specific data sheet for exact timer and locations) 11b = Timer source (see device-specific data sheet for exact timer and locations)
9	ADC12SHP	RW	0h	ADC12_A sample-and-hold pulse-mode select. This bit selects the source of the sampling signal (SAMPCON) to be either the output of the sampling timer or the sample-input signal directly. 0b = SAMPCON signal is sourced from the sample-input signal. 1b = SAMPCON signal is sourced from the sampling timer.
8	ADC12ISSH	RW	0h	ADC12_A invert signal sample-and-hold 0b = The sample-input signal is not inverted. 1b = The sample-input signal is inverted.
7-5	ADC12DIVx	RW	0h	ADC12_A clock divider 000b = Divide by 1 001b = Divide by 2 010b = Divide by 3 011b = Divide by 4 100b = Divide by 5 101b = Divide by 6 110b = Divide by 7 111b = Divide by 8
4-3	ADC12SSELx	RW	0h	ADC12_A clock source select 00b = ADC12OSC (MODOSC) 01b = ACLK 10b = MCLK 11b = SMCLK
2-1	ADC12CONSEQx	RW	0h	ADC12_A conversion sequence mode select 00b = Single-channel, single-conversion 01b = Sequence-of-channels 10b = Repeat-single-channel 11b = Repeat-sequence-of-channels
0	ADC12BUSY	R	0h	ADC12_A busy. This bit indicates an active sample or conversion operation. 0b = No operation is active. 1b = A sequence, sample, or conversion is active.

Copyright © 2009–2013, Texas Instruments Incorporated

20.3.3 ADC12CTL2 Register

ADC12_A Control Register 2

Figure 20-15. ADC12CTL2 Register

15	14	13	12	11	10	9	8
Reserved							ADC12PDIV
r-0	r-0	r-0	r-0	r-0	r-0	r-0	rw-0
7	6	5	4	3	2	1	0
ADC12TCOFF	Reserved	ADC12RES		ADC12DF	ADC12SR	ADC12REFOUT	ADC12REFBURST
rw-(0)	r-0	rw-(1)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Can be modified only when ADC12ENC = 0

Table 20-6. ADC12CTL2 Register Description

Bit	Field	Type	Reset	Description
15-9	Reserved	R	0h	Reserved. Always reads as 0.
8	ADC12PDIV	RW	0h	ADC12_A predivider. This bit predivides the selected ADC12_A clock source. 0b = Predivide by 1 1b = Predivide by 4
7	ADC12TCOFF	RW	0h	ADC12_A temperature sensor off. If the bit is set, the temperature sensor turned off. This is used to save power. In devices with the REF module, this bit is only valid if the REFMSTR bit of the REF module is set to 0. In the F54xx devices (non-A), the REF module is not available. 0b = Temperature sensor on 1b = Temperature sensor off
6	Reserved	R	0h	Reserved. Always reads as 0.
5-4	ADC12RES	RW	2h	ADC12_A resolution. This bit defines the conversion result resolution. 00b = 8 bit (9 clock cycle conversion time) 01b = 10 bit (11 clock cycle conversion time) 10b = 12 bit (13 clock cycle conversion time) 11b = Reserved
3	ADC12DF	RW	0h	ADC12_A data read-back format. Data is always stored in the binary unsigned format. 0b = Binary unsigned. Theoretically, the analog input voltage -VREF results in 0000h, the analog input voltage +VREF results in 0FFFh. 1b = Signed binary (2s complement), left aligned. Theoretically, the analog input voltage -VREF results in 8000h, the analog input voltage +VREF results in 7FF0h.
2	ADC12SR	RW	0h	ADC12_A sampling rate. This bit selects the reference buffer drive capability for the maximum sampling rate. Setting ADC12SR reduces the current consumption of the reference buffer. 0b = Reference buffer supports up to approximately 200 ksps. 1b = Reference buffer supports up to approximately 50 ksps.
1	ADC12REFOUT	RW	0h	Reference output. In devices with the REF module, this bit is only valid if the REFMSTR bit of the REF module is set to 0. In the F54xx devices (non-A), the REF module is not available. 0b = Reference output off 1b = Reference output on
0	ADC12REFBURST	RW	0h	Reference burst 0b = Reference buffer on continuously 1b = Reference buffer on only during sample-and-conversion

Copyright © 2009–2013, Texas Instruments Incorporated

20.3.4 ADC12MEMx Register

ADC12_A Conversion Memory Register

Figure 20-16. ADC12MEMx Register

15	14	13	12	11	10	9	8
Conversion Results							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
Conversion Results							
rw	rw	rw	rw	rw	rw	rw	rw

Table 20-7. ADC12MEMx Register Description

Bit	Field	Type	Reset	Description
15-0	Conversion Results	RW	undefined	Binary unsigned format: This data format is used if ADC12DF = 0. The 12-bit conversion results are right justified. Bit 11 is the MSB. Bits 15–12 are 0 in 12-bit mode, bits 15–10 are 0 in 10-bit mode, and bits 15–8 are 0 in 8-bit mode. Writing to the conversion memory registers corrupts the results. 2s-complement format: This data format is used if ADC12DF = 1. The 12-bit conversion results are left justified, 2s-complement format. Bit 15 is the MSB. Bits 3–0 are 0 in 12-bit mode, bits 5–0 are 0 in 10-bit mode, and bits 7–0 are 0 in 8-bit mode. The data is stored in the right-justified format and is converted to the left-justified 2s-complement format during read back.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3.5 ADC12MCTLx Register

ADC12_A Conversion Memory Control Register

Figure 20-17. ADC12MCTLx Register

7	6	5	4	3	2	1	0
ADC12EOS	ADC12SREFx			ADC12INCHx			
rw	rw	rw	rw	rw	rw	rw	rw

Can be modified only when ADC12ENC = 0

Table 20-8. ADC12MCTLx Register Description

Bit	Field	Type	Reset	Description
7	ADC12EOS	RW	0h	End of sequence. Indicates the last conversion in a sequence. 0b = Not end of sequence 1b = End of sequence
6-4	ADC12SREFx	RW	0h	Select reference 000b = V(R+) = AVCC and V(R-) = AVSS 001b = V(R+) = VREF+ and V(R-) = AVSS 010b = V(R+) = VeREF+ and V(R-) = AVSS 011b = V(R+) = VeREF+ and V(R-) = AVSS 100b = V(R+) = AVCC and V(R-) = VREF-/VeREF- 101b = V(R+) = VREF+ and V(R-) = VREF-/VeREF- 110b = V(R+) = VeREF+ and V(R-) = VREF-/VeREF- 111b = V(R+) = VeREF+ and V(R-) = VREF-/VeREF-
3-0	ADC12INCHx	RW	0h	Input channel select 0000b = A0 0001b = A1 0010b = A2 0011b = A3 0100b = A4 0101b = A5 0110b = A6 0111b = A7 1000b = VeREF+ 1001b = VREF-/VeREF- 1010b = Temperature diode 1011b = (AVCC – AVSS) / 2 1100b = A12. On devices with the Battery Backup System, VBAT can be measured internally by the ADC. 1101b = A13 1110b = A14 1111b = A15

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3.6 ADC12IE Register

ADC12_A Interrupt Enable Register

Figure 20-18. ADC12IE Register

15	14	13	12	11	10	9	8
ADC12IE15	ADC12IE14	ADC12IE13	ADC12IE12	ADC12IE11	ADC12IE10	ADC12IE9	ADC12IE8
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC12IE7	ADC12IE6	ADC12IE5	ADC12IE4	ADC12IE3	ADC12IE2	ADC12IE1	ADC12IE0
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 20-9. ADC12IE Register Description

Bit	Field	Type	Reset	Description
15	ADC12IE15	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG15 bit. 0b = Interrupt disabled 1b = Interrupt enabled
14	ADC12IE14	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG14 bit. 0b = Interrupt disabled 1b = Interrupt enabled
13	ADC12IE13	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG13 bit. 0b = Interrupt disabled 1b = Interrupt enabled
12	ADC12IE12	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG12 bit. 0b = Interrupt disabled 1b = Interrupt enabled
11	ADC12IE11	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG11 bit. 0b = Interrupt disabled 1b = Interrupt enabled
10	ADC12IE10	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG10 bit. 0b = Interrupt disabled 1b = Interrupt enabled
9	ADC12IE9	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG9 bit. 0b = Interrupt disabled 1b = Interrupt enabled
8	ADC12IE8	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG8 bit. 0b = Interrupt disabled 1b = Interrupt enabled
7	ADC12IE7	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG7 bit. 0b = Interrupt disabled 1b = Interrupt enabled
6	ADC12IE6	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG6 bit. 0b = Interrupt disabled 1b = Interrupt enabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 20-9. ADC12IE Register Description (continued)

Bit	Field	Type	Reset	Description
5	ADC12IE5	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG5 bit. 0b = Interrupt disabled 1b = Interrupt enabled
4	ADC12IE4	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG4 bit. 0b = Interrupt disabled 1b = Interrupt enabled
3	ADC12IE3	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG3 bit. 0b = Interrupt disabled 1b = Interrupt enabled
2	ADC12IE2	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG2 bit. 0b = Interrupt disabled 1b = Interrupt enabled
1	ADC12IE1	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG1 bit. 0b = Interrupt disabled 1b = Interrupt enabled
0	ADC12IE0	RW	0h	Interrupt enable. This bit enables or disables the interrupt request for the ADC12IFG0 bit. 0b = Interrupt disabled 1b = Interrupt enabled

Copyright © 2009–2013, Texas Instruments Incorporated

20.3.7 ADC12IFG Register

ADC12_A Interrupt Flag Register

Figure 20-19. ADC12IFG Register

15	14	13	12	11	10	9	8
ADC12IFG15	ADC12IFG14	ADC12IFG13	ADC12IFG12	ADC12IFG11	ADC12IFG10	ADC12IFG9	ADC12IFG8
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
ADC12IFG7	ADC12IFG6	ADC12IFG5	ADC12IFG4	ADC12IFG3	ADC12IFG2	ADC12IFG1	ADC12IFG0
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 20-10. ADC12IFG Register Description

Bit	Field	Type	Reset	Description
15	ADC12IFG15	RW	0h	ADC12MEM15 interrupt flag. This bit is set when ADC12MEM15 is loaded with a conversion result. This bit is reset if the ADC12MEM15 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
14	ADC12IFG14	RW	0h	ADC12MEM14 interrupt flag. This bit is set when ADC12MEM14 is loaded with a conversion result. This bit is reset if the ADC12MEM14 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
13	ADC12IFG13	RW	0h	ADC12MEM13 interrupt flag. This bit is set when ADC12MEM13 is loaded with a conversion result. This bit is reset if the ADC12MEM13 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
12	ADC12IFG12	RW	0h	ADC12MEM12 interrupt flag. This bit is set when ADC12MEM12 is loaded with a conversion result. This bit is reset if the ADC12MEM12 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
11	ADC12IFG11	RW	0h	ADC12MEM11 interrupt flag. This bit is set when ADC12MEM11 is loaded with a conversion result. This bit is reset if the ADC12MEM11 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
10	ADC12IFG10	RW	0h	ADC12MEM10 interrupt flag. This bit is set when ADC12MEM10 is loaded with a conversion result. This bit is reset if the ADC12MEM10 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
9	ADC12IFG9	RW	0h	ADC12MEM9 interrupt flag. This bit is set when ADC12MEM9 is loaded with a conversion result. This bit is reset if the ADC12MEM9 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
8	ADC12IFG8	RW	0h	ADC12MEM8 interrupt flag. This bit is set when ADC12MEM8 is loaded with a conversion result. This bit is reset if the ADC12MEM8 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 20-10. ADC12IFG Register Description (continued)

Bit	Field	Type	Reset	Description
7	ADC12IFG7	RW	0h	ADC12MEM7 interrupt flag. This bit is set when ADC12MEM7 is loaded with a conversion result. This bit is reset if the ADC12MEM7 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
6	ADC12IFG6	RW	0h	ADC12MEM6 interrupt flag. This bit is set when ADC12MEM6 is loaded with a conversion result. This bit is reset if the ADC12MEM6 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
5	ADC12IFG5	RW	0h	ADC12MEM5 interrupt flag. This bit is set when ADC12MEM5 is loaded with a conversion result. This bit is reset if the ADC12MEM5 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
4	ADC12IFG4	RW	0h	ADC12MEM4 interrupt flag. This bit is set when ADC12MEM4 is loaded with a conversion result. This bit is reset if the ADC12MEM4 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
3	ADC12IFG3	RW	0h	ADC12MEM3 interrupt flag. This bit is set when ADC12MEM3 is loaded with a conversion result. This bit is reset if the ADC12MEM3 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
2	ADC12IFG2	RW	0h	ADC12MEM2 interrupt flag. This bit is set when ADC12MEM2 is loaded with a conversion result. This bit is reset if the ADC12MEM2 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
1	ADC12IFG1	RW	0h	ADC12MEM1 interrupt flag. This bit is set when ADC12MEM1 is loaded with a conversion result. This bit is reset if the ADC12MEM1 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending
0	ADC12IFG0	RW	0h	ADC12MEM0 interrupt flag. This bit is set when ADC12MEM0 is loaded with a conversion result. This bit is reset if the ADC12MEM0 is accessed, or it may be reset with software. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

20.3.8 ADC12IV Register

ADC12_A Interrupt Vector Register

Figure 20-20. ADC12IV Register

15	14	13	12	11	10	9	8
ADC12IVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
ADC12IVx							
r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 20-11. ADC12IV Register Description

Bit	Field	Type	Reset	Description
15-0	ADC12IVx	R	0h	ADC12_A interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: ADC12MEMx overflow; Interrupt Flag: –; Interrupt Priority: Highest 04h = Interrupt Source: Conversion time overflow; Interrupt Flag: – 06h = Interrupt Source: ADC12MEM0 interrupt flag; Interrupt Flag: ADC12IFG0 08h = Interrupt Source: ADC12MEM1 interrupt flag; Interrupt Flag: ADC12IFG1 0Ah = Interrupt Source: ADC12MEM2 interrupt flag; Interrupt Flag: ADC12IFG2 0Ch = Interrupt Source: ADC12MEM3 interrupt flag; Interrupt Flag: ADC12IFG3 0Eh = Interrupt Source: ADC12MEM4 interrupt flag; Interrupt Flag: ADC12IFG4 10h = Interrupt Source: ADC12MEM5 interrupt flag; Interrupt Flag: ADC12IFG5 12h = Interrupt Source: ADC12MEM6 interrupt flag; Interrupt Flag: ADC12IFG6 14h = Interrupt Source: ADC12MEM7 interrupt flag; Interrupt Flag: ADC12IFG7 16h = Interrupt Source: ADC12MEM8 interrupt flag; Interrupt Flag: ADC12IFG8 18h = Interrupt Source: ADC12MEM9 interrupt flag; Interrupt Flag: ADC12IFG9 1Ah = Interrupt Source: ADC12MEM10 interrupt flag; Interrupt Flag: ADC12IFG10 1Ch = Interrupt Source: ADC12MEM11 interrupt flag; Interrupt Flag: ADC12IFG11 1Eh = Interrupt Source: ADC12MEM12 interrupt flag; Interrupt Flag: ADC12IFG12 20h = Interrupt Source: ADC12MEM13 interrupt flag; Interrupt Flag: ADC12IFG13 22h = Interrupt Source: ADC12MEM14 interrupt flag; Interrupt Flag: ADC12IFG14 24h = Interrupt Source: ADC12MEM15 interrupt flag; Interrupt Flag: ADC12IFG15; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Comp_B

Comp_B is an analog voltage comparator. This chapter describes the Comp_B. Comp_B covers general comparator functionality for up to 16 channels.

Topic **Page**

21.1 Comp_B Introduction **564**
21.2 Comp_B Operation **565**
21.3 Comp_B Registers **570**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.1 Comp_B Introduction

The Comp_B module supports precision slope analog-to-digital conversions, supply voltage supervision, and monitoring of external analog signals.

Features of Comp_B include:

- Inverting and noninverting terminal input multiplexer
- Software-selectable RC filter for the comparator output
- Output provided to Timer_A capture input
- Software control of the port input buffer
- Interrupt capability
- Selectable reference voltage generator, voltage hysteresis generator
- Reference voltage input from shared reference
- Ultra-low-power comparator mode
- Interrupt driven measurement system – low-power operation support

The Comp_B block diagram is shown in Figure 21-1.

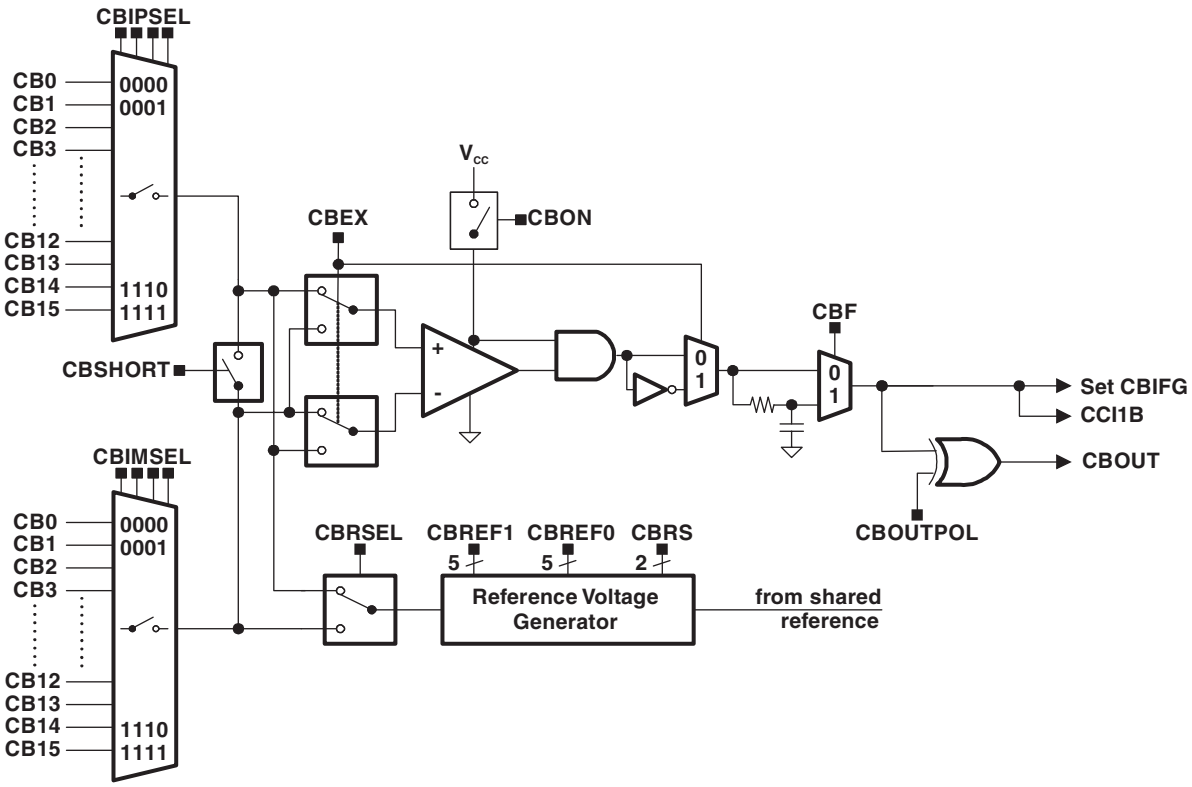


Figure 21-1. Comp_B Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.2 Comp_B Operation

The Comp_B module is configured by user software. The setup and operation of Comp_B is discussed in the following sections.

21.2.1 Comparator

The comparator compares the analog voltages at the + and – input terminals. If the + terminal is more positive than the – terminal, the comparator output CBOUT is high. The comparator can be switched on or off using control bit CBON. The comparator should be switched off when not in use to reduce current consumption. When the comparator is switched off, CBOUT is always low. The bias current of the comparator is programmable.

21.2.2 Analog Input Switches

The analog input switches connect or disconnect the two comparator input terminals to associated port pins using the CBIPSELx and CBIMSELx bits. The comparator terminal inputs can be controlled individually. The CBIPSELx/CBIMSELx bits allow:

- Application of an external signal to the + and – terminals of the comparator
- Routing of an internal reference voltage to an associated output port pin
- Application of an external current source (for example, a resistor) to the + or – terminal of the comparator
- The mapping of both terminals of the internal multiplexer to the outside

Internally, the input switch is constructed as a T-switch to suppress distortion in the signal path.

NOTE: Comparator Input Connection

When the comparator is on, the input terminals should be connected to a signal, power, or ground. Otherwise, floating levels may cause unexpected interrupts and increased current consumption.

The CBEX bit controls the input multiplexer, permuting the input signals of the comparator's + and – terminals. Additionally, when the comparator terminals are permuted, the output signal from the comparator is inverted too. This allows the user to determine or compensate for the comparator input offset voltage.

21.2.3 Port Logic

The Px.y pins associated with a comparator channel are enabled by the CBIPSELx or CBIMSELx bits to disable its digital components while used as comparator input. Only one of the comparator input pins is selected as input to the comparator by the input multiplexer at a time.

21.2.4 Input Short Switch

The CBSHORT bit shorts the Comp_B inputs. This can be used to build a simple sample-and-hold for the comparator as shown in Figure 21-2.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

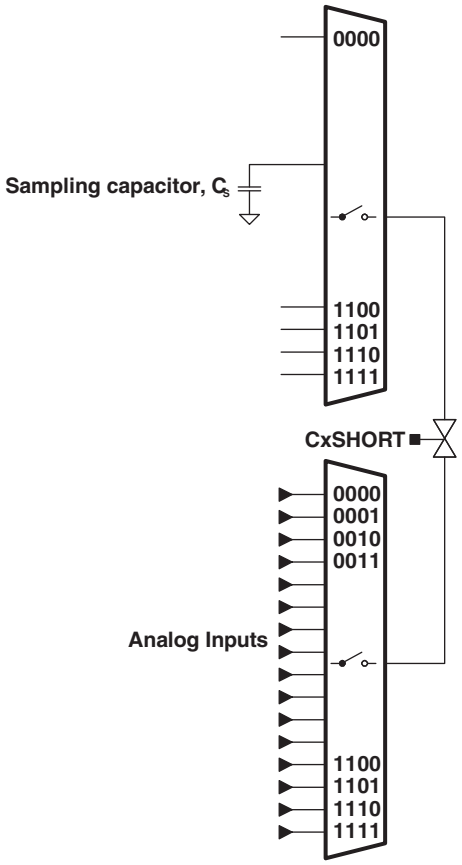


Figure 21-2. Comp_B Sample-And-Hold

The required sampling time is proportional to the size of the sampling capacitor (C_S), the resistance of the input switches in series with the short switch (R_i), and the resistance of the external source (R_S). The total internal resistance (R_I) is typically in the range of 1 kΩ. The sampling capacitor C_S should be greater than 100 pF. The time constant, Tau, to charge the sampling capacitor C_S can be calculated with the following equation:

$$Tau = (R_I + R_S) \times C_S$$

Depending on the required accuracy, 3 to 10 Tau should be used as a sampling time. With 3 Tau the sampling capacitor is charged to approximately 95% of the input signals voltage level, with 5 Tau it is charged to more than 99%, and with 10 Tau the sampled voltage is sufficient for 12-bit accuracy.

21.2.5 Output Filter

The output of the comparator can be used with or without internal filtering. When control bit CBF is set, the output is filtered with an on-chip RC filter. The delay of the filter can be adjusted in four different steps.

All comparator outputs are oscillating if the voltage difference across the input terminals is small. Internal and external parasitic effects and cross coupling on and between signal lines, power supply lines, and other parts of the system are responsible for this behavior as shown in Figure 21-3. The comparator output oscillation reduces the accuracy and resolution of the comparison result. Selecting the output filter can reduce errors associated with comparator oscillation.

Copyright © 2009–2013, Texas Instruments Incorporated

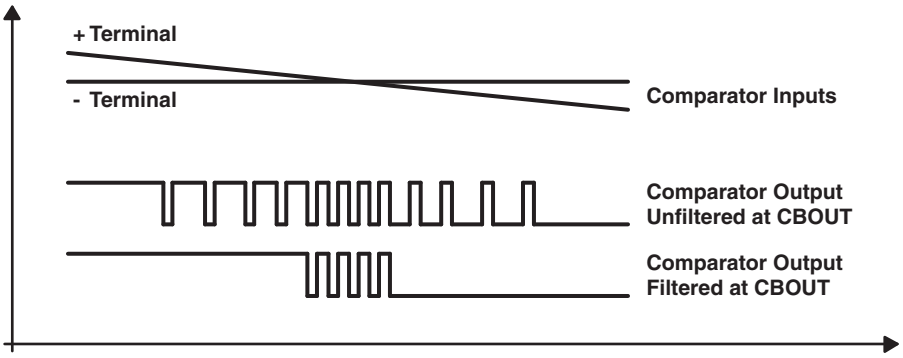


Figure 21-3. RC-Filter Response at the Output of the Comparator

21.2.6 Reference Voltage Generator

The Comp_B reference block diagram is shown in Figure 21-4.

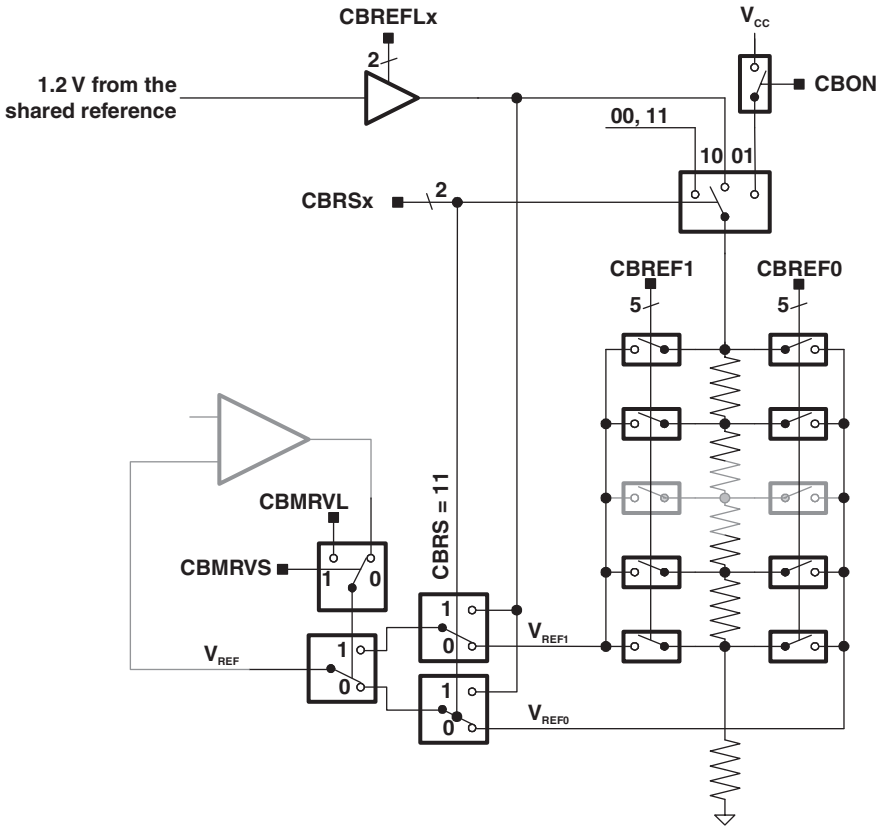


Figure 21-4. Reference Generator Block Diagram

The voltage reference generator is used to generate VREF, which can be applied to either comparator input terminal. The CBREF1x (VREF1) and CBREF0x (VREF0) bits control the output of the voltage generator. The CBRSEL bit selects the comparator terminal to which VREF is applied. If external signals are applied to both comparator input terminals, the internal reference generator should be turned off to reduce current consumption. The voltage reference generator can generate a fraction of the device's V_{CC} or of the voltage reference of the integrated precision voltage reference source. Vref1 is used while CBOUT is 1 and Vref0 is used while CBOUT is 0. This allows the generation of a hysteresis without using external components.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.2.7 Comp_B, Port Disable Register CBPD

The comparator input and output functions are multiplexed with the associated I/O port pins, which are digital CMOS gates. When analog signals are applied to digital CMOS gates, parasitic current can flow from V_{CC} to GND. This parasitic current occurs if the input voltage is near the transition level of the gate. Disabling the port pin buffer eliminates the parasitic current flow and therefore reduces overall current consumption.

The CBPDx bits, when set, disable the corresponding Px.y input buffer as shown in Figure 21-5. When current consumption is critical, any Px.y pin connected to analog signals should be disabled with their associated CBPDx bits.

Selecting an input pin to the comparator multiplexer with the CBIPSEL or CBIMSEL bits automatically disables the input buffer for that pin, regardless of the state of the associated CBPDx bit.

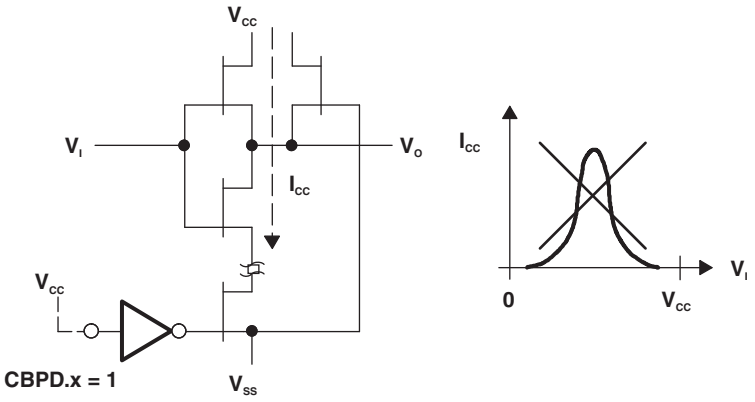


Figure 21-5. Transfer Characteristic and Power Dissipation in a CMOS Inverter/Buffer

21.2.8 Comp_B Interrupts

One interrupt flag and one interrupt vector is associated with the Comp_B.

The interrupt flag CBIFG is set on either the rising or falling edge of the comparator output, selected by the CBIES bit. If both the CBIE and the GIE bits are set, then the CBIFG interrupt flag generates an interrupt request.

21.2.9 Comp_B Used to Measure Resistive Elements

The Comp_B can be optimized to precisely measure resistive elements using single slope analog-to-digital conversion. For example, temperature can be converted into digital data using a thermistor, by comparing the thermistor's capacitor discharge time to that of a reference resistor as shown in Figure 21-6. A reference resister Rref is compared to Rmeas.

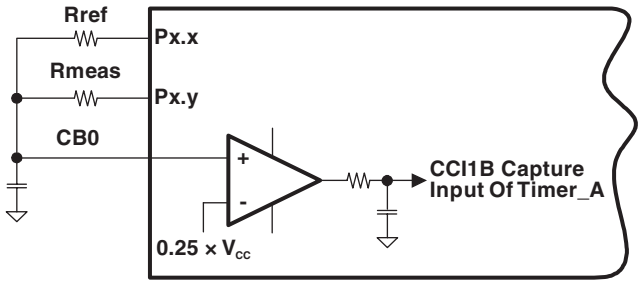


Figure 21-6. Temperature Measurement System

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The resources used to calculate the temperature sensed by Rmeas are:

- Two digital I/O pins charge and discharge the capacitor.
- I/O is set to output high (V_{CC}) to charge capacitor, reset to discharge.
- I/O is switched to high-impedance input with CBPDx set when not in use.
- One output charges and discharges the capacitor via Rref.
- One output discharges capacitor via Rmeas.
- The + terminal is connected to the positive terminal of the capacitor.
- The – terminal is connected to a reference level, for example 0.25 × V_{CC}.
- The output filter should be used to minimize switching noise.
- CBOUT is used to gate Timer_A CCI1B, capturing capacitor discharge time.

More than one resistive element can be measured. Additional elements are connected to CB0 with available I/O pins and switched to high impedance when not being measured.

The thermistor measurement is based on a ratiometric conversion principle. The ratio of two capacitor discharge times is calculated as shown in Figure 21-7.

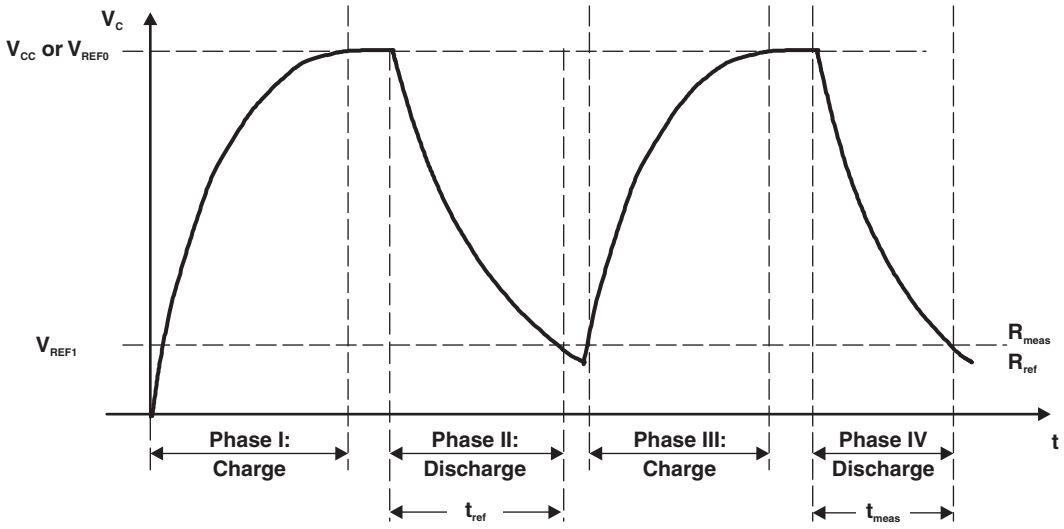


Figure 21-7. Timing for Temperature Measurement Systems

The V_{CC} voltage and the capacitor value should remain constant during the conversion, but are not critical since they cancel in the ratio:

$$\frac{N_{meas}}{N_{ref}} = \frac{-R_{meas} \times C \times \ln \frac{V_{ref1}}{V_{CC}}}{-R_{ref} \times C \times \ln \frac{V_{ref1}}{V_{CC}}}$$

$$\frac{N_{meas}}{N_{ref}} = \frac{R_{meas}}{R_{ref}}$$

$$R_{meas} = R_{ref} \times \frac{N_{meas}}{N_{ref}}$$

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3 Comp_B Registers

The Comp_B registers are listed in Table 21-1. The base address of the Comp_B module can be found in the device-specific data sheet.

Table 21-1. Comp_B Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	CBCTL0	Comp_B control register 0	Read/write	Word	0000h	Section 21.3.1
02h	CBCTL1	Comp_B control register 1	Read/write	Word	0000h	Section 21.3.2
04h	CBCTL2	Comp_B control register 2	Read/write	Word	0000h	Section 21.3.3
06h	CBCTL3	Comp_B control register 3	Read/write	Word	0000h	Section 21.3.4
0Ch	CBINT	Comp_B interrupt register	Read/write	Word	0000h	Section 21.3.5
0Eh	CBIV	Comp_B interrupt vector word	Read	Word	0000h	Section 21.3.6

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.1 CBCTL0 Register

Comp_B Control Register 0

Figure 21-8. CBCTL0 Register

15	14	13	12	11	10	9	8
CBIMEN	Reserved			CBIMSEL			
rw-0	r-0	r-0	r-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
CBIPEN	Reserved			CBIPSEL			
rw-0	r-0	r-0	r-0	rw-0	rw-0	rw-0	rw-0

Table 21-2. CBCTL0 Register Description

Bit	Field	Type	Reset	Description
15	CBIMEN	RW	0h	Channel input enable for the V– terminal of the comparator. 0b = Selected analog input channel for V– terminal is disabled. 1b = Selected analog input channel for V– terminal is enabled.
14-12	Reserved	R	0h	Reserved. Always reads as 0.
11-8	CBIMSEL	RW	0h	Channel input selected for the V– terminal of the comparator if CBIMEN is set to 1.
7	CBIPEN	RW	0h	Channel input enable for the V+ terminal of the comparator. 0b = Selected analog input channel for V+ terminal is disabled. 1b = Selected analog input channel for V+ terminal is enabled.
6-4	Reserved	R	0h	Reserved. Always reads as 0.
3-0	CBIPSEL	RW	0h	Channel input selected for the V+ terminal of the comparator if CBIPEN is set to 1.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.2 CBCTL1 Register

Comp_B Control Register 1

Figure 21-9. CBCTL1 Register

15	14	13	12	11	10	9	8
Reserved			CBMRVS	CBMRVL	CBON	CBPWRMD	
r-0	r-0	r-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
CBFDLY		CBEX	CBSHORT	CBIES	CBF	CBOUTPOL	CBOUT
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	r-0

Table 21-3. CBCTL1 Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12	CBMRVS	RW	0h	This bit defines if the comparator output selects between VREF0 or VREF1 if CBRS = 00, 01, or 10. 0b = Comparator output state selects between VREF0 or VREF1. 1b = CBMRVL selects between VREF0 or VREF1.
11	CBMRVL	RW	0h	This bit is valid of CBMRVS is set to 1. 0b = VREF0 is selected if CBRS = 00, 01, or 10. 1b = VREF1 is selected if CBRS = 00, 01, or 10.
10	CBON	RW	0h	On. This bit turns the comparator on. When the comparator is turned off the Comp_B consumes no power. 0b = Off 1b = On
9-8	CBPWRMD	RW	0h	Power mode. Not all modes are supported in all products. See devices specific data sheet for details. 00b = High-speed mode (optional) 01b = Normal mode (optional) 10b = Ultra-low-power mode (optional) 11b = Reserved
7-6	CBFDLY	RW	0h	Filter delay. The filter delay can be selected in 4 steps. See the device-specific data sheet for details. 00b = Typical filter delay of 450 ns 01b = Typical filter delay of 900 ns 10b = Typical filter delay of 1800 ns 11b = Typical filter delay of 3600 ns
5	CBEX	RW	0h	Exchange. This bit permutes the comparator 0 inputs and inverts the comparator 0 output.
4	CBSHORT	RW	0h	Input short. This bit shorts the + and – input terminals. 0b = Inputs not shorted 1b = Inputs shorted
3	CBIES	RW	0h	Interrupt edge select for CBIIFG and CBIFG 0b = Rising edge for CBIFG, falling edge for CBIIFG 1b = Falling edge for CBIFG, rising edge for CBIIFG
2	CBF	RW	0h	Output filter 0b = Comp_B output is not filtered 1b = Comp_B output is filtered
1	CBOUTPOL	RW	0h	Output polarity. This bit defines the CBOUT polarity. 0b = Noninverted 1b = Inverted

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 21-3. CBCTL1 Register Description (continued)

Bit	Field	Type	Reset	Description
0	CBOUT	R	0h	Output value. This bit reflects the value of the Comp_B output. Writing this bit has no effect on the comparator output.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.3 CBCTL2 Register

Comp_B Control Register 2

Figure 21-10. CBCTL2 Register

15	14	13	12	11	10	9	8
CBREFACC	CBREFL		CBREF1				
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
CBRS		CBRSEL	CBREF0				
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 21-4. CBCTL2 Register Description

Bit	Field	Type	Reset	Description
15	CBREFACC	RW	0h	Reference accuracy. A reference voltage is requested only if CBREFL > 0. 0b = Static mode 1b = Clocked (low-power, low-accuracy) mode
14-13	CBREFL	RW	0h	Reference voltage level 00b = Reference voltage is disabled. No reference voltage is requested. 01b = 1.5 V 10b = 2.0 V 11b = 2.5 V
12-8	CBREF1	RW	0h	Reference resistor tap 1. This register defines the tap of the resistor string while CBOUT = 1.
7-6	CBRS	RW	0h	Reference source. This bit define if the reference voltage is derived from VCC or from the precise shared reference. 00b = No current is drawn by the reference curcuitry. 01b = VCC applied to the resistor ladder 10b = Shared reference voltage applied to the resistor ladder. 11b = Shared reference voltage supplied to V(CREF). Resistor ladder is off.
5	CBRSEL	RW	0h	Reference select. This bit selects which terminal the V(CCREF) is applied to. 0b = When CBEX = 0: V(REF) is applied to the + terminal; When CBEX = 1: V(REF) is applied to the – terminal 1b = When CBEX = 0: V(REF) is applied to the – terminal; When CBEX = 1: V(REF) is applied to the + terminal
4-0	CBREF0	RW	0h	Reference resistor tap 0. This register defines the tap of the resistor string while CBOUT = 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.4 CBCTL3 Register

Comp_B Control Register 3

Figure 21-11. CBCTL3 Register

15	14	13	12	11	10	9	8
CBPD15	CBPD14	CBPD13	CBPD14	CBPD11	CBPD10	CBPD9	CBPD8
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
CBPD7	CBPD6	CBPD5	CBPD4	CBPD3	CBPD2	CBPD1	CBPD0
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 21-5. CBCTL3 Register Description

Bit	Field	Type	Reset	Description
15	CBPD15	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD15 disables the port of the comparator channel 15. 0b = Input buffer enabled 1b = Input buffer disabled
14	CBPD14	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD14 disables the port of the comparator channel 14. 0b = Input buffer enabled 1b = Input buffer disabled
13	CBPD13	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD13 disables the port of the comparator channel 13. 0b = Input buffer enabled 1b = Input buffer disabled
12	CBPD12	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD12 disables the port of the comparator channel 12. 0b = Input buffer enabled 1b = Input buffer disabled
11	CBPD11	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD11 disables the port of the comparator channel 11. 0b = Input buffer enabled 1b = Input buffer disabled
10	CBPD10	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD10 disables the port of the comparator channel 10. 0b = Input buffer enabled 1b = Input buffer disabled
9	CBPD9	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD9 disables the port of the comparator channel 9. 0b = Input buffer enabled 1b = Input buffer disabled
8	CBPD8	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD8 disables the port of the comparator channel 8. 0b = Input buffer enabled 1b = Input buffer disabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 21-5. CBCTL3 Register Description (continued)

Bit	Field	Type	Reset	Description
7	CBPD7	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD7 disables the port of the comparator channel 7. 0b = Input buffer enabled 1b = Input buffer disabled
6	CBPD6	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD6 disables the port of the comparator channel 6. 0b = Input buffer enabled 1b = Input buffer disabled
5	CBPD5	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD5 disables the port of the comparator channel 5. 0b = Input buffer enabled 1b = Input buffer disabled
4	CBPD4	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD4 disables the port of the comparator channel 4. 0b = Input buffer enabled 1b = Input buffer disabled
3	CBPD3	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD3 disables the port of the comparator channel 3. 0b = Input buffer enabled 1b = Input buffer disabled
2	CBPD2	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD2 disables the port of the comparator channel 2. 0b = Input buffer enabled 1b = Input buffer disabled
1	CBPD1	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD1 disables the port of the comparator channel 1. 0b = Input buffer enabled 1b = Input buffer disabled
0	CBPD0	RW	0h	Port disable. This bit individually disables the input buffer for the pins of the port associated with Comp_B. The bit CBPD0 disables the port of the comparator channel 0. 0b = Input buffer enabled 1b = Input buffer disabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.5 CBINT Register

Comp_B Interrupt Control Register

Figure 21-12. CBINT Register

15	14	13	12	11	10	9	8
Reserved						CBIIE	CBIE
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0
7	6	5	4	3	2	1	0
Reserved						CBIIFG	CBIFG
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0

Table 21-6. CBINT Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9	CBIIE	RW	0h	Comp_B output interrupt enable inverted polarity 0b = Interrupt is disabled 1b = Interrupt is enabled
8	CBIE	RW	0h	Comp_B output interrupt enable 0b = Interrupt is disabled 1b = Interrupt is enabled
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	CBIIFG	RW	0h	Comp_B output inverted interrupt flag. The bit CBIES defines the transition of the output setting this bit. 0b = No interrupt pending 1b = Output interrupt pending
0	CBIFG	RW	0h	Comp_B output interrupt flag. The bit CBIES defines the transition of the output setting this bit. 0b = No interrupt pending 1b = Output interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

21.3.6 CBIV Register

Comp_B Interrupt Vector Word Register

Figure 21-13. CBIV Register

15	14	13	12	11	10	9	8
CBIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
CBIV							
r0	r0	r0	r0	r0	r-0	r-0	r0

Table 21-7. CBIV Register Description

Bit	Field	Type	Reset	Description
15-0	CBIV	R	0h	Comp_B interrupt vector word register. The interrupt vector register reflects only interrupt flags whose interrupt enable bit are set. Reading the CBIV register clears the pending interrupt flag with the highest priority. 00h = No interrupt pending 02h = Interrupt Source: CBOUT interrupt; Interrupt Flag: CBIFG; Interrupt Priority: Highest 04h = Interrupt Source: CBOUT interrupt inverted polarity; Interrupt Flag: CBIIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Universal Serial Communication Interface – UART Mode

The universal serial communication interface (USCI) supports multiple serial communication modes with one hardware module. This chapter discusses the operation of the asynchronous UART mode.

Topic **Page**

22.1 Universal Serial Communication Interface (USCI) Overview 580
22.2 USCI Introduction – UART Mode 581
22.3 USCI Operation – UART Mode 583
22.4 USCI_A UART Mode Registers 599

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.1 Universal Serial Communication Interface (USCI) Overview

The USCI modules support multiple serial communication modes. Different USCI modules support different modes. Each different USCI module is named with a different letter. For example, USCI_A is different from USCI_B, etc. If more than one identical USCI module is implemented on one device, those modules are named with incrementing numbers. For example, if one device has two USCI_A modules, they are named USCI_A0 and USCI_A1. See the device-specific data sheet to determine which USCI modules, if any, are implemented on which devices.

USCI_Ax modules support:

- UART mode
- Pulse shaping for IrDA communications
- Automatic baud-rate detection for LIN communications
- SPI mode

USCI_Bx modules support:

- I^2C mode
- SPI mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.2 USCI Introduction – UART Mode

In asynchronous mode, the USCI_Ax modules connect the device to an external system via two external pins, UCAxRXD and UCAxTXD. UART mode is selected when the UCSYNC bit is cleared.

UART mode features include:

- 7- or 8-bit data with odd, even, or non-parity
- Independent transmit and receive shift registers
- Separate transmit and receive buffer registers
- LSB-first or MSB-first data transmit and receive
- Built-in idle-line and address-bit communication protocols for multiprocessor systems
- Receiver start-edge detection for auto wake up from LPMx modes
- Programmable baud rate with modulation for fractional baud-rate support
- Status flags for error detection and suppression
- Status flags for address detection
- Independent interrupt capability for receive and transmit

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Figure 22-1 shows the USCI_Ax when configured for UART mode.

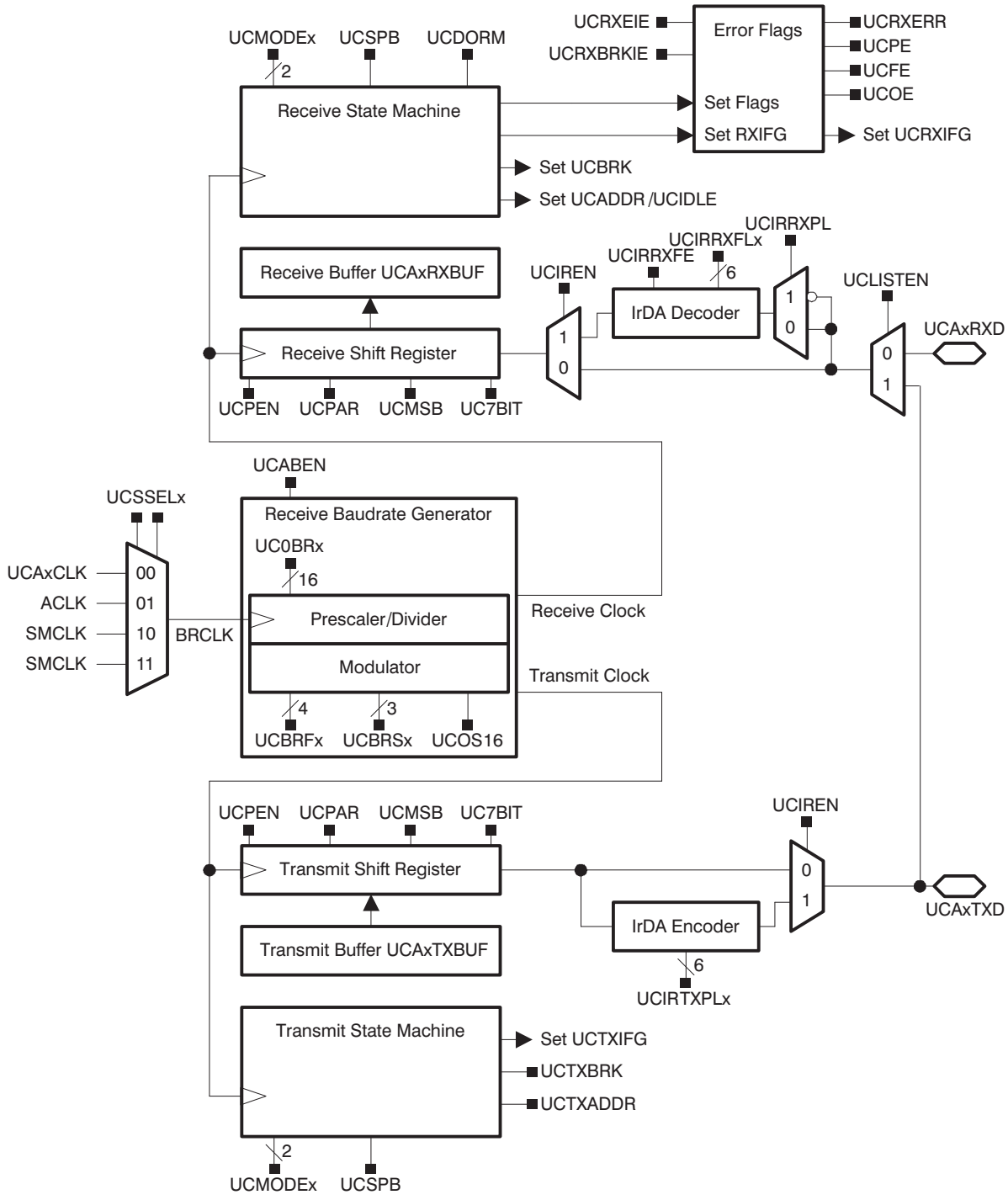


Figure 22-1. USCI_Ax Block Diagram – UART Mode (UCSYNC = 0)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3 USCI Operation – UART Mode

In UART mode, the USCI transmits and receives characters at a bit rate asynchronous to another device. Timing for each character is based on the selected baud rate of the USCI. The transmit and receive functions use the same baud-rate frequency.

22.3.1 USCI Initialization and Reset

The USCI is reset by a PUC or by setting the UCSWRST bit. After a PUC, the UCSWRST bit is automatically set, keeping the USCI in a reset condition. When set, the UCSWRST bit resets the UCRXIE, UCTXIE, UCRXIFG, UCRXERR, UCBRK, UCPE, UCOE, UCFE, UCSTOE, and UCBTOE bits, and sets the UCTXIFG bit. Clearing UCSWRST releases the USCI for operation.

To avoid unpredictable behavior, configure or reconfigure the USCI_A module only when UCSWRST is set.

NOTE: Initializing or reconfiguring the USCI module

The recommended USCI initialization/reconfiguration process is:

1. Set UCSWRST (`BIS.B #UCSWRST,&UCAxCTL1`).
2. Initialize all USCI registers with UCSWRST = 1 (including UCAxCTL1).
3. Configure ports.
4. Clear UCSWRST via software (`BIC.B #UCSWRST,&UCAxCTL1`).
5. Enable interrupts (optional) via UCRXIE and/or UCTXIE.

22.3.2 Character Format

The UART character format (see Figure 22-2) consists of a start bit, seven or eight data bits, an even/odd/no parity bit, an address bit (address-bit mode), and one or two stop bits. The UCMSB bit controls the direction of the transfer and selects LSB or MSB first. LSB first is typically required for UART communication.

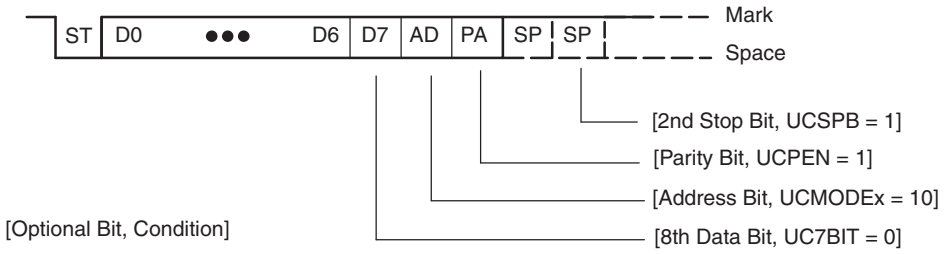


Figure 22-2. Character Format

22.3.3 Asynchronous Communication Format

When two devices communicate asynchronously, no multiprocessor format is required for the protocol. When three or more devices communicate, the USCI supports the idle-line and address-bit multiprocessor communication formats.

22.3.3.1 Idle-Line Multiprocessor Format

When UCMODEx = 01, the idle-line multiprocessor format is selected. Blocks of data are separated by an idle time on the transmit or receive lines (see Figure 22-3). An idle receive line is detected when ten or more continuous ones (marks) are received after the one or two stop bits of a character. The baud-rate generator is switched off after reception of an idle line until the next start edge is detected. When an idle line is detected, the UCIDLE bit is set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The first character received after an idle period is an address character. The UCIDLE bit is used as an address tag for each block of characters. In idle-line multiprocessor format, this bit is set when a received character is an address.

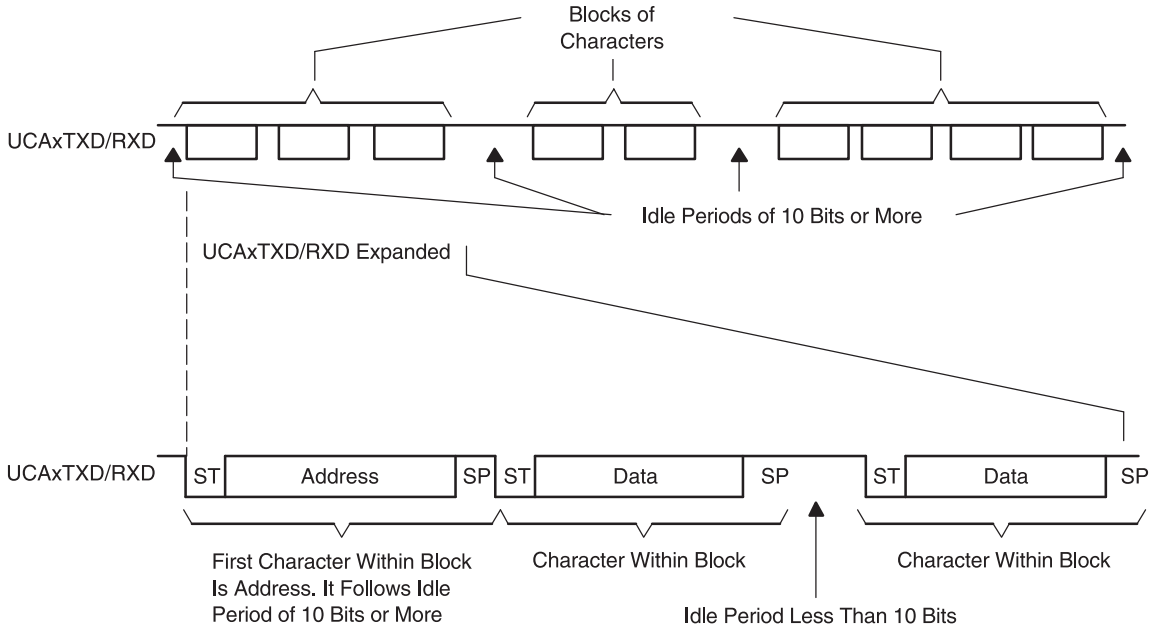


Figure 22-3. Idle-Line Format

The UCDORM bit is used to control data reception in the idle-line multiprocessor format. When UCDORM = 1, all non-address characters are assembled but not transferred into the UCAxRXBUF, and interrupts are not generated. When an address character is received, the character is transferred into UCAxRXBUF, UCRXIFG is set, and any applicable error flag is set when UCRXEIE = 1. When UCRXEIE = 0 and an address character is received but has a framing error or parity error, the character is not transferred into UCAxRXBUF and UCRXIFG is not set.

If an address is received, user software can validate the address and must reset UCDORM to continue receiving data. If UCDORM remains set, only address characters are received. When UCDORM is cleared during the reception of a character, the receive interrupt flag is set after the reception completed. The UCDORM bit is not modified by the USCI hardware automatically.

For address transmission in idle-line multiprocessor format, a precise idle period can be generated by the USCI to generate address character identifiers on UCAxTXD. The double-buffered UCTXADDR flag indicates if the next character loaded into UCAxTXBUF is preceded by an idle line of 11 bits. UCTXADDR is automatically cleared when the start bit is generated.

22.3.3.1.1 Transmitting an Idle Frame

The following procedure sends out an idle frame to indicate an address character followed by associated data:

1. Set UCTXADDR, then write the address character to UCAxTXBUF. UCAxTXBUF must be ready for new data (UCTXIFG = 1).

   This generates an idle period of exactly 11 bits followed by the address character. UCTXADDR is reset automatically when the address character is transferred from UCAxTXBUF into the shift register.
2. Write desired data characters to UCAxTXBUF. UCAxTXBUF must be ready for new data (UCTXIFG = 1).

   The data written to UCAxTXBUF is transferred to the shift register and transmitted as soon as the shift register is ready for new data.

   The idle-line time must not be exceeded between address and data transmission or between data transmissions. Otherwise, the transmitted data is misinterpreted as an address.

Copyright © 2009–2013, Texas Instruments Incorporated

22.3.3.2 Address-Bit Multiprocessor Format

When UCMODEx = 10, the address-bit multiprocessor format is selected. Each processed character contains an extra bit used as an address indicator (see Figure 22-4). The first character in a block of characters carries a set address bit that indicates that the character is an address. The USCI UCADDR bit is set when a received character has its address bit set and is transferred to UCAxRXBUF.

The UCDORM bit is used to control data reception in the address-bit multiprocessor format. When UCDORM is set, data characters with address bit = 0 are assembled by the receiver but are not transferred to UCAxRXBUF and no interrupts are generated. When a character containing a set address bit is received, the character is transferred into UCAxRXBUF, UCRXIFG is set, and any applicable error flag is set when UCRXEIE = 1. When UCRXEIE = 0 and a character containing a set address bit is received but has a framing error or parity error, the character is not transferred into UCAxRXBUF and UCRXIFG is not set.

If an address is received, user software can validate the address and must reset UCDORM to continue receiving data. If UCDORM remains set, only address characters with address bit = 1 are received. The UCDORM bit is not modified by the USCI hardware automatically.

When UCDORM = 0, all received characters set the receive interrupt flag UCRXIFG. If UCDORM is cleared during the reception of a character, the receive interrupt flag is set after the reception is completed.

For address transmission in address-bit multiprocessor mode, the address bit of a character is controlled by the UCTXADDR bit. The value of the UCTXADDR bit is loaded into the address bit of the character transferred from UCAxTXBUF to the transmit shift register. UCTXADDR is automatically cleared when the start bit is generated.

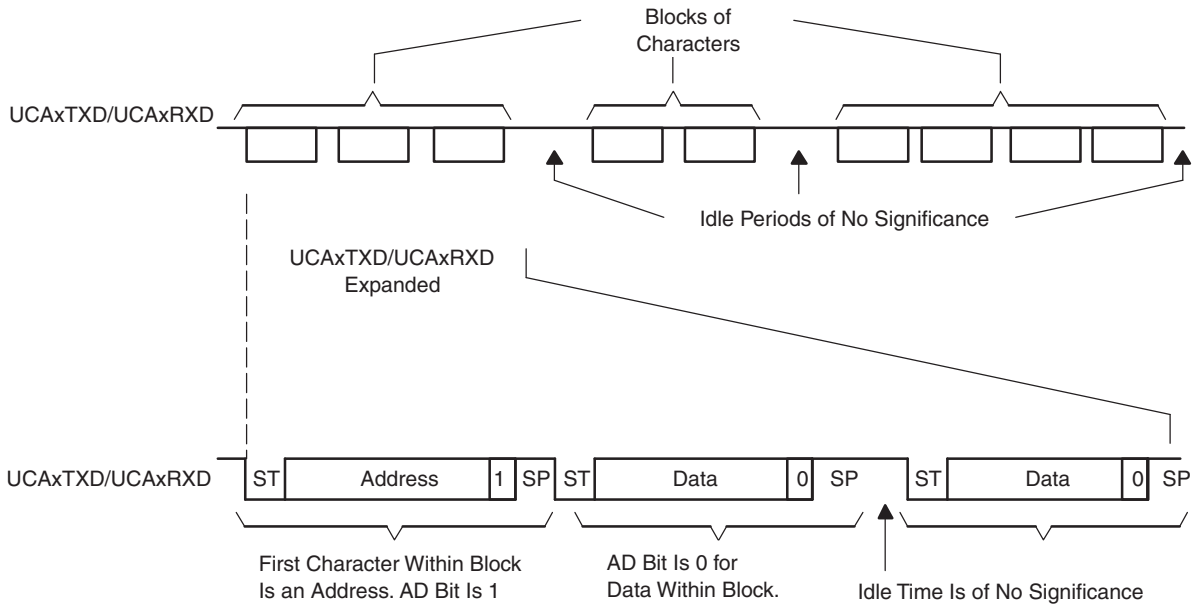


Figure 22-4. Address-Bit Multiprocessor Format

22.3.3.2.1 Break Reception and Generation

When UCMODEx = 00, 01, or 10, the receiver detects a break when all data, parity, and stop bits are low, regardless of the parity, address mode, or other character settings. When a break is detected, the UCBRK bit is set. If the break interrupt enable bit (UCBRKIE) is set, the receive interrupt flag UCRXIFG is also set. In this case, the value in UCAxRXBUF is 0h, because all data bits were zero.

To transmit a break, set the UCTXBRK bit, then write 0h to UCAxTXBUF. UCAxTXBUF must be ready for new data (UCTXIFG = 1). This generates a break with all bits low. UCTXBRK is automatically cleared when the start bit is generated.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.4 Automatic Baud-Rate Detection

When UCMODEx = 11, UART mode with automatic baud-rate detection is selected. For automatic baud-rate detection, a data frame is preceded by a synchronization sequence that consists of a break and a synch field. A break is detected when 11 or more continuous zeros (spaces) are received. If the length of the break exceeds 21 bit times the break timeout error flag UCBTOE is set. The USCI can not transmit data while receiving the break/sync field. The synch field follows the break as shown in Figure 22-5.

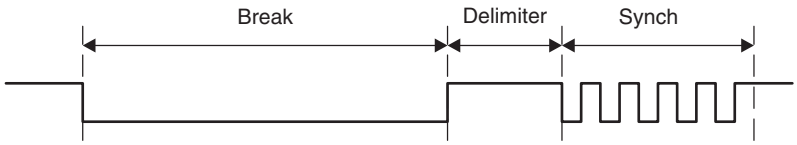


Figure 22-5. Auto Baud-Rate Detection – Break/Synch Sequence

For LIN conformance, the character format should be set to eight data bits, LSB first, no parity, and one stop bit. No address bit is available.

The synch field consists of the data 055h inside a byte field (see Figure 22-6). The synchronization is based on the time measurement between the first falling edge and the last falling edge of the pattern. The transmit baud-rate generator is used for the measurement if automatic baud-rate detection is enabled by setting UCABDEN. Otherwise, the pattern is received but not measured. The result of the measurement is transferred into the baud-rate control registers (UCAxBR0, UCAxBR1, and UCAxMCTL). If the length of the synch field exceeds the measurable time, the synch timeout error flag UCSTOE is set.

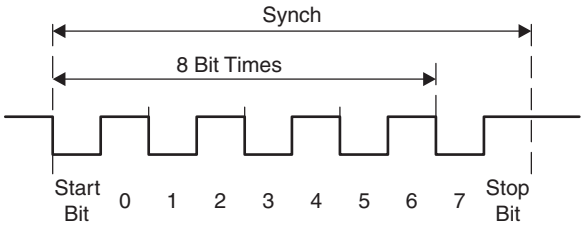


Figure 22-6. Auto Baud-Rate Detection – Synch Field

The UCDORM bit is used to control data reception in this mode. When UCDORM is set, all characters are received but not transferred into the UCAxRXBUF, and interrupts are not generated. When a break/synch field is detected, the UCBRK flag is set. The character following the break/synch field is transferred into UCAxRXBUF and the UCRXIFG interrupt flag is set. Any applicable error flag is also set. If the UCBRKIE bit is set, reception of the break/synch sets the UCRXIFG. The UCBRK bit is reset by user software or by reading the receive buffer UCAxRXBUF.

When a break/synch field is received, user software must reset UCDORM to continue receiving data. If UCDORM remains set, only the character after the next reception of a break/synch field is received. The UCDORM bit is not modified by the USCI hardware automatically.

When UCDORM = 0, all received characters set the receive interrupt flag UCRXIFG. If UCDORM is cleared during the reception of a character, the receive interrupt flag is set after the reception is complete.

The counter used to detect the baud rate is limited to 07FFFh (32767) counts. This means the minimum baud rate detectable is 488 baud in oversampling mode and 30 baud in low-frequency mode.

The automatic baud-rate detection mode can be used in a full-duplex communication system with some restrictions. The USCI can not transmit data while receiving the break/sync field and, if a 0h byte with framing error is received, any data transmitted during this time gets corrupted. The latter case can be discovered by checking the received data and the UCFE bit.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.4.1 Transmitting a Break/Synch Field

The following procedure transmits a break/synch field:

1. Set UCTXBRK with UMODEx = 11.
2. Write 055h to UCAxTXBUF. UCAxTXBUF must be ready for new data (UCTXIFG = 1).

   This generates a break field of 13 bits followed by a break delimiter and the synch character. The length of the break delimiter is controlled with the UCDELIMx bits. UCTXBRK is reset automatically when the synch character is transferred from UCAxTXBUF into the shift register.
3. Write desired data characters to UCAxTXBUF. UCAxTXBUF must be ready for new data (UCTXIFG = 1).

   The data written to UCAxTXBUF is transferred to the shift register and transmitted as soon as the shift register is ready for new data.

22.3.5 IrDA Encoding and Decoding

When UCIREN is set, the IrDA encoder and decoder are enabled and provide hardware bit shaping for IrDA communication.

22.3.5.1 IrDA Encoding

The encoder sends a pulse for every zero bit in the transmit bit stream coming from the UART (see Figure 22-7). The pulse duration is defined by UCIRTXPLx bits specifying the number of one-half clock periods of the clock selected by UCIRTXCLK.

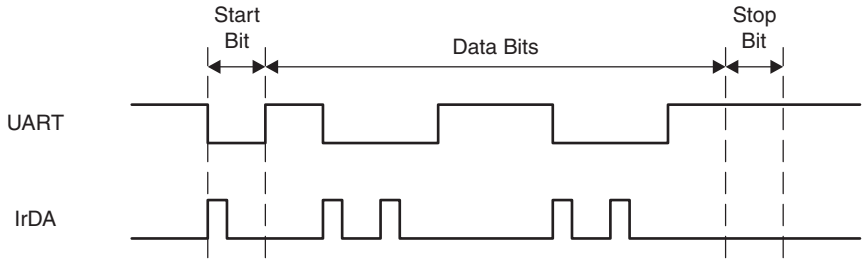


Figure 22-7. UART vs IrDA Data Format

To set the pulse time of 3/16 bit period required by the IrDA standard, the BITCLK16 clock is selected with UCIRTXCLK = 1 ,and the pulse length is set to six one-half clock cycles with UCIRTXPLx = 6 – 1 = 5.

When UCIRTXCLK = 0, the pulse length t_{PULSE} is based on BRCLK and is calculated as:

$UCIRTXPLx = t_{PULSE} \times 2 \times f_{BRCLK} - 1$

When UCIRTXCLK = 0 ,the prescaler UCBRx must to be set to a value greater or equal to 5.

22.3.5.2 IrDA Decoding

The decoder detects high pulses when UCIRRXPL = 0. Otherwise, it detects low pulses. In addition to the analog deglitch filter, an additional programmable digital filter stage can be enabled by setting UCIRRXFE. When UCIRRXFE is set, only pulses longer than the programmed filter length are passed. Shorter pulses are discarded. The equation to program the filter length UCIRRXFLx is:

$UCIRRXFLx = (t_{PULSE} - t_{WAKE}) \times 2 \times f_{BRCLK} - 4$

Where:

t_{PULSE} = Minimum receive pulse width

t_{WAKE} = Wake time from any low-power mode. Zero when the device is in active mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.6 Automatic Error Detection

Glitch suppression prevents the USCI from being accidentally started. Any pulse on UCAxRXD shorter than the deglitch time t_t (approximately 150 ns) is ignored (see the device-specific data sheet for parameters).

When a low period on UCAxRXD exceeds t_t, a majority vote is taken for the start bit. If the majority vote fails to detect a valid start bit, the USCI halts character reception and waits for the next low period on UCAxRXD. The majority vote is also used for each bit in a character to prevent bit errors.

The USCI module automatically detects framing errors, parity errors, overrun errors, and break conditions when receiving characters. The bits UCFE, UCPE, UCOE, and UCBRK are set when their respective condition is detected. When the error flags UCFE, UCPE, or UCOE are set, UCRXERR is also set. The error conditions are described in Table 22-1.

Table 22-1. Receive Error Conditions

Error Condition	Error Flag	Description
Framing error	UCFE	A framing error occurs when a low stop bit is detected. When two stop bits are used, both stop bits are checked for framing error. When a framing error is detected, the UCFE bit is set.
Parity error	UCPE	A parity error is a mismatch between the number of 1s in a character and the value of the parity bit. When an address bit is included in the character, it is included in the parity calculation. When a parity error is detected, the UCPE bit is set.
Receive overrun	UCOE	An overrun error occurs when a character is loaded into UCAxRXBUF before the prior character has been read. When an overrun occurs, the UCOE bit is set.
Break condition	UCBRK	When not using automatic baud-rate detection, a break is detected when all data, parity, and stop bits are low. When a break condition is detected, the UCBRK bit is set. A break condition can also set the interrupt flag UCRXIFG if the break interrupt enable UCBRKIE bit is set.

When UCRXEIE = 0 and a framing error or parity error is detected, no character is received into UCAxRXBUF. When UCRXEIE = 1, characters are received into UCAxRXBUF and any applicable error bit is set.

When any of the UCFE, UCPE, UCOE, UCBRK, or UCRXERR bit is set, the bit remains set until user software resets it or UCAxRXBUF is read. UCOE must be reset by reading UCAxRXBUF. Otherwise, it does not function properly. To detect overflows reliably the following flow is recommended. After a character was received and UCAxRXIFG is set, first read UCAxSTAT to check the error flags including the overflow flag UCOE. Read UCAxRXBUF next. This clears all error flags except UCOE, if UCAxRXBUF was overwritten between the read access to UCAxSTAT and to UCAxRXBUF. Therefore, the UCOE flag should be checked after reading UCAxRXBUF to detect this condition. Note that, in this case, the UCRXERR flag is not set.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.7 USCI Receive Enable

The USCI module is enabled by clearing the UCSWRST bit and the receiver is ready and in an idle state. The receive baud rate generator is in a ready state but is not clocked nor producing any clocks.

The falling edge of the start bit enables the baud rate generator and the UART state machine checks for a valid start bit. If no valid start bit is detected the UART state machine returns to its idle state and the baud rate generator is turned off again. If a valid start bit is detected, a character is received.

When the idle-line multiprocessor mode is selected with UCMODEx = 01 the UART state machine checks for an idle line after receiving a character. If a start bit is detected another character is received. Otherwise the UCIDLE flag is set after 10 ones are received and the UART state machine returns to its idle state and the baud rate generator is turned off.

22.3.7.1 Receive Data Glitch Suppression

Glitch suppression prevents the USCI from being accidentally started. Any glitch on UCAxRXD shorter than the deglitch time t_t (approximately 150 ns) is ignored by the USCI, and further action is initiated as shown in Figure 22-8 (see the device-specific data sheet for parameters).

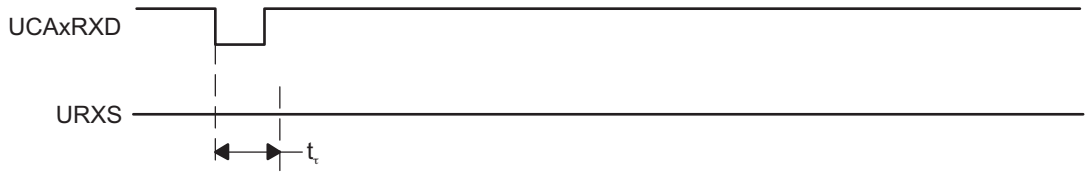


Figure 22-8. Glitch Suppression, USCI Receive Not Started

When a glitch is longer than t_t, or a valid start bit occurs on UCAxRXD, the USCI receive operation is started and a majority vote is taken (see Figure 22-9). If the majority vote fails to detect a start bit, the USCI halts character reception.

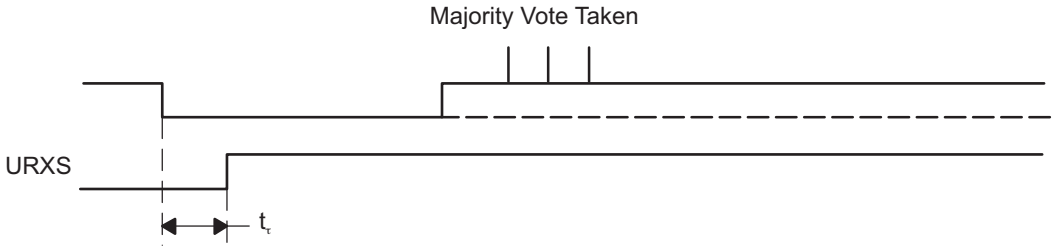


Figure 22-9. Glitch Suppression, USCI Activated

22.3.8 USCI Transmit Enable

The USCI module is enabled by clearing the UCSWRST bit and the transmitter is ready and in an idle state. The transmit baud-rate generator is ready but is not clocked nor producing any clocks.

A transmission is initiated by writing data to UCAxTXBUF. When this occurs, the baud-rate generator is enabled, and the data in UCAxTXBUF is moved to the transmit shift register on the next BITCLK after the transmit shift register is empty. UCTXIFG is set when new data can be written into UCAxTXBUF.

Transmission continues as long as new data is available in UCAxTXBUF at the end of the previous byte transmission. If new data is not in UCAxTXBUF when the previous byte has transmitted, the transmitter returns to its idle state and the baud-rate generator is turned off.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.9 UART Baud-Rate Generation

The USCI baud-rate generator is capable of producing standard baud rates from nonstandard source frequencies. It provides two modes of operation selected by the UCOS16 bit. The baud-rate is generate using the BRCLK that can be sourced by the external clock UCAxCLK, or the internal clocks ACLK or SMCLK depending on the UCSSELx settings.

22.3.9.1 Low-Frequency Baud-Rate Generation

The low-frequency mode is selected when UCOS16 = 0. This mode allows generation of baud rates from low frequency clock sources (for example, 9600 baud from a 32768-Hz crystal). By using a lower input frequency, the power consumption of the module is reduced. Using this mode with higher frequencies and higher prescaler settings causes the majority votes to be taken in an increasingly smaller window and, thus, decrease the benefit of the majority vote.

In low-frequency mode, the baud-rate generator uses one prescaler and one modulator to generate bit clock timing. This combination supports fractional divisors for baud-rate generation. In this mode, the maximum USCI baud rate is one-third the UART source clock frequency BRCLK.

Timing for each bit is shown in Figure 22-10. For each bit received, a majority vote is taken to determine the bit value. These samples occur at the N/2 – 1/2, N/2, and N/2 + 1/2 BRCLK periods, where N is the number of BRCLKs per BITCLK.

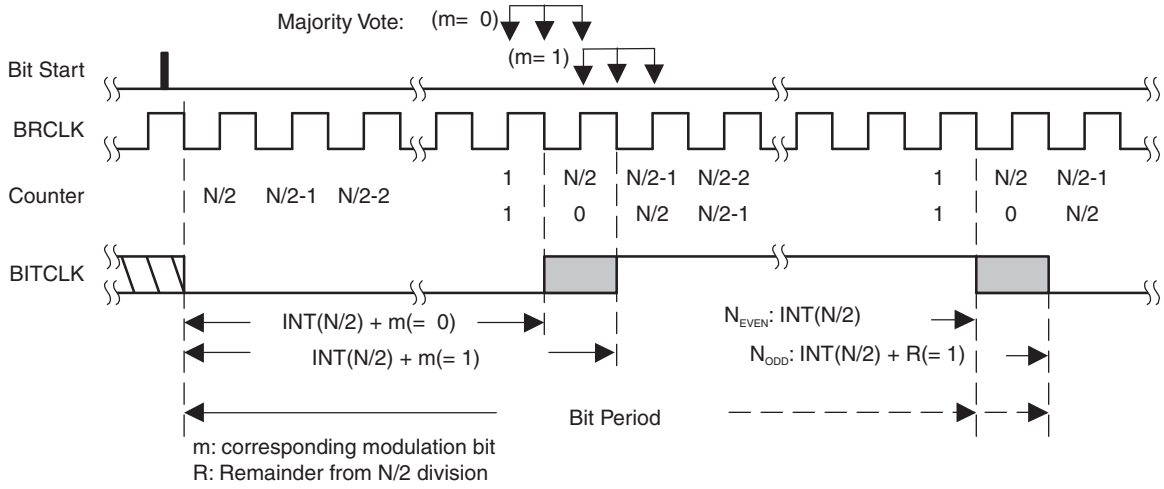


Figure 22-10. BITCLK Baud-Rate Timing With UCOS16 = 0

Modulation is based on the UCBRSx setting (see Table 22-2). A 1 in the table indicates that m = 1 and the corresponding BITCLK period is one BRCLK period longer than a BITCLK period with m = 0. The modulation wraps around after eight bits but restarts with each new start bit.

Table 22-2. BITCLK Modulation Pattern

UCBRSx	Bit 0 (Start Bit)	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0
3	0	1	0	1	0	1	0	0
4	0	1	0	1	0	1	0	1
5	0	1	1	1	0	1	0	1
6	0	1	1	1	0	1	1	1
7	0	1	1	1	1	1	1	1

Copyright © 2009–2013, Texas Instruments Incorporated

22.3.9.2 Oversampling Baud-Rate Generation

The oversampling mode is selected when UCOS16 = 1. This mode supports sampling a UART bit stream with higher input clock frequencies. This results in majority votes that are always 1/16 of a bit clock period apart. This mode also easily supports IrDA pulses with a 3/16 bit time when the IrDA encoder and decoder are enabled.

This mode uses one prescaler and one modulator to generate the BITCLK16 clock that is 16 times faster than the BITCLK. An additional divider and modulator stage generates BITCLK from BITCLK16. This combination supports fractional divisions of both BITCLK16 and BITCLK for baud-rate generation. In this mode, the maximum USCI baud rate is 1/16 the UART source clock frequency BRCLK. When UCBRx is set to 0 or 1, the first prescaler and modulator stage is bypassed and BRCLK is equal to BITCLK16 – in this case, no modulation for the BITCLK16 is possible and, thus, the UCBRFx bits are ignored.

Modulation for BITCLK16 is based on the UCBRFx setting (see Table 22-3). A 1 in the table indicates that the corresponding BITCLK16 period is one BRCLK period longer than the periods m = 0. The modulation restarts with each new bit timing.

Modulation for BITCLK is based on the UCBRSx setting (see Table 22-2) as previously described.

Table 22-3. BITCLK16 Modulation Pattern

UCBRFx	No. of BITCLK16 Clocks After Last Falling BITCLK Edge															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
00h	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01h	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02h	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
03h	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1
04h	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
05h	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1
06h	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
07h	0	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1
08h	0	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1
09h	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1
0Ah	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1
0Bh	0	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1
0Ch	0	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1
0Dh	0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1
0Eh	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
0Fh	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.10 Setting a Baud Rate

For a given BRCLK clock source, the baud rate used determines the required division factor N:

$N = f_{BRCLK}/\text{Baudrate}$

The division factor N is often a noninteger value, thus, at least one divider and one modulator stage is used to meet the factor as closely as possible.

If N is equal or greater than 16, the oversampling baud-rate generation mode can be chosen by setting UCOS16.

22.3.10.1 Low-Frequency Baud-Rate Mode Setting

In low-frequency mode, the integer portion of the divisor is realized by the prescaler:

UCBRx = INT(N)

and the fractional portion is realized by the modulator with the following nominal formula:

UCBRSx = round[(N – INT(N)) × 8]

Incrementing or decrementing the UCBRSx setting by one count may give a lower maximum bit error for any given bit. To determine if this is the case, a detailed error calculation must be performed for each bit for each UCBRSx setting.

22.3.10.2 Oversampling Baud-Rate Mode Setting

In the oversampling mode, the prescaler is set to:

UCBRx = INT(N/16)

and the first stage modulator is set to:

UCBRFx = round([(N/16) – INT(N/16)] × 16)

When greater accuracy is required, the UCBRSx modulator can also be implemented with values from 0 to 7. To find the setting that gives the lowest maximum bit error rate for any given bit, a detailed error calculation must be performed for all settings of UCBRSx from 0 to 7 with the initial UCBRFx setting, and with the UCBRFx setting incremented and decremented by one.

22.3.11 Transmit Bit Timing

The timing for each character is the sum of the individual bit timings. Using the modulation features of the baud-rate generator reduces the cumulative bit error. The individual bit error can be calculated using the following steps.

22.3.11.1 Low-Frequency Baud-Rate Mode Bit Timing

In low-frequency mode, calculate the length of bit i $T_{bit,TX}[i]$ based on the UCBRx and UCBRSx settings:

$T_{bit,TX}[i] = (1/f_{BRCLK})(UCBRx + m_{UCBRSx}[i])$

Where:

$m_{UCBRSx}[i]$ = Modulation of bit i from Table 22-2

22.3.11.2 Oversampling Baud-Rate Mode Bit Timing

In oversampling baud-rate mode, calculate the length of bit i $T_{bit,TX}[i]$ based on the baud-rate generator UCBRx, UCBRFx and UCBRSx settings:

$$T_{bit,TX}[i] = \frac{1}{f_{BRCLK}}\left((16 + m_{UCBRSx}[i]) \times UCBRx + \sum_{j=0}^{15} m_{UCBRFx}[j]\right)$$

Where:

$\sum_{j=0}^{15} m_{UCBRFx}[j]$ = Sum of ones from the corresponding row in Table 22-3

$m_{UCBRSx}[i]$ = Modulation of bit i from Table 22-2

Copyright © 2009–2013, Texas Instruments Incorporated

This results in an end-of-bit time $t_{bit,TX}[i]$ equal to the sum of all previous and the current bit times:

$$T_{bit,TX}[i] = \sum_{j=0}^{i} T_{bit,TX}[j]$$

To calculate bit error, this time is compared to the ideal bit time $t_{bit,ideal,TX}[i]$:

$t_{bit,ideal,TX}[i] = (1/Baudrate)(i + 1)$

This results in an error normalized to one ideal bit time (1/baudrate):

$Error_{TX}[i] = (t_{bit,TX}[i] – t_{bit,ideal,TX}[i]) \times Baudrate \times 100\%$

22.3.12 Receive Bit Timing

Receive timing error consists of two error sources. The first is the bit-to-bit timing error similar to the transmit bit timing error. The second is the error between a start edge occurring and the start edge being accepted by the USCI module. Figure 22-11 shows the asynchronous timing errors between data on the UCAxRXD pin and the internal baud-rate clock. This results in an additional synchronization error. The synchronization error t_{SYNC} is between –0.5 BRCLKs and +0.5 RCLKs, independent of the selected baud-rate generation mode.

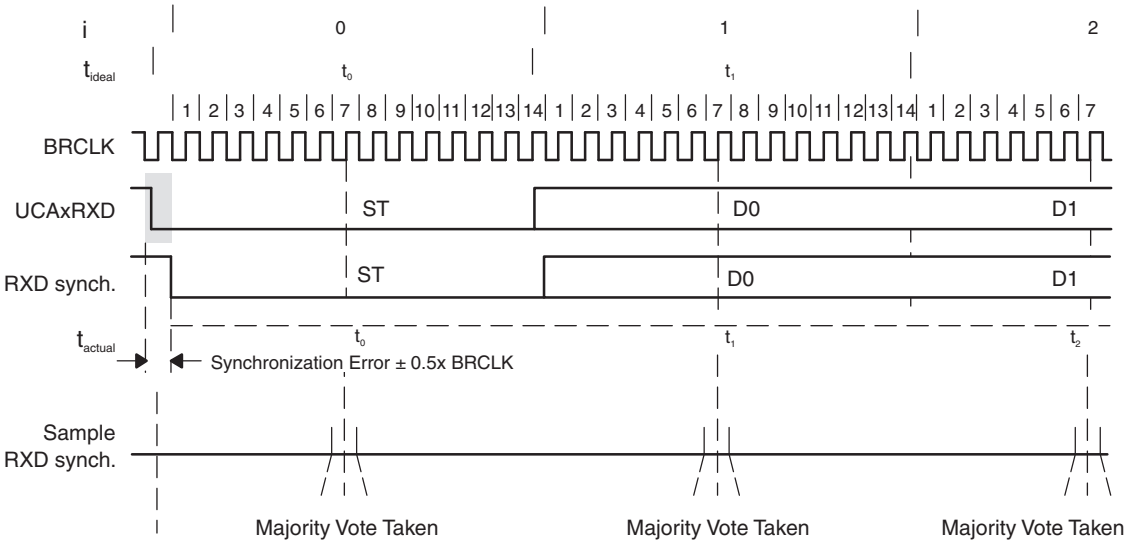


Figure 22-11. Receive Error

The ideal sampling time $t_{bit,ideal,RX}[i]$ is in the middle of a bit period:

$t_{bit,ideal,RX}[i] = (1/Baudrate)(i + 0.5)$

The real sampling time, $t_{bit,RX}[i]$, is equal to the sum of all previous bits according to the formulas shown in the transmit timing section, plus one-half BITCLK for the current bit i, plus the synchronization error t_{SYNC}.

This results in the following $t_{bit,RX}[i]$ for the low-frequency baud-rate mode:

$$t_{bit,RX}[i] = t_{SYNC} + \sum_{j=0}^{i-1} T_{bit,RX}[j] + \frac{1}{f_{BRCLK}} \left(INT(½UCBRx) + m_{UCBRSx}[i] \right)$$

Where:

$T_{bit,RX}[i] = (1/f_{BRCLK})(UCBRx + m_{UCBRSx}[i])$

$m_{UCBRSx}[i]$ = Modulation of bit i from Table 22-2

Copyright © 2009–2013, Texas Instruments Incorporated

For the oversampling baud-rate mode, the sampling time $t_{bit,RX}[i]$ of bit i is calculated by:

$$t_{bit,RX}[i] = t_{SYNC} + \sum_{j=0}^{i-1} T_{bit,RX}[j] + \frac{1}{f_{BRCLK}}\left((8 + m_{UCBRSx}[i]) \times UCBRx + \sum_{j=0}^{7+m_{UCBRSx}[i]} m_{UCBRFx}[j]\right)$$

Where:

$$T_{bit,RX}[i] = \frac{1}{f_{BRCLK}}\left((16 + m_{UCBRSx}[i]) \times UCBRx + \sum_{j=0}^{15} m_{UCBRFx}[j]\right)$$

$\sum_{j=0}^{7+m_{UCBRSx}[i]} m_{UCBRFx}[j]$ = Sum of ones from columns 0 to (7 + $m_{UCBRSx}[i]$) from the corresponding row in Table 22-3.

$m_{UCBRSx}[i]$ = Modulation of bit i from Table 22-2

This results in an error normalized to one ideal bit time (1/baudrate) according to the following formula:

$Error_{RX}[i] = (t_{bit,RX}[i] – t_{bit,ideal,RX}[i]) \times Baudrate \times 100\%$

22.3.13 Typical Baud Rates and Errors

Standard baud-rate data for UCBRx, UCBRSx, and UCBRFx are listed in Table 22-4 and Table 22-5 for a 32,768-Hz crystal sourcing ACLK and typical SMCLK frequencies. Please ensure that the selected BRCLK frequency does not exceed the device specific maximum USCI input frequency (see the device-specific data sheet).

The receive error is the accumulated time versus the ideal scanning time in the middle of each bit. The worst-case error is given for the reception of an 8-bit character with parity and one stop bit including synchronization error.

The transmit error is the accumulated timing error versus the ideal time of the bit period. The worst-case error is given for the transmission of an 8-bit character with parity and stop bit.

Table 22-4. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 0

BRCLK Frequency (Hz)	Baud Rate (baud)	UCBRx	UCBRSx	UCBRFx	Maximum TX Error (%)		Maximum RX Error (%)	
32,768	1200	27	2	0	-2.8	1.4	-5.9	2.0
32,768	2400	13	6	0	-4.8	6.0	-9.7	8.3
32,768	4800	6	7	0	-12.1	5.7	-13.4	19.0
32,768	9600	3	3	0	-21.1	15.2	-44.3	21.3
1,000,000	9600	104	1	0	-0.5	0.6	-0.9	1.2
1,000,000	19200	52	0	0	-1.8	0	-2.6	0.9
1,000,000	38400	26	0	0	-1.8	0	-3.6	1.8
1,000,000	57600	17	3	0	-2.1	4.8	-6.8	5.8
1,000,000	115200	8	6	0	-7.8	6.4	-9.7	16.1
1,048,576	9600	109	2	0	-0.2	0.7	-1.0	0.8
1,048,576	19200	54	5	0	-1.1	1.0	-1.5	2.5
1,048,576	38400	27	2	0	-2.8	1.4	-5.9	2.0
1,048,576	57600	18	1	0	-4.6	3.3	-6.8	6.6
1,048,576	115200	9	1	0	-1.1	10.7	-11.5	11.3
4,000,000	9600	416	6	0	-0.2	0.2	-0.2	0.4
4,000,000	19200	208	3	0	-0.2	0.5	-0.3	0.8
4,000,000	38400	104	1	0	-0.5	0.6	-0.9	1.2
4,000,000	57600	69	4	0	-0.6	0.8	-1.8	1.1
4,000,000	115200	34	6	0	-2.1	0.6	-2.5	3.1
4,000,000	230400	17	3	0	-2.1	4.8	-6.8	5.8
4,194,304	9600	436	7	0	-0.3	0	-0.3	0.2

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 22-4. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 0 (continued)

BRCLK Frequency (Hz)	Baud Rate (baud)	UCBRx	UCBRSx	UCBRFx	Maximum TX Error (%)		Maximum RX Error (%)	
4,194,304	19200	218	4	0	-0.2	0.2	-0.3	0.6
4,194,304	57600	72	7	0	-1.1	0.6	-1.3	1.9
4,194,304	115200	36	3	0	-1.9	1.5	-2.7	3.4
8,000,000	9600	833	2	0	-0.1	0	-0.2	0.1
8,000,000	19200	416	6	0	-0.2	0.2	-0.2	0.4
8,000,000	38400	208	3	0	-0.2	0.5	-0.3	0.8
8,000,000	57600	138	7	0	-0.7	0	-0.8	0.6
8,000,000	115200	69	4	0	-0.6	0.8	-1.8	1.1
8,000,000	230400	34	6	0	-2.1	0.6	-2.5	3.1
8,000,000	460800	17	3	0	-2.1	4.8	-6.8	5.8
8,388,608	9600	873	7	0	-0.1	0.06	-0.2	0,1
8,388,608	19200	436	7	0	-0.3	0	-0.3	0.2
8,388,608	57600	145	5	0	-0.5	0.3	-1.0	0.5
8,388,608	115200	72	7	0	-1.1	0.6	-1.3	1.9
12,000,000	9600	1250	0	0	0	0	-0.05	0.05
12,000,000	19200	625	0	0	0	0	-0.2	0
12,000,000	38400	312	4	0	-0.2	0	-0.2	0.2
12,000,000	57600	208	2	0	-0.5	0.2	-0.6	0.5
12,000,000	115200	104	1	0	-0.5	0.6	-0.9	1.2
12,000,000	230400	52	0	0	-1.8	0	-2.6	0.9
12,000,000	460800	26	0	0	-1.8	0	-3.6	1.8
16,000,000	9600	1666	6	0	-0.05	0.05	-0.05	0.1
16,000,000	19200	833	2	0	-0.1	0.05	-0.2	0.1
16,000,000	38400	416	6	0	-0.2	0.2	-0.2	0.4
16,000,000	57600	277	7	0	-0.3	0.3	-0.5	0.4
16,000,000	115200	138	7	0	-0.7	0	-0.8	0.6
16,000,000	230400	69	4	0	-0.6	0.8	-1.8	1.1
16,000,000	460800	34	6	0	-2.1	0.6	-2.5	3.1
16,777,216	9600	1747	5	0	-0.04	0.03	-0.08	0.05
16,777,216	19200	873	7	0	-0.09	0.06	-0.2	0.1
16,777,216	57600	291	2	0	-0.2	0.2	-0.5	0.2
16,777,216	115200	145	5	0	-0.5	0.3	-1.0	0.5
20,000,000	9600	2083	2	0	-0.05	0.02	-0.09	0.02
20,000,000	19200	1041	6	0	-0.06	0.06	-0.1	0.1
20,000,000	38400	520	7	0	-0.2	0.06	-0.2	0.2
20,000,000	57600	347	2	0	-0.06	0.2	-0.3	0.3
20,000,000	115200	173	5	0	-0.4	0.3	-0.8	0.5
20,000,000	230400	86	7	0	-1.0	0.6	-1.0	1.7
20,000,000	460800	43	3	0	-1.4	1.3	-3.3	1.8

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 22-5. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 1

BRCLK Frequency (Hz)	Baud Rate (baud)	UCBRx	UCBRSx	UCBRFx	Maximum TX Error (%)		Maximum RX Error (%)	
1,000,000	9600	6	0	8	-1.8	0	-2.2	0.4
1,000,000	19200	3	0	4	-1.8	0	-2.6	0.9
1,048,576	9600	6	0	13	-2.3	0	-2.2	0.8
1,048,576	19200	3	1	6	-4.6	3.2	-5.0	4.7
4,000,000	9600	26	0	1	0	0.9	0	1.1
4,000,000	19200	13	0	0	-1.8	0	-1.9	0.2
4,000,000	38400	6	0	8	-1.8	0	-2.2	0.4
4,000,000	57600	4	5	3	-3.5	3.2	-1.8	6.4
4,000,000	115200	2	3	2	-2.1	4.8	-2.5	7.3
4,194,304	9600	27	0	5	0	0.2	0	0.5
4,194,304	19200	13	0	10	-2.3	0	-2.4	0.1
4,194,304	57600	4	4	7	-2.5	2.5	-1.3	5.1
4,194,304	115200	2	6	3	-3.9	2.0	-1.9	6.7
8,000,000	9600	52	0	1	-0.4	0	-0.4	0.1
8,000,000	19200	26	0	1	0	0.9	0	1.1
8,000,000	38400	13	0	0	-1.8	0	-1.9	0.2
8,000,000	57600	8	0	11	0	0.88	0	1.6
8,000,000	115200	4	5	3	-3.5	3.2	-1.8	6.4
8,000,000	230400	2	3	2	-2.1	4.8	-2.5	7.3
8,388,608	9600	54	0	10	0	0.2	-0.05	0.3
8,388,608	19200	27	0	5	0	0.2	0	0.5
8,388,608	57600	9	0	2	0	2.8	-0.2	3.0
8,388,608	115200	4	4	7	-2.5	2.5	-1.3	5.1
12,000,000	9600	78	0	2	0	0	-0.05	0.05
12,000,000	19200	39	0	1	0	0	0	0.2
12,000,000	38400	19	0	8	-1.8	0	-1.8	0.1
12,000,000	57600	13	0	0	-1.8	0	-1.9	0.2
12,000,000	115200	6	0	8	-1.8	0	-2.2	0.4
12,000,000	230400	3	0	4	-1.8	0	-2.6	0.9
16,000,000	9600	104	0	3	0	0.2	0	0.3
16,000,000	19200	52	0	1	-0.4	0	-0.4	0.1
16,000,000	38400	26	0	1	0	0.9	0	1.1
16,000,000	57600	17	0	6	0	0.9	-0.1	1.0
16,000,000	115200	8	0	11	0	0.9	0	1.6
16,000,000	230400	4	5	3	-3.5	3.2	-1.8	6.4
16,000,000	460800	2	3	2	-2.1	4.8	-2.5	7.3
16,777,216	9600	109	0	4	0	0.2	-0.02	0.3
16,777,216	19200	54	0	10	0	0.2	-0.05	0.3
16,777,216	57600	18	0	3	-1.0	0	-1.0	0.3
16,777,216	115200	9	0	2	0	2.8	-0.2	3.0
20,000,000	9600	130	0	3	-0.2	0	-0.2	0.04
20,000,000	19200	65	0	2	0	0.4	-0.03	0.4
20,000,000	38400	32	0	9	0	0.4	0	0.5
20,000,000	57600	21	0	11	-0.7	0	-0.7	0.3
20,000,000	115200	10	0	14	0	2.5	-0.2	2.6
20,000,000	230400	5	0	7	0	2.5	0	3.5

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 22-5. Commonly Used Baud Rates, Settings, and Errors, UCOS16 = 1 (continued)

BRCLK Frequency (Hz)	Baud Rate (baud)	UCBRx	UCBRSx	UCBRFx	Maximum TX Error (%)		Maximum RX Error (%)	
20,000,000	460800	2	6	10	-3.2	1.8	-2.8	4.6

22.3.14 Using the USCI Module in UART Mode With Low-Power Modes

The USCI module provides automatic clock activation for use with low-power modes. When the USCI clock source is inactive because the device is in a low-power mode, the USCI module automatically activates it when needed, regardless of the control-bit settings for the clock source. The clock remains active until the USCI module returns to its idle condition. After the USCI module returns to the idle condition, control of the clock source reverts to the settings of its control bits.

22.3.15 USCI Interrupts

The USCI has only one interrupt vector that is shared for transmission and for reception. USCI_Ax and USC_Bx do not share the same interrupt vector.

22.3.15.1 USCI Transmit Interrupt Operation

The UCTXIFG interrupt flag is set by the transmitter to indicate that UCAxTXBUF is ready to accept another character. An interrupt request is generated if UCTXIE and GIE are also set. UCTXIFG is automatically reset if a character is written to UCAxTXBUF.

UCTXIFG is set after a PUC or when UCSWRST = 1. UCTXIE is reset after a PUC or when UCSWRST = 1.

22.3.15.2 USCI Receive Interrupt Operation

The UCRXIFG interrupt flag is set each time a character is received and loaded into UCAxRXBUF. An interrupt request is generated if UCRXIE and GIE are also set. UCRXIFG and UCRXIE are reset by a system reset PUC signal or when UCSWRST = 1. UCRXIFG is automatically reset when UCAxRXBUF is read.

Additional interrupt control features include:

- When UCAxRXEIE = 0, erroneous characters do not set UCRXIFG.
- When UCDORM = 1, nonaddress characters do not set UCRXIFG in multiprocessor modes. In plain UART mode, no characters are set UCRXIFG.
- When UCBRKIE = 1, a break condition sets the UCBRK bit and the UCRXIFG flag.

22.3.15.3 UCAxIV, Interrupt Vector Generator

The USCI interrupt flags are prioritized and combined to source a single interrupt vector. The interrupt vector register UCAxIV is used to determine which flag requested an interrupt. The highest-priority enabled interrupt generates a number in the UCAxIV register that can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled interrupts do not affect the UCAxIV value.

Any access, read or write, of the UCAxIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.3.15.3.1 UCAxIV Software Example

The following software example shows the recommended use of UCAxIV. The UCAxIV value is added to the PC to automatically jump to the appropriate routine. The following example is given for USCI_A0.

```
USCI_UART_ISR
         ADD    &UCA0IV, PC   ; Add offset to jump table
         RETI                 ; Vector 0: No interrupt
         JMP    RXIFG_ISR     ; Vector 2: RXIFG
TXIFG_ISR                     ; Vector 4: TXIFG
         ...                  ; Task starts here
         RETI                 ; Return
RXIFG_ISR                     ; Vector 2
         ...                  ; Task starts here
         RETI                 ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4 USCI_A UART Mode Registers

The USCI registers applicable in UART mode listed in Table 22-6. The base address can be found in the device-specific data sheet. The address offsets are listed in Table 22-6.

Table 22-6. USCI_A UART Mode Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCAxCTLW0	USCI_Ax Control Word 0	Read/write	Word	0001h	
00h	UCAxCTL1	USCI_Ax Control 1	Read/write	Byte	01h	Section 22.4.2
01h	UCAxCTL0	USCI_Ax Control 0	Read/write	Byte	00h	Section 22.4.1
06h	UCAxBRW	USCI_Ax Baud Rate Control Word	Read/write	Word	0000h	
06h	UCAxBR0	USCI_Ax Baud Rate Control 0	Read/write	Byte	00h	Section 22.4.3
07h	UCAxBR1	USCI_Ax Baud Rate Control 1	Read/write	Byte	00h	Section 22.4.4
08h	UCAxMCTL	USCI_Ax Modulation Control	Read/write	Byte	00h	Section 22.4.5
09h		Reserved - reads zero	Read	Byte	00h	
0Ah	UCAxSTAT	USCI_Ax Status	Read/write	Byte	00h	Section 22.4.6
0Bh		Reserved - reads zero	Read	Byte	00h	
0Ch	UCAxRXBUF	USCI_Ax Receive Buffer	Read/write	Byte	00h	Section 22.4.7
0Dh		Reserved - reads zero	Read	Byte	00h	
0Eh	UCAxTXBUF	USCI_Ax Transmit Buffer	Read/write	Byte	00h	Section 22.4.8
0Fh		Reserved - reads zero	Read	Byte	00h	
10h	UCAxABCTL	USCI_Ax Auto Baud Rate Control	Read/write	Byte	00h	Section 22.4.11
11h		Reserved - reads zero	Read	Byte	00h	
12h	UCAxIRCTL	USCI_Ax IrDA Control	Read/write	Word	0000h	
12h	UCAxIRTCTL	USCI_Ax IrDA Transmit Control	Read/write	Byte	00h	Section 22.4.9
13h	UCAxIRRCTL	USCI_Ax IrDA Receive Control	Read/write	Byte	00h	Section 22.4.10
1Ch	UCAxICTL	USCI_Ax Interrupt Control	Read/write	Word	0000h	
1Ch	UCAxIE	USCI_Ax Interrupt Enable	Read/write	Byte	00h	Section 22.4.12
1Dh	UCAxIFG	USCI_Ax Interrupt Flag	Read/write	Byte	00h	Section 22.4.13
1Eh	UCAxIV	USCI_Ax Interrupt Vector	Read	Word	0000h	Section 22.4.14

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.1 UCAxCTL0 Register

USCI_Ax Control Register 0

Figure 22-12. UCAxCTL0 Register

7	6	5	4	3	2	1	0
UCPEN	UCPAR	UCMSB	UC7BIT	UCSPB	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 22-7. UCAxCTL0 Register Description

Bit	Field	Type	Reset	Description
7	UCPEN	RW	0h	Parity enable 0b = Parity disabled 1b = Parity enabled. Parity bit is generated (UCAxTXD) and expected (UCAxRXD). In address-bit multiprocessor mode, the address bit is included in the parity calculation.
6	UCPAR	RW	0h	Parity select. UCPAR is not used when parity is disabled. 0b = Odd parity 1b = Even parity
5	UCMSB	RW	0h	MSB first select. Controls the direction of the receive and transmit shift register. 0b = LSB first 1b = MSB first
4	UC7BIT	RW	0h	Character length. Selects 7-bit or 8-bit character length. 0b = 8-bit data 1b = 7-bit data
3	UCSPB	RW	0h	Stop bit select. Number of stop bits. 0b = One stop bit 1b = Two stop bits
2-1	UCMODEx	RW	0h	USCI mode. The UCMODEx bits select the asynchronous mode when UCSYNC = 0. 00b = UART mode 01b = Idle-line multiprocessor mode 10b = Address-bit multiprocessor mode 11b = UART mode with automatic baud-rate detection
0	UCSYNC	RW	0h	Synchronous mode enable 0b = Asynchronous mode 1b = Synchronous mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.2 UCAxCTL1 Register

USCI_Ax Control Register 1

Figure 22-13. UCAxCTL1 Register

7	6	5	4	3	2	1	0
UCSSELx		UCRXEIE	UCBRKIE	UCDORM	UCTXADDR	UCTXBRK	UCSWRST
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1

Modify only when UCSWRST = 1.

Table 22-8. UCAxCTL1 Register Description

Bit	Field	Type	Reset	Description
7-6	UCSSELx	RW	0h	USCI clock source select. These bits select the BRCLK source clock. 00b = UCAxCLK (external USCI clock) 01b = ACLK 10b = SMCLK 11b = SMCLK
5	UCRXEIE	RW	0h	Receive erroneous-character interrupt enable 0b = Erroneous characters rejected and UCRXIFG is not set. 1b = Erroneous characters received set UCRXIFG.
4	UCBRKIE	RW	0h	Receive break character interrupt enable 0b = Received break characters do not set UCRXIFG. 1b = Received break characters set UCRXIFG.
3	UCDORM	RW	0h	Dormant. Puts USCI into sleep mode. 0b = Not dormant. All received characters set UCRXIFG. 1b = Dormant. Only characters that are preceded by an idle-line or with address bit set UCRXIFG. In UART mode with automatic baud-rate detection, only the combination of a break and synch field sets UCRXIFG.
2	UCTXADDR	RW	0h	Transmit address. Next frame to be transmitted is marked as address, depending on the selected multiprocessor mode. 0b = Next frame transmitted is data. 1b = Next frame transmitted is an address.
1	UCTXBRK	RW	0h	Transmit break. Transmits a break with the next write to the transmit buffer. In UART mode with automatic baud-rate detection, 055h must be written into UCAxTXBUF to generate the required break/synch fields. Otherwise, 0h must be written into the transmit buffer. 0b = Next frame transmitted is not a break. 1b = Next frame transmitted is a break or a break/synch.
0	UCSWRST	RW	1h	Software reset enable 0b = Disabled. USCI reset released for operation. 1b = Enabled. USCI logic held in reset state.

Copyright © 2009–2013, Texas Instruments Incorporated

22.4.3 UCAxBR0 Register

USCI_Ax Baud Rate Control Register 0

Figure 22-14. UCAxBR0 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 22-9. UCAxBR0 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Low byte of clock prescaler setting of the baud-rate generator. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value UCBRx.

22.4.4 UCAxBR1 Register

USCI_Ax Baud Rate Control Register 1

Figure 22-15. UCAxBR1 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 22-10. UCAxBR1 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	High byte of clock prescaler setting of the baud-rate generator. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value UCBRx.

22.4.5 UCAxMCTL Register

USCI_Ax Modulation Control Register

Figure 22-16. UCAxMCTL Register

7	6	5	4	3	2	1	0
UCBRFx				UCBRSx			UCOS16
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 22-11. UCAxMCTL Register Description

Bit	Field	Type	Reset	Description
7-4	UCBRFx	RW	0h	First modulation stage select. These bits determine the modulation pattern for BITCLK16 when UCOS16 = 1. Ignored with UCOS16 = 0. Table 22-2 shows the modulation pattern.
3-1	UCBRSx	RW	0h	Second modulation stage select. These bits determine the modulation pattern for BITCLK. Table 22-2 shows the modulation pattern.
0	UCOS16	RW	0h	Oversampling mode enabled 0b = Disabled 1b = Enabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.6 UCAxSTAT Register

USCI_Ax Status Register

Figure 22-17. UCAxSTAT Register

7	6	5	4	3	2	1	0
UCLISTEN	UCFE	UCOE	UCPE	UCBRK	UCRXERR	UCADDR/ UCIDLE	UCBUSY
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	r-0

Modify only when UCSWRST = 1.

Table 22-12. UCAxSTAT Register Description

Bit	Field	Type	Reset	Description
7	UCLISTEN	RW	0h	Listen enable. The UCLISTEN bit selects loopback mode. 0b = Disabled 1b = Enabled. UCAxTXD is internally fed back to the receiver.
6	UCFE	RW	0h	Framing error flag. UCFE is cleared when UCAxRXBUF is read. 0b = No error 1b = Character received with low stop bit
5	UCOE	RW	0h	Overrun error flag. This bit is set when a character is transferred into UCAxRXBUF before the previous character was read. UCOE is cleared automatically when UCxRXBUF is read, and must not be cleared by software. Otherwise, it does not function correctly. 0b = No error 1b = Overrun error occurred
4	UCPE	RW	0h	Parity error flag. When UCPEN = 0, UCPE is read as 0. UCPE is cleared when UCAxRXBUF is read. 0b = No error 1b = Character received with parity error
3	UCBRK	RW	0h	Break detect flag. UCBRK is cleared when UCAxRXBUF is read. 0b = No break condition 1b = Break condition occurred
2	UCRXERR	RW	0h	Receive error flag. This bit indicates a character was received with error(s). When UCRXERR = 1, on or more error flags, UCFE, UCPE, or UCOE is also set. UCRXERR is cleared when UCAxRXBUF is read. 0b = No receive errors detected 1b = Receive error detected
1	UCADDR/UCIDLE	RW	0h	UCADDR: Address received in address-bit multiprocessor mode. UCADDR is cleared when UCAxRXBUF is read. 0b = Received character is data. 1b = Received character is an address. UCIDLE: Idle line detected in idle-line multiprocessor mode. UCIDLE is cleared when UCAxRXBUF is read. 0b = No idle line detected 1b = Idle line detected
0	UCBUSY	R	0h	USCI busy. This bit indicates if a transmit or receive operation is in progress. 0b = USCI inactive 1b = USCI transmitting or receiving

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.7 UCAxRXBUF Register

USCI_Ax Receive Buffer Register

Figure 22-18. UCAxRXBUF Register

7	6	5	4	3	2	1	0
UCRXBUFx							
r	r	r	r	r	r	r	r

Table 22-13. UCAxRXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCRXBUFx	R	undefined	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCAxRXBUF resets the receive-error bits, the UCADDR or UCIDLE bit, and UCRXIFG. In 7-bit data mode, UCAxRXBUF is LSB justified and the MSB is always reset.

22.4.8 UCAxTXBUF Register

USCI_Ax Transmit Buffer Register

Figure 22-19. UCAxTXBUF Register

7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 22-14. UCAxTXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCTXBUFx	RW	undefined	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted on UCAxTXD. Writing to the transmit data buffer clears UCTXIFG. The MSB of UCAxTXBUF is not used for 7-bit data and is reset.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.9 UCAxIRTCTL Register

USCI_Ax IrDA Transmit Control Register

Figure 22-20. UCAxIRTCTL Register

7	6	5	4	3	2	1	0
UCIRTXPLx						UCIRTXCLK	UCIREN
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 22-15. UCAxIRTCTL Register Description

Bit	Field	Type	Reset	Description
7-2	UCIRTXPLx	RW	0h	Transmit pulse length. Pulse length t(PULSE) = (UCIRTXPLx + 1) / [2 × f(IRTXCLK)]
1	UCIRTXCLK	RW	0h	IrDA transmit pulse clock select 0b = BRCLK 1b = BITCLK16 when UCOS16 = 1. Otherwise, BRCLK.
0	UCIREN	RW	0h	IrDA encoder and decoder enable 0b = IrDA encoder and decoder disabled 1b = IrDA encoder and decoder enabled

22.4.10 UCAxIRRCTL Register

USCI_Ax IrDA Receive Control Register

Figure 22-21. UCAxIRRCTL Register

7	6	5	4	3	2	1	0
UCIRRXFLx						UCIRRXPL	UCIRRXFE
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 22-16. UCAxIRRCTL Register Description

Bit	Field	Type	Reset	Description
7-2	UCIRRXFLx	RW	0h	Receive filter length. The minimum pulse length for receive is given by: t(MIN) = (UCIRRXFLx + 4) / (2 × f(BRCLK))
1	UCIRRXPL	RW	0h	IrDA receive input UCAxRXD polarity 0b = IrDA transceiver delivers a high pulse when a light pulse is seen. 1b = IrDA transceiver delivers a low pulse when a light pulse is seen.
0	UCIRRXFE	RW	0h	IrDA receive filter enabled 0b = Receive filter disabled 1b = Receive filter enabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.11 UCAxABCTL Register

USCI_Ax Auto Baud Rate Control Register

Figure 22-22. UCAxABCTL Register

7	6	5	4	3	2	1	0
Reserved		UCDELIMx		UCSTOE	UCBTOE	Reserved	UCABDEN
r-0	r-0	rw-0	rw-0	rw-0	rw-0	r-0	rw-0

Modify only when UCSWRST = 1.

Table 22-17. UCAxABCTL Register Description

Bit	Field	Type	Reset	Description
7-6	Reserved	R	0h	Reserved. Always reads as 0.
5-4	UCDELIMx	RW	0h	Break and synch delimiter length 00b = 1 bit time 01b = 2 bit times 10b = 3 bit times 11b = 4 bit times
3	UCSTOE	RW	0h	Synch field time out error 0b = No error 1b = Length of synch field exceeded measurable time.
2	UCBTOE	RW	0h	Break time out error 0b = No error 1b = Length of break field exceeded 22 bit times.
1	Reserved	R	0h	Reserved. Always reads as 0.
0	UCABDEN	RW	0h	Automatic baud-rate detect enable 0b = Baud-rate detection disabled. Length of break and synch field is not measured. 1b = Baud-rate detection enabled. Length of break and synch field is measured and baud-rate settings are changed accordingly.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

22.4.12 UCAxIE Register

USCI_Ax Interrupt Enable Register

Figure 22-23. UCAxIE Register

7	6	5	4	3	2	1	0
Reserved						UCTXIE	UCRXIE
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0

Table 22-18. UCAxIE Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIE	RW	0h	Transmit interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE	RW	0h	Receive interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

22.4.13 UCAxIFG Register

USCI_Ax Interrupt Flag Register

Figure 22-24. UCAxIFG Register

7	6	5	4	3	2	1	0
Reserved						UCTXIFG	UCRXIFG
r-0	r-0	r-0	r-0	r-0	r-0	rw-1	rw-0

Table 22-19. UCAxIFG Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIFG	RW	1h	Transmit interrupt flag. UCTXIFG is set when UCAxTXBUF empty. 0b = No interrupt pending 1b = Interrupt pending
0	UCRXIFG	RW	0h	Receive interrupt flag. UCRXIFG is set when UCAxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

22.4.14 UCAxIV Register

USCI_Ax Interrupt Vector Register

Figure 22-25. UCAxIV Register

15	14	13	12	11	10	9	8
UCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCIVx							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 22-20. UCAxIV Register Description

Bit	Field	Type	Reset	Description
15-0	UCIVx	R	0h	USCI interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Data received; Interrupt Flag: UCRXIFG; Interrupt Priority: Highest 04h = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Universal Serial Communication Interface – SPI Mode

The universal serial communication interface (USCI) supports multiple serial communication modes with one hardware module. This chapter discusses the operation of the synchronous peripheral interface (SPI) mode.

Topic **Page**

23.1 Universal Serial Communication Interface (USCI) Overview 610
23.2 USCI Introduction – SPI Mode 611
23.3 USCI Operation – SPI Mode 613
23.4 USCI_A SPI Mode Registers 618
23.5 USCI_B SPI Mode Registers 626

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.1 Universal Serial Communication Interface (USCI) Overview

The universal serial communication interface (USCI) modules support multiple serial communication modes. Different USCI modules support different modes. Each different USCI module is named with a different letter. For example, USCI_A is different from USCI_B, etc. If more than one identical USCI module is implemented on one device, those modules are named with incrementing numbers. For example, if one device has two USCI_A modules, they are named USCI_A0 and USCI_A1. See the device-specific data sheet to determine which USCI modules, if any, are implemented on which devices.

USCI_Ax modules support:

- UART mode
- Pulse shaping for IrDA communications
- Automatic baud-rate detection for LIN communications
- SPI mode

USCI_Bx modules support:

- I^2C mode
- SPI mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.2 USCI Introduction – SPI Mode

In synchronous mode, the USCI connects the device to an external system via three or four pins: UCxSIMO, UCxSOMI, UCxCLK, and UCxSTE. SPI mode is selected when the UCSYNC bit is set, and SPI mode (3-pin or 4-pin) is selected with the UCMODEx bits.

SPI mode features include:

- 7-bit or 8-bit data length
- LSB-first or MSB-first data transmit and receive
- 3-pin and 4-pin SPI operation
- Master or slave modes
- Independent transmit and receive shift registers
- Separate transmit and receive buffer registers
- Continuous transmit and receive operation
- Selectable clock polarity and phase control
- Programmable clock frequency in master mode
- Independent interrupt capability for receive and transmit
- Slave operation in LPM4

Figure 23-1 shows the USCI when configured for SPI mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

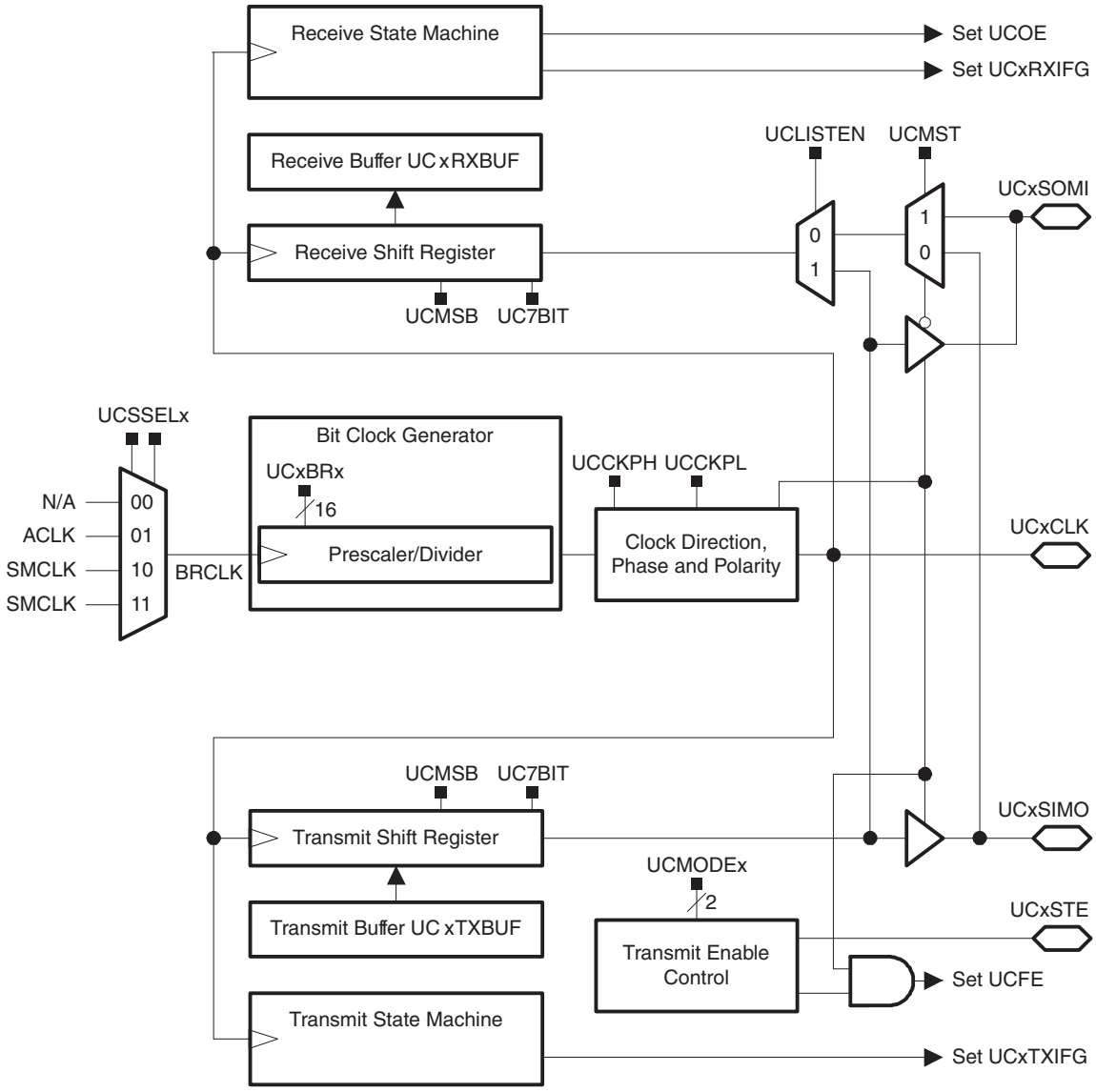


Figure 23-1. USCI Block Diagram – SPI Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.3 USCI Operation – SPI Mode

In SPI mode, serial data is transmitted and received by multiple devices using a shared clock provided by the master. An additional pin, UCxSTE, is provided to enable a device to receive and transmit data and is controlled by the master.

Three or four signals are used for SPI data exchange:

- UCxSIMO – slave in, master out
  Master mode: UCxSIMO is the data output line.
  Slave mode: UCxSIMO is the data input line.
- UCxSOMI – slave out, master in
  Master mode: UCxSOMI is the data input line.
  Slave mode: UCxSOMI is the data output line.
- UCxCLK – USCI SPI clock
  Master mode: UCxCLK is an output.
  Slave mode: UCxCLK is an input.
- UCxSTE – slave transmit enable
  Used in 4-pin mode to allow multiple masters on a single bus. Not used in 3-pin mode.
  Table 23-1 describes the UCxSTE operation.

Table 23-1. UCxSTE Operation

UCMODEx	UCxSTE Active State	UCxSTE	Slave	Master
01	High	0	Inactive	Active
		1	Active	Inactive
10	Low	0	Active	Inactive
		1	Inactive	Active

23.3.1 USCI Initialization and Reset

The USCI is reset by a PUC or by the UCSWRST bit. After a PUC, the UCSWRST bit is automatically set, keeping the USCI in a reset condition. When set, the UCSWRST bit resets the UCRXIE, UCTXIE, UCRXIFG, UCOE, and UCFE bits, and sets the UCTXIFG flag. Clearing UCSWRST releases the USCI for operation.

To avoid unpredictable behavior, configure or reconfigure the USCI module only when UCSWRST is set.

NOTE: Initializing or reconfiguring the USCI module

The recommended USCI initialization/reconfiguration process is:

1. Set UCSWRST (`BIS.B #UCSWRST,&UCxCTL1`).
2. Initialize all USCI registers with UCSWRST = 1 (including UCxCTL1).
3. Configure ports.
4. Clear UCSWRST via software (`BIC.B #UCSWRST,&UCxCTL1`).
5. Enable interrupts (optional) via UCRXIE and/or UCTXIE.

23.3.2 Character Format

The USCI module in SPI mode supports 7-bit and 8-bit character lengths selected by the UC7BIT bit. In 7-bit data mode, UCxRXBUF is LSB justified and the MSB is always reset. The UCMSB bit controls the direction of the transfer and selects LSB or MSB first.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

NOTE: Default character format

The default SPI character transmission is LSB first. For communication with other SPI interfaces, MSB-first mode may be required.

NOTE: Character format for Figures

Figures throughout this chapter use MSB-first format.

23.3.3 Master Mode

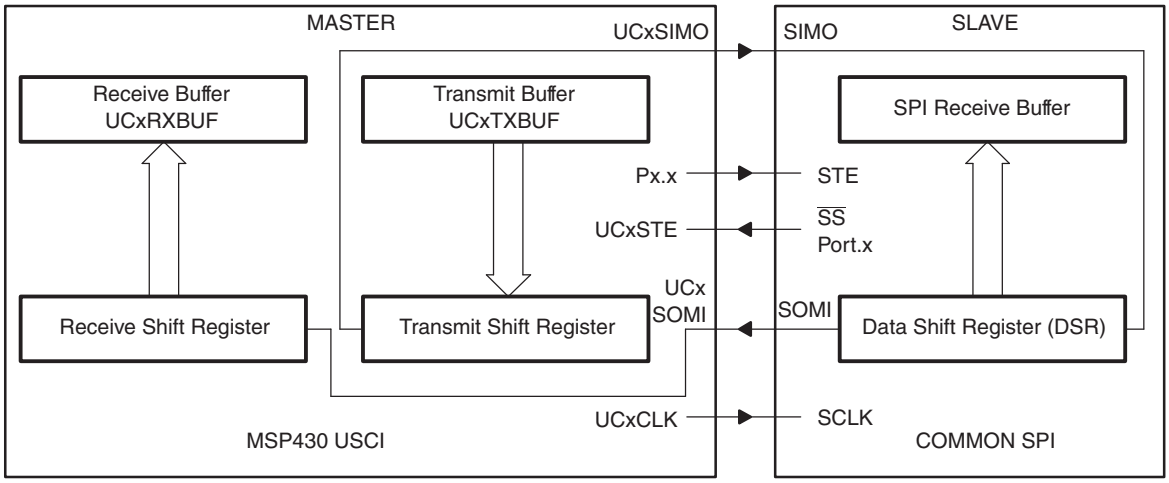


Figure 23-2. USCI Master and External Slave

Figure 23-2 shows the USCI as a master in both 3-pin and 4-pin configurations. The USCI initiates data transfer when data is moved to the transmit data buffer UCxTXBUF. The UCxTXBUF data is moved to the transmit (TX) shift register when the TX shift register is empty, initiating data transfer on UCxSIMO starting with either the MSB or LSB, depending on the UCMSB setting. Data on UCxSOMI is shifted into the receive shift register on the opposite clock edge. When the character is received, the receive data is moved from the receive (RX) shift register to the received data buffer UCxRXBUF and the receive interrupt flag UCRXIFG is set, indicating the RX/TX operation is complete.

A set transmit interrupt flag, UCTXIFG, indicates that data has moved from UCxTXBUF to the TX shift register and UCxTXBUF is ready for new data. It does not indicate RX/TX completion.

To receive data into the USCI in master mode, data must be written to UCxTXBUF, because receive and transmit operations operate concurrently.

23.3.3.1 4-Pin SPI Master Mode

In 4-pin master mode, UCxSTE is used to prevent conflicts with another master and controls the master as described in Table 23-1. When UCxSTE is in the master-inactive state:

- UCxSIMO and UCxCLK are set to inputs and no longer drive the bus.
- The error bit UCFE is set, indicating a communication integrity violation to be handled by the user.
- The internal state machines are reset and the shift operation is aborted.

If data is written into UCxTXBUF while the master is held inactive by UCxSTE, it is transmit as soon as UCxSTE transitions to the master-active state. If an active transfer is aborted by UCxSTE transitioning to the master-inactive state, the data must be rewritten into UCxTXBUF to be transferred when UCxSTE transitions back to the master-active state. The UCxSTE input signal is not used in 3-pin master mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.3.4 Slave Mode

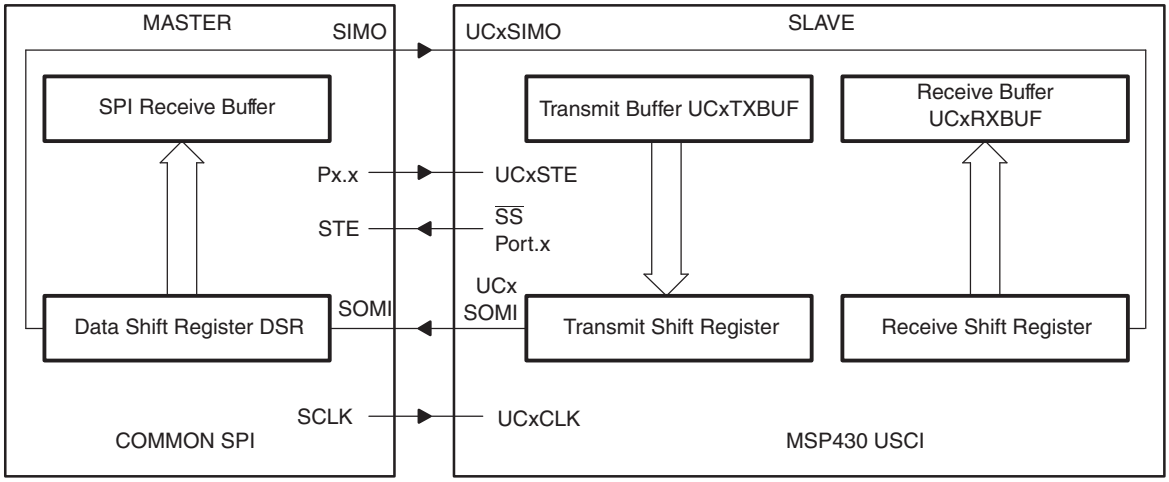


Figure 23-3. USCI Slave and External Master

Figure 23-3 shows the USCI as a slave in both 3-pin and 4-pin configurations. UCxCLK is used as the input for the SPI clock and must be supplied by the external master. The data-transfer rate is determined by this clock and not by the internal bit clock generator. Data written to UCxTXBUF and moved to the TX shift register before the start of UCxCLK is transmitted on UCxSOMI. Data on UCxSIMO is shifted into the receive shift register on the opposite edge of UCxCLK and moved to UCxRXBUF when the set number of bits are received. When data is moved from the RX shift register to UCxRXBUF, the UCRXIFG interrupt flag is set, indicating that data has been received. The overrun error bit UCOE is set when the previously received data is not read from UCxRXBUF before new data is moved to UCxRXBUF.

23.3.4.1 4-Pin SPI Slave Mode

In 4-pin slave mode, UCxSTE is used by the slave to enable the transmit and receive operations and is provided by the SPI master. When UCxSTE is in the slave-active state, the slave operates normally. When UCxSTE is in the slave- inactive state:

- Any receive operation in progress on UCxSIMO is halted.
- UCxSOMI is set to the input direction.
- The shift operation is halted until the UCxSTE line transitions into the slave transmit active state.

The UCxSTE input signal is not used in 3-pin slave mode.

23.3.5 SPI Enable

When the USCI module is enabled by clearing the UCSWRST bit, it is ready to receive and transmit. In master mode, the bit clock generator is ready, but is not clocked nor producing any clocks. In slave mode, the bit clock generator is disabled and the clock is provided by the master.

A transmit or receive operation is indicated by UCBUSY = 1.

A PUC or set UCSWRST bit disables the USCI immediately and any active transfer is terminated.

23.3.5.1 Transmit Enable

In master mode, writing to UCxTXBUF activates the bit clock generator, and the data begins to transmit.

In slave mode, transmission begins when a master provides a clock and, in 4-pin mode, when the UCxSTE is in the slave-active state.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.3.5.2 Receive Enable

The SPI receives data when a transmission is active. Receive and transmit operations operate concurrently.

23.3.6 Serial Clock Control

UCxCLK is provided by the master on the SPI bus. When UCMST = 1, the bit clock is provided by the USCI bit clock generator on the UCxCLK pin. The clock used to generate the bit clock is selected with the UCSSELx bits. When UCMST = 0, the USCI clock is provided on the UCxCLK pin by the master, the bit clock generator is not used, and the UCSSELx bits are don't care. The SPI receiver and transmitter operate in parallel and use the same clock source for data transfer.

The 16-bit value of UCBRx in the bit rate control registers (UCxxBR1 and UCxxBR0) is the division factor of the USCI clock source, BRCLK. The maximum bit clock that can be generated in master mode is BRCLK. Modulation is not used in SPI mode, and UCAxMCTL should be cleared when using SPI mode for USCI_A. The UCAxCLK/UCBxCLK frequency is given by:

$f_{BitClock} = f_{BRCLK}/UCBRx$

23.3.6.1 Serial Clock Polarity and Phase

The polarity and phase of UCxCLK are independently configured via the UCCKPL and UCCKPH control bits of the USCI. Timing for each case is shown in Figure 23-4.

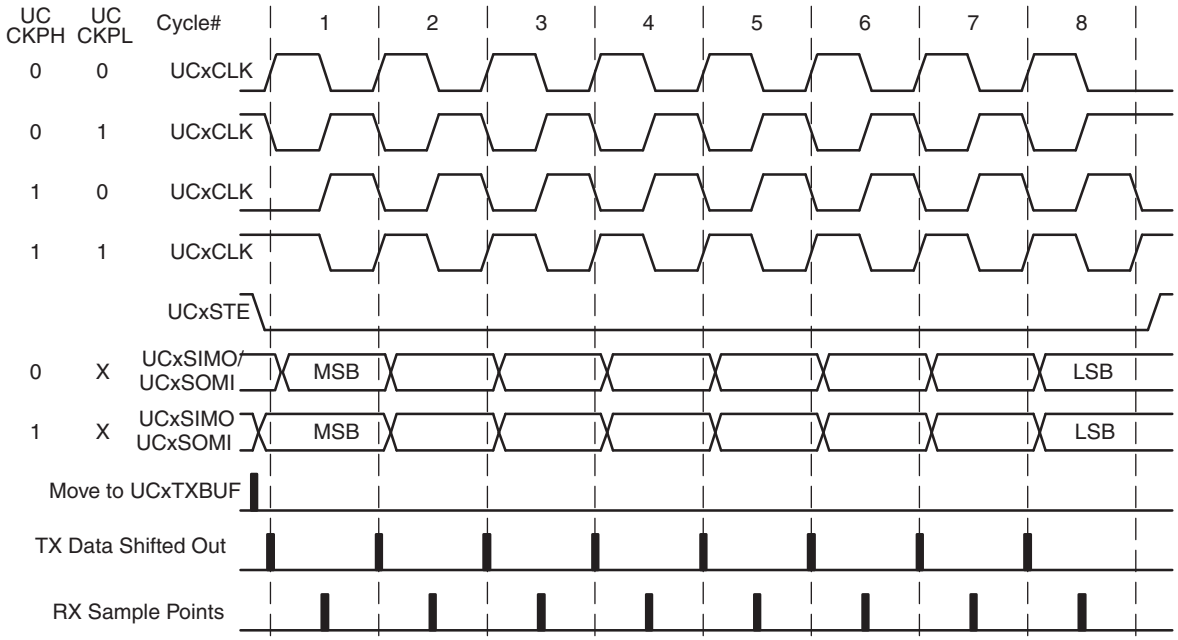


Figure 23-4. USCI SPI Timing With UCMSB = 1

23.3.7 Using the SPI Mode With Low-Power Modes

The USCI module provides automatic clock activation for use with low-power modes. When the USCI clock source is inactive because the device is in a low-power mode, the USCI module automatically activates it when needed, regardless of the control-bit settings for the clock source. The clock remains active until the USCI module returns to its idle condition. After the USCI module returns to the idle condition, control of the clock source reverts to the settings of its control bits.

In SPI slave mode, no internal clock source is required because the clock is provided by the external master. It is possible to operate the USCI in SPI slave mode while the device is in LPM4 and all clock sources are disabled. The receive or transmit interrupt can wake up the CPU from any low-power mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.3.8 SPI Interrupts

The USCI has only one interrupt vector that is shared for transmission and for reception. USCI_Ax and USC_Bx do not share the same interrupt vector.

23.3.8.1 SPI Transmit Interrupt Operation

The UCTXIFG interrupt flag is set by the transmitter to indicate that UCxTXBUF is ready to accept another character. An interrupt request is generated if UCTXIE and GIE are also set. UCTXIFG is automatically reset if a character is written to UCxTXBUF. UCTXIFG is set after a PUC or when UCSWRST = 1. UCTXIE is reset after a PUC or when UCSWRST = 1.

NOTE: Writing to UCxTXBUF in SPI mode

Data written to UCxTXBUF when UCTXIFG = 0 may result in erroneous data transmission.

23.3.8.2 SPI Receive Interrupt Operation

The UCRXIFG interrupt flag is set each time a character is received and loaded into UCxRXBUF. An interrupt request is generated if UCRXIE and GIE are also set. UCRXIFG and UCRXIE are reset by a system reset PUC signal or when UCSWRST = 1. UCRXIFG is automatically reset when UCxRXBUF is read.

23.3.8.3 UCxIV, Interrupt Vector Generator

The USCI interrupt flags are prioritized and combined to source a single interrupt vector. The interrupt vector register UCxIV is used to determine which flag requested an interrupt. The highest-priority enabled interrupt generates a number in the UCxIV register that can be evaluated or added to the program counter (PC) to automatically enter the appropriate software routine. Disabled interrupts do not affect the UCxIV value.

Any access, read or write, of the UCxIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt.

23.3.8.3.1 UCxIV Software Example

The following software example shows the recommended use of UCxIV. The UCxIV value is added to the PC to automatically jump to the appropriate routine. The following example is given for USCI_B0.

```
USCI_SPI_ISR
         ADD      &UCB0IV, PC  ; Add offset to jump table
         RETI                  ; Vector 0: No interrupt
         JMP      RXIFG_ISR    ; Vector 2: RXIFG
TXIFG_ISR                      ; Vector 4: TXIFG
         ...                   ; Task starts here
         RETI                  ; Return
RXIFG_ISR                      ; Vector 2
         ...                   ; Task starts here
         RETI                  ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4 USCI_A SPI Mode Registers

The USCI_A registers that are applicable in SPI mode are listed in Table 23-2. The base addresses can be found in the device-specific data sheet. The address offsets are listed in Table 23-2.

Table 23-2. USCI_A SPI Mode Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCAxCTLW0	USCI_Ax Control Word 0	Read/write	Word	0001h	
00h	UCAxCTL1	USCI_Ax Control 1	Read/write	Byte	01h	Section 23.4.2
01h	UCAxCTL0	USCI_Ax Control 0	Read/write	Byte	00h	Section 23.4.1
06h	UCAxBRW	USCI_Ax Bit Rate Control Word	Read/write	Word	0000h	
06h	UCAxBR0	USCI_Ax Bit Rate Control 0	Read/write	Byte	00h	Section 23.4.3
07h	UCAxBR1	USCI_Ax Bit Rate Control 1	Read/write	Byte	00h	Section 23.4.4
08h	UCAxMCTL	USCI_Ax Modulation Control	Read/write	Byte	00h	Section 23.4.5
0Ah	UCAxSTAT	USCI_Ax Status	Read/write	Byte	00h	Section 23.4.6
0Bh		Reserved - reads zero	Read	Byte	00h	
0Ch	UCAxRXBUF	USCI_Ax Receive Buffer	Read/write	Byte	00h	Section 23.4.7
0Dh		Reserved - reads zero	Read	Byte	00h	
0Eh	UCAxTXBUF	USCI_Ax Transmit Buffer	Read/write	Byte	00h	Section 23.4.8
0Fh		Reserved - reads zero	Read	Byte	00h	
1Ch	UCAxICTL	USCI_Ax Interrupt Control	Read/write	Word	0200h	
1Ch	UCAxIE	USCI_Ax Interrupt Enable	Read/write	Byte	00h	Section 23.4.9
1Dh	UCAxIFG	USCI_Ax Interrupt Flag	Read/write	Byte	02h	Section 23.4.10
1Eh	UCAxIV	USCI_Ax Interrupt Vector	Read	Word	0000h	Section 23.4.11

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4.1 UCAxCTL0 Register

USCI_Ax Control Register 0

Figure 23-5. UCAxCTL0 Register

7	6	5	4	3	2	1	0
UCCKPH	UCCKPL	UCMSB	UC7BIT	UCMST	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 23-3. UCAxCTL0 Register Description

Bit	Field	Type	Reset	Description
7	UCCKPH	RW	0h	Clock phase select 0b = Data is changed on the first UCLK edge and captured on the following edge. 1b = Data is captured on the first UCLK edge and changed on the following edge.
6	UCCKPL	RW	0h	Clock polarity select 0b = The inactive state is low. 1b = The inactive state is high.
5	UCMSB	RW	0h	MSB first select. Controls the direction of the receive and transmit shift register. 0b = LSB first 1b = MSB first
4	UC7BIT	RW	0h	Character length. Selects 7-bit or 8-bit character length. 0b = 8-bit data 1b = 7-bit data
3	UCMST	RW	0h	Master mode select 0b = Slave mode 1b = Master mode
2-1	UCMODEx	RW	0h	USCI mode. The UCMODEx bits select the synchronous mode when UCSYNC = 1. 00b = 3-pin SPI 01b = 4-pin SPI with UCxSTE active high: Slave enabled when UCxSTE = 1 10b = 4-pin SPI with UCxSTE active low: Slave enabled when UCxSTE = 0 11b = I2C mode
0	UCSYNC	RW	0h	Synchronous mode enable 0b = Asynchronous mode 1b = Synchronous mode

Copyright © 2009–2013, Texas Instruments Incorporated

23.4.2 UCAxCTL1 Register

USCI_Ax Control Register 1

Figure 23-6. UCAxCTL1 Register

7	6	5	4	3	2	1	0
UCSSELx		Reserved					UCSWRST
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1

Modify only when UCSWRST = 1.

Table 23-4. UCAxCTL1 Register Description

Bit	Field	Type	Reset	Description
7-6	UCSSELx	RW	0h	USCI clock source select. These bits select the BRCLK source clock in master mode. UCxCLK is always used in slave mode. 00b = Reserved 01b = ACLK 10b = SMCLK 11b = SMCLK
5-1	Reserved	RW	0h	Reserved. Always write as 0.
0	UCSWRST	RW	1h	Software reset enable 0b = Disabled. USCI reset released for operation. 1b = Enabled. USCI logic held in reset state.

Copyright © 2009–2013, Texas Instruments Incorporated

23.4.3 UCAxBR0 Register

USCI_Ax Bit Rate Control Register 0

Figure 23-7. UCAxBR0 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 23-5. UCAxBR0 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler low byte. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value UCBRx.

23.4.4 UCAxBR1 Register

USCI_Ax Bit Rate Control Register 1

Figure 23-8. UCAxBR1 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 23-6. UCAxBR1 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler high byte. The 16-bit value of (UCAxBR0 + UCAxBR1 × 256) forms the prescaler value UCBRx.

23.4.5 UCAxMCTL Register

USCI_Ax Modulation Control Register

Figure 23-9. UCAxMCTL Register

7	6	5	4	3	2	1	0
Reserved							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 23-7. UCAxMCTL Register Description

Bit	Field	Type	Reset	Description
7-0	Reserved	R	0h	Reserved. Always write as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4.6 UCAxSTAT Register

USCI_Ax Status Register

Figure 23-10. UCAxSTAT Register

7	6	5	4	3	2	1	0
UCLISTEN	UCFE	UCOE	Reserved				UCBUSY
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	r-0

Modify only when UCSWRST = 1.

Table 23-8. UCAxSTAT Register Description

Bit	Field	Type	Reset	Description
7	UCLISTEN	RW	0h	Listen enable. The UCLISTEN bit selects loopback mode. 0b = Disabled 1b = Enabled. The transmitter output is internally fed back to the receiver.
6	UCFE	RW	0h	Framing error flag. This bit indicates a bus conflict in 4-wire master mode. UCFE is not used in 3-wire master or any slave mode. 0b = No error 1b = Bus conflict occurred.
5	UCOE	RW	0h	Overrun error flag. This bit is set when a character is transferred into UCxRXBUF before the previous character was read. UCOE is cleared automatically when UCxRXBUF is read, and must not be cleared by software. Otherwise, it does not function correctly. 0b = No error 1b = Overrun error occurred
4-1	Reserved	R	0h	Reserved. Always reads as 0.
0	UCBUSY	R	0h	USCI busy. This bit indicates if a transmit or receive operation is in progress. 0b = USCI inactive 1b = USCI transmitting or receiving

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4.7 UCAxRXBUF Register

USCI_Ax Receive Buffer Register

Figure 23-11. UCAxRXBUF Register

7	6	5	4	3	2	1	0
UCRXBUFx							
r	r	r	r	r	r	r	r

Table 23-9. UCAxRXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCRXBUFx	R	undefined	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCRXBUF resets the receive-error bits and UCRXIFG. In 7-bit data mode, UCRXBUF is LSB justified and the MSB is always reset.

23.4.8 UCAxTXBUF Register

USCI_Ax Transmit Buffer Register

Figure 23-12. UCAxTXBUF Register

7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 23-10. UCAxTXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCTXBUFx	RW	undefined	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted. Writing to the transmit data buffer clears UCTXIFG. The MSB of UCAxTXBUF is not used for 7-bit data and is reset.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4.9 UCAxIE Register

USCI_Ax Interrupt Enable Register

Figure 23-13. UCAxIE Register

7	6	5	4	3	2	1	0
Reserved						UCTXIE	UCRXIE
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0

Table 23-11. UCAxIE Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIE	RW	0h	Transmit interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE	RW	0h	Receive interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

23.4.10 UCAxIFG Register

USCI_Ax Interrupt Flag Register

Figure 23-14. UCAxIFG Register

7	6	5	4	3	2	1	0
Reserved						UCTXIFG	UCRXIFG
r-0	r-0	r-0	r-0	r-0	r-0	rw-1	rw-0

Table 23-12. UCAxIFG Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIFG	RW	1h	Transmit interrupt flag. UCTXIFG is set when UCAxTXBUF empty. 0b = No interrupt pending 1b = Interrupt pending
0	UCRXIFG	RW	0h	Receive interrupt flag. UCRXIFG is set when UCAxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.4.11 UCAxIV Register

USCI_Ax Interrupt Vector Register

Figure 23-15. UCAxIV Register

15	14	13	12	11	10	9	8
UCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCIVx							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 23-13. UCAxIV Register Description

Bit	Field	Type	Reset	Description
15-0	UCIVx	R	0h	USCI interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Data received; Interrupt Flag: UCRXIFG; Interrupt Priority: Highest 04h = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5 USCI_B SPI Mode Registers

The USCI_B registers applicable in SPI mode are listed in Table 23-14. The base addresses can be found in the device-specific data sheet. The address offsets are listed in Table 23-14.

Table 23-14. USCI_B SPI Mode Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCBxCTLW0	USCI_Bx Control Word 0	Read/write	Word	0101h	
00h	UCBxCTL1	USCI_Bx Control 1	Read/write	Byte	01h	Section 23.5.2
01h	UCBxCTL0	USCI_Bx Control 0	Read/write	Byte	01h	Section 23.5.1
06h	UCBxBRW	USCI_Bx Bit Rate Control Word	Read/write	Word	0000h	
06h	UCBxBR0	USCI_Bx Bit Rate Control 0	Read/write	Byte	00h	Section 23.5.3
07h	UCBxBR1	USCI_Bx Bit Rate Control 1	Read/write	Byte	00h	Section 23.5.4
08h	UCBxMCTL	USCI_Bx Modulation Control	Read/write	Byte	00h	Section 23.5.5
0Ah	UCBxSTAT	USCI_Bx Status	Read/write	Byte	00h	Section 23.5.6
0Bh		Reserved - reads zero	Read	Byte	00h	
0Ch	UCBxRXBUF	USCI_Bx Receive Buffer	Read/write	Byte	00h	Section 23.5.7
0Dh		Reserved - reads zero	Read	Byte	00h	
0Eh	UCBxTXBUF	USCI_Bx Transmit Buffer	Read/write	Byte	00h	Section 23.5.8
0Fh		Reserved - reads zero	Read	Byte	00h	
1Ch	UCBxICTL	USCI_Bx Interrupt Control	Read/write	Word	0200h	
1Ch	UCBxIE	USCI_Bx Interrupt Enable	Read/write	Byte	00h	Section 23.5.9
1Dh	UCBxIFG	USCI_Bx Interrupt Flag	Read/write	Byte	02h	Section 23.5.10
1Eh	UCBxIV	USCI_Bx Interrupt Vector	Read	Word	0000h	Section 23.5.11

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5.1 UCBxCTL0 Register

USCI_Bx Control Register 0

Figure 23-16. UCBxCTL0 Register

7	6	5	4	3	2	1	0
UCCKPH	UCCKPL	UCMSB	UC7BIT	UCMST	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-1

Modify only when UCSWRST = 1.

Table 23-15. UCBxCTL0 Register Description

Bit	Field	Type	Reset	Description
7	UCCKPH	RW	0h	Clock phase select 0b = Data is changed on the first UCLK edge and captured on the following edge. 1b = Data is captured on the first UCLK edge and changed on the following edge.
6	UCCKPL	RW	0h	Clock polarity select 0b = The inactive state is low. 1b = The inactive state is high.
5	UCMSB	RW	0h	MSB first select. Controls the direction of the receive and transmit shift register. 0b = LSB first 1b = MSB first
4	UC7BIT	RW	0h	Character length. Selects 7-bit or 8-bit character length. 0b = 8-bit data 1b = 7-bit data
3	UCMST	RW	0h	Master mode select 0b = Slave mode 1b = Master mode
2-1	UCMODEx	RW	0h	USCI mode. The UCMODEx bits select the synchronous mode when UCSYNC = 1. 00b = 3-pin SPI 01b = 4-pin SPI with UCxSTE active high: Slave enabled when UCxSTE = 1 10b = 4-pin SPI with UCxSTE active low: Slave enabled when UCxSTE = 0 11b = I2C mode
0	UCSYNC	RW	1h	Synchronous mode enable 0b = Asynchronous mode 1b = Synchronous mode

Copyright © 2009–2013, Texas Instruments Incorporated

23.5.2 UCBxCTL1 Register

USCI_Bx Control Register 1

Figure 23-17. UCBxCTL1 Register

7	6	5	4	3	2	1	0
UCSSELx		Reserved					UCSWRST
rw-0	rw-0	r0	rw-0	rw-0	rw-0	rw-0	rw-1

Modify only when UCSWRST = 1.

Table 23-16. UCBxCTL1 Register Description

Bit	Field	Type	Reset	Description
7-6	UCSSELx	RW	0h	USCI clock source select. These bits select the BRCLK source clock in master mode. UCxCLK is always used in slave mode. 00b = Reserved 01b = ACLK 10b = SMCLK 11b = SMCLK
5-1	Reserved	RW	0h	Reserved. Always write as 0.
0	UCSWRST	RW	1h	Software reset enable 0b = Disabled. USCI reset released for operation. 1b = Enabled. USCI logic held in reset state.

Copyright © 2009–2013, Texas Instruments Incorporated

23.5.3 UCBxBR0 Register

USCI_Bx Bit Rate Control Register 0

Figure 23-18. UCBxBR0 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 23-17. UCBxBR0 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler low byte. The 16-bit value of (UCBxBR0 + UCBxBR1 × 256) forms the prescaler value UCBRx.

23.5.4 UCBxBR1 Register

USCI_Bx Bit Rate Control Register 1

Figure 23-19. UCBxBR1 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 23-18. UCBxBR1 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler high byte. The 16-bit value of (UCBxBR0 + UCBxBR1 × 256) forms the prescaler value UCBRx.

23.5.5 UCBxMCTL Register

USCI_Bx Modulation Control Register

Figure 23-20. UCBxMCTL Register

7	6	5	4	3	2	1	0
Reserved							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 23-19. UCBxMCTL Register Description

Bit	Field	Type	Reset	Description
7-0	Reserved	R	0h	Reserved. Always write as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5.6 UCBxSTAT Register

USCI_Bx Status Register

Figure 23-21. UCBxSTAT Register

7	6	5	4	3	2	1	0
UCLISTEN	UCFE	UCOE	Reserved				UCBUSY
rw-0	rw-0	rw-0	r0	r0	r0	r0	r-0

Modify only when UCSWRST = 1.

Table 23-20. UCBxSTAT Register Description

Bit	Field	Type	Reset	Description
7	UCLISTEN	RW	0h	Listen enable. The UCLISTEN bit selects loopback mode. 0b = Disabled 1b = Enabled. The transmitter output is internally fed back to the receiver.
6	UCFE	RW	0h	Framing error flag. This bit indicates a bus conflict in 4-wire master mode. UCFE is not used in 3-wire master or any slave mode. 0b = No error 1b = Bus conflict occurred.
5	UCOE	RW	0h	Overrun error flag. This bit is set when a character is transferred into UCxRXBUF before the previous character was read. UCOE is cleared automatically when UCxRXBUF is read, and must not be cleared by software. Otherwise, it does not function correctly. 0b = No error 1b = Overrun error occurred
4-1	Reserved	R	0h	Reserved. Always reads as 0.
0	UCBUSY	R	0h	USCI busy. This bit indicates if a transmit or receive operation is in progress. 0b = USCI inactive 1b = USCI transmitting or receiving

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5.7 UCBxRXBUF Register

USCI_Bx Receive Buffer Register

Figure 23-22. UCBxRXBUF Register

7	6	5	4	3	2	1	0
UCRXBUFx							
r	r	r	r	r	r	r	r

Table 23-21. UCBxRXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCRXBUFx	R	undefined	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCRXBUF resets the receive-error bits and UCRXIFG. In 7-bit data mode, UCRXBUF is LSB justified and the MSB is always reset.

23.5.8 UCBxTXBUF Register

USCI_Bx Transmit Buffer Register

Figure 23-23. UCBxTXBUF Register

7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 23-22. UCBxTXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCTXBUFx	RW	undefined	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted. Writing to the transmit data buffer clears UCTXIFG. The MSB of UCBxTXBUF is not used for 7-bit data and is reset.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5.9 UCBxIE Register

USCI_Bx Interrupt Enable Register

Figure 23-24. UCBxIE Register

7	6	5	4	3	2	1	0
Reserved						UCTXIE	UCRXIE
r-0	r-0	r-0	r-0	r-0	r-0	rw-0	rw-0

Table 23-23. UCBxIE Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIE	RW	0h	Transmit interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE	RW	0h	Receive interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

23.5.10 UCBxIFG Register

USCI_Bx Interrupt Flag Register

Figure 23-25. UCBxIFG Register

7	6	5	4	3	2	1	0
Reserved						UCTXIFG	UCRXIFG
r-0	r-0	r-0	r-0	r-0	r-0	rw-1	rw-0

Table 23-24. UCBxIFG Register Description

Bit	Field	Type	Reset	Description
7-2	Reserved	R	0h	Reserved. Always reads as 0.
1	UCTXIFG	RW	1h	Transmit interrupt flag. UCTXIFG is set when UCBxTXBUF empty. 0b = No interrupt pending 1b = Interrupt pending
0	UCRXIFG	RW	0h	Receive interrupt flag. UCRXIFG is set when UCBxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

23.5.11 UCBxIV Register

USCI_Bx Interrupt Vector Register

Figure 23-26. UCBxIV Register

15	14	13	12	11	10	9	8
UCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCIVx							
r0	r0	r0	r-0	r-0	r-0	r-0	r0

Table 23-25. UCBxIV Register Description

Bit	Field	Type	Reset	Description
15-0	UCIVx	R	0h	USCI interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Data received; Interrupt Flag: UCRXIFG; Interrupt Priority: Highest 04h = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Universal Serial Communication Interface – I²C Mode

The universal serial communication interface (USCI) supports multiple serial communication modes with one hardware module. This chapter discusses the operation of the I²C mode.

Topic **Page**

24.1 Universal Serial Communication Interface (USCI) Overview 635
24.2 USCI Introduction – I²C Mode 636
24.3 USCI Operation – I²C Mode 637
24.4 USCI_B I2C Mode Registers 656

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.1 Universal Serial Communication Interface (USCI) Overview

The USCI modules support multiple serial communication modes. Different USCI modules support different modes. Each different USCI module is named with a different letter. For example, USCI_A is different from USCI_B, etc. If more than one identical USCI module is implemented on one device, those modules are named with incrementing numbers. For example, if one device has two USCI_A modules, they are named USCI_A0 and USCI_A1. See the device-specific data sheet to determine which USCI modules, if any, are implemented on each device.

USCI_Ax modules support:

- UART mode
- Pulse shaping for IrDA communications
- Automatic baud-rate detection for LIN communications
- SPI mode

USCI_Bx modules support:

- I^2C mode
- SPI mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.2 USCI Introduction – I²C Mode

In I²C mode, the USCI module provides an interface between the device and I²C-compatible devices connected by the two-wire I²C serial bus. External components attached to the I²C bus serially transmit and/or receive serial data to/from the USCI module through the 2-wire I²C interface.

The I²C mode features include:

- Compliance to the Philips Semiconductor I²C specification v2.1
- 7-bit and 10-bit device addressing modes
- General call
- START/RESTART/STOP
- Multi-master transmitter/receiver mode
- Slave receiver/transmitter mode
- Standard mode up to 100 kbps and fast mode up to 400 kbps support
- Programmable UCxCLK frequency in master mode
- Designed for low power
- Slave receiver START detection for auto wake up from LPMx modes (wake up from LPMx.5 is not supported)
- Slave operation in LPM4

Figure 24-1 shows the USCI when configured in I²C mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

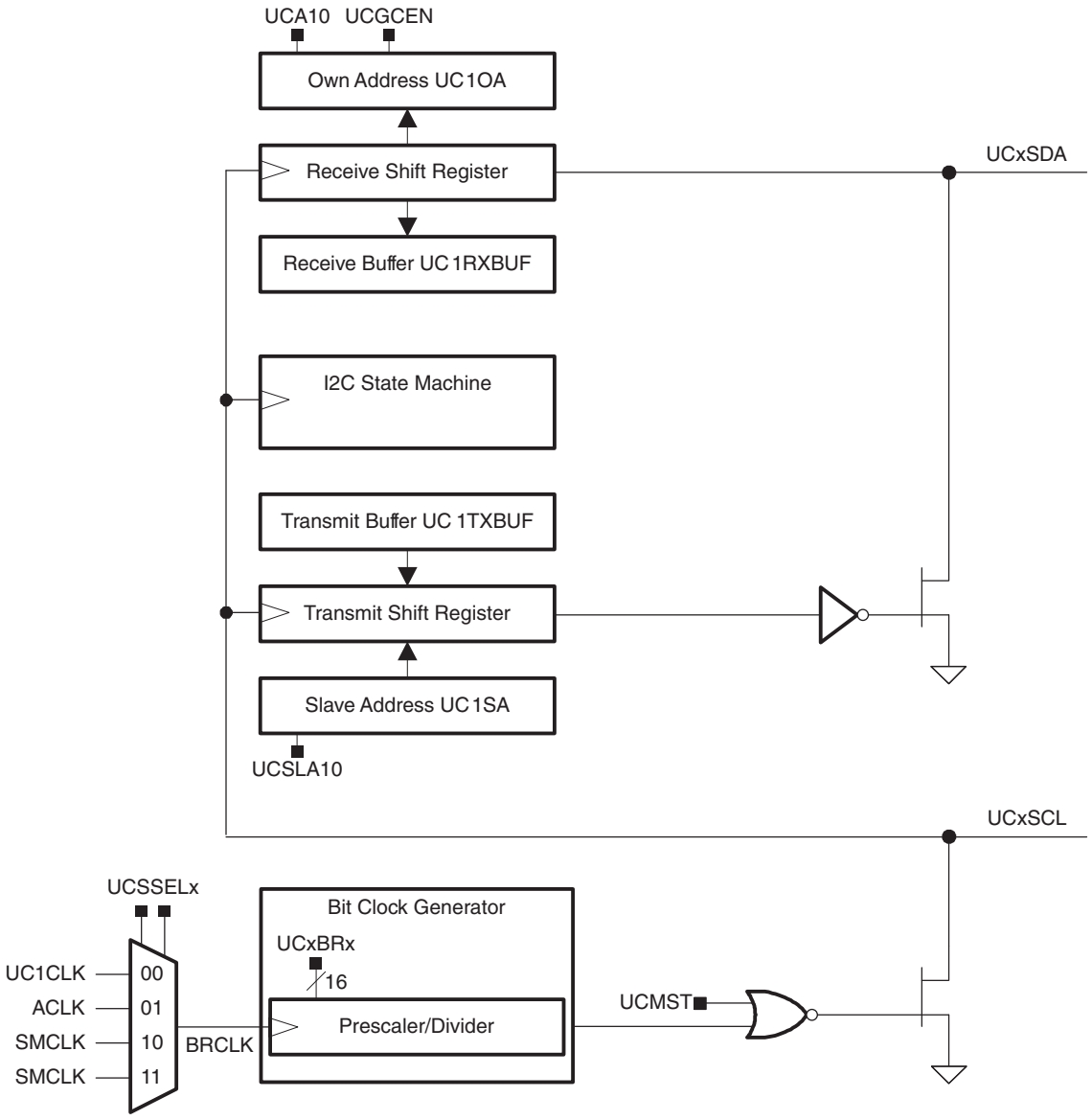


Figure 24-1. USCI Block Diagram – I²C Mode

24.3 USCI Operation – I²C Mode

The I²C mode supports any slave or master I²C-compatible device. Figure 24-2 shows an example of an I²C bus. Each I²C device is recognized by a unique address and can operate as either a transmitter or a receiver. A device connected to the I²C bus can be considered as the master or the slave when performing data transfers. A master initiates a data transfer and generates the clock signal SCL. Any device addressed by a master is considered a slave.

I²C data is communicated using the serial data (SDA) pin and the serial clock (SCL) pin. Both SDA and SCL are bidirectional and must be connected to a positive supply voltage using a pullup resistor.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

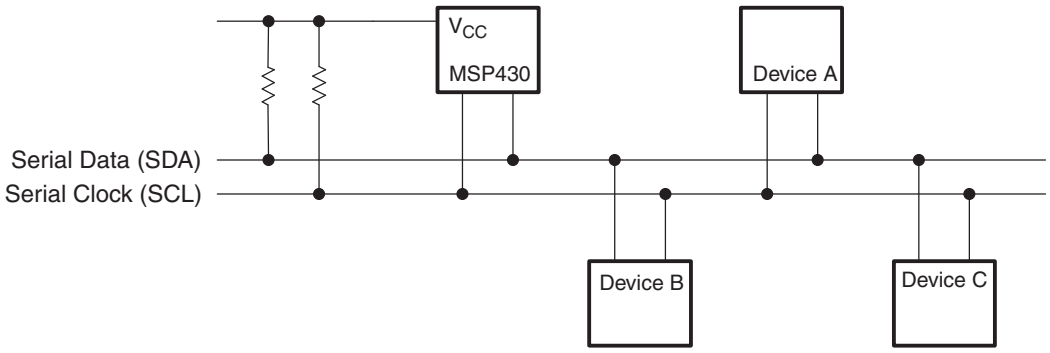


Figure 24-2. I²C Bus Connection Diagram

NOTE: SDA and SCL levels

The SDA and SCL pins must not be pulled up above the device V_{CC} level.

24.3.1 USCI Initialization and Reset

The USCI is reset by a PUC or by setting the UCSWRST bit. After a PUC, the UCSWRST bit is automatically set, keeping the USCI in a reset condition. To select I²C operation, the UCMODEx bits must be set to 11. After module initialization, it is ready for transmit or receive operation. Clearing UCSWRST releases the USCI for operation.

To avoid unpredictable behavior, configure or reconfigure the USCI module only when UCSWRST is set.

Setting UCSWRST in I²C mode has the following effects:

- I²C communication stops.
- SDA and SCL are high impedance.
- UCBxI2CSTAT, bits 6–0 are cleared.
- Registers UCBxIE and UCBxIFG are cleared.
- All other bits and register remain unchanged.

NOTE: Initializing or re-configuring the USCI module

The recommended USCI initialization/reconfiguration process is:

1. Set UCSWRST (`BIS.B #UCSWRST,&UCxCTL1`).
2. Initialize all USCI registers with UCSWRST = 1.
3. Configure ports.
4. Clear UCSWRST via software (`BIC.B #UCSWRST,&UCxCTL1`).
5. Enable interrupts (optional).

24.3.2 I²C Serial Data

One clock pulse is generated by the master device for each data bit transferred. The I²C mode operates with byte data. Data is transferred MSB first as shown in Figure 24-3.

The first byte after a START condition consists of a 7-bit slave address and the R/$\overline{W}$ bit. When R/$\overline{W}$ = 0, the master transmits data to a slave. When R/$\overline{W}$ = 1, the master receives data from a slave. The ACK bit is sent from the receiver after each byte on the ninth SCL clock.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

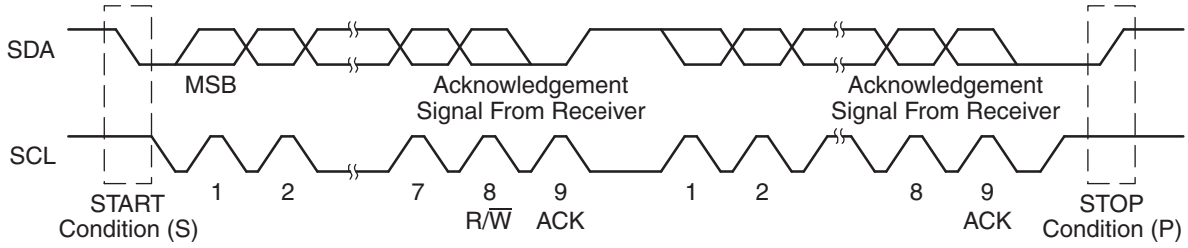


Figure 24-3. I²C Module Data Transfer

START and STOP conditions are generated by the master and are shown in Figure 24-3. A START condition is a high-to-low transition on the SDA line while SCL is high. A STOP condition is a low-to-high transition on the SDA line while SCL is high. The bus busy bit, UCBBUSY, is set after a START and cleared after a STOP.

Data on SDA must be stable during the high period of SCL (see Figure 24-4). The high and low state of SDA can only change when SCL is low, otherwise START or STOP conditions are generated.

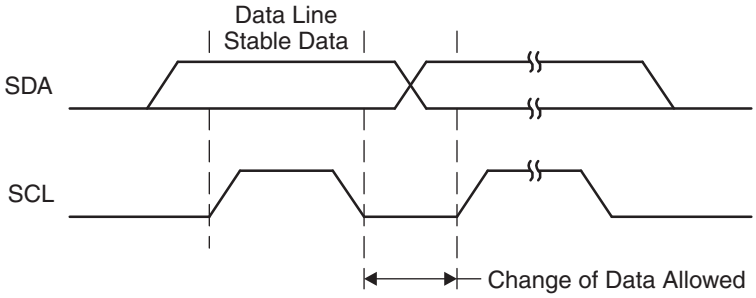


Figure 24-4. Bit Transfer on I²C Bus

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.3 I²C Addressing Modes

The I²C mode supports 7-bit and 10-bit addressing modes.

24.3.3.1 7-Bit Addressing

In the 7-bit addressing format (see Figure 24-5), the first byte is the 7-bit slave address and the R/$\overline{W}$ bit. The ACK bit is sent from the receiver after each byte.

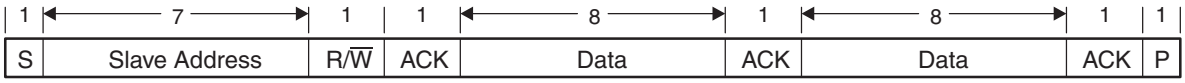


Figure 24-5. I²C Module 7-Bit Addressing Format

24.3.3.2 10-Bit Addressing

In the 10-bit addressing format (see Figure 24-6), the first byte is made up of 11110b plus the two MSBs of the 10-bit slave address and the R/$\overline{W}$ bit. The ACK bit is sent from the receiver after each byte. The next byte is the remaining eight bits of the 10-bit slave address, followed by the ACK bit and the 8-bit data. See I2C Slave 10-bit Addressing Mode and I2C Master 10-bit Addressing Mode for details how to use the 10-bit addressing mode with the USCI module.

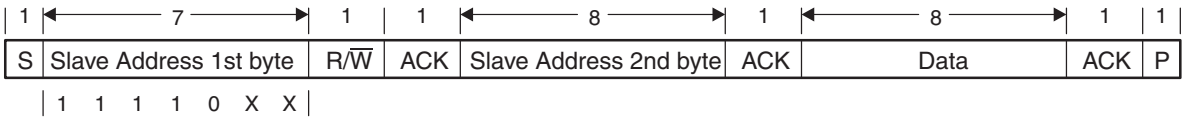


Figure 24-6. I²C Module 10-Bit Addressing Format

24.3.3.3 Repeated Start Conditions

The direction of data flow on SDA can be changed by the master, without first stopping a transfer, by issuing a repeated START condition. This is called a RESTART. After a RESTART is issued, the slave address is again sent out with the new data direction specified by the R/$\overline{W}$ bit. The RESTART condition is shown in Figure 24-7.

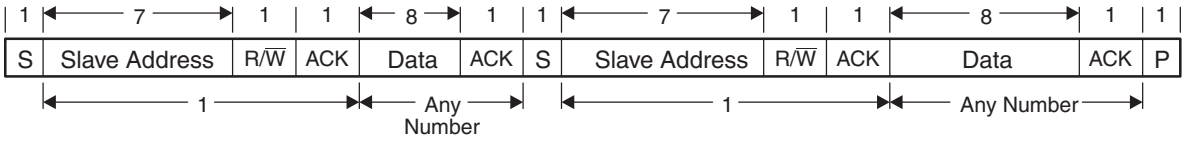


Figure 24-7. I²C Module Addressing Format With Repeated START Condition

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4 I²C Module Operating Modes

In I²C mode, the USCI module can operate in master transmitter, master receiver, slave transmitter, or slave receiver mode. The modes are discussed in the following sections. Time lines are used to illustrate the modes.

Figure 24-8 shows how to interpret the time-line figures. Data transmitted by the master is represented by grey rectangles; data transmitted by the slave is represented by white rectangles. Data transmitted by the USCI module, either as master or slave, is shown by rectangles that are taller than the others.

Actions taken by the USCI module are shown in grey rectangles with an arrow indicating where in the data stream the action occurs. Actions that must be handled with software are indicated with white rectangles with an arrow pointing to where in the data stream the action must take place.

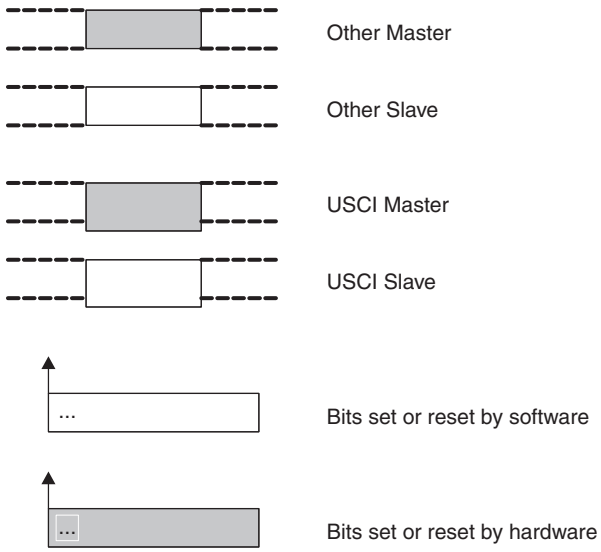


Figure 24-8. I²C Time-Line Legend

24.3.4.1 Slave Mode

The USCI module is configured as an I²C slave by selecting the I²C mode with UCMODEx = 11 and UCSYNC = 1 and clearing the UCMST bit.

Initially, the USCI module must to be configured in receiver mode by clearing the UCTR bit to receive the I²C address. Afterwards, transmit and receive operations are controlled automatically, depending on the R/$\overline{W}$ bit received together with the slave address.

The USCI slave address is programmed with the UCBxI2COA register. When UCA10 = 0, 7-bit addressing is selected. When UCA10 = 1, 10-bit addressing is selected. The UCGCEN bit selects if the slave responds to a general call.

When a START condition is detected on the bus, the USCI module receives the transmitted address and compare it against its own address stored in UCBxI2COA. The UCSTTIFG flag is set when address received matches the USCI slave address.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.1.1 I²C Slave Transmitter Mode

Slave transmitter mode is entered when the slave address transmitted by the master is identical to its own address with a set R/$\overline{W}$ bit. The slave transmitter shifts the serial data out on SDA with the clock pulses that are generated by the master device. The slave device does not generate the clock, but it does hold SCL low while intervention of the CPU is required after a byte has been transmitted.

If the master requests data from the slave, the USCI module is automatically configured as a transmitter and UCTR and UCTXIFG become set. The SCL line is held low until the first data to be sent is written into the transmit buffer UCBxTXBUF. Then the address is acknowledged, the UCSTTIFG flag is cleared, and the data is transmitted. As soon as the data is transferred into the shift register, the UCTXIFG is set again. After the data is acknowledged by the master, the next data byte written into UCBxTXBUF is transmitted or, if the buffer is empty, the bus is stalled during the acknowledge cycle by holding SCL low until new data is written into UCBxTXBUF. If the master sends a NACK succeeded by a STOP condition, the UCSTPIFG flag is set. If the NACK is succeeded by a repeated START condition, the USCI I²C state machine returns to its address-reception state.

Figure 24-9 shows the slave transmitter operation.

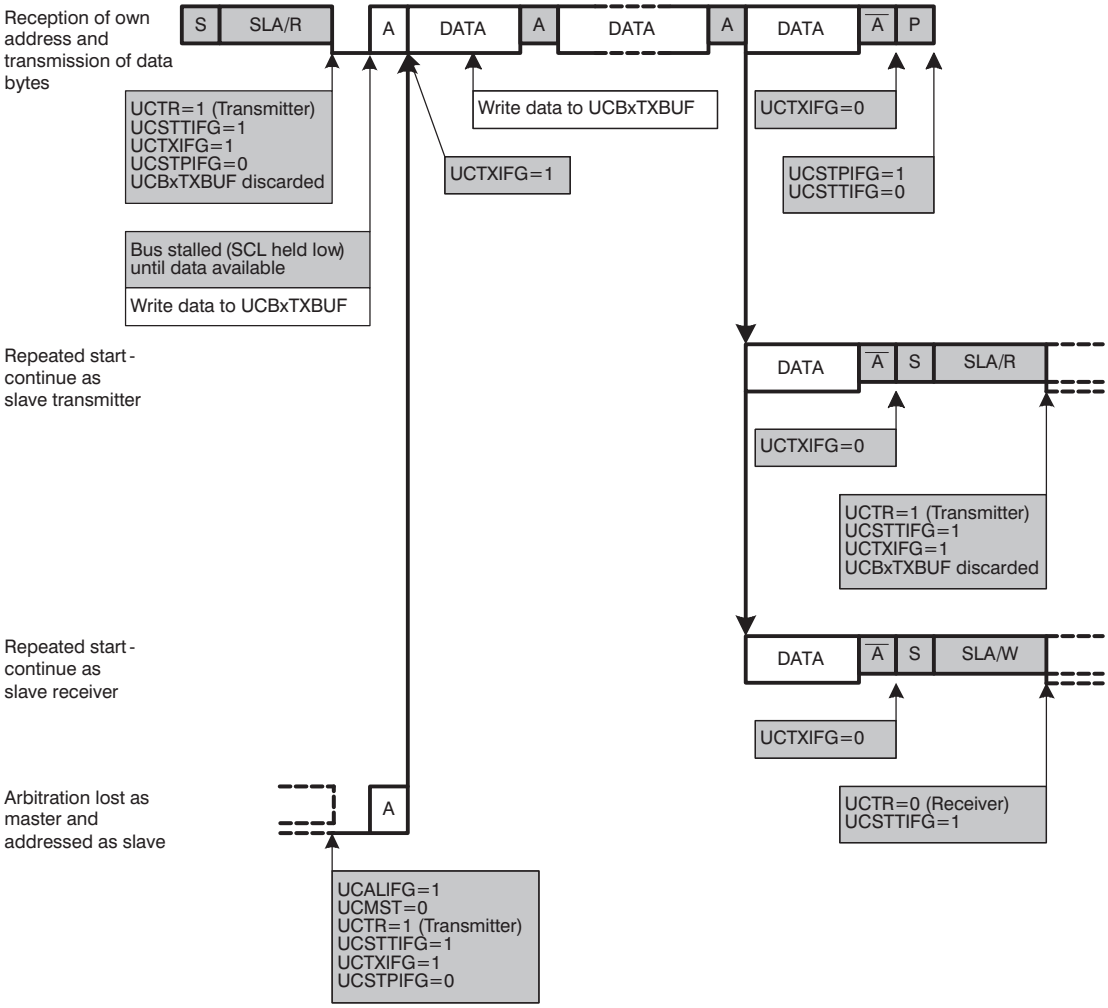


Figure 24-9. I²C Slave Transmitter Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.1.2 I²C Slave Receiver Mode

Slave receiver mode is entered when the slave address transmitted by the master is identical to its own address and a cleared R/$\overline{W}$ bit is received. In slave receiver mode, serial data bits received on SDA are shifted in with the clock pulses that are generated by the master device. The slave device does not generate the clock, but it can hold SCL low if intervention of the CPU is required after a byte has been received.

If the slave should receive data from the master, the USCI module is automatically configured as a receiver and UCTR is cleared. After the first data byte is received, the receive interrupt flag UCRXIFG is set. The USCI module automatically acknowledges the received data and can receive the next data byte.

If the previous data was not read from the receive buffer UCBxRXBUF at the end of a reception, the bus is stalled by holding SCL low. As soon as UCBxRXBUF is read, the new data is transferred into UCBxRXBUF, an acknowledge is sent to the master, and the next data can be received.

Setting the UCTXNACK bit causes a NACK to be transmitted to the master during the next acknowledgment cycle. A NACK is sent even if UCBxRXBUF is not ready to receive the latest data. If the UCTXNACK bit is set while SCL is held low, the bus is released, a NACK is transmitted immediately, and UCBxRXBUF is loaded with the last received data. Because the previous data was not read, that data is lost. To avoid loss of data, the UCBxRXBUF must be read before UCTXNACK is set**.**

When the master generates a STOP condition, the UCSTPIFG flag is set.

If the master generates a repeated START condition, the USCI I²C state machine returns to its address reception state.

Figure 24-10 shows the I²C slave receiver operation.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

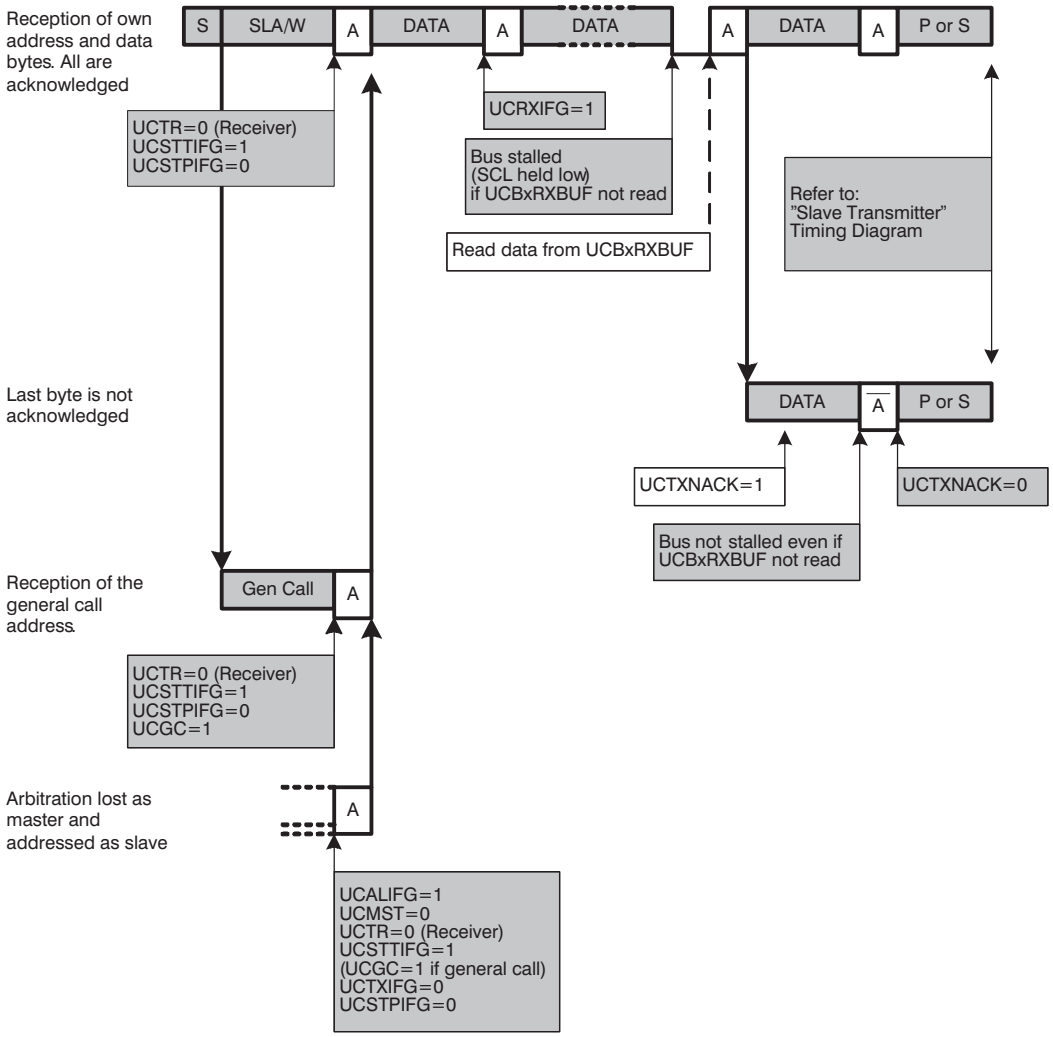


Figure 24-10. I²C Slave Receiver Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.1.3 I²C Slave 10-Bit Addressing Mode

The 10-bit addressing mode is selected when UCA10 = 1 and is as shown in Figure 24-11. In 10-bit addressing mode, the slave is in receive mode after the full address is received. The USCI module indicates this by setting the UCSTTIFG flag while the UCTR bit is cleared. To switch the slave into transmitter mode, the master sends a repeated START condition together with the first byte of the address but with the R/$\overline{W}$ bit set. This sets the UCSTTIFG flag if it was previously cleared by software, and the USCI modules switches to transmitter mode with UCTR = 1.

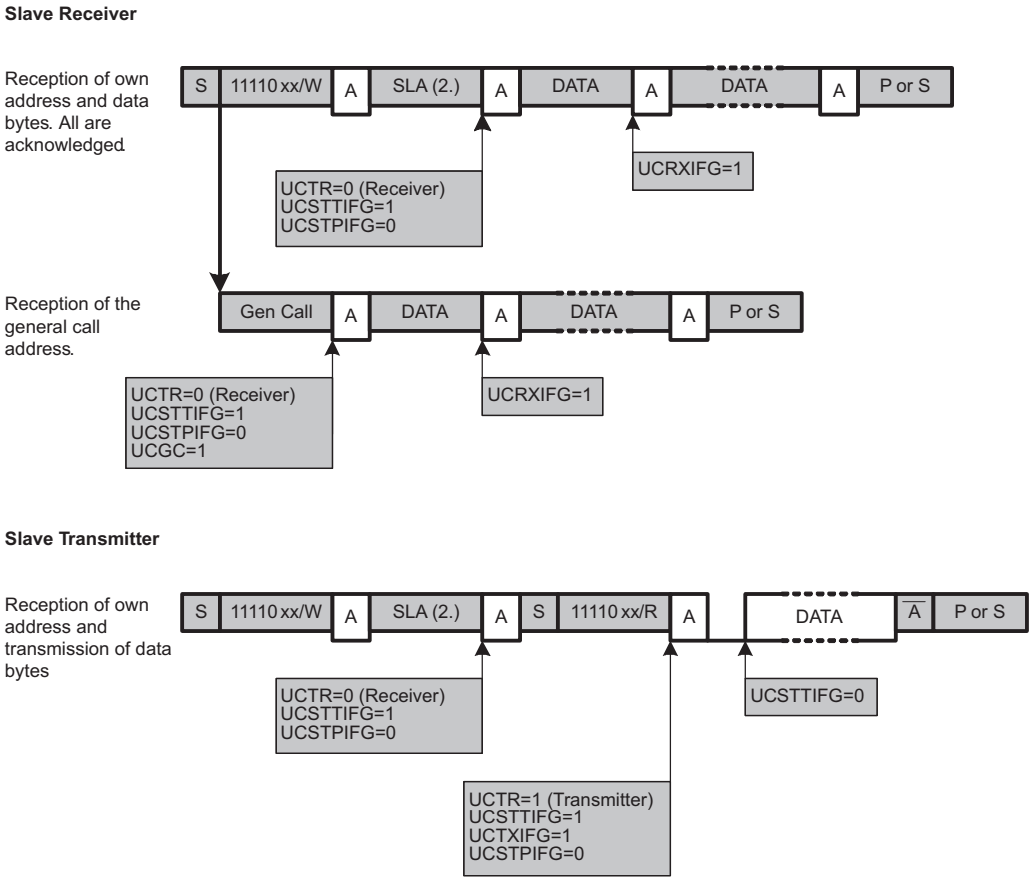


Figure 24-11. I²C Slave 10-Bit Addressing Mode

24.3.4.2 Master Mode

The USCI module is configured as an I²C master by selecting the I²C mode with UCMODEx = 11 and UCSYNC = 1 and setting the UCMST bit. When the master is part of a multi-master system, UCMM must be set and its own address must be programmed into the UCBxI2COA register. When UCA10 = 0, 7-bit addressing is selected. When UCA10 = 1, 10-bit addressing is selected. The UCGCEN bit selects if the USCI module responds to a general call.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.2.1 I²C Master Transmitter Mode

After initialization, master transmitter mode is initiated by writing the desired slave address to the UCBxI2CSA register, selecting the size of the slave address with the UCSLA10 bit, setting UCTR for transmitter mode, and setting UCTXSTT to generate a START condition.

The USCI module checks if the bus is available, generates the START condition, and transmits the slave address. The UCTXIFG bit is set when the START condition is generated and the first data to be transmitted can be written into UCBxTXBUF. As soon as the slave acknowledges the address, the UCTXSTT bit is cleared.

NOTE: Handling of TXIFG in a multi-master system

In a multi-master system (UCMM =1), if the bus is unavailable, the USCI module waits and checks for bus release. Bus unavailability can occur even after the UCTXSTT bit has been set. While waiting for the bus to become available, the USCI may update the TXIFG based on SCL clock line activity. Checking the UCTXSTT bit to verify if the START condition has been sent ensures that the TXIFG is being serviced correctly.

The data written into UCBxTXBUF is transmitted if arbitration is not lost during transmission of the slave address. UCTXIFG is set again as soon as the data is transferred from the buffer into the shift register. If there is no data loaded to UCBxTXBUF before the acknowledge cycle, the bus is held during the acknowledge cycle with SCL low until data is written into UCBxTXBUF. Data is transmitted or the bus is held, as long as the UCTXSTP bit or UCTXSTT bit is not set.

Setting UCTXSTP generates a STOP condition after the next acknowledge from the slave. If UCTXSTP is set during the transmission of the slave's address or while the USCI module waits for data to be written into UCBxTXBUF, a STOP condition is generated, even if no data was transmitted to the slave. When transmitting a single byte of data, the UCTXSTP bit must be set while the byte is being transmitted or anytime after transmission begins, without writing new data into UCBxTXBUF. Otherwise, only the address is transmitted. When the data is transferred from the buffer to the shift register, UCTXIFG is set, indicating data transmission has begun, and the UCTXSTP bit may be set.

Setting UCTXSTT generates a repeated START condition. In this case, UCTR may be set or cleared to configure transmitter or receiver, and a different slave address may be written into UCBxI2CSA if desired.

If the slave does not acknowledge the transmitted data, the not-acknowledge interrupt flag UCNACKIFG is set. The master must react with either a STOP condition or a repeated START condition. If data was already written into UCBxTXBUF, it is discarded. If this data should be transmitted after a repeated START, it must be written into UCBxTXBUF again. Any set UCTXSTT is also discarded. To trigger a repeated START, UCTXSTT must be set again.

Figure 24-12 shows the I²C master transmitter operation.

Copyright © 2009–2013, Texas Instruments Incorporated

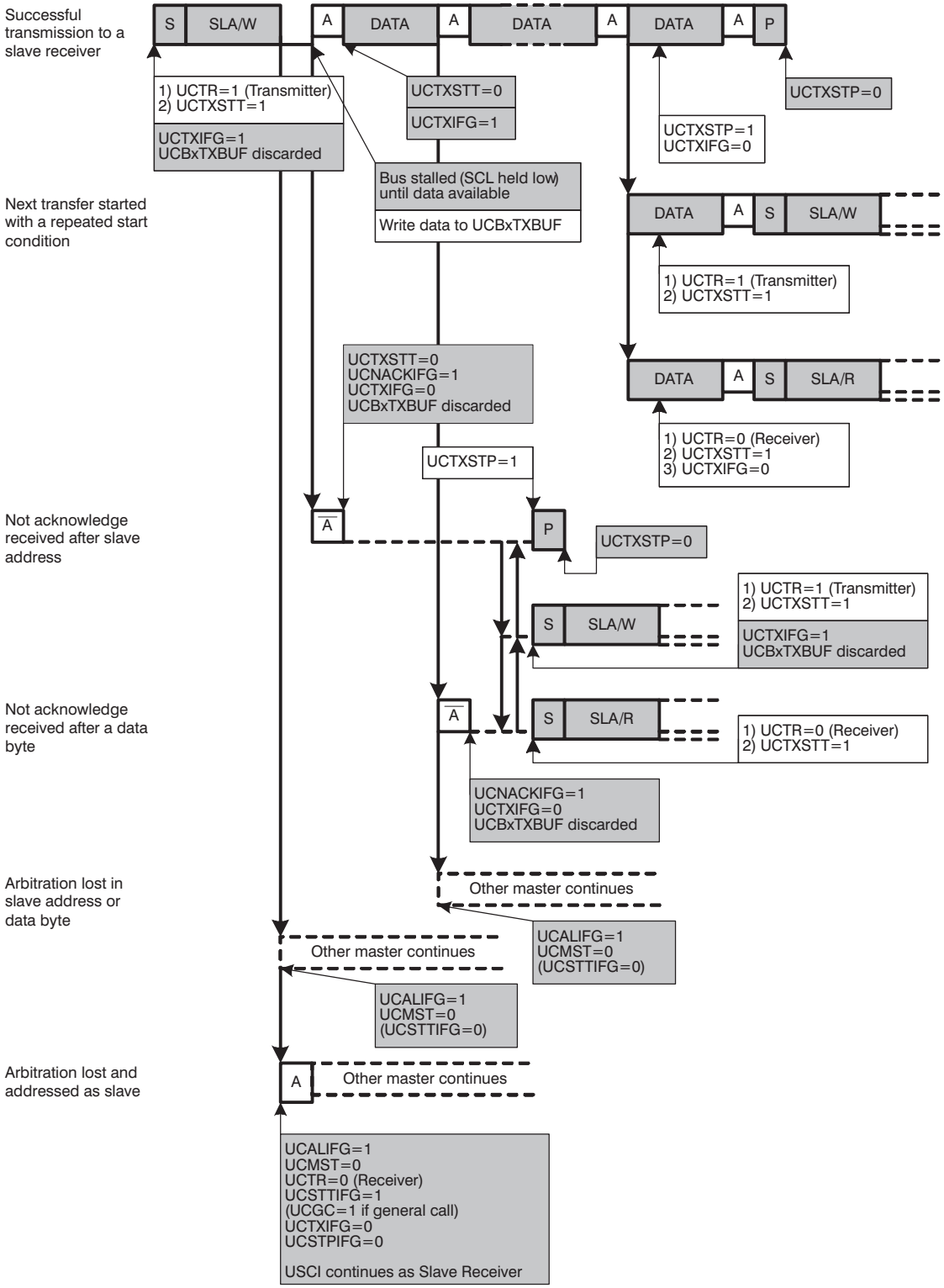


Figure 24-12. I²C Master Transmitter Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.2.2 I²C Master Receiver Mode

After initialization, master receiver mode is initiated by writing the desired slave address to the UCBxI2CSA register, selecting the size of the slave address with the UCSLA10 bit, clearing UCTR for receiver mode, and setting UCTXSTT to generate a START condition.

The USCI module checks if the bus is available, generates the START condition, and transmits the slave address. As soon as the slave acknowledges the address, the UCTXSTT bit is cleared.

After the acknowledge of the address from the slave, the first data byte from the slave is received and acknowledged and the UCRXIFG flag is set. Data is received from the slave, as long as UCTXSTP or UCTXSTT is not set. If UCBxRXBUF is not read, the master holds the bus during reception of the last data bit and until the UCBxRXBUF is read.

If the slave does not acknowledge the transmitted address, the not-acknowledge interrupt flag UCNACKIFG is set. The master must react with either a STOP condition or a repeated START condition.

Setting the UCTXSTP bit generates a STOP condition. After setting UCTXSTP, a NACK followed by a STOP condition is generated after reception of the data from the slave, or immediately if the USCI module is currently waiting for UCBxRXBUF to be read.

If a master wants to receive a single byte only, the UCTXSTP bit must be set while the byte is being received. For this case, the UCTXSTT may be polled to determine when it is cleared:

```
            BIS.B   #UCTXSTT, &UCB0CTL1  ;Transmit START cond.
POLL_STT    BIT.B   #UCTXSTT, &UCB0CTL1  ;Poll UCTXSTT bit
            JC      POLL_STT             ;When cleared,
            BIS.B   #UCTXSTP, &UCB0CTL1  ;transmit STOP cond.
```

Setting UCTXSTT generates a repeated START condition. In this case, UCTR may be set or cleared to configure transmitter or receiver, and a different slave address may be written into UCBxI2CSA if desired.

NOTE: Repeated START

The UCTXSTT bit must be set before the last data byte is received; that is, immediately after the UCRXIFG is set and the UCRXBUF with the second to last byte is read, the UCTXSTT bit should be set.

NOTE: Consecutive master transactions without repeated START

When performing multiple consecutive I²C master transactions without the repeated START feature, the current transaction must be completed before the next one is initiated. This can be done by ensuring that the transmit STOP condition flag UCTXSTP is cleared before the next I²C transaction is initiated with setting UCTXSTT = 1. Otherwise, the current transaction might be affected.

Figure 24-13 shows the I²C master receiver operation.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

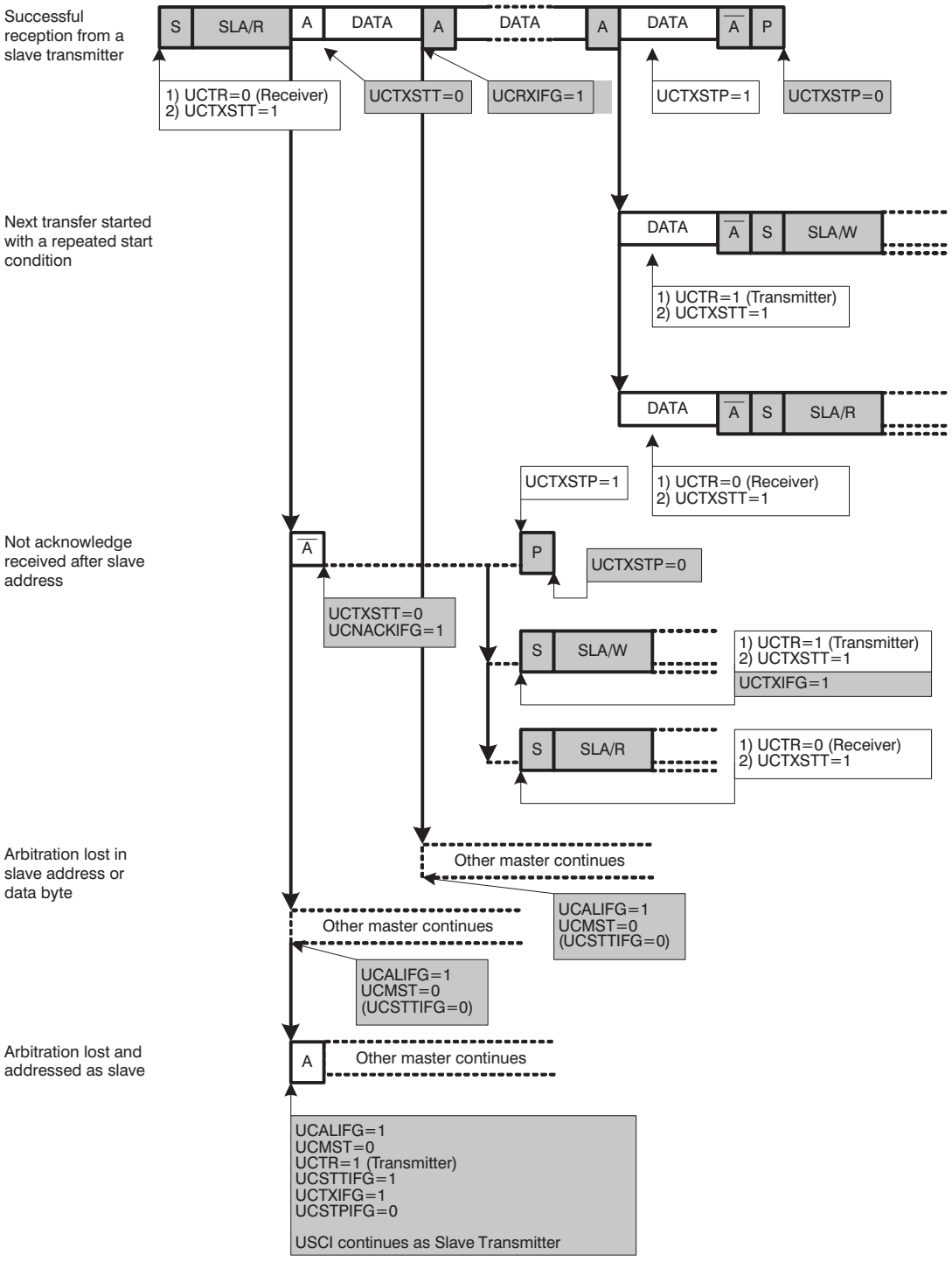


Figure 24-13. I²C Master Receiver Mode

24.3.4.2.3 I²C Master 10-Bit Addressing Mode

The 10-bit addressing mode is selected when UCSLA10 = 1 and is shown in Figure 24-14.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

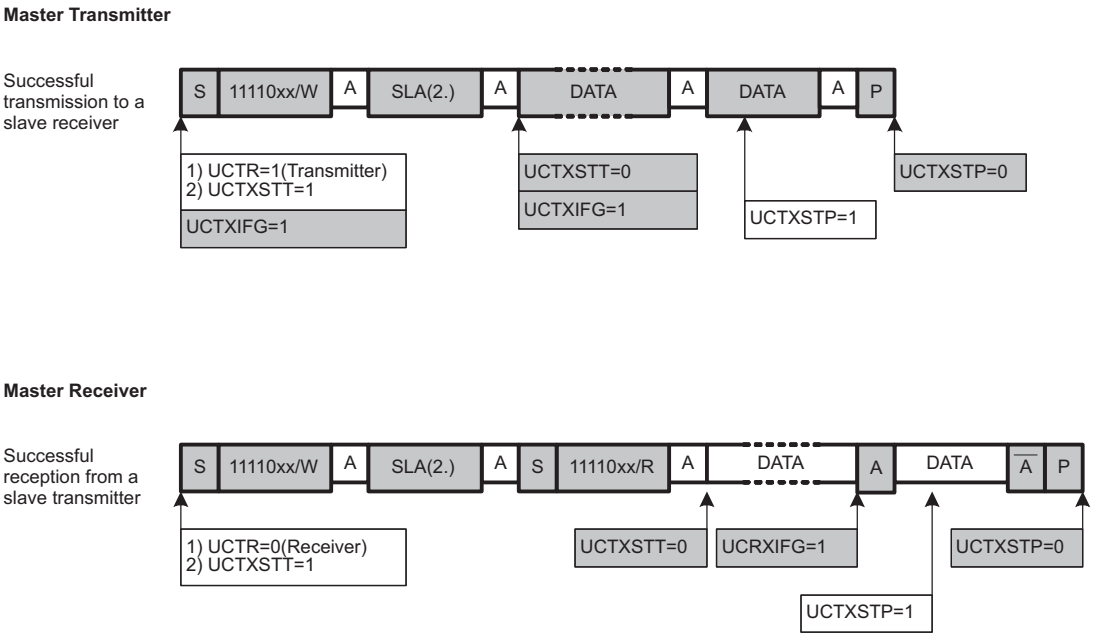


Figure 24-14. I²C Master 10-Bit Addressing Mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.4.3 Arbitration

If two or more master transmitters simultaneously start a transmission on the bus, an arbitration procedure is invoked. Figure 24-15 shows the arbitration procedure between two devices. The arbitration procedure uses the data presented on SDA by the competing transmitters. The first master transmitter that generates a logic high is overruled by the opposing master generating a logic low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. The master transmitter that lost arbitration switches to the slave receiver mode and sets the arbitration lost flag UCALIFG. If two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

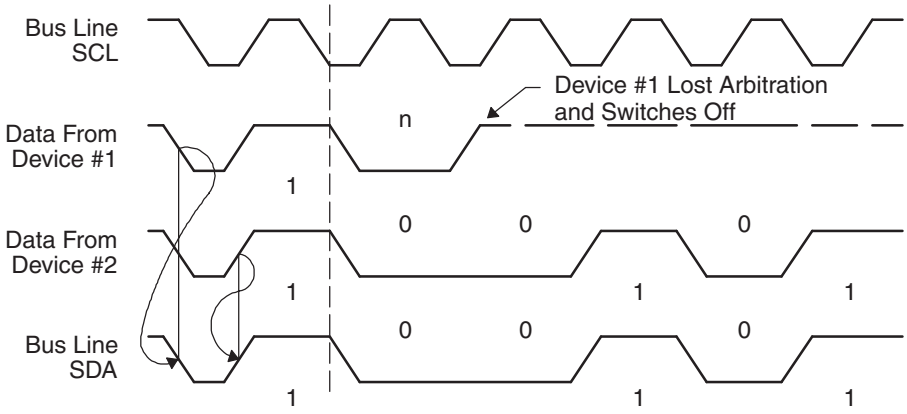


Figure 24-15. Arbitration Procedure Between Two Master Transmitters

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

If the arbitration procedure is in progress when a repeated START condition or STOP condition is transmitted on SDA, the master transmitters involved in arbitration must send the repeated START condition or STOP condition at the same position in the format frame. Arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

24.3.5 I²C Clock Generation and Synchronization

The I²C clock SCL is provided by the master on the I²C bus. When the USCI is in master mode, BITCLK is provided by the USCI bit clock generator and the clock source is selected with the UCSSELx bits. In slave mode, the bit clock generator is not used and the UCSSELx bits are don't care.

The 16-bit value of UCBRx in registers UCBxBR1 and UCBxBR0 is the division factor of the USCI clock source, BRCLK. The maximum bit clock that can be used in single master mode is $f_{BRCLK}/4$. In multi-master mode, the maximum bit clock is $f_{BRCLK}/8$. The BITCLK frequency is given by:

$f_{BitClock} = f_{BRCLK}/UCBRx$

The minimum high and low periods of the generated SCL are:

$t_{LOW,MIN} = t_{HIGH,MIN} = (UCBRx/2)/f_{BRCLK}$ when UCBRx is even

$t_{LOW,MIN} = t_{HIGH,MIN} = (UCBRx – 1/2)/f_{BRCLK}$ when UCBRx is odd

The USCI clock source frequency and the prescaler setting UCBRx must to be chosen such that the minimum low and high period times of the I²C specification are met.

During the arbitration procedure the clocks from the different masters must be synchronized. A device that first generates a low period on SCL overrules the other devices, forcing them to start their own low periods. SCL is then held low by the device with the longest low period. The other devices must wait for SCL to be released before starting their high periods. Figure 24-16 shows the clock synchronization. This allows a slow slave to slow down a fast master.

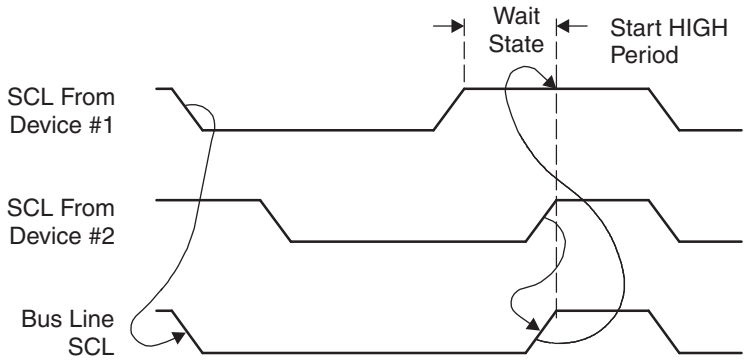


Figure 24-16. Synchronization of Two I²C Clock Generators During Arbitration

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.5.1 Clock Stretching

The USCI module supports clock stretching and also makes use of this feature as described in the Operation Mode sections.

The UCSCLLOW bit can be used to observe if another device pulls SCL low while the USCI module already released SCL due to the following conditions:

- USCI is acting as master and a connected slave drives SCL low.
- USCI is acting as master and another master drives SCL low during arbitration.

The UCSCLLOW bit is also active if the USCI holds SCL low because it is waiting as transmitter for data being written into UCBxTXBUF or as receiver for the data being read from UCBxRXBUF.

The UCSCLLOW bit might get set for a short time with each rising SCL edge because the logic observes the external SCL and compares it to the internally generated SCL.

24.3.6 Using the USCI Module in I²C Mode With Low-Power Modes

The USCI module provides automatic clock activation for use with low-power modes. When the USCI clock source is inactive because the device is in a low-power mode, the USCI module automatically activates it when needed, regardless of the control-bit settings for the clock source. The clock remains active until the USCI module returns to its idle condition. After the USCI module returns to the idle condition, control of the clock source reverts to the settings of its control bits.

In I²C slave mode, no internal clock source is required because the clock is provided by the external master. It is possible to operate the USCI in I²C slave mode while the device is in LPM4 and all internal clock sources are disabled. The receive or transmit interrupts can wake up the CPU from any low-power mode.

24.3.7 USCI Interrupts in I²C Mode

The USCI has only one interrupt vector that is shared for transmission, reception, and the state change. USCI_Ax and USC_Bx do not share the same interrupt vector.

Each interrupt flag has its own interrupt enable bit. When an interrupt is enabled and the GIE bit is set, the interrupt flag generates an interrupt request. DMA transfers are controlled by the UCTXIFG and UCRXIFG flags on devices with a DMA controller.

24.3.7.1 I²C Transmit Interrupt Operation

The UCTXIFG interrupt flag is set by the transmitter to indicate that UCBxTXBUF is ready to accept another character. An interrupt request is generated if UCTXIE and GIE are also set. UCTXIFG is automatically reset if a character is written to UCBxTXBUF or if a NACK is received. UCTXIFG is set when UCSWRST = 1 and the I²C mode is selected. UCTXIE is reset after a PUC or when UCSWRST = 1.

24.3.7.2 I²C Receive Interrupt Operation

The UCRXIFG interrupt flag is set when a character is received and loaded into UCBxRXBUF. An interrupt request is generated if UCRXIE and GIE are also set. UCRXIFG and UCRXIE are reset after a PUC signal or when UCSWRST = 1. UCRXIFG is automatically reset when UCxRXBUF is read.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.7.3 I²C State Change Interrupt Operation

Table 24-1 describes the I²C state change interrupt flags.

Table 24-1. I²C State Change Interrupt Flags

Interrupt Flag	Interrupt Condition
UCALIFG	Arbitration-lost. Arbitration can be lost when two or more transmitters start a transmission simultaneously, or when the USCI operates as master but is addressed as a slave by another master in the system. The UCALIFG flag is set when arbitration is lost. When UCALIFG is set, the UCMST bit is cleared and the I²C controller becomes a slave.
UCNACKIFG	Not-acknowledge interrupt. This flag is set when an acknowledge is expected but is not received. UCNACKIFG is automatically cleared when a START condition is received.
UCSTTIFG	START condition detected interrupt. This flag is set when the I²C module detects a START condition together with its own address while in slave mode. UCSTTIFG is used in slave mode only and is automatically cleared when a STOP condition is received.
UCSTPIFG	STOP condition detected interrupt. This flag is set when the I²C module detects a STOP condition while in slave mode. UCSTPIFG is used in slave mode only and is automatically cleared when a START condition is received.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.3.7.4 UCBxIV, Interrupt Vector Generator

The USCI interrupt flags are prioritized and combined to source a single interrupt vector. The interrupt vector register UCBxIV is used to determine which flag requested an interrupt. The highest-priority enabled interrupt generates a number in the UCBxIV register that can be evaluated or added to the PC to automatically enter the appropriate software routine. Disabled interrupts do not affect the UCBxIV value.

Any access, read or write, of the UCBxIV register automatically resets the highest-pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt.

24.3.7.4.1 UCBxIV Software Example

The following software example shows the recommended use of UCBxIV. The UCBxIV value is added to the PC to automatically jump to the appropriate routine. The example is given for USCI_B0.

```
USCI_I2C_ISR
         ADD      &UCB0IV, PC  ; Add offset to jump table
         RETI                  ; Vector 0: No interrupt
         JMP      ALIFG_ISR    ; Vector 2: ALIFG
         JMP      NACKIFG_ISR  ; Vector 4: NACKIFG
         JMP      STTIFG_ISR   ; Vector 6: STTIFG
         JMP      STPIFG_ISR   ; Vector 8: STPIFG
         JMP      RXIFG_ISR    ; Vector 10: RXIFG
TXIFG_ISR                     ; Vector 12
         ...                  ; Task starts here
         RETI                 ; Return
ALIFG_ISR                     ; Vector 2
         ...                  ; Task starts here
         RETI                 ; Return
NACKIFG_ISR                   ; Vector 4
         ...                  ; Task starts here
         RETI                 ; Return
STTIFG_ISR                    ; Vector 6
         ...                  ; Task starts here
         RETI                 ; Return
STPIFG_ISR                    ; Vector 8
         ...                  ; Task starts here
         RETI                 ; Return
RXIFG_ISR                     ; Vector 10
         ...                  ; Task starts here
         RETI                 ; Return
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4 USCI_B I2C Mode Registers

The USCI registers applicable in I2C mode are listed in Table 24-2. The base address can be found in the device-specific data sheet. The address offsets are listed in Table 24-2.

Table 24-2. USCI_B Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCBxCTLW0	USCI_Bx Control Word 0	Read/write	Word	0101h	
00h	UCBxCTL1	USCI_Bx Control 1	Read/write	Byte	01h	Section 24.4.2
01h	UCBxCTL0	USCI_Bx Control 0	Read/write	Byte	01h	Section 24.4.1
06h	UCBxBRW	USCI_Bx Bit Rate Control Word	Read/write	Word	0000h	
06h	UCBxBR0	USCI_Bx Bit Rate Control 0	Read/write	Byte	00h	Section 24.4.3
07h	UCBxBR1	USCI_Bx Bit Rate Control 1	Read/write	Byte	00h	Section 24.4.4
0Ah	UCBxSTAT	USCI_Bx Status	Read/write	Byte	00h	Section 24.4.5
0Bh		Reserved - reads zero	Read	Byte	00h	
0Ch	UCBxRXBUF	USCI_Bx Receive Buffer	Read/write	Byte	00h	Section 24.4.6
0Dh		Reserved - reads zero	Read	Byte	00h	
0Eh	UCBxTXBUF	USCI_Bx Transmit Buffer	Read/write	Byte	00h	Section 24.4.7
0Fh		Reserved - reads zero	Read	Byte	00h	
10h	UCBxI2COA	USCI_Bx I2C Own Address	Read/write	Word	0000h	Section 24.4.8
12h	UCBxI2CSA	USCI_Bx I2C Slave Address	Read/write	Word	0000h	Section 24.4.9
1Ch	UCBxICTL	USCI_Bx Interrupt Control	Read/write	Word	0200h	
1Ch	UCBxIE	USCI_Bx Interrupt Enable	Read/write	Byte	00h	Section 24.4.10
1Dh	UCBxIFG	USCI_Bx Interrupt Flag	Read/write	Byte	02h	Section 24.4.11
1Eh	UCBxIV	USCI_Bx Interrupt Vector	Read	Word	0000h	Section 24.4.12

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.1 UCBxCTL0 Register

USCI_Bx Control Register 0

Figure 24-17. UCBxCTL0 Register

7	6	5	4	3	2	1	0
UCA10	UCSLA10	UCMM	Reserved	UCMST	UCMODEx		UCSYNC
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	r-1

Modify only when UCSWRST = 1.

Table 24-3. UCBxCTL0 Register Description

Bit	Field	Type	Reset	Description
7	UCA10	RW	0h	Own addressing mode select 0b = Own address is a 7-bit address 1b = Own address is a 10-bit address
6	UCSLA10	RW	0h	Slave addressing mode select 0b = Address slave with 7-bit address 1b = Address slave with 10-bit address
5	UCMM	RW	0h	Multi-master environment select 0b = Single master environment. There is no other master in the system. The address compare unit is disabled. 1b = Multi-master environment
4	Reserved	R	0h	Reserved. Always reads as 0.
3	UCMST	RW	0h	Master mode select. When a master loses arbitration in a multi-master environment (UCMM = 1), the UCMST bit is automatically cleared and the module acts as slave. 0b = Slave mode 1b = Master mode
2-1	UCMODEx	RW	0h	USCI mode. The UCMODEx bits select the synchronous mode when UCSYNC = 1. 00b = 3-pin SPI 01b = 4-pin SPI (master/slave enabled if STE = 1) 10b = 4-pin SPI (master/slave enabled if STE = 0) 11b = I2C mode
0	UCSYNC	R	1h	Synchronous mode enable 0b = Asynchronous mode 1b = Synchronous mode

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.2 UCBxCTL1 Register

USCI_Bx Control Register 1

Figure 24-18. UCBxCTL1 Register

7	6	5	4	3	2	1	0
UCSSELx		Reserved	UCTR	UCTXNACK	UCTXSTP	UCTXSTT	UCSWRST
rw-0	rw-0	r0	rw-0	rw-0	rw-0	rw-0	rw-1

Modify only when UCSWRST = 1.

Table 24-4. UCBxCTL1 Register Description

Bit	Field	Type	Reset	Description
7-6	UCSSELx	RW	0h	USCI clock source select. These bits select the BRCLK source clock. 00b = UCLKI 01b = ACLK 10b = SMCLK 11b = SMCLK
5	Reserved	RW	0h	Reserved. Always reads as 0.
4	UCTR	RW	0h	Transmitter or receiver 0b = Receiver 1b = Transmitter
3	UCTXNACK	RW	0h	Transmit a NACK. UCTXNACK is automatically cleared after a NACK is transmitted. 0b = Acknowledge normally 1b = Generate NACK
2	UCTXSTP	RW	0h	Transmit STOP condition in master mode. Ignored in slave mode. In master receiver mode, the STOP condition is preceded by a NACK. UCTXSTP is automatically cleared after STOP is generated. 0b = No STOP generated 1b = Generate STOP
1	UCTXSTT	RW	0h	Transmit START condition in master mode. Ignored in slave mode. In master receiver mode, a repeated START condition is preceded by a NACK. UCTXSTT is automatically cleared after START condition and address information is transmitted. Ignored in slave mode. 0b = Do not generate START condition 1b = Generate START condition
0	UCSWRST	RW	1h	Software reset enable 0b = Disabled. USCI reset released for operation. 1b = Enabled. USCI logic held in reset state.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.3 UCBxBR0 Register

USCI_Bx Baud Rate Control Register 0

Figure 24-19. UCBxBR0 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 24-5. UCBxBR0 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler low byte. The 16-bit value of (UCxxBR0 + UCxxBR1 × 256) forms the prescaler value UCBRx.

24.4.4 UCBxBR1 Register

USCI_Bx Baud Rate Control Register 1

Figure 24-20. UCBxBR1 Register

7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Modify only when UCSWRST = 1.

Table 24-6. UCBxBR1 Register Description

Bit	Field	Type	Reset	Description
7-0	UCBRx	RW	undefined	Bit clock prescaler high byte. The 16-bit value of (UCxxBR0 + UCxxBR1 × 256) forms the prescaler value UCBRx.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.5 UCBxSTAT Register

USCI_Bx Status Register

Figure 24-21. UCBxSTAT Register

7	6	5	4	3	2	1	0
Reserved	UCSCLLOW	UCGC	UCBBUSY	Reserved			
rw-0	r-0	rw-0	r-0	r0	r0	r0	r0

Table 24-7. UCBxSTAT Register Description

Bit	Field	Type	Reset	Description
7	Reserved	RW	0h	Reserved. Always reads as 0.
6	UCSCLLOW	R	0h	SCL low 0b = SCL is not held low. 1b = SCL is held low.
5	UCGC	RW	0h	General call address received. UCGC is automatically cleared when a START condition is received. 0b = No general call address received 1b = General call address received
4	UCBBUSY	R	0h	Bus busy 0b = Bus inactive 1b = Bus busy
3-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.6 UCBxRXBUF Register

USCI_Bx Receive Buffer Register

Figure 24-22. UCBxRXBUF Register

7	6	5	4	3	2	1	0
UCRXBUFx							
r	r	r	r	r	r	r	r

Table 24-8. UCBxRXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCRXBUFx	R	undefined	The receive-data buffer is user accessible and contains the last received character from the receive shift register. Reading UCBxRXBUF resets UCRXIFG.

24.4.7 UCBxTXBUF Register

USCI_Bx Transmit Buffer Register

Figure 24-23. UCBxTXBUF Register

7	6	5	4	3	2	1	0
UCTXBUFx							
rw	rw	rw	rw	rw	rw	rw	rw

Table 24-9. UCBxTXBUF Register Description

Bit	Field	Type	Reset	Description
7-0	UCTXBUFx	RW	undefined	The transmit data buffer is user accessible and holds the data waiting to be moved into the transmit shift register and transmitted. Writing to the transmit data buffer clears UCTXIFG.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.8 UCBxI2COA Register

USCIBx I2C Own Address Register

Figure 24-24. UCBxI2COA Register

15	14	13	12	11	10	9	8
UCGCEN	Reserved					I2COAx	
rw-0	r0	r0	r0	r0	r0	rw-0	rw-0
7	6	5	4	3	2	1	0
I2COAx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Modify only when UCSWRST = 1.

Table 24-10. UCBxI2COA Register Description

Bit	Field	Type	Reset	Description
15	UCGCEN	RW	0h	General call response enable 0b = Do not respond to a general call 1b = Respond to a general call
14-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	I2COAx	RW	0h	I2C own address. The I2COAx bits contain the local address of the USCI_Bx I2C controller. The address is right justified. In 7-bit addressing mode, bit 6 is the MSB and bits 9-7 are ignored. In 10-bit addressing mode, bit 9 is the MSB.

24.4.9 UCBxI2CSA Register

USCI_Bx I2C Slave Address Register

Figure 24-25. UCBxI2CSA Register

15	14	13	12	11	10	9	8
Reserved						I2CSAx	
r0	r0	r0	r0	r0	r0	rw-0	rw-0
7	6	5	4	3	2	1	0
I2CSAx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 24-11. UCBxI2CSA Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved. Always reads as 0.
9-0	I2CSAx	RW	0h	I2C slave address. The I2CSAx bits contain the slave address of the external device to be addressed by the USCI_Bx module. It is only used in master mode. The address is right justified. In 7-bit slave addressing mode, bit 6 is the MSB and bits 9-7 are ignored. In 10-bit slave addressing mode, bit 9 is the MSB.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.10 UCBxIE Register

USCI_Bx I2C Interrupt Enable Register

Figure 24-26. UCBxIE Register

7	6	5	4	3	2	1	0
Reserved		UCNACKIE	UCALIE	UCSTPIE	UCSTTIE	UCTXIE	UCRXIE
r-0	r-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 24-12. UCBxIE Register Description

Bit	Field	Type	Reset	Description
7-6	Reserved	R	0h	Reserved. Always reads as 0.
5	UCNACKIE	RW	0h	Not-acknowledge interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
4	UCALIE	RW	0h	Arbitration lost interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
3	UCSTPIE	RW	0h	STOP condition interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
2	UCSTTIE	RW	0h	START condition interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
1	UCTXIE	RW	0h	Transmit interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE	RW	0h	Receive interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.11 UCBxIFG Register

USCI_Bx I2C Interrupt Flag Register

Figure 24-27. UCBxIFG Register

7	6	5	4	3	2	1	0
Reserved		UCNACKIFG	UCALIFG	UCSTPIFG	UCSTTIFG	UCTXIFG	UCRXIFG
r-0	r-0	rw-0	rw-0	rw-0	rw-0	rw-1	rw-0

Table 24-13. UCBxIFG Register Description

Bit	Field	Type	Reset	Description
7-6	Reserved	R	0h	Reserved. Always reads as 0.
5	UCNACKIFG	RW	0h	Not-acknowledge received interrupt flag. UCNACKIFG is automatically cleared when a START condition is received. 0b = No interrupt pending 1b = Interrupt pending
4	UCALIFG	RW	0h	Arbitration lost interrupt flag 0b = No interrupt pending 1b = Interrupt pending
3	UCSTPIFG	RW	0h	STOP condition interrupt flag. UCSTPIFG is automatically cleared when a START condition is received. 0b = No interrupt pending 1b = Interrupt pending
2	UCSTTIFG	RW	0h	START condition interrupt flag. UCSTTIFG is automatically cleared if a STOP condition is received. 0b = No interrupt pending 1b = Interrupt pending
1	UCTXIFG	RW	0h	USCI transmit interrupt flag. UCTXIFG is set when UCBxTXBUF is empty. 0b = No interrupt pending 1b = Interrupt pending
0	UCRXIFG	RW	0h	USCI receive interrupt flag. UCRXIFG is set when UCBxRXBUF has received a complete character. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

24.4.12 UCBxIV Register

USCI_Bx Interrupt Vector Register

Figure 24-28. UCBxIV Register

15	14	13	12	11	10	9	8
UCIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCIVx							
r0	r0	r0	r0	r-0	r-0	r-0	r0

Table 24-14. UCBxIV Register Description

Bit	Field	Type	Reset	Description
15-0	UCIVx	R	0h	USCI interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Arbitration lost; Interrupt Flag: UCALIFG; Interrupt Priority: Highest 04h = Interrupt Source: Not acknowledgment; Interrupt Flag: UCNACKIFG 06h = Interrupt Source: Start condition received; Interrupt Flag: UCSTTIFG 08h = Interrupt Source: Stop condition received; Interrupt Flag: UCSTPIFG 0Ah = Interrupt Source: Data received; Interrupt Flag: UCRXIFG 0Ch = Interrupt Source: Transmit buffer empty; Interrupt Flag: UCTXIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

CC1101-Based Radio Module (RF1A)

This chapter describes the RF1A module for frequencies below 1 GHz, which is based on the stand-alone radio device CC1101.

Topic **Page**

25.1 Radio Module Introduction **667**
25.2 Radio Interface Operation **668**
25.3 CC1101-Based Radio Core **680**
25.4 RF1A Registers **732**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.1 Radio Module Introduction

The RF1A radio module integrates the sub-1-GHz radio core of the CC1101 into the MSP430 system.

The CC1101-based radio core features a low intermediate frequency (IF) receiver. The received radio frequency (RF) signal is amplified by the low-noise amplifier (LNA) and down-converted in quadrature (I and Q) to the IF. At IF, the in-phase/quadrature-phase (I/Q) signals are digitized by the ADCs. Automatic gain control (AGC), fine channel filtering, and demodulation bit and packet synchronization is performed digitally.

The transmitter part is based on direct synthesis of the RF frequency. The frequency synthesizer includes a completely on-chip LC voltage-controlled oscillator (VCO) and a 90° phase shifter for generating the I and Q local oscillator (LO) signals to the down-conversion mixers in receive mode. A crystal oscillator with a 26-MHz crystal generates the reference frequency for the synthesizer and clocks for the ADC and the digital part. The digital baseband includes support for channel configuration, packet handling, and data buffering.

The features of the CC1101-based radio module are:

- Frequency bands: 300 MHz to 348 MHz, 389 MHz to 464 MHz, and 779 MHz to 928 MHz
- Programmable data rate from 0.8 kBaud to 500 kBaud
- High sensitivity (-110 dBm at 1.2 kBaud, 868 MHz, 1% packet error rate)
- Excellent receiver selectivity and blocking performance
- Programmable output power up to 10 dBm for all supported frequencies
- 2-FSK, 2-GFSK, and MSK supported as well as OOK and flexible ASK shaping
- Flexible support for packet oriented systems: On-chip support for sync word detection, address check, flexible packet length, and automatic CRC handling
- Support for automatic clear channel assessment (CCA) before transmitting (for listen-before-talk systems)
- Digital received signal strength indicator (RSSI) output
- Suited for systems targeting compliance with EN 300 220 (Europe) and FCC CFR Part 15 (US)

A simplified block diagram of the RF1A radio module based on the CC1101 is shown in Figure 25-1.

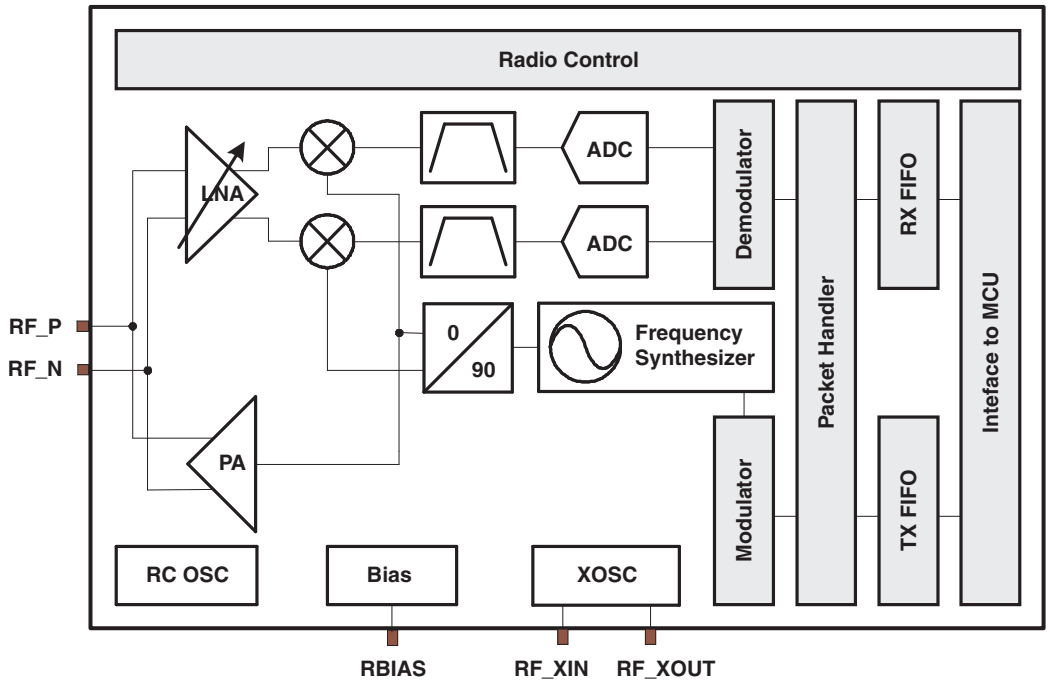


Figure 25-1. Simplified Block Diagram of the CC1101-Based Radio Module

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.2 Radio Interface Operation

The radio core is configured with user software via the radio interface. The radio interface is discussed in the following sections. Section 25.3 describes the radio core and its operation in detail.

25.2.1 Radio Interface

Figure 25-2 gives an overview of the radio interface and its interfaces to the radio core.

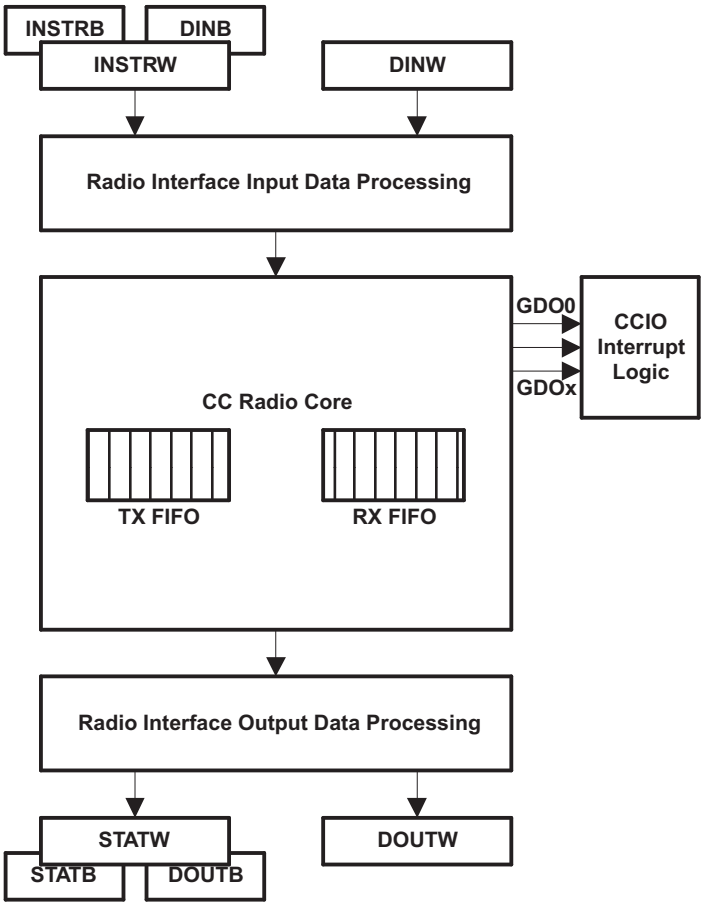


Figure 25-2. Radio Interface Overview

The communication between the radio interface and the radio core takes place via the logical channels shown in Figure 25-3. Every instruction presented to the core results in a status update sent to the interface, and every data byte or word sent to the radio core results in a data byte or word transmitted back to the interface. Depending on the instruction, some data in either direction can be "dummy" or "don't care" data. To avoid "dummy" writes, an auto-read feature is implemented.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

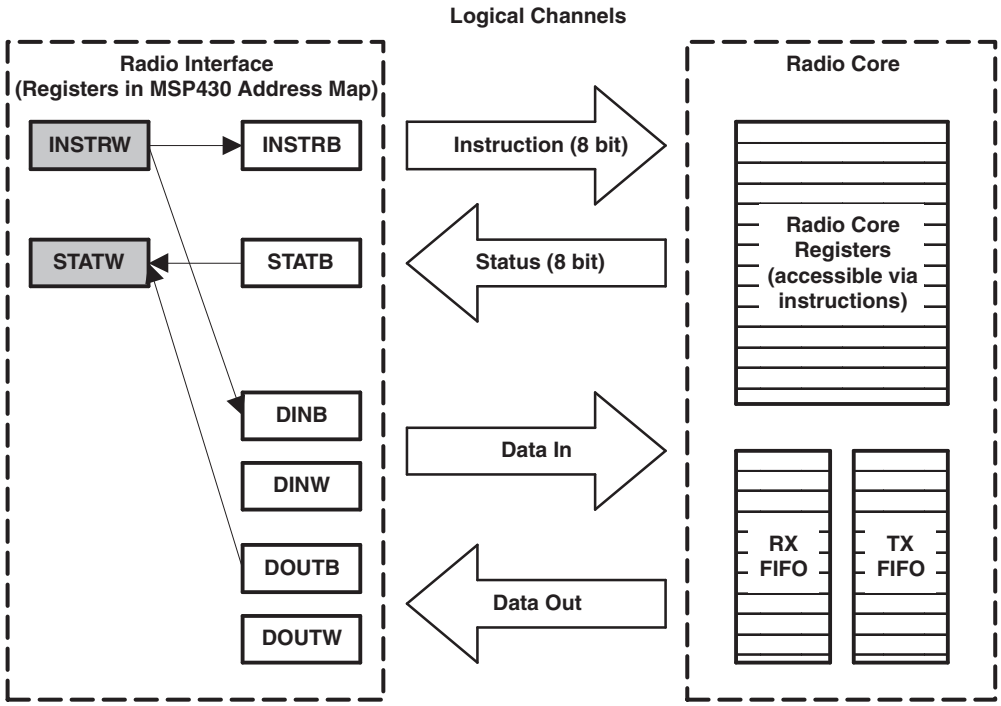


Figure 25-3. Logical Channels Between Radio Interface and Radio Core

25.2.1.1 Instruction and Status Register

The radio core is controlled using instructions written into the RF1AINSTRxW or RF1AINSTRxB registers. Additional data can be provided via the RF1ADINB or RF1ADINW registers (referred to as RF1ADIN registers in the following description). Resulting data can be read from RF1ADOUTxB or RF1ADOUTxW registers (referred to as RF1ADOUT registers in the following description). With each write access to the instruction registers (except for the SRES command strobe), the radio core status is updated and can be read via the RF1ASTATxB or RF1ASTATxW registers (referred to as RF1ASTAT registers in the following description).

Using the RF1AINSTRW register allows the application to provide the instruction together with the first data byte. Any further parameters need to be written into the RF1ADIN registers.

RF1AINSTRB can be used for single-byte instructions (like command strobes) and for all other instructions when all the required parameters are provided separately via the RF1ADIN registers.

The radio core status is updated with each transfer between the radio interface and the core. It can always be read via the byte status register RF1ASTATB.

The RF1ASTATW register can be used in conjunction with the word instruction register RF1AINSTRW. This provides the radio core status updated with the last instruction together with first data byte returned from the radio. If the instruction was written into the byte instruction register RF1AINSTRB reading RF1ASTATW causes an output data error, and the OUTERR flag is set.

The SNOP command strobe can be used to get an updated status without causing further actions.

The SRES command strobe does not provide an updated status. The interface and the radio core are reset.

An instruction is ended by either providing the complete instruction including all required data bytes (for instruction requiring a finite number of bytes) or by writing a new instruction into the instruction register. If an instruction requiring a given (and finite) number of bytes is aborted by writing a new instruction, the operand error flag OPERR is set. The aborted instruction might be partially executed already, causing an unpredictable state of the radio core.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.2.1.2 Data Registers

Parameters can be passed to the radio core using the word data register RF1ADINW or the byte data register RF1ADINB when the radio core is ready to accept additional data. This is indicated with RFDINIFG = 1. If the radio core is not ready to accept additional data and data is written into one of the RF1ADIN registers, the CPU is stalled for a given number of cycles or until the processing of previous data is completed. If the error condition still remains after the given number of clock cycles, the OPOVERR flag is set and the written data is ignored (also see Section 25.2.1.3).

The OPERR flag is set if the instruction does not expect more data; that is, if too many operands are provided. The excess operands are ignored.

Using the word data register RF1ADINW, word parameters can be passed to radio core registers using little-endian numbers (the default representation in MSP430 devices) even when the radio core expects big-endian numbers.

When writing words with RFENDIAN = 0, little-endian numbers are converted to the representation required by the radio core, whereas with RFENDIAN = 1, the data is passed unchanged and in the original order (starting at the lowest address first) to the radio core.

The CC1101-based radio core expects big-endian numbers. Here RFENDIAN = 1 might be used to ease the porting of code from a two-chip RF solution. RFENDIAN = 1 should also be used if transmit or receive data and the power amplifier table data is passed to the radio core using RF1ADINW.

A 16-bit word is passed to the radio core by writing the word into RF1ADINW.

A byte is passed to the radio core by writing it into RF1ADINB. A byte access to the low byte of the RF1ADINW register behaves identical to accessing RF1ADINB.

NOTE: Number of Bytes

Care must be taken that the exact number of bytes required by the radio core is passed. For example, if only one byte is provided, do not use the 16-bit word access. If the radio core expects a given number of bytes and this number is exceeded, the operand error flag OPERR is set. In this case, any excess data is ignored.

Data can be read from the radio core using the word data register RF1ADOUTW or the byte data register RF1ADOUTB. For word reads, the endianness of the data can be selected with RFENDIAN, similar to the process described previously.

A 16-bit word can be read from the radio core by reading it from RF1ADOUTW if previously a 16-bit data word was written into the radio using RF1ADINW or a 2-byte auto-read (see below) was initiated.

A byte is read from the radio core by reading it from RF1ADOUTB if previously a 8-bit data byte was written into the radio using RF1ADINB or RF1AINSTRW or if a 1-byte auto-read (see below) was previously initiated. A byte access to the low byte of the RF1ADOUTW register behaves identical to accessing RF1ADOUTB.

NOTE: Number of Bytes

Care needs to be taken that the exact number of bytes provided by the radio core is read - otherwise the OUTERR flag is set.

For example, if only one byte was written into the radio using RF1ADINB the resulting output data must not be read via RF1ADOUTW, which is a 16-bit word access.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.2.1.3 Delayed Write or Read

If a read or write access from the CPU to one of the radio interface registers causes an error condition, the CPU is stalled until the error condition is resolved for up to 16 CPU clock cycles. In correctly written software (1), the delay is triggered only when the synchronization between the CPU and the RF clock domain requires more than one CPU clock cycle. The delay of up to 16 CPU clock cycles accounts for the worst case conditions and ensures that the synchronization can be completed under all conditions before the read or write access is completed.

If the cycles expire without the error condition being resolved, the read or write access is executed anyway, to avoid possible deadlock situations. This happens only with erroneous software and the corresponding error flags are set.

25.2.1.4 Auto-Read

If the parameters provided to the radio core are don't care (for example, if the registers are only read) the auto-read feature can be used to avoid providing dummy data via the RF1ADIN registers.

To auto-read the next byte, the registers RF1AINSTR1B, RF1ASTAT1W, RF1ASTAT1B, RF1ADOUT1B, and RF1ADOUT1W can be used instead of the registers RF1AINSTRB, RF1ASTATW, RF1ASTATB, RF1ADOUTB, and RF1ADOUTW. Using these registers allows reading of the next byte without having to do a dummy write to RF1ADINB.

To auto-read the next two bytes (that is, one word) the registers RF1AINSTR2B, RF1AINSTR2W, RF1ASTAT2B, RF1ASTAT2W, RF1ADOUT2B, and RF1ADOUT2W can be used instead of the registers RF1AINSTRB, RF1AINSTRW, RF1ASTATB, RF1ASTATW, RF1ADOUTB, and RF1ADOUTW. Using these registers allows reading the next word (or the next two bytes) without having to do a dummy write to RF1ADINW.

(1) Correctly written software is software that does not implement deadlock situations like reading data from RF1ADOUT without having written data into RF1ADIN or using an auto-read register access previously.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-1. One-Byte Auto-Read Registers

Register	Auto-Read Register	Example C-Code Without Auto-Read	Example C-Code With Auto-Read

RF1AINSTRB	RF1AINSTR1B	`RF1AINSTRB = instr;`
		`RF1ADINB = 0; //dummy write`
		`byte_dat = RF1ADOUTB;`
			`RF1AINSTR1B = instr;`
			`// No dummy write!`
			`byte_dat = RF1ADOUTB;`

RF1AINSTRW	RF1AINSTR1W	`RF1AINSTRW = instr<<8;`
		(1)`// No dummy write`
		`byte_dat = RF1ADOUTB;`
			`RF1AINSTR1W = instr<<8;`
			`// No dummy write!`
			`byte_dat = RF1ADOUTB;`

RF1ASTATB	RF1ASTAT1B	`rf_stat = RF1ASTATB;`
		`RF1ADINB = 0; //dummy write`
		`byte_dat = RF1ADOUTB;`
			`rf_stat = RF1ASTAT1B;`
			`// No dummy write!`
			`byte_dat= RF1ADOUTB;`

RF1ASTATW	RF1ASTAT1W	`rf_stat_dat = RF1ASTATW;`
		`RF1ADINB = 0; //dummy write`
		`byte_dat = RF1ADOUTB;`
			`rf_stat_dat = RF1ASTAT1W;`
			`// No dummy write!`
			`byte_dat= RF1ADOUTB;`

RF1ADOUTB	RF1ADOUT1B	`first_byte = RF1ADOUTB;`
		`RF1ADINB = 0; //dummy write`
		`byte_dat = RF1ADOUTB;`
			`first_byte = RF1ADOUT1B;`
			`// No dummy write!`
			`byte_dat = RF1ADOUTB;`

RF1ADOUTW	RF1ADOUT1B	`first_word = RF1ADOUTW;`
		`RF1ADINB = 0; //dummy write`
		`byte_dat = RF1ADOUTB;`
			`first_word = RF1ADOUT1W;`
			`// No dummy write!`
			`byte_dat = RF1ADOUTB;`

(1) No dummy byte write is required, because one dummy byte is already written with RF1AINSTRW.

Table 25-2. Two-Byte (One-Word) Auto-Read Registers

Register	Auto-Read Register	Example C-Code Without Auto-Read	Example C-Code With Auto-Read

RF1AINSTRB	RF1AINSTR2B	`RF1AINSTRB = instr;`
		`RF1ADINW = 0; //dummy write`
		`word_dat = RF1ADOUTW;`
			`RF1AINSTR2B = instr;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

RF1AINSTRW	RF1AINSTR2W	`RF1AINSTRW = instr<<8;`
		(1)`RF1ADINB = 0;//dummy write`
		`word_dat = RF1ADOUTW;`
			`RF1AINSTR2W = instr<<8;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

RF1ASTATB	RF1ASTAT2B	`rf_stat = RF1ASTATB;`
		`RF1ADINW = 0; //dummy write`
		`word_dat = RF1ADOUTW;`
			`rf_stat = RF1ASTAT2B;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

RF1ASTATW	RF1ASTAT2W	`rf_stat_dat = RF1ASTATW;`
		`RF1ADINW = 0; //dummy write`
		`word_dat = RF1ADOUTW;`
			`rf_stat_dat = RF1ASTAT2W;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

RF1ADOUTB	RF1ADOUT2B	`first_byte = RF1ADOUTB;`
		`RF1ADINW = 0; //dummy write`
		`word_dat = RF1ADOUTW;`
			`first_byte = RF1ADOUT2B;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

RF1ADOUTW	RF1ADOUT2B	`first_word = RF1ADOUTW;`
		`RF1ADINW = 0; //dummy write`
		`word_dat = RF1ADOUTW;`
			`first_word = RF1ADOUT2W;`
			`// No dummy write!`
			`word_dat = RF1ADOUTW;`

(1) Only one dummy *byte* write is required, because one dummy byte is already written with RF1AINSTRW.

25.2.1.5 Error Flags

Table 25-3 lists the interface error conditions and its flags. There is also a error vector generator register, RF1AERRV, that allows decoding the error condition using the same mechanisms as with the interrupt vector word registers. Any read access of the RF1AERRV register automatically resets the highest pending error flag. If another error flag is set, the RFERRIFG interrupt flag remains set, and another interrupt is immediately generated after servicing the initial interrupt. A write access to the RF1AERRV register automatically resets all error flags. In addition, all error flags can be cleared via software.

The flags are provided to simplify software debugging. Ideally, a production ready software should never see any radio interface errors.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-3. Radio Interface Error Conditions

Error Condition	Error Flag	Description
Operand error	OPERR	Not enough or too many operands were provided for an instruction. If not enough operands were provided, the instruction was aborted by writing a new instruction before all required operands were passed to the radio core. The aborted instruction might already be partially executed, causing an unpredictable state of the radio core. If too many operands were provided, the excess operands are ignored.
Output data not available error	OUTERR	Not enough data available for the executed read access.
Operand overwrite error	OPOVERR	Attempt to overwrite operands in RF1ADIN registers that are still being processed by the radio core. Written data is ignored.
Low-voltage error	LVERR	Attempt to activate the radio core (go to a state other than SLEEP and IDLE) with a core voltage level PMMCOREVx = 00b or 01b. The transition from IDLE to the next state is prevented until PMMCOREVx ≥ 10b and the LVERR flag is cleared by software. The command that triggered the LVERR to be set is ignored and must be retransmitted to the radio core after the error condition is resolved; that is, after the core voltage level setting is increased to PMMCOREVx ≥ 10b and after ensuring that the voltage reached the desired level.

25.2.1.5.1 Examples

```
// OPERR - not enough operands
RF1AINSTRB = SNGLREGWR+0x00; // Write radio core register IOCFG2, expects 1 byte
RF1AINSTRB = SNOP;           // Error!
//=> OPERR flag set,
//   no operand provided for previous instruction

// OPERR - too many operands
RF1AINSTRB = SNGLREGWR+0x00; // Write radio core register IOCFG2, expects 1 byte
RF1ADINB = 0x00;             // Okay.
RF1ADINB = 0x01;             // Error!
//=> OPERR flag set,
//   only 1 byte expected for SNGLREGWR instruction,
//   data ignored

// OPOVERR
RF1AINSTRB = REGWR+0x00; // Write radio core registers starting with IOCFG2
RF1ADINB = 0x00;
RF1ADINB = 0x01;
//=> OPOVERR flag set and data ignored,
//   if synchronization of data between the interface and the core takes too long

// OUTERR
RF1AINSTRB = REGRD+0x00; // Read radio core registers starting with IOCFG2
data= RF1ADOUTB;         // Error!
//=> OUTERR flag set, because dummy data write is missing
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.2.2 Radio Interface Interrupts

The radio interface provides a number of interrupt flags to control the data-flow between the radio core and the CPU. The interrupt flags are listed in Table 25-4.

Table 25-4. Radio Interface Interrupt Flags

Interrupt Flag	Interrupt Condition
RFINSTRIFG	The radio core is ready to accept the next instruction; that is, the previous instruction was completely processed and all required data was provided.
RFDINIFG	The radio core is ready to accept additional data.
RFSTATIFG	The radio core updated the status accessible via the RF1ASTAT registers. If the instruction was provided as a word via RF1AINSTRW, the flag is set after the first data byte is also available.
RFDOUTIFG	Data was provided by the radio core and can be read via the RF1ADOUT registers. If the corresponding parameters were provided as 16-bit data, the flag is set only after 16-bit data is available. With the auto-read feature, the flag is set after the selected amount of data is available. With each read access of the RF1ADOUT registers, the flag is cleared; if there is still data available after the read access, the RFDOUTIFG is set again.
RFERRIFG	An error occurred interfacing to the radio core. The error condition can be encoded using the error flags. The error interrupt flag is set as long as one of the error flags (OPERR, OUTERR, OPOVERR, or LVERR) is set. It is cleared automatically when all error flags are cleared.

25.2.3 Radio Core Interrupts

The radio core provides interrupt signals to the radio interface. There are three programmable output signals GDO0, GDO1, and GDO2 that can also be routed to pins as well as hardwired output signals going into the interrupt logic.

For each interrupt signal there is an associated interrupt flag RFIFGx together with an interrupt enable RFIEx, an interrupt edge select RFIESx, and an input bit RFINx. The input bit RFINx allows to query the actual status of a signal, the interrupt edge select bit RFIESx allows to trigger an interrupt on the positive (RFIES = 0) or on the negative (RFIES = 1) edge of the associated signal and RFIEx enables the associated RFIFGx to cause an interrupt.

NOTE: Changing RFIES

Changing RFIES bits can result in setting the corresponding RFIFG interrupt flags.

The radio core interrupt flags are prioritized and combined to source a single interrupt vector together with the radio interface interrupts. The interrupt vector register RF1AIV is used to determine which radio core interrupt flag requested an interrupt. The highest priority enabled interrupt generates a number in the RF1AIV register that can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled interrupts do not affect the RF1AIV value.

Any read access of the RF1AIV register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is generated immediately after servicing the initial interrupt. A write access to the RF1AIV register automatically resets all pending interrupt flags.

Table 25-5 lists the available CC1101 radio core interrupt sources and the associated flags. It also lists for the hardwired interrupt sources what event causes a low-to-high transition and what causes a high-to-low transition.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-5. CC1101 Radio Core Interrupt Mapping

Interrupt Flag	Interrupt Condition
RFIFG0	Based on GDO0 signal - programmable using IOCFG0 (0x02) register of radio core.
RFIFG1	Based on GDO1 signal - programmable using IOCFG1 (0x01) register of radio core.
RFIFG2	Based on GDO2 signal - programmable using IOCFG2 (0x00) register of radio core.
RFIFG3	Positive edge: RX FIFO filled or above the RX FIFO threshold. Negative edge: RX FIFO drained below RX FIFO threshold. (Equal to GDOx_CFG=0)
RFIFG4	Positive edge: RX FIFO filled or above the RX FIFO threshold or end of packet is reached. Negative edge: RX FIFO empty. (Equal to GDOx_CFG=1)
RFIFG5	Positive edge: TX FIFO filled or above the TX FIFO threshold. Negative edge: TX FIFO below TX FIFO threshold. (Equal to GDOx_CFG=2)
RFIFG6	Positive edge: TX FIFO full. Negative edge: TX FIFO below TX FIFO threshold. (Equal to GDOx_CFG=3)
RFIFG7	Positive edge: RX FIFO overflowed. Negative edge: RX FIFO flushed. (Equal to GDOx_CFG=4)
RFIFG8	Positive edge: TX FIFO underflowed. Negative edge: TX FIFO flushed. (Equal to GDOx_CFG=5)
RFIFG9	Positive edge: Sync word sent or received. Negative edge: End of packet or in RX when optional address check fails or RX FIFO overflows or in TX when TX FIFO underflows. (Equal to GDOx_CFG=6)
RFIFG10	Positive edge: Packet received with CRC OK. Negative edge: First byte read from RX FIFO. (Equal to GDOx_CFG=7)
RFIFG11	Positive edge: Preamble quality reached (PQI) is above programmed PQT value. Negative edge: (LPW) (Equal to GDOx_CFG=8)
RFIFG12	Positive edge: Clear channel assessment when RSSI level is below threshold (dependent on the current CCA_MODE setting). Negative edge: RSSI level is above threshold. (Equal to GDOx_CFG=9)
RFIFG13	Positive edge: Carrier sense. RSSI level is above threshold. Negative edge: RSSI level is below threshold. (Equal to GDOx_CFG=14)
RFIFG14	Positive edge: WOR event 0 Negative edge: WOR event 0 + 1 ACLK. (Equal to GDOx_CFG=36)
RFIFG15	Positive edge: WOR event 1 Negative edge: RF oscillator stable or next WOR event0 triggered. (Equal to GDOx_CFG=37)

25.2.4 Using Radio With Low-Power Modes

The radio works seamlessly in LPM0 to LPM2. To enable radio operation in LPM3 or LPM4, the application must set the PMMHPMRE bit in the PMMCTL0 register of the PMM module to 1. This enables the PMM to provide sufficient current while the radio is active. When the radio goes into SLEEP state, the current consumption drops to the usual LPM3 numbers; however, while the radio is active, the current consumption reflects the respective radio state.

When the radio needs to transition from the SLEEP state to an active state (for example, from IDLE, to RX or TX) the CPU must not go into a low-power mode until the radio is active. This is reflected by the RF_RDYn signal transitioning from 1 to 0. This can be observed best on the programmable RFINx signals; for example, the GDO2 signal fed to RFIN2 is programmed by default to RF_RDYn.

25.2.5 Radio Interrupt Handling

The radio module has one entry in the device's interrupt vector table and two interrupt vector word registers RF1AIFIV and RF1AIV that are used to determine which interface interrupt flag or radio core interrupt flag requested an interrupt.

The highest priority enabled interface or core interrupt generates a number in the RF1AIFIV or RF1AIV registers (see register description). This number can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled interrupts do not affect the RF1AIFIV and RF1AIV values.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Any read access of the RF1AIFIV register automatically resets the highest pending radio interface interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. A write access to the RF1AIFIV register automatically resets all pending radio interface interrupt flags. In addition, all flags can be cleared via software.

Any read access of the RF1AIV register automatically resets the highest pending radio core interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. A write access to the RF1AIV register automatically resets all pending radio core interrupt flags. In addition, all flags can be cleared via software.

25.2.5.1 RF1AIFIV and RF1AIV Software Example

The following software example shows the recommended use of RF1AIFIV, RF1AIV, and the handling overhead. The RF1AIFIV or RF1AIV values are added to the PC to automatically jump to the appropriate routine. The software can define the priority of the interface interrupts versus the priority of the core interrupts by ordering the access to the interrupt vector word registers accordingly.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself.

```
; Interrupt handler for radio interrupt flags.
; Radio interface interrupts have higher priority

RADIO_HND                           ; Interrupt latency           6
                                    ; Radio Interface Interrupts
            ADD        &RF1AIFIV,PC ; Add offset to Jump table    3
            JMP    RF_CORE_HND      ; Vector 0: No I/F interrupt  2
            JMP    ..._HND          ; Vector 2: ...               2
            JMP    ..._HND          ; Vector 4: ...               2
            ...                     ; ...
            JMP    ..._HND          ; Vector 12: ...              2
                                    ; Vector 14: ...
            ...                     ; Task starts here
            RETI                                                  5

..._HND                             ; Vector xyz: ...
            ...                     ; Task starts here
            RETI                                                  5

RF_CORE_HND                         ; Radio Core Interrupts
            ADD        &RF1AIV,PC   ; Add offset to Jump table    3
            RETI                    ; Vector 0: No interrupt      5
            JMP    ..._HND          ; Vector 2: ...               2
            JMP    ..._HND          ; Vector 4: ...               2
            ...                     ; ...
            JMP    ..._HND          ; Vector 30: ...              2
                                    ; Vector 32: ...
            ...                     ; Task starts here
            RETI                                                  5

..._HND                             ; Vector xyz: ...
            ...                     ; Task starts here
            RETI                                                  5

; Interrupt handler for radio interrupt flags.
; Radio core interrupts have higher priority

RADIO_HND                           ; Interrupt latency           6
                                    ; Radio Core Interrupts
            ADD        &RF1AIV,PC   ; Add offset to Jump table    3
            JMP    RF_IF_HND        ; Vector 0: No Core interr.   2
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.


```
                JMP    ..._HND           ; Vector 2: ...               2
                JMP    ..._HND           ; Vector 4: ...               2
                ...                      ; ...
                JMP    ..._HND           ; Vector 30: ...              2
                                         ; Vector 32: ...
                ...                      ; Task starts here
                RETI                                                   5

...._HND                                 ; Vector xyz: ...
                ...                      ; Task starts here
                RETI                                                   5

RF_IF_HND                                ; Radio Interface Interrupts
                ADD        &RF1AIFIV,PC  ; Add offset to Jump table    3
                RETI                     ; Vector 0: No interrupt      5
                JMP    ..._HND           ; Vector 2: ...               2
                JMP    ..._HND           ; Vector 4: ...               2
                ...                      ; ...
                JMP    ..._HND           ; Vector 12: ...              2
                                         ; Vector 14: ...
                ...                      ; Task starts here
                RETI                                                   5

...._HND                                 ; Vector xyz: ...
                ...                      ; Task starts here
                RETI                                                   5
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.2.6 Software Considerations

The radio interface supports three different possibilities to access the radio core:

1. Straight access to interface registers

   This method relies on the implemented delay that can cause the CPU to stall for up to 16 cycles (see Section 25.2.1.3). The advantage of this method is that it is very straight forward and requires the least code overhead. The draw-back of this solution is that the worst case interrupt latency is increase by 16 MCLK cycles. In erroneous software, an error flag is set that can be caught in the RF1A interrupt service routine.

2. Polling interface interrupt flags

   The interface interrupt flags (for example the RFINSTRIFG) are polled before accessing the corresponding interface register (for example, RF1AINSTRB). This method does not add any interrupt latency because by polling the flag it is ensured that the corresponding register can be accessed without stalling the CPU. The draw-back is the requirement of having to execute polling loop before each access to the radio interface registers.

3. Using the interface interrupt

   Instead of polling the interrupt flags the access to the interface registers is executed in the radio interface interrupt service routine. For example all radio interface interrupts are disabled by default. If then the instruction register should be written, one would write the instruction in a global variable that can be read within the interrupt service routine and would set the corresponding interrupt enable RFINSTRIE = 1. Within the interrupt service routine one decodes that the RFINSTRIFG is set and then loads the RFAINSTRB register with the value store in the global variable.

   This method also adds no interrupt latency as above and avoids the polling loops. The draw-back is that it requires a rather sophisticated interrupt service routine.

Of course the methods mentioned above can be mixed in a given application.

One possibility to ease porting of software between a two-chip solution MSP430 plus CC1101 and a CC430-based one-chip solution is abstracting the access to the radio core using C-Defines and C-Functions. Then same "functions" can be used independently of being used in a two-chip or a single-chip application.

The basic idea is to have some C-Defines and C-Functions as shown below that can be used independently of being used in a two-chip or a single-chip application.

```
#ifdef CC430_RF1A

// C-Defines to "abstract" the CC430 radio interface for a one-chip solution (using option 1)

#define RF_IFCTL0(ctl) RF1AIFCTL0= ctl
#define RF_INSTRB(instr) RF1AINSTRB= instr
#define RF_INSTRW(instr) RF1AINSTRW= instr
#define RF_INSTR1B(instr) RF1AINSTR1B= instr
#define RF_STATB() RF1ASTATB
#define RF_DINB(din) RF1ADINB= din
#define RF_DINW(din ) RF1ADINW= din
#define RF_DOUTB() RF1ADOUT0B
#define RF_DOUT1B() RF1ADOUT1B
#define RF_DOUT2B() RF1ADOUT2B
#define RF_DOUTW() RF1ADOUT0W
//...

// Example of a define using option 2
//#define RF_INSTRB(instr) {while( !(RF1AIFCTL1 & RFINSTRIFG)); RF1AINSTRB= instr; }


#else

// C-Functions to mimic the CC430 radio interface for a two-chip solution

void RF_IFCTL0(unsigned int ctl);
void RF_INSTRB(char instr);
void RF_INSTRW(unsigned int instr);
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.


```
void RF_INSTR1B(char instr);
char RF_STATB();
void RF_DINB(char din);
void RF_DINW(unsigned int din);
char RF_DOUTB();
char RF_DOUT1B();
char RF_DOUT2B();
unsigned int RF_DOUTW();
//...

#endif
```

The C-Functions for the two-chip solution need to provide the functionality of the C-Defines, taking the interface toward the CC1101 into account. For example, the RF_INSTR function for a device with the USART module might look like this:

```
void RF_INSTRB(char instr)
{
  // Indicate start of new instr. by toggling CS
  TI_CC_CSn_PxOUT |= TI_CC_CSn_PIN;                    // /CS disable
  TI_CC_CSn_PxOUT &= ~TI_CC_CSn_PIN;                   // /CS enable
  while (TI_CC_SPI_USART1_PxIN&TI_CC_SPI_USART1_SOMI); // Wait for CCxxxx ready
  IFG2 &= ~URXIFG1;                                    // Clear RX flag
  U1TXBUF = instr;                                     // Send instr
  while (!(IFG2&URXIFG1));                             // Wait until status received
  rf_statb= U1RXBUF;                                   // Read status
  rf_dout_cnt= 0;                                      // No data returned

  // if strobe command disable /CS immediately
  if ((instr >= 0x30) && (instr <= 0x3d))
  {
    TI_CC_CSn_PxOUT |= TI_CC_CSn_PIN; // /CS disable
  } // if
} // RF_INSTRB
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3 CC1101-Based Radio Core

25.3.1 Differences From CC1101

The radio core differs from the CC1101 device:

- Forward error correction (FEC) and interleaving are not supported.
- After a reset (PUC) the radio is in SLEEP state instead of IDLE.
- Strobe command SRES resets the radio and sets the state machine to SLEEP instead of IDLE.
- Strobe command SXOFF causes the radio core to transition into the SLEEP state instead of the XOFF state.
- ACLK is clock source for the WOR timer, not the integrated CC1101 R/C oscillator.
- Instructions can be executed while the radio core is in SLEEP state.
- An oscillator fault detection for the RF crystal oscillator is available via the UCS module. A fault of RF oscillator must be "acknowledged" with a SRES command strobe.
- Support of synchronous and asynchronous operation via Timer_A (see the device-specific data sheet).

  The input is taken from Timer_A (as specified in the data sheet) except GDO0_CFG is set to 0x2D. If GDO1_CFG or GDO2_CFG is set to 0x2D, the signal indicates, when 0, that GDO0 is configured as input (when GDO0_CFG = 0x2D) or that serial TX data is taken from the Timer_A (when GDO0_CFG is not 0x2D); that is, same functionality as on CC1101 for GDO1 and GDO2.
- The register 0x30 PARTNUM - CHIP ID reads as 0 (0x00).
  The register 0x31 VERSION - CHIP ID reads as 6 (0x06).
- The RC_PD bit in register 0x20 WORCTRL is used as a ACLK_PD bit. Set to 1, the ACLK input to the WOR timer is disabled (ACLK is not requested). Reset to 0, the ACLK input to the WOR timer is enabled (ACLK is requested). The SWOR strobe automatically clears this bit and enables the WOR timer.
- The default settings for WORCTRL (now: 0xF8), WOREVT0 (now: 0x00) and WOREVT1 (now: 0x80) changed due to the changed frequency used for the WOR timer (32 kHz vs 26 MHz / 750 or 27 MHz / 750).
- The following signals are added to the GDOx multiplexers:
  - 30 (0x1E): RSSI_VALID
  - 31 (0x1F): RX_TIMEOUT
- In register 0x18 MCSM0, bits 2 and 3 (used as PO_TIMEOUT in CC1101) are reserved; that is, "r0".
- The PA_PD and LNA_PD signals fed to the GDOx multiplexers behave as intended: PA_PD is low only in TX states, LNA_PD is low only in RX states. Otherwise, the signals are 1 (even in SLEEP state).
- The SFTX or SFRX command strobes can be issued in SLEEP state, in addition to IDLE, TX_UNDERFLOW, and RX_OVERFLOW. The FIFOs are not automatically flushed when going to SLEEP state. This allows putting the radio into sleep state and reading previously received data from the RX FIFO or preparing the data in the TX FIFO for the next transmission.
- In register 0x01 IOCFG1 bit 7, GDO_DS is reserved; that is, "r0".
- Registers 0x27 RCCTRL1 and 0x28 RCCTRL0 are reserved; that is, "r0".
- The default setting for IOCFG0.GDO0_CFGx changed from 0x3F (RFCLK / 192) to 0x2E (3-state).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.2 Instruction Set for CC1101-Based Radio Core

Table 25-6 and Table 25-7 list the available instructions to communicate with the CC1101-based radio core. Table 25-6 lists the available command strobes, and Table 25-7 lists all other available instructions. The following nomenclature is used:

- i:[……] gives the value that needs to be written into the radio interface instruction register.
- s:[……] is the value that is returned in the radio interface status register after the instruction was transferred to the radio core.
- A i:[……] in the "Inputs" column always corresponds with a s:[……] in the "Outputs" column.
- [ssss ssss] is the status byte as described in Table 25-8.
- d:[……] in the "Inputs" column is a byte value written into one of the RFA1DIN registers.
- d:[……] in the "Outputs" column is a byte value that can be read via the RFA1DOUT registers.
- A d:[……] in the "Inputs" column always corresponds with a d:[……] in the "Outputs" column.
- [---- ----] is "don't care" data.

The most significant bit of the command strobes in Table 25-6 shown as "x" can be either set to 0 to receive a status byte indicating the number of bytes available in the TX FIFO or it can be set to 1 to receive a status byte indicating the number of bytes available in the RX FIFO. Also see the status byte description in Table 25-8.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-6. CC1101-Based Radio Core Instruction Set – Command Strobes

Instruction Mnemonic	Inputs	Outputs	Description
SRES	i:[x011 0000]	n/a	Command strobe: reset radio core Nothing returned. Accordingly STATIFG is not set.
SFSTXON	i:[x011 0001]	s:[ssss ssss]	Command strobe: enable and calibrate frequency synthesizer (if MCSM0.FS_AUTOCAL = 1). If in RX (with CCA), go to a wait state where only the synthesizer is running (for quick RX or TX turnaround). Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SXOFF	i:[x011 0010]	s:[ssss ssss]	Command strobe: radio core to enter SLEEP state Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SCAL	i:[x011 0011]	s:[ssss ssss]	Command strobe: calibrate frequency synthesizer and turn it off. SCAL can be strobed from IDLE mode without setting manual calibration mode (MCSM0.FS_AUTOCAL = 0). Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SRX	i:[x011 0100]	s:[ssss ssss]	Command strobe: enable RX. Perform calibration first if coming from IDLE and MCSM0.FS_AUTOCAL = 1. If in RX state and CCA is enabled, only go to TX if channel is clear. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
STX	i:[x011 0101]	s:[ssss ssss]	Command strobe: enable TX if in IDLE state and perform calibration first if MCSM0.FS_AUTOCAL = 1. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SIDLE	i:[x011 0110]	s:[ssss ssss]	Command strobe: exit RX or TX, turn off frequency synthesizer, and exit WOR mode, if applicable. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SWOR	i:[x011 1000]	s:[ssss ssss]	Command strobe: start automatic RX polling sequence (Wake-on-Radio, WOR) as described in . Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SPWD	i:[x011 1001]	s:[ssss ssss]	Command strobe: radio core to enter SLEEP state Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SFRX	i:[x011 1010]	s:[ssss ssss]	Command strobe: flush the RX FIFO buffer. Only issue SFRX in IDLE or RX_OVERFLOW states. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SFTX	i:[x011 1011]	s:[ssss ssss]	Command strobe: flush the TX FIFO buffer. Only issue SFTX in IDLE or TX_UNDERFLOW states. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SWORRST	i:[x011 1100]	s:[ssss ssss]	Command strobe: reset WOR timer to Event1 value. Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.
SNOP	i:[x011 1101]	s:[ssss ssss]	Command strobe: no operation. Can be used to read the radio core status byte (see Section 25.3.2.1). Returns status byte with bytes available in TX FIFO when x = 0 and with bytes available in RX FIFO when x=1.

Copyright © 2009–2013, Texas Instruments Incorporated

Table 25-7. CC1101-Based Radio Core Instruction Set

Instruction Mnemonic	Inputs	Outputs	Description
SNGLREGRD	i:[10aa aaaa] d:[---- ----]	s:[ssss ssss] d:[dddd dddd]	Read register at address [a] with a ≤ 0x2E. Returns status [s] and content of register [d]. Status contains bytes available in RX FIFO.
SNGLREGWR	i:[00aa aaaa] d:[dddd dddd]	s:[ssss ssss] d:[ssss ssss]	Write data [d] into register at address [a] with a ≤ 0x2E. Returns status [s]. Status contains bytes available in TX FIFO.
REGRD	i:[11aa aaaa] d:[---- ----] ...	s:[ssss ssss] d:[dddd dddd] ...	Read registers starting at address [a] with a ≤ 0x2E. Returns status [s] and content of registers [d]. Status contains bytes available in RX FIFO.
REGWR	i:[01aa aaaa] d:[dddd dddd] ...	s:[ssss ssss] d:[ssss ssss] ...	Write data [d] into registers starting at address [a] with a ≤ 0x2E. Returns status [s]. Status contains bytes available in TX FIFO.
STATREGRD	i:[11aa aaaa] d:[---- ----]	s:[ssss ssss] d:[dddd dddd]	Read single radio core status register. Returns status [s] and content of radio core status register [a] with 0x30 ≤ a ≤ 0x3D. Status contains bytes available in RX FIFO.
SNGLPATABRD	i:[1011 1110] d:[---- ----]	s:[ssss ssss] d:[dddd dddd]	Read single byte from table with power amplifier settings. Returns status [s] and one byte of PA table [d]. Status contains bytes available in RX FIFO.
SNGLPATABWR	i:[0011 1110] d:[dddd dddd]	s:[ssss ssss] d:[ssss ssss]	Write single byte [d] into table with power amplifier settings. Returns status [s]. Status contains bytes available in TX FIFO.
PATABRD	i:[1111 1110] d:[---- ----] ...	s:[ssss ssss] d:[dddd dddd] ...	Read table with power amplifier settings. Returns status [s] and content of table [d]. Status contains bytes available in RX FIFO.
PATABWR	i:[0111 1110] d:[dddd dddd] ...	s:[ssss ssss] d:[ssss ssss] ...	Write table with power amplifier settings with data [d]. Returns status [s]. Status contains bytes available in TX FIFO.
SNGLRXRD	i:[1011 1111] d:[---- ----]	s:[ssss ssss] d:[dddd dddd]	Read single byte from receive FIFO. Returns status [s] and received byte [d]. Status contains bytes available in RX FIFO.
SNGLTXWR	i:[0011 1111] d:[dddd dddd]	s:[ssss ssss] d:[ssss ssss]	Write single byte [d] into transmit FIFO. Returns status [s]. Status contains bytes available in TX FIFO.
RXFIFORD	i:[1111 1111] d:[---- ----] ...	s:[ssss ssss] d:[dddd dddd] ...	Read data from receive FIFO. Returns status [s] and received bytes [d]. Status contains bytes available in RX FIFO.
TXFIFOWR	i:[0111 1111] d:[dddd dddd] ...	s:[ssss ssss] d:[ssss ssss] ...	Write data bytes [d] into transmit FIFO. Returns status [s]. Status contains bytes available in TX FIFO.

25.3.2.1 Status Byte

With each instruction sent to the radio core, the status is updated and can be read back using the RF1ASTAT registers. Table 25-8 summarizes the information available in the status byte.

Copyright © 2009–2013, Texas Instruments Incorporated

Table 25-8. Radio Core Status Byte Summary

7	6	5	4	3	2	1	0
RF_RDYn	**RF_STATEx**			**FIFO_BYTES_AVAILx**			

RF_RDYn	Bit 7	Radio core ready		
		0	Radio core is ready. Crystal oscillator has stabilized.	
		1	Radio core is not ready. Crystal oscillator not stable.	
RF_STATEx	Bits 6-4	State of the radio core main state machine.		
		000	IDLE	Idle state. Also reported for some transitional states.
		001	RX	Receive mode
		010	TX	Transmit mode
		011	FSTXON	Fast TX ready
		100	CALIBRATE	Frequency synthesizer calibration is running.
		101	SETTLING	PLL is settling.
		110	RX_OVERFLOW	RX FIFO overflow
		111	TX_UNDERFLOW	TX FIFO underflow
FIFO_BYTES_AVAILx	Bits 3-0	Number of bytes available in the RX FIFO or TX FIFO. Depending on the MSB of the instruction, these bits indicate either the number of bytes available for read from the RX FIFO (MSB = 1) or the number of bytes that can be written to the TX FIFO (MSB = 0). When FIFO_BYTES_AVAILx = 1111, then 15 or more bytes are available or free.		

25.3.2.2 Power Amplifier Table Access

The power amplifier table PATABLE is an 8-byte table that defines the PA control settings to be used for each of the eight PA power values (selected by the PA_POWERx bits in radio core register FREND0). The table is written and read from the lowest setting (0) to the highest (7), one byte at a time. An index counter is used to control the access to the table. This counter is incremented each time a byte is read or written to the table using one of the instructions SNGLPATABRD, SNGLPATABWR, PATABRD, and PATABWR. When the highest value is reached the counter wraps around to zero. The counter is reset when any other instruction than a power table instruction is written into the RF1AINSTR register.

25.3.3 Radio-Core Operation

25.3.3.1 Data Rate Programming

The data rate used when transmitting, or the data rate expected in receive, is programmed by the MDMCFG3.DRATE_M and the MDMCFG4.DRATE_E configuration registers. The data rate is given by Equation 12. As the formula shows, the programmed data rate depends on the crystal frequency.

$$R_{DATA} = \frac{(256+DRATE_M) \times 2^{DRATE_E}}{2^{28}} \times f_{XOSC} \quad (12)$$

Equation 13 can be used to find suitable values for a given data rate:

$$DRATE_E = \left\lfloor \log_2 \left(\frac{R_{DATA} \times 2^{20}}{f_{XOSC}} \right) \right\rfloor$$

$$DRATE_M = \frac{R_{DATA} \times 2^{28}}{f_{XOSC} \times 2^{DRATE_E}} - 256 \quad (13)$$

If DRATE_M is rounded to the nearest integer and becomes 256, increment DRATE_E and use DRATE_M = 0.

The data rate can be set from 0.8 kBaud to 500 kBaud with the minimum step as shown in Table 25-9.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-9. Data Rate Step Size

DATA RATE (kBaud)			DATA RATE STEP SIZE (kBaud)
MINIMUM	TYPICAL	MAXIMUM	
0.8	1.2 or 2.4	3.17	0.0062
3.17	4.8	6.35	0.0124
6.35	9.6	12.7	0.0248
12.7	19.6	25.4	0.0496
25.4	38.4	50.8	0.0992
50.8	76.8	101.6	0.1984
101.6	153.6	203.1	0.3967
203.1	250	406.3	0.7935

25.3.3.2 Receiver Channel Filter Bandwidth

To meet different channel width requirements, the receiver channel filter is programmable. The MDMCFG4.CHANBW_E and MDMCFG4.CHANBW_M configuration registers control the receiver channel filter bandwidth, which scales with the crystal oscillator frequency. Equation 14 gives the relation between the register settings and the channel filter bandwidth.

$$BW_{channel} = \frac{f_{XOSC}}{8 \times (4 + CHANBW_M) \times 2^{CHANBW_E}} \quad (14)$$

Table 25-10 lists the supported channel filter bandwidths.

Table 25-10. Channel Filter Bandwidths (kHz) (Assuming a 26-MHz Crystal)

MDMCFG4. CHANBW_M	MDMCFG4.CHANBW_E			
	00	01	10	11
00	812	406	203	102
01	650	325	162	81
10	541	270	135	68
11	464	232	116	58

For best performance, the channel filter bandwidth should be selected so that the signal bandwidth occupies at most 80% of the channel filter bandwidth. The channel center tolerance due to crystal inaccuracy should also be subtracted from the channel filter bandwidth, as shown in the following example.

With the channel filter bandwidth set to 500 kHz, the signal should stay within 80% of 500 kHz, which is 400 kHz. Assuming 915-MHz frequency and ±20-ppm frequency uncertainty for both the transmitting device and the receiving device, the total frequency uncertainty is ±40 ppm of 915 MHz, which is ±37 kHz. If the whole transmitted signal bandwidth is to be received within 400 kHz, the transmitted signal bandwidth should be maximum 400 kHz – (2 × 37 kHz), which is 326 kHz.

By compensating for a frequency offset between the transmitter and the receiver, the filter bandwidth can be reduced and the sensitivity can be improved, see more in *DN005 - CC11xx Sensitivity versus Frequency Offset and Crystal Accuracy* (SWRA122) and in Section 25.3.3.3.1.

25.3.3.3 Demodulator, Symbol Synchronizer, and Data Decision

The radio core contains an advanced and highly configurable demodulator. Channel filtering and frequency offset compensation are performed digitally. To generate the RSSI level (see Section 25.3.3.6.3 for more information) the signal level in the channel is estimated. Data filtering is also included for enhanced performance.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.3.1 Frequency Offset Compensation

The CC1101-based radio has a very fine frequency resolution (refer to the frequency synthesizer characteristics in the device-specific data sheet). This feature can be used to compensate for frequency offset and drift.

When using 2-FSK, 2-GFSK, or MSK modulation, the demodulator compensates for the offset between the transmitter and receiver frequency, within certain limits, by estimating the center of the received data. The frequency offset compensation configuration is controlled from the FOCCFG register. By compensating for a large frequency offset between the transmitter and the receiver, the sensitivity can be improved as shown in *DN005 - CC11xx Sensitivity versus Frequency Offset and Crystal Accuracy* (SWRA122).

The tracking range of the algorithm is selectable as fractions of the channel bandwidth with the FOCCFG.FOC_LIMIT configuration register.

If the FOCCFG.FOC_BS_CS_GATE bit is set, the offset compensator freezes until carrier sense asserts. This may be useful when the radio is in RX for long periods with no traffic, because the algorithm may drift to the boundaries when trying to track noise.

The tracking loop has two gain factors, which affect the settling time and noise sensitivity of the algorithm. FOCCFG.FOC_PRE_K sets the gain before the sync word is detected, and FOCCFG.FOC_POST_K selects the gain after the sync word has been found.

NOTE: Frequency offset compensation is not supported for ASK or OOK modulation.

The estimated frequency offset value is available in the FREQEST status register. This can be used for permanent frequency offset compensation. By writing the value from FREQEST into FSCTRL0.FREQOFF, the frequency synthesizer is automatically adjusted according to the estimated frequency offset. More details regarding this permanent frequency compensation algorithm can be found in *DN015 - Permanent Frequency Offset Compensation* (SWRA159).

25.3.3.3.2 Bit Synchronization

The bit synchronization algorithm extracts the clock from the incoming symbols. The algorithm requires that the expected data rate is programmed as described in Section 25.3.3.1. Resynchronization is performed continuously to adjust for error in the incoming symbol rate.

25.3.3.3.3 Byte Synchronization

Byte synchronization is achieved by a continuous sync word search. The sync word is a 16-bit configurable field (can be repeated to get a 32-bit field) that is automatically inserted at the start of the packet by the modulator in transmit mode. The demodulator uses this field to find the byte boundaries in the stream of bits. The sync word also functions as a system identifier, because only packets with the correct predefined sync word are received if the sync word detection in RX is enabled in register MDMCFG2 (see Sync Word Qualifier). The sync word detector correlates against the user-configured 16- or 32-bit sync word. The correlation threshold can be set to 15/16, 16/16, or 30/32 bits match. The sync word can be further qualified using the preamble quality indicator mechanism described below and/or a carrier sense condition. The sync word is configured through the SYNC1 and SYNC0 registers.

To make false detections of sync words less likely, a mechanism called preamble quality indication (PQI) can be used to qualify the sync word. A threshold value for the preamble quality must be exceeded in order for a detected sync word to be accepted. See Section 25.3.3.6.2 for more details.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.4 Packet Handling Hardware Support

The radio has built-in hardware support for packet oriented radio protocols.

In transmit mode, the packet handler can be configured to add the following elements to the packet stored in the TX FIFO:

- A programmable number of preamble bytes
- A 2-byte synchronization (sync) word. Can be duplicated to give a 4-byte sync word (recommended). It is not possible to insert only preamble or insert only a sync word.
- A CRC checksum computed over the data field.

The recommended setting is 4-byte preamble and 4-byte sync word, except for 500-kBaud data rate, when the recommended preamble length is 8 bytes. In addition, whitening of the data with a PN9 sequence can be implemented on the data field and the optional 2-byte CRC checksum.

In receive mode, the packet handling support deconstructs the data packet by implementing the following (if enabled):

- Preamble detection
- Sync word detection
- CRC computation and CRC check
- One byte address check
- Packet length check (length byte checked against a programmable maximum length)
- Dewhitening

Optionally, two status bytes (see Table 25-11 and Table 25-12) with RSSI value, Link Quality Indication, and CRC status can be appended in the RX FIFO.

Table 25-11. Received Packet Status Byte 1 (First Byte Appended After Data)

BIT	FIELD NAME	DESCRIPTION
7:0	RSSI	RSSI value

Table 25-12. Received Packet Status Byte 2 (Second Byte Appended After Data)

BIT	FIELD NAME	DESCRIPTION
7	CRC_OK	1: CRC for received data OK (or CRC disabled) 0: CRC error in received data
6:0	LQI	Indicating the link quality

NOTE: Register fields that control the packet handling features should be altered only when the radio is in the IDLE state.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.4.1 Data Whitening

From a radio perspective, the ideal over-the-air data are random and dc free. This results in the smoothest power distribution over the occupied bandwidth. This also gives the regulation loops in the receiver uniform operation conditions (no data dependencies).

Real-world data often contain long sequences of zeros and ones. Performance can then be improved by whitening the data before transmitting, and dewhitening the data in the receiver. This can be enabled by setting PKTCTRL0.WHITE_DATA = 1. All data, except the preamble and the sync word, are then XORed with a 9-bit pseudo-random (PN9) sequence before being transmitted, as shown in Figure 25-4. At the receiver end, the data are XORed with the same pseudo-random sequence. This way, the whitening is reversed, and the original data appear in the receiver. The PN9 sequence is initialized to all ones.

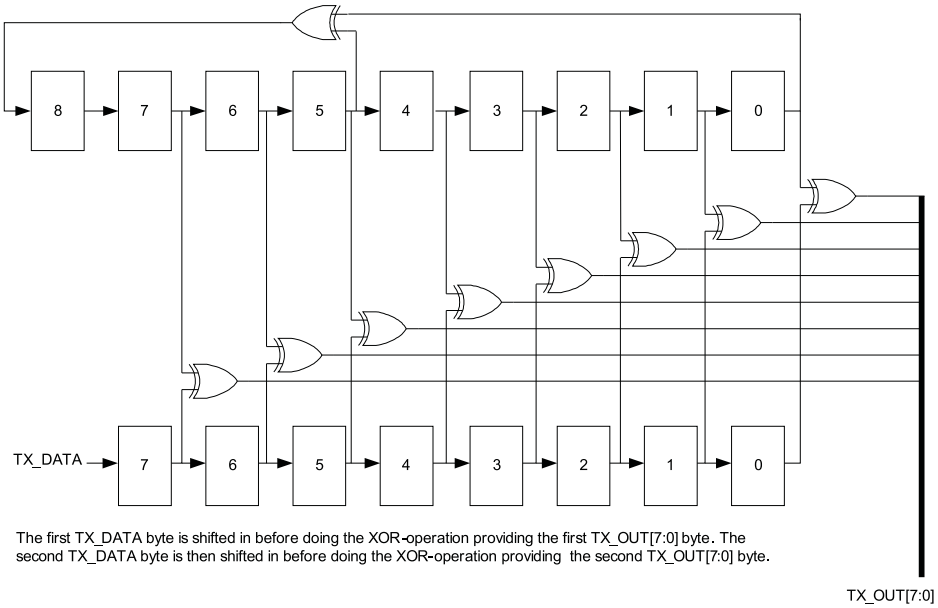


Figure 25-4. Data Whitening in TX Mode

25.3.3.4.2 Packet Format

The format of the data packet can be configured and consists of the following items (see Figure 25-5):

- Preamble
- Synchronization word
- Optional length byte
- Optional address byte
- Payload
- Optional 2-byte CRC

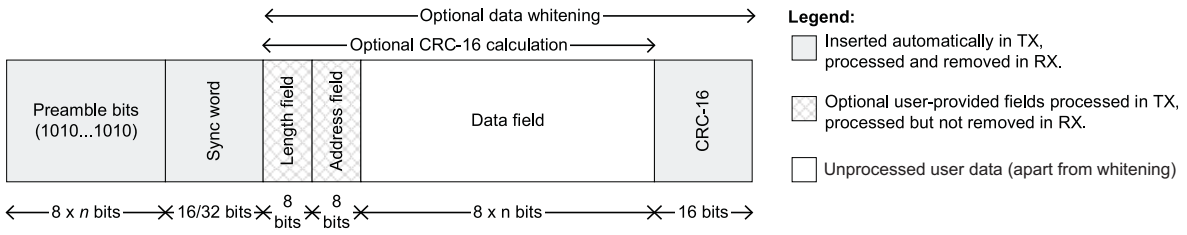


Figure 25-5. Packet Format

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The preamble pattern is an alternating sequence of ones and zeros (10101010…). The minimum length of the preamble is programmed with the MDMCFG1.NUM_PREAMBLE value. When enabling TX, the modulator starts transmitting the preamble. When the programmed number of preamble bytes has been transmitted, the modulator sends the sync word and then data from the TX FIFO if data is available. If the TX FIFO is empty, the modulator continues to send preamble bytes until the first byte is written to the TX FIFO. The modulator then sends the sync word and then the data bytes.

The synchronization word is a two-byte value set in the SYNC1 and SYNC0 registers. The sync word provides byte synchronization of the incoming packet. A one-byte synch word can be emulated by setting the SYNC1 value to the preamble pattern. It is also possible to emulate a 32-bit sync word by setting MDMCFG2.SYNC_MODE to 3 or 7. The sync word is then repeated twice.

The radio supports both constant packet length protocols and variable length protocols. Variable or fixed packet length mode can be used for packets up to 255 bytes. For longer packets, infinite packet length mode must be used.

Fixed packet length mode is selected by setting PKTCTRL0.LENGTH_CONFIG = 0. The desired packet length is set by the PKTLEN register.

In variable packet length mode, PKTCTRL0.LENGTH_CONFIG = 1, the packet length is configured by the first byte after the sync word. The packet length is defined as the payload data, excluding the length byte and the optional CRC. The PKTLEN register is used to set the maximum packet length allowed in RX. Any packet received with a length byte with a value greater than PKTLEN is discarded.

With PKTCTRL0.LENGTH_CONFIG = 2, the packet length is set to infinite, and transmission and reception continues until turned off manually. As described in the next section, this can be used to support packet formats with different length configuration than natively supported. One should make sure that TX mode is not turned off during the transmission of the first half of any byte.

NOTE: The minimum packet length supported (excluding the optional length byte and CRC) is one byte of payload data.

25.3.3.4.2.1 Arbitrary Length Field Configuration

The packet length register, PKTLEN, can be reprogrammed during receive and transmit. In combination with fixed packet length mode (PKTCTRL0.LENGTH_CONFIG = 0) this opens the possibility to have a different length field configuration than supported for variable length packets (in variable packet length mode the length byte is the first byte after the sync word). At the start of reception, the packet length is set to a large value. The CPU reads out enough bytes to interpret the length field in the packet. Then the PKTLEN value is set according to this value. The end of packet occurs when the byte counter in the packet handler is equal to the PKTLEN register. Thus, the CPU must be able to program the correct length, before the internal counter reaches the packet length.

25.3.3.4.2.2 Packet Length Greater Than 255

Also the packet automation control register, PKTCTRL0, can be reprogrammed during TX and RX. This opens the possibility to transmit and receive packets that are longer than 256 bytes and still be able to use the packet handling hardware support. At the start of the packet, the infinite packet length mode (PKTCTRL0.LENGTH_CONFIG = 2) must be active. On the TX side, the PKTLEN register is set to mod(length, 256). On the RX side the CPU reads out enough bytes to interpret the length field in the packet and sets the PKTLEN register to mod(length, 256). When less than 256 bytes remain of the packet, the CPU disables infinite packet length mode and activates fixed packet length mode. When the internal byte counter reaches the PKTLEN value, the transmission or reception ends (the radio enters the state determined by TXOFF_MODE or RXOFF_MODE). Automatic CRC appending and checking can also be used (by setting PKTCTRL0.CRC_EN = 1).

Copyright © 2009–2013, Texas Instruments Incorporated

When, for example, a 600-byte packet is to be transmitted, the CPU should do the following (see also Figure 25-6).

1. Set PKTCTRL0.LENGTH_CONFIG = 2.
2. Preprogram the PKTLEN register to mod(600, 256) = 88.
3. Transmit at least 345 bytes (600 – 255), for example by filling the 64-byte TX FIFO six times (384 bytes transmitted).
4. Set PKTCTRL0.LENGTH_CONFIG = 0.
5. The transmission ends when the packet counter reaches 88. A total of 600 bytes are transmitted.

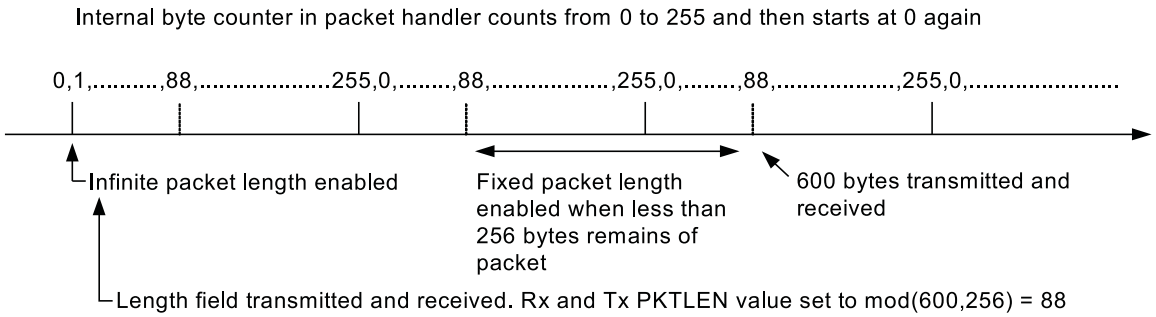


Figure 25-6. Packet Length Greater Than 255

25.3.3.4.3 Packet Filtering in Receive Mode

The radio supports three different types of packet filtering: address filtering, maximum length filtering, and CRC filtering.

25.3.3.4.3.1 Address Filtering

Setting PKTCTRL1.ADR_CHK to any other value than zero enables the packet address filter. The packet handler engine compares the destination address byte in the packet with the programmed node address in the ADDR register and the 0x00 broadcast address when PKTCTRL1.ADR_CHK = 10 or both 0x00 and 0xFF broadcast addresses when PKTCTRL1.ADR_CHK = 11. If the received address matches a valid address, the packet is received and written into the RX FIFO. If the address match fails, the packet is discarded and receive mode restarted (regardless of the MCSM1.RXOFF_MODE setting).

If the received address matches a valid address when using infinite packet length mode and address filtering is enabled, 0xFF is written into the RX FIFO followed by the address byte and then the payload data.

25.3.3.4.3.2 Maximum Length Filtering

In variable packet length mode, PKTCTRL0.LENGTH_CONFIG = 1, the PKTLEN.PACKET_LENGTH register value is used to set the maximum allowed packet length. If the received length byte has a larger value than this, the packet is discarded and receive mode restarted (regardless of the MCSM1.RXOFF_MODE setting).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.4.3.3 CRC Filtering

The filtering of a packet when CRC check fails is enabled by setting PKTCTRL1.CRC_AUTOFLUSH = 1. The CRC auto flush function flushes the entire RX FIFO if the CRC check fails. After auto flushing the RX FIFO, the next state depends on the MCSM1.RXOFF_MODE setting.

When using the auto flush function, the maximum packet length is 63 bytes in variable packet length mode and 64 bytes in fixed packet length mode. Note that the maximum allowed packet length is reduced by two bytes when PKTCTRL1.APPEND_STATUS is enabled, to make room in the RX FIFO for the two status bytes appended at the end of the packet. Because the entire RX FIFO is flushed when the CRC check fails, the previously received packet must be read out of the FIFO before receiving the current packet. The CPU must not read from the current packet until the CRC has been checked as OK.

25.3.3.4.4 Packet Handling in Transmit Mode

The payload that is to be transmitted must be written into the TX FIFO. The first byte written must be the length byte when variable packet length is enabled. The length byte has a value equal to the payload of the packet (including the optional address byte). If address recognition is enabled on the receiver, the second byte written to the TX FIFO must be the address byte. If fixed packet length is enabled, then the first byte written to the TX FIFO should be the address (if the receiver uses address recognition).

The modulator first sends the programmed number of preamble bytes. If data is available in the TX FIFO, the modulator sends the two-byte (optionally four-byte) sync word and then the payload in the TX FIFO. If CRC is enabled, the checksum is calculated over all the data pulled from the TX FIFO and the result is sent as two extra bytes following the payload data. If the TX FIFO runs empty before the complete packet has been transmitted, the radio enters TX_UNDERFLOW state. The only way to exit this state is by issuing an SFTX strobe. Writing to the TX FIFO after it has underflowed does not restart TX mode.

If whitening is enabled, everything following the sync words is whitened. Whitening is enabled by setting PKTCTRL0.WHITE_DATA = 1.

25.3.3.4.5 Packet Handling in Receive Mode

In receive mode, the demodulator and packet handler searches for a valid preamble and the sync word. When found, the demodulator has obtained both bit and byte synchronism and receives the first payload byte.

If whitening is enabled, the data is dewhitened at this stage.

When variable packet length mode is enabled, the first byte is the length byte. The packet handler stores this value as the packet length and receives the number of bytes indicated by the length byte. If fixed packet length mode is used, the packet handler accepts the programmed number of bytes.

Next, the packet handler optionally checks the address and only continues the reception if the address matches. If automatic CRC check is enabled, the packet handler computes CRC and matches it with the appended CRC checksum.

At the end of the payload, the packet handler optionally writes two extra packet status bytes (see Table 25-11 and Table 25-12) that contain CRC status, link quality indication, and RSSI value.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.4.6 Packet Handling in Firmware

When implementing a packet oriented radio protocol in firmware, the CPU needs to know when a packet has been received or transmitted. Additionally, for packets longer than 64 bytes the RX FIFO needs to be read while in RX and the TX FIFO needs to be refilled while in TX. This means that the CPU needs to know the number of bytes that can be read from or written to the RX FIFO and TX FIFO respectively. There are two possible solutions to get the necessary status information:

Interrupt Driven Solution

In both RX and TX one can use the RFIFG9 radio core interrupts to request an interrupt when a sync word has been received or transmitted or when a complete packet has been received or transmitted (alternatively one of the programmable radio core interrupt flags RFIFG0, RFIFG1, or RFIFG2 can be set to GDOx_CFG = 0x06). In addition, there are two radio core interrupts associated with the RX FIFO (RFIFG3 and RFIFG4 or alternatively the programmable radio core interrupt flags with GDOx_CFG = 0x00 and GDOx_CFG = 0x01) and two that are associated with the TX FIFO (RFIFG5 and RFIFG6 or alternatively the programmable radio core interrupt flags with GDOx_CFG = 0x02 and GDOx_CFG = 0x03) that can be used as interrupt sources to provide information on how many bytes are in the RX FIFO and TX FIFO respectively. Using the interrupt flags is the recommended approach.

Status Register Polling Solution

The PKTSTATUS register can be polled to get information about the current GDO2 and GDO0 values respectively. The RXBYTES and TXBYTES registers can be polled to get information about the number of bytes in the RX FIFO and TX FIFO respectively. Alternatively, the number of bytes in the RX FIFO and TX FIFO can be read from the status byte returned each time an instruction is sent to the radio core.

25.3.3.5 Modulation Formats

The radio supports amplitude, frequency, and phase shift modulation formats. The desired modulation format is set in the MDMCFG2.MOD_FORMAT register.

Optionally, the data stream can be Manchester coded by the modulator and decoded by the demodulator. This option is enabled by setting MDMCFG2.MANCHESTER_EN = 1.

25.3.3.5.1 Frequency Shift Keying

The radio has the possibility to use Gaussian shaped 2-FSK (2-GFSK). The 2-FSK signal is then shaped by a Gaussian filter with BT = 0.5, producing a 2-GFSK modulated signal. This spectrum-shaping feature improves adjacent channel power (ACP) and occupied bandwidth.

In ‘true’ 2-FSK systems with abrupt frequency shifting, the spectrum is inherently broad. By making the frequency shift ‘softer’, the spectrum can be made significantly narrower. Thus, higher data rates can be transmitted in the same bandwidth using 2-GFSK.

When 2-FSK or 2-GFSK modulation is used, the DEVIATN register specifies the expected frequency deviation of incoming signals in RX and should be the same as the TX deviation for demodulation to be performed reliably and robustly.

The frequency deviation is programmed with the DEVIATION_M and DEVIATION_E values in the DEVIATN register. The value has an exponent and mantissa form, and the deviation is given in Equation 15:

$$f_{dev} = \frac{f_{xosc}}{2^{17}} \times (8 + \text{DEVIATION_M}) \times 2^{\text{DEVIATION_E}} \quad (15)$$

The symbol encoding is shown in Table 25-13.

Table 25-13. Symbol Encoding for 2-FSK and 2-GFSK Modulation

FORMAT	SYMBOL	CODING
2-FSK and 2-GFSK	0	– Deviation
	1	+ Deviation

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.5.2 Minimum Shift Keying

When using MSK [1] , the complete transmission (preamble, sync word, and payload) is MSK modulated.

Phase shifts are performed with a constant transition time. The fraction of a symbol period used to change the phase can be modified with the DEVIATN.DEVIATION_M setting. This is equivalent to changing the shaping of the symbol. The DEVIATN register setting has no effect in RX when using MSK.

When using MSK, Manchester encoding and decoding should be disabled by setting MDMCFG2.MANCHESTER_EN = 0.

The implemented MSK modulation format inverts the sync word and data compared to, for example, signal generators.

25.3.3.5.3 Amplitude Modulation

The radio supports two different forms of amplitude modulation: on-off keying (OOK) and amplitude shift keying (ASK).

OOK modulation simply turns on or off the PA to modulate 1 and 0, respectively.

The ASK variant allows programming of the modulation depth (the difference between 1 and 0) and shaping of the pulse amplitude. Pulse shaping produces a more bandwidth-constrained output spectrum.

When using OOK or ASK, the AGC settings from the SmartRF® Studio preferred FSK and MSK settings are not optimum. application note *DN022 - CC11xx OOK/ASK Register Settings* (SWRA215) provides guidelines on how to find optimum OOK or ASK settings from the preferred settings in SmartRF Studio. The DEVIATN register setting has no effect in either TX or RX when using OOK or ASK.

25.3.3.6 Received Signal Qualifiers and Link Quality Information

The radio has qualifiers that can be used to increase the likelihood that a valid sync word is detected:

- Sync Word Qualifier
- Preamble Quality Threshold
- RSSI
- Carrier Sense
- Clear Channel Assessment
- Link Quality Indicator

25.3.3.6.1 Sync Word Qualifier

If sync word detection in RX is enabled in register MDMCFG2, the radio does not start filling the RX FIFO and performing the packet filtering described in Section 25.3.3.4.3.3 before a valid sync word has been detected. The sync word qualifier mode is set by MDMCFG2.SYNC_MODE and is summarized in Table 25-14. Carrier sense is described in Section 25.3.3.6.4.

[1] Identical to offset QPSK with half-sine shaping (data coding may differ)

Table 25-14. Sync Word Qualifier Mode

MDMCFG2.SYNC_MODE	SYNC WORD QUALIFIER MODE
000	No preamble or sync
001	15/16 sync word bits detected
010	16/16 sync word bits detected
011	30/32 sync word bits detected
100	No preamble or sync, carrier sense above threshold
101	15/16 + carrier sense above threshold
110	16/16 + carrier sense above threshold
111	30/32 + carrier sense above threshold

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.6.2 Preamble Quality Threshold (PQT)

The preamble quality threshold (PQT) sync-word qualifier adds the requirement that the received sync word must be preceded with a preamble with a quality above the programmed threshold.

Another use of the preamble quality threshold is as a qualifier for the optional RX termination timer. See RX Termination Timer for details.

The preamble quality estimator increases an internal counter by one each time a bit is received that is different from the previous bit, and decreases the counter by 8 each time a bit is received that is the same as the last bit. The threshold is configured with the register field PKTCTRL1.PQT. A threshold of 4 × PQT for this counter is used to gate sync word detection. By setting the value to zero, the preamble quality qualifier of the synch word is disabled.

A preamble quality reached signal can be observed using RFIFG11 or using one of the GDO signals by setting IOCFGx.GDOx_CFG = 8. It is also possible to determine if preamble quality is reached by checking the PQT_REACHED bit in the PKTSTATUS register. This signal or bit asserts when the received signal exceeds the PQT.

25.3.3.6.3 Received Signal Strength Indicator (RSSI)

The RSSI value is an estimate of the signal power level in the chosen channel. This value is based on the current gain setting in the RX chain and the measured signal level in the channel.

In RX mode, the RSSI value can be read continuously from the RSSI status register until the demodulator detects a sync word (when sync word detection is enabled). At that point the RSSI readout value is frozen until the next time the radio enters the RX state.

NOTE: It takes some time from the radio enters RX mode until a valid RSSI value is present in the RSSI register. Please refer to *DN505 - RSSI interpretation and timing* (SWRA114) for details on how the RSSI response time can be estimated.

The RSSI value is in dBm with ½-dB resolution. The RSSI update rate, f_{RSSI}, depends on the receiver filter bandwidth ($BW_{channel}$ defined in Section 25.3.3.2) and AGCCTRL0.FILTER_LENGTH.

$$f_{RSSI} = \frac{2 \times BW_{channel}}{8 \times 2^{FILTER_LENGTH}} \quad (16)$$

If PKTCTRL1.APPEND_STATUS is enabled the last RSSI value of the packet is automatically added to the first byte appended after the payload.

The RSSI value read from the RSSI status register is a 2s complement number. The following procedure can be used to convert the RSSI reading to an absolute power level (RSSI_dBm).

1. Read the RSSI status register
2. Convert the reading from a hexadecimal number to a decimal number (RSSI_dec)
3. If RSSI_dec ≥ 128, then RSSI_dBm = (RSSI_dec – 256) / 2 – RSSI_offset
4. If RSSI_dec < 128, then RSSI_dBm = (RSSI_dec) / 2 – RSSI_offset

See the device-specific data sheet for typical RSSI offset values and for typical RSSI values vs input power levels at various frequencies.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.6.4 Carrier Sense (CS)

Carrier sense (CS) is used as a sync word qualifier and for CCA and can be asserted based on two conditions, which can be individually adjusted:

- CS is asserted when the RSSI is above a programmable absolute threshold and deasserted when RSSI is below the same threshold (with hysteresis).
- CS is asserted when the RSSI has increased with a programmable number of dB from one RSSI sample to the next and deasserted when RSSI has decreased with the same number of dB. This setting is not dependent on the absolute signal level and is thus useful to detect signals in environments with time varying noise floor.

Carrier sense can be used as a sync word qualifier that requires the signal level to be higher than the threshold for a sync word search to be performed. The signal can also be observed using RFIFG13 or using one of the GDO signals by setting IOCFGx.GDOx_CFG = 14 and in the status register bit PKTSTATUS.CS.

Other uses of carrier sense include the TX-if-CCA function (see Clear Channel Assessment (CCA)) and the optional fast RX termination (see RX Termination Timer).

CS can be used to avoid interference from other RF sources in the ISM bands.

25.3.3.6.4.1 CS Absolute Threshold

The absolute threshold related to the RSSI value depends on the following register fields:

- AGCCTRL2.MAX_LNA_GAIN
- AGCCTRL2.MAX_DVGA_GAIN
- AGCCTRL1.CARRIER_SENSE_ABS_THR
- AGCCTRL2.MAGN_TARGET

For a given AGCCTRL2.MAX_LNA_GAIN and AGCCTRL2.MAX_DVGA_GAIN setting the absolute threshold can be adjusted ±7 dB in steps of 1 dB using CARRIER_SENSE_ABS_THR.

The MAGN_TARGET setting is a compromise between blocker tolerance or selectivity and sensitivity. The value sets the desired signal level in the channel into the demodulator. Increasing this value reduces the headroom for blockers, and therefore close-in selectivity.

It is strongly recommended to use SmartRF Studio to generate the correct MAGN_TARGET setting.

Table 25-15 and Table 25-16 show the typical RSSI readout values at the CS threshold at 2.4 kBaud and 250 kBaud data rate respectively. The default CARRIER_SENSE_ABS_THR = 0 (0 dB) and MAGN_TARGET = 3 (33 dB) have been used.

For other data rates the user must generate similar tables to find the CS absolute threshold.

Table 25-15. Typical RSSI Value in dBm at CS Threshold With Default MAGN_TARGET at 2.4 kBaud, 868 MHz

		MAX_DVGA_GAIN[1:0]			
		00	**01**	**10**	**11**
MAX_LNA_GAIN[2:0]	**000**	–97.5	–91.5	–85.5	–79.5
	001	–94	–88	–82.5	–76
	010	–90.5	–84.5	–78.5	–72.5
	011	–88	–82.5	–76.5	–70.5
	100	–85.5	–80	–73.5	–68
	101	–84	–78	–72	–66
	110	–82	–76	–70	–64
	111	–79	–73.5	–67	–61

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-16. Typical RSSI Value in dBm at CS Threshold With Default MAGN_TARGET at 250 kBaud, 868 MHz

		MAX_DVGA_GAIN[1:0]			
		00	**01**	**10**	**11**
MAX_LNA_GAIN[2:0]	**000**	–90.5	–84.5	–78.5	–72.5
	001	–88	–82	–76	–70
	010	–84.5	–78.5	–72	–66
	011	–82.5	–76.5	–70	–64
	100	–80.5	–74.5	–68	–62
	101	–78	–72	–66	–60
	110	–76.5	–70	–64	–58
	111	–74.5	–68	–62	–56

If the threshold is set high (that is, only strong signals are wanted), the threshold should be adjusted upwards by first reducing the MAX_LNA_GAIN value and then the MAX_DVGA_GAIN value. This reduces power consumption in the receiver front end, because the highest gain settings are avoided.

25.3.3.6.4.2 CS Relative Threshold

The relative threshold detects sudden changes in the measured signal level. This setting is not dependent on the absolute signal level and is thus useful to detect signals in environments with a time varying noise floor. The register field AGCCTRL1.CARRIER_SENSE_REL_THR is used to enable or disable relative CS, and to select threshold of 6-dB, 10-dB, or 14-dB RSSI change.

25.3.3.6.5 Clear Channel Assessment (CCA)

The CCA is used to indicate if the current channel is free or busy. The current CCA state can be observed by using RFIFG12 or using the GDO signals by setting IOCFGx.GDOx_ CFG = 0x09.

MCSM1.CCA_MODE selects the mode to use when determining CCA.

When the STX or SFSTXON command strobe is given while the radio is in the RX state, the TX or FSTXON state is only entered if the clear channel requirements are fulfilled. The radio otherwise remains in RX (if the channel becomes available, the radio does not enter TX or FSTXON state before a new strobe command is sent). This feature is called TX-if-CCA. Four CCA requirements can be programmed:

- Always (CCA disabled, always goes to TX)
- If RSSI is below threshold
- Unless currently receiving a packet
- Both the above (RSSI below threshold and not currently receiving a packet)

25.3.3.6.6 Link Quality Indicator (LQI)

The Link Quality Indicator is a metric of the current quality of the received signal. If PKTCTRL1.APPEND_STATUS is enabled, the value is automatically added to the last byte appended after the payload. The value can also be read from the LQI status register. The LQI gives an estimate of how easily a received signal can be demodulated by accumulating the magnitude of the error between ideal constellations and the received signal over the 64 symbols immediately following the sync word. LQI is best used as a relative measurement of the link quality (a low value indicates a better link than what a high value does), since the value is dependent on the modulation format.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.7 Radio Control

The radio has a built-in state machine that is used to switch between different operational states (modes). The change of state is done either by using command strobes or by internal events such as TX FIFO underflow.

The complete radio control state diagram is shown in Figure 25-7. The numbers refer to the state number readable in the MARCSTATE status register. This register is primarily for test purposes.

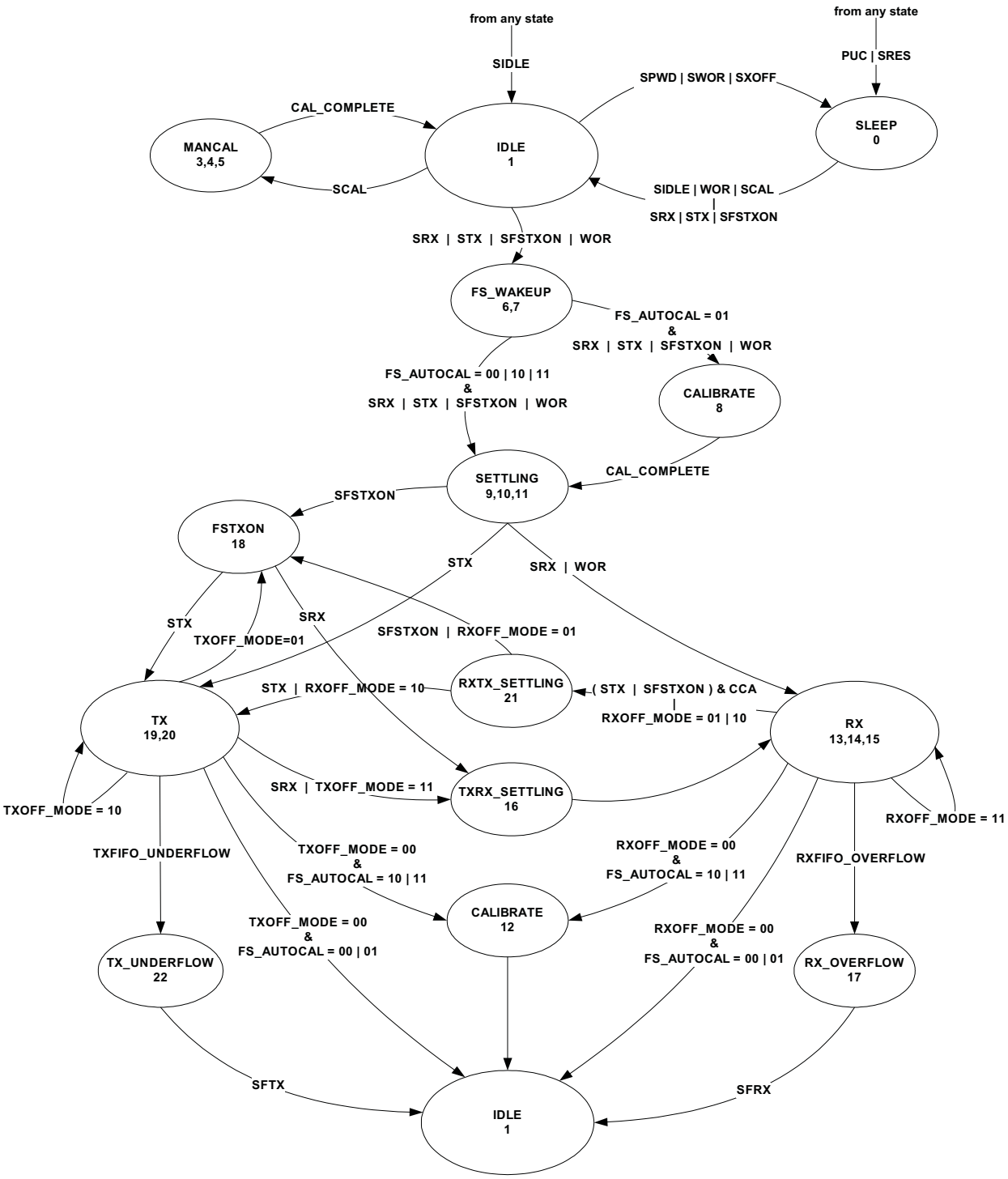


Figure 25-7. Complete Radio-Control State Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.7.0.1 Manual Reset

Use the SRES command strobe to globally reset the radio core . By issuing this strobe, all internal registers are reset to their default states and the radio core enters the SLEEP state.

25.3.3.7.1 Crystal Control

The RF XT2 crystal oscillator is either automatically controlled or always on, if MCSM0.XOSC_FORCE_ON=1 or XT2OFF=0.

In the automatic mode, the oscillator is turned off if the SPWD, SWOR or SXOFF command strobe is issued. The state machine then goes to SLEEP. This can be done only from the IDLE state. When the radio state machine transitions from the SLEEP state to an active state the RF XT2 oscillator is turned back on. A maximum of 810 us can pass before the RF XT2 oscillator provides a stable clock source - mainly depending on the crystal ESR and load capacitances. To safely transition from SLEEP state (with the oscillator disabled) to an active state using a strobe command the following flow should be used:

- Issue the SIDLE strobe command.
- The CPU must remain active until the RF_RDYn signal transitions to 0 – this can be observed best on the programmable RFINx signals; for example, the GDO2 signal fed to RFIN2 is programmed by default to RF_RDYn.
- Delay any further interaction with the radio until the oscillator is stable - best with a delay loop that delays further program execution by 810us.

If the RF XT2 oscillator is forced on (with MCSM0.XOSC_FORCE_ON=1 or XT2OFF=0), the crystal stays on, even in the SLEEP state.

After a reset the crystal oscillator is off because the state machine is in the SLEEP state, MCSM0.XOSC_FORCE_ON=0 and XT2OFF=1.

25.3.3.7.2 Active Modes

The radio has two active modes: receive and transmit. These modes are activated directly by the CPU by using the SRX and STX command strobes.

The frequency synthesizer must be calibrated regularly. There is one manual calibration option (using the SCAL strobe), and three automatic calibration options, controlled by the MCSM0.FS_AUTOCAL setting:

- Calibrate when going from IDLE to either RX or TX (or FSTXON)
- Calibrate when going from either RX or TX to IDLE automatically
- Calibrate every fourth time when going from either RX or TX to IDLE automatically

If the radio goes from TX or RX to IDLE by issuing an SIDLE strobe, calibration is not performed. The calibration takes a constant number of crystal oscillator cycles (see Table 25-17 for timing details).

When RX is activated, the radio remains in receive mode until a packet is successfully received or the RX termination timer expires (see RX Termination Timer). The probability that a false sync word is detected can be reduced by using PQT, CS, maximum sync word length, and sync word qualifier mode as described in Section 25.3.3.6.

After a packet is successfully received, the radio controller goes to the state indicated by the MCSM1.RXOFF_MODE setting. The possible destinations are:

- IDLE
- FSTXON: Frequency synthesizer on and ready at the TX frequency. Activate TX with STX.
- TX: Start sending preamble
- RX: Start search for a new packet

NOTE: When MCSM1.RXOFF_MODE=11 and a packet has been received, it takes some time before a valid RSSI value is present in the RSSI register again even if the radio never exited RX mode. This time is the same as the RSSI response time discussed in *DN505 - RSSI interpretation and timing* (SWRA114).

Copyright © 2009–2013, Texas Instruments Incorporated

Similarly, when TX is active the radio remains in the TX state until the current packet has been successfully transmitted. Then the state changes as indicated by the MCSM1.TXOFF_MODE setting. The possible destinations are the same as for RX.

The CPU can manually change the state from RX to TX and vice versa by using the command strobes. If the radio controller is currently in transmit and the SRX strobe is used, the current transmission is ended and the transition to RX is done.

If the radio controller is in RX when the STX or SFSTXON command strobes are used, the TX-if-CCA function is used. If the channel is not clear, the radio remains in RX. The MCSM1.CCA_MODE setting controls the conditions for clear channel assessment (see Clear Channel Assessment (CCA) for details).

The SIDLE command strobe can always be used to force the radio controller to go to the IDLE state.

25.3.3.7.3 Wake On Radio (WOR) Timer

The Wake on Radio (WOR) timer functionality can be used to wake up the radio core periodically and listen for incoming packets.

When the SWOR strobe command is sent the radio core goes to the SLEEP state. The WOR timer function can be enabled with WORCTRL.ACLK_PD = 0 before the SWOR strobe is used otherwise the SWOR strobe automatically clears it and the WOR timer starts. The WOR timer has two events, Event 0 and Event 1. Event 1 follows Event 0 after a programmed timeout. The WOR timer generates interrupt signals on Event 0 and Event 1 setting RFIFG14 and RFIFG15, respectively.

The time between two consecutive Event 0 is programmed with a mantissa value given by WOREVT1.EVENT0 and WOREVT0.EVENT0, and an exponent value set by WORCTRL.WOR_RES. The equation is:

$$t_{Event0} = 1 / f_{ACLK} \times EVENT0 \times 2^{5 \times WOR_RES} \quad (17)$$

The Event 1 timeout is programmed with WORCTRL.EVENT1. Figure 25-8 shows the timing relationship between Event 0 timeout and Event 1 timeout.

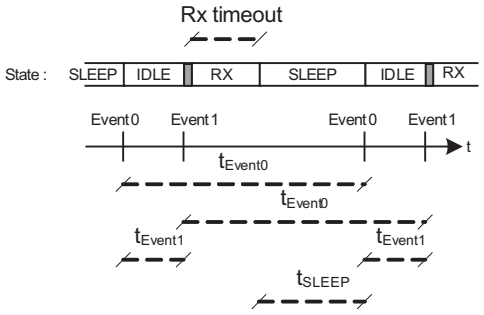


Figure 25-8. Event 0 and Event 1 Relationship

The time from entering the SLEEP state until the next Event 0 (t_{SLEEP} in Figure 25-8) should be larger than 11.72 ms when f_{ACLK} = 32.768kHz. If t_{SLEEP} is less than 11.72 ms, there is a chance that the consecutive Event 0 can occur (1 / f_{ACLK}) × 128 seconds too early.

The program flow to use the WOR feature is as follows:

- The RFIFG14 interrupt flag can be used to wake the CPU on WOR timer Event 0. Then the CPU must strobe SIDLE to wake the radio core.
- The CPU must remain active until the RF_RDYn signal transitions to 0 – this can be observed best on the programmable RFINx signals; for example, the GDO2 signal fed to RFIN2 is programmed by default to RF_RDYn.
- On WOR timer Event 1 the radio automatically transistions into the RX state.
- After a programmable time in RX, the radio returns to the SLEEP state, unless a packet is received. See Figure 25-8 and RX Termination Timer for details on how the timeout works.
- The radio core can signal the CPU that a packet has been received by using the radio core interrupts. If a packet is received, the MCSM1.RXOFF_MODE determines the behavior at the end of the received

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

packet. When the CPU has acknowledged that a packet was received, it can put the radio back into SLEEP with the SWOR strobe from the IDLE state. The RX FIFO can be read during SLEEP.

- Set the radio core into the SLEEP or IDLE state to exit WOR mode and disable the WOR timer with WORCTRL.ACLK_PD = 1.

If the WOR feature is used together with low-power mode LPM3 the guidelines given in section Section 25.2.4 "Using Radio with Low-Power Modes" must be followed.

25.3.3.7.4 Timing

The radio controller controls most of the timing, such as synthesizer calibration, PLL lock time, and RX or TX turnaround times. Timing from IDLE to RX and IDLE to TX is constant, dependent on the auto calibration setting. RX-to-TX and TX-to-RX turnaround times are constant. The calibration time is constant 18739 clock periods. Table 25-17 shows timing in crystal clock cycles for key state transitions. RF XT2 oscillator start-up times are variable.

Note that in a frequency hopping spread spectrum or a multi-channel protocol, the calibration time can be reduced from 721 µs to approximately 150 µs (see Section 25.3.4.2).

Table 25-17. State Transition Timing

DESCRIPTION	RF XT2 PERIODS	26-MHz CRYSTAL
IDLE to RX, no calibration	2298	88.4 µs
IDLE to RX, with calibration	~21037	809 µs
IDLE to TX or FSTXON, no calibration	2298	88.4 µs
IDLE to TX or FSTXON, with calibration	~21037	809 µs
TX to RX switch	560	21.5 µs
RX to TX switch	250	9.6 µs
RX or TX to IDLE, no calibration	2	0.1 µs
RX or TX to IDLE, with calibration	~18739	721 µs
Manual calibration	~18739	721 µs

25.3.3.7.5 RX Termination Timer

The radio has optional functions for automatic termination of RX after a programmable time. The main use for this functionality is wake-on-radio (WOR), but it may be useful for other applications. The termination timer starts when in RX state. The timeout is programmable with the MCSM2.RX_TIME setting. When the timer expires, the radio controller checks the condition for staying in RX. If the condition is not met, RX terminates.

The programmable conditions are:

- MCSM2.RX_TIME_QUAL = 0

  Continue receive if sync word has been found
- MCSM2.RX_TIME_QUAL = 1

  Continue receive if sync word has been found or preamble quality is above threshold (PQT)

If the system can expect the transmission to have started when enabling the receiver, the MCSM2.RX_TIME_RSSI function can be used. The radio controller then terminates RX if the first valid carrier sense sample indicates no carrier (RSSI below threshold) (see Section 25.3.3.6.4 for details on Carrier Sense).

For ASK or OOK modulation, lack of carrier sense is only considered valid after eight symbol periods. Thus, the MCSM2.RX_TIME_RSSI function can be used in ASK or OOK mode when the distance between "1" symbols is 8 or less.

Copyright © 2009–2013, Texas Instruments Incorporated

If RX terminates due to no carrier sense when the MCSM2.RX_TIME_RSSI function is used, or if no sync word was found when using the MCSM2.RX_TIME timeout function, the radio goes back to IDLE if WOR is disabled and back to SLEEP if WOR is enabled. Otherwise, the MCSM1.RXOFF_MODE setting determines the state to go to when RX ends. This means that the radio does not automatically go back to SLEEP once a sync word has been received. It is therefore recommended to always wake up the CPU on sync word detection when using WOR mode (RFIFG9 or a GDO signal with GDOx_CFG=6).

25.3.3.8 Data FIFO

The radio contains two 64 byte FIFOs, one for received data and one for data to be transmitted. The instructions SNGLRXRD and RXFIFORD can be used to read from the RX FIFO and the instructions SNGLTXWR and TXFIFOWR can be used write to the TX FIFO.

The FIFO controller detects overflow in the RX FIFO and underflow in the TX FIFO.

When writing to the TX FIFO, it is the responsibility of the application software to avoid TX FIFO overflow. A TX FIFO overflow results in an error in the TX FIFO content.

Likewise, when reading the RX FIFO, the software must avoid reading the RX FIFO past its empty value, because an RX FIFO underflow results in an error in the data read out of the RX FIFO.

The status byte that is return each time an instruction is sent to the radio core contains the fill grade of the RX FIFO if the access is a read operation and the fill grade of the TX FIFO if the access is a write operation.

The number of bytes in the RX FIFO and TX FIFO can be read from the status registers RXBYTES.NUM_RXBYTES and TXBYTES.NUM_TXBYTES respectively. If a received data byte is written to the RX FIFO at the exact same time as the last byte in the RX FIFO is read, the RX FIFO pointer is not properly updated and the last read byte is duplicated. To avoid this problem one should never empty the RX FIFO before the last byte of the packet is received.

For packet lengths less than 64 bytes it is recommended to wait until the complete packet has been received before reading the RX FIFO.

If the packet length is larger than 64 bytes the CPU must determine how many bytes can be read from the RX FIFO (RXBYTES.NUM_RXBYTES-1) and the following software routine can be used:

1. Read RXBYTES.NUM_RXBYTES repeatedly at a rate ensured to be at least twice that at which RF bytes are received until the same value is returned twice. Store value in n.
2. If n < # of bytes remaining in packet, read n – 1 bytes from the RX FIFO.
3. Repeat steps 1 and 2 until n = # of bytes remaining in packet.
4. Read the remaining bytes from the RX FIFO.

The 4-bit FIFOTHR.FIFO_THR setting is used to program threshold points in the FIFOs. Table 25-18 lists the 16 FIFO_THR settings and the corresponding thresholds for the RX and TX FIFOs. The threshold value is coded in opposite directions for the RX FIFO and TX FIFO. This gives equal margin to the overflow and underflow conditions when the threshold is reached.

A signal asserts when the number of bytes in the FIFO is equal to or higher than the programmed threshold. This signal can be observed using the GDOx signals as well as the corresponding radio core interrupt flags RFIFGx.

Figure 25-10 shows the number of bytes in both the RX FIFO and TX FIFO when the threshold signal toggles, in the case of FIFO_THR = 13. Figure 25-9 shows the signal as the respective FIFO is filled above the threshold, and then drained below.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

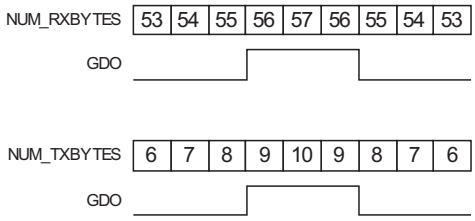


Figure 25-9. FIFO_THR = 13 vs Number of Bytes in FIFO (GDOx_CFG = 0x00 in RX and GDOx_CFG = 0x02 in TX)

Table 25-18. FIFO_THR Settings and the Corresponding FIFO Thresholds

FIFO_THR	BYTES IN TX FIFO	BYTES IN RX FIFO
0 (0000)	61	4
1 (0001)	57	8
2 (0010)	53	12
3 (0011)	49	16
4 (0100)	45	20
5 (0101)	41	24
6 (0110)	37	28
7 (0111)	33	32
8 (1000)	29	36
9 (1001)	25	40
10 (1010)	21	44
11 (1011)	17	48
12 (1100)	13	52
13 (1101)	9	56
14 (1110)	5	60
15 (1111)	1	64

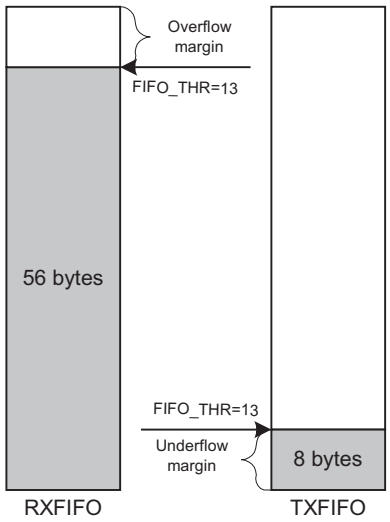


Figure 25-10. Example of FIFOs at Threshold

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.3.9 Frequency Programming

The frequency programming is designed to minimize the programming needed in a channel-oriented system.

To set up a system with channel numbers, the desired channel spacing is programmed with the MDMCFG0.CHANSPC_M and MDMCFG1.CHANSPC_E registers. The channel spacing registers are mantissa and exponent respectively.

The base or start frequency is set by the 24-bit frequency word located in the FREQ2, FREQ1, and FREQ0 registers. This word is typically set to the center of the lowest channel frequency that is to be used.

The desired channel number is programmed with the 8-bit channel number register, CHANNR.CHAN, which is multiplied by the channel offset. The resultant carrier frequency is given by:

$$f_{carrier} = \frac{f_{XOSC}}{2^{16}} \times \left(FREQ + CHAN \times \left(\left(256 + CHANSPC_M\right) \times 2^{CHANSPC_E-2}\right)\right) \quad (18)$$

With a 26-MHz crystal the maximum channel spacing is 405 kHz. To get, for example, 1-MHz channel spacing one solution is to use 333-kHz channel spacing and select each third channel in CHANNR.CHAN.

The preferred IF frequency is programmed with the FSCTRL1.FREQ_IF register. The IF frequency is given by:

$$f_{IF} = \frac{f_{XOSC}}{2^{10}} \times FREQ_IF \quad (19)$$

NOTE: The SmartRF Studio software automatically calculates the optimum FSCTRL1.FREQ_IF register setting based on channel spacing and channel filter bandwidth.

If any frequency programming register is altered when the frequency synthesizer is running, the synthesizer may give an undesired response. Hence, the frequency programming should only be updated when the radio is in the IDLE state.

25.3.3.10 VCO

The VCO is completely integrated on-chip.

25.3.3.10.1 VCO and PLL Self-Calibration

The VCO characteristics vary with temperature and supply voltage changes, as well as the desired operating frequency. To ensure reliable operation, the radio includes a frequency synthesizer self-calibration circuitry. This calibration should be done regularly, and must be performed after turning on power and before using a new frequency (or channel). The number of RF XT2 oscillator cycles for completing the PLL calibration is given in Table 25-17.

The calibration can be initiated automatically or manually. The synthesizer can be automatically calibrated each time the synthesizer is turned on, or each time the synthesizer is turned off automatically. This is configured with the MCSM0.FS_AUTOCAL register setting. In manual mode, the calibration is initiated when the SCAL command strobe is activated in the IDLE mode.

The calibration values are maintained in SLEEP mode, so the calibration is still valid after waking up from SLEEP mode unless supply voltage or temperature has changed significantly.

To check that the PLL is in lock, the user can program register IOCFGx.GDOx_CFG to 0x0A and use the lock detector output as an interrupt source for the RF interrupt flags RFIFGx with x = 0,1, or 2. A positive transition means that the PLL is in lock. As an alternative the user can read register FSCAL1. The PLL is in lock if the register content is different from 0x3F (see also the errata notes). For more robust operation the source code could include a check so that the PLL is re-calibrated until PLL lock is achieved if the PLL does not lock the first time.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.11 Output Power Programming

The RF output power level has two levels of programmability, as illustrated in Figure 25-11. Firstly, the special PATABLE register can hold up to eight user selected output power settings. Secondly, the 3-bit FREND0.PA_POWER value selects the PATABLE entry to use. This two-level functionality provides flexible PA power ramp up and ramp down at the start and end of transmission, as well as ASK modulation shaping. All the PA power settings in the PATABLE from index 0 up to the FREND0.PA_POWER value are used.

The power ramping at the start and at the end of a packet can be turned off by setting FREND0.PA_POWER to zero and then program the desired output power to index 0 in the PATABLE.

If OOK modulation is used, the logic 0 and logic 1 power levels shall be programmed to index 0 and 1 respectively.

Refer to the device-specific data sheet for recommended PATABLE settings for various output levels and frequency bands. Note that PA settings from 0x61 to 0x6F are not allowed. The data sheet also list output power and current consumption for the default PATABLE setting (0xC6).

25.3.3.12 Shaping and PA Ramping

With ASK modulation, up to eight power settings are used for shaping. The modulator contains a counter that counts up when transmitting a one and down when transmitting a zero. The counter counts at a rate equal to 8 times the symbol rate. The counter saturates at FREND0.PA_POWER and 0 respectively. This counter value is used as an index for a lookup in the power table. Thus, to utilize the whole table, FREND0.PA_POWER should be 7 when ASK is active. The shaping of the ASK signal is dependent on the configuration of the PATABLE.

Figure 25-12 shows some examples of ASK shaping.

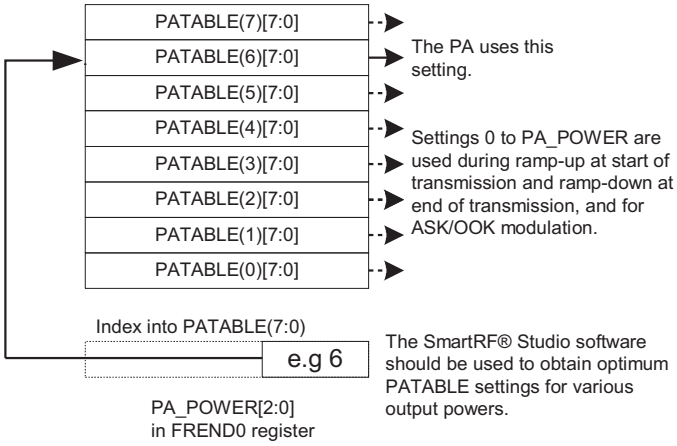


Figure 25-11. PA_POWER and PATABLE

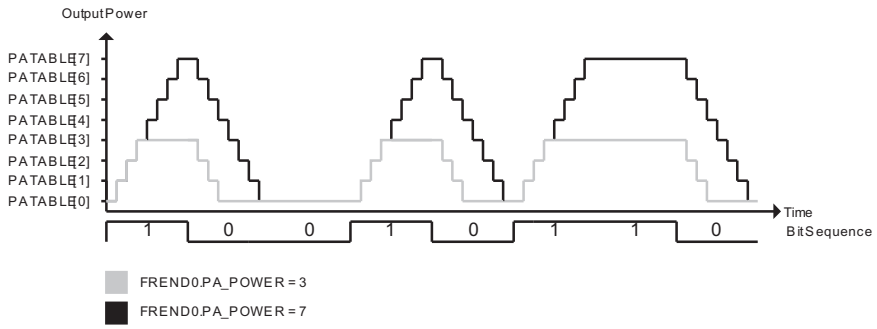


Figure 25-12. Shaping of ASK Signal

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.3.13 Asynchronous and Synchronous Serial Operation

Several features and modes of operation have been included in the radio core to provide backward compatibility with previous radios and other existing RF communication systems. For new systems, it is recommended to use the built-in packet handling features, as they can give more robust communication, significantly offload the CPU, and simplify software development.

25.3.3.13.1 Asynchronous Serial Operation

For backward compatibility with existing systems the asynchronous data transfer is included. When asynchronous transfer is enabled, several of the support mechanisms, such as packet handling, buffering in the FIFO, and so on, are disabled. The asynchronous transfer mode does not allow the use of the data whitener, and it is not possible to use Manchester encoding. MSK is not supported for asynchronous transfer.

Setting PKTCTRL0.PKT_FORMAT to 3 enables asynchronous serial mode.

The TX data is taken from a Timer_A capture/compare output (see the device-specific data sheet) or the GDO0 pin can be used for data input (TX data) with GDO0_CFGx = 0x2D. In this case the pin is automatically configured as an input when TX is active. The RX data is provided to a Timer_A capture/compare input (see the device-specific data sheet) and can be on GDO0, GDO1, or GDO2 according to their GDOx_CFG settings.

The modulator of the radio samples the level of the asynchronous input 8 times faster than the programmed data rate. The timing requirement for the asynchronous stream is that the error in the bit period must be less than one eighth of the programmed data rate.

25.3.3.13.2 Synchronous Serial Operation

Setting PKTCTRL0.PKT_FORMAT to 1 enables synchronous serial mode. In the synchronous serial mode, data is transferred on a two wire serial interface. The radio core provides a clock that is used to set up new data on the data input line or sample data on the data output line. The TX data is taken from a Timer_A capture/compare output (see the device-specific data sheet) or the GDO0 pin can be used for data input (TX data) with GDO0_CFGx = 0x2D. In this case the pin is automatically configured as an input when TX is active. The RX data is provided to a Timer_A capture/compare input (see the device-specific data sheet) and can be on GDO0, GDO1, or GDO2 according to their GDOx_CFG settings.

Preamble and sync word insertion or detection may or may not be active, dependent on the sync mode set by the MDMCFG2.SYNC_MODE. If preamble and sync word is disabled, all other packet handler features must be disabled and the preamble and sync word insertion and detection must be handled in software. If preamble and sync word insertion or detection is left on, all packet handling features can be used except of the address filtering feature.

When using the packet handling features in synchronous serial mode, the radio inserts and detects the preamble and sync word, and the application software only provides or receives the data payload. This is equivalent to the recommended FIFO operation mode.

25.3.4 System Considerations and Guidelines

25.3.4.1 SRD Regulations

International regulations and national laws regulate the use of radio receivers and transmitters. Short range devices (SRDs) for license-free operation below 1 GHz are usually operated in the 433 MHz, 868 MHz, or 915 MHz frequency bands. The CC1101 radio is specifically designed for such use with its 300 MHz to 348 MHz, 389 MHz to 464 MHz, and 779 MHz to 928 MHz operating ranges. The most important regulations when using a CC1101-based radio in the 433 MHz, 868 MHz, or 915 MHz frequency bands are EN 300 220 (Europe) and FCC CFR47 Part 15 (USA). A summary of the most important aspects of these regulations can be found in *SRD Regulations for Licence Free Transceiver Operation* (SWRA090).

NOTE: Compliance with regulations is dependent on complete system performance. It is the customer's responsibility to ensure that the system complies with regulations.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.4.2 Frequency Hopping and Multi-Channel Systems

The 315 MHz, 433 MHz, 868 MHz, or 915 MHz bands are shared by many systems both in industrial, office, and home environments. It is therefore recommended to use frequency hopping spread spectrum (FHSS) or a multi-channel protocol because the frequency diversity makes the system more robust with respect to interference from other systems operating in the same frequency band. FHSS also combats multipath fading.

The CC1101-based radio is highly suited for FHSS or multi-channel systems due to its agile frequency synthesizer and effective communication interface. Using the packet handling support and data buffering is also beneficial in such systems, as these features significantly offload the host controller.

Charge pump current, VCO current, and VCO capacitance array calibration data is required for each frequency when implementing frequency hopping. There are three ways of obtaining the calibration data from the radio:

1. Frequency hopping with calibration for each hop. The PLL calibration time is approximately 720 µs. The blanking interval between each frequency hop is then approximately 810 µs.
2. Fast frequency hopping without calibration for each hop can be done by performing the necessary calibrating at startup and saving the resulting FSCAL3, FSCAL2, and FSCAL1 register values in memory. The VCO capacitance calibration FSCAL1 register value must be found for each RF frequency to be used. The VCO current calibration value and the charge pump current calibration value available in FSCAL2 and FSCAL3 respectively are not dependent on the RF frequency, so the same value can therefore be used for all RF frequencies for these two registers. Between each frequency hop, the calibration process can then be replaced by writing the FSCAL3, FSCAL2 and FSCAL1 register values that corresponds to the next RF frequency. The PLL turn on time is approximately 90 µs. The blanking interval between each frequency hop is then approximately 90 µs.
3. Run calibration on a single frequency at startup. Next, write 0 to FSCAL3[5:4] to disable the charge-pump calibration. After writing to FSCAL3[5:4], strobe SRX (or STX) with MCSM0.FS_AUTOCAL = 1 for each new frequency hop. That is, VCO current and VCO capacitance calibration are done but not charge-pump current calibration. When charge pump current calibration is disabled, the calibration time is reduced from approximately 720 µs to approximately 150 µs. The blanking interval between each frequency hop is then approximately 240 µs.

There is a trade off between blanking time and memory space needed for storing calibration data in non-volatile memory. Solution 2 above gives the shortest blanking interval, but requires more memory space to store calibration values. This solution also requires that the supply voltage and temperature do not vary much in order to have a robust solution. Solution 3 gives approximately 570 µs smaller blanking interval than solution 1.

The recommended settings for TEST0.VCO_SEL_CAL_EN change with frequency. Therefore, SmartRF Studio should be used to determine the correct settings for a specific frequency before doing a calibration, regardless of which calibration method is used.

25.3.4.3 Wideband Modulation Not Using Spread Spectrum

Digital modulation systems under FFC Part 15.247 include 2-FSK and 2-GFSK modulation. A maximum peak output power of 1 W (+30 dBm) is allowed if the 6-dB bandwidth of the modulated signal exceeds 500 kHz. In addition, the peak power spectral density conducted to the antenna shall not be greater than 8 dBm in any 3-kHz band.

Operating at high data rates and frequency separation, the CC1101-based radio is suited for systems targeting compliance with digital modulation system as defined by FFC part 15.247. An external power amplifier is needed to increase the output above 10 dBm.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.4.4 Wireless MBUS

The wireless MBUS standard is a communication standard for meters and wireless readout of meters, and specifies the physical and the data link layer. Power consumption is a critical parameter for the meter side, since the communication link should be operative for the full lifetime of the meter without changing the battery. MSP430 with an CC1101-based radio is an excellent choice for the Wireless MBUS standard. For more information regarding see *AN067 - Wireless MBUS Implementation with CC1101 and MSP430* (SWRA234) . Since the Wireless MBUS standard operates in the 868-870 ISM band, the radio requirements must also comply with the ETSI EN 300 220 and CEPT/ERC/REC 70-03 E standards.

25.3.4.5 Data Burst Transmissions

The high maximum data rate of the CC1101-based radio allows burst transmissions. A low average data rate link (for example, 10 kBaud), can be realized using a higher over-the-air data rate. Buffering the data and transmitting in bursts at high data rate (for example, 500 kBaud) reduces the time in active mode and, therefore, reduces the average current consumption significantly. Reducing the time in active mode reduces the likelihood of collisions with other systems in the same frequency range.

NOTE: The sensitivity and thus transmission range is reduced for high data rate bursts compared to lower data rates.

25.3.4.6 Continuous Transmissions

In data streaming applications, the CC1101-based radio allows continuous transmissions at 500-kBaud effective data rate. As the modulation is done with a closed-loop PLL, there is no limitation on the length of a transmission (open-loop modulation used in some transceivers often prevents this continuous data streaming and reduces the effective data rate).

25.3.4.7 Spectrum Efficient Modulation

The CC1101-based radio also allows the use of Gaussian shaped 2-FSK (2-GFSK). This spectrum-shaping feature improves adjacent channel power (ACP) and occupied bandwidth. In true 2-FSK systems with abrupt frequency shifting, the spectrum is inherently broad. By making the frequency shift softer, the spectrum can be made significantly narrower. Thus, higher data rates can be transmitted in the same bandwidth using 2-GFSK.

25.3.4.8 Low Cost Systems

As the CC1101-based radio provides 0.8 to 500 kBaud multi-channel performance without any external SAW or loop filters, a very low-cost system can be made.

A differential antenna eliminates the need for a balun, and the dc biasing can be achieved in the antenna topology.

25.3.4.9 Battery Operated Systems

In low-power applications, the SLEEP state with the crystal oscillator core switched off should be used when the radio is not active. The crystal oscillator core can be left running in the SLEEP state if start-up time is critical.

The WOR functionality should be used in low power applications.

25.3.4.10 Increasing Output Power

In some applications, it may be necessary to extend the link range. Adding an external power amplifier is the most effective way to do this.

The power amplifier should be inserted between the antenna and the balun, and two T/R switches are needed to disconnect the PA in RX mode (see Figure 25-13).

Copyright © 2009–2013, Texas Instruments Incorporated

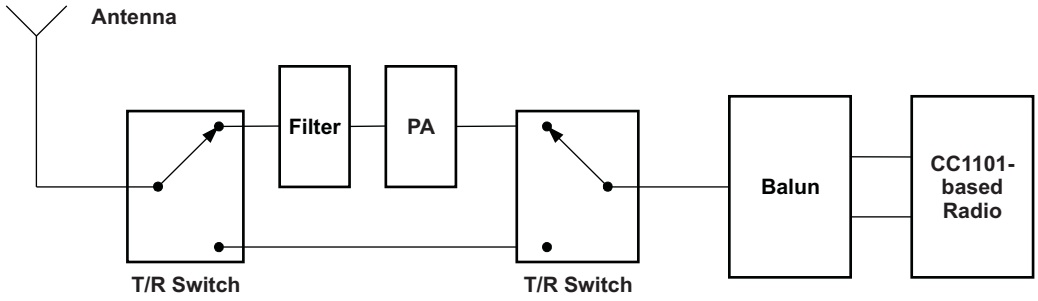


Figure 25-13. Block Diagram of CC1101-based Radio With External Power Amplifier

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.5 Radio Core Registers

The configuration of radio core is done by programming 8-bit registers. The optimum configuration based on selected system parameters is most easily found by using the SmartRF Studio software. Complete descriptions of the registers are given in the following tables. After reset (PUC or command strobe SRES), all the registers have default values as shown in the tables. The optimum register setting might differ from the default value.

There are 47 normal 8-bit configuration registers, listed in Table 25-19. Many of these registers are for test purposes only and need not be written for normal operation.

There are also 12 status registers, listed in Table 25-20. These registers, which are read-only, contain information about the status of radio core.

Copyright © 2009–2013, Texas Instruments Incorporated

Table 25-19. Configuration Registers

ADDRESS	REGISTER	DESCRIPTION
0x00	IOCFG2	GDO2 output configuration
0x01	IOCFG1	GDO1 output configuration
0x02	IOCFG0	GDO0 output configuration
0x03	FIFOTHR	RX FIFO and TX FIFO thresholds
0x04	SYNC1	Sync word, high byte
0x05	SYNC0	Sync word, low byte
0x06	PKTLEN	Packet length
0x07	PKTCTRL1	Packet automation control
0x08	PKTCTRL0	Packet automation control
0x09	ADDR	Device address
0x0A	CHANNR	Channel number
0x0B	FSCTRL1	Frequency synthesizer control
0x0C	FSCTRL0	Frequency synthesizer control
0x0D	FREQ2	Frequency control word, high byte
0x0E	FREQ1	Frequency control word, middle byte
0x0F	FREQ0	Frequency control word, low byte
0x10	MDMCFG4	Modem configuration
0x11	MDMCFG3	Modem configuration
0x12	MDMCFG2	Modem configuration
0x13	MDMCFG1	Modem configuration
0x14	MDMCFG0	Modem configuration
0x15	DEVIATN	Modem deviation setting
0x16	MCSM2	Main radio control state machine configuration
0x17	MCSM1	Main radio control state machine configuration
0x18	MCSM0	Main radio control state machine configuration
0x19	FOCCFG	Frequency offset compensation configuration
0x1A	BSCFG	Bit synchronization configuration
0x1B	AGCCTRL2	AGC control
0x1C	AGCCTRL1	AGC control
0x1D	AGCCTRL0	AGC control
0x1E	WOREVT1	High byte Event 0 timeout
0x1F	WOREVT0	Low byte Event 0 timeout
0x20	WORCTRL	Wake-on-radio control
0x21	FREND1	Front-end RX configuration
0x22	FREND0	Front-end TX configuration
0x23	FSCAL3	Frequency synthesizer calibration
0x24	FSCAL2	Frequency synthesizer calibration
0x25	FSCAL1	Frequency synthesizer calibration
0x26	FSCAL0	Frequency synthesizer calibration
0x27		Reserved - read as "0"
0x28		Reserved - read as "0"
0x29	FSTEST	Frequency synthesizer calibration control
0x2A	PTEST	Production test
0x2B	AGCTEST	AGC test
0x2C	TEST2	Various test settings
0x2D	TEST1	Various test settings
0x2E	TEST0	Various test settings

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-20. Status Registers

ADDRESS	REGISTER	DESCRIPTION
0x30	PARTNUM	Part number
0x31	VERSION	Current version number
0x32	FREQEST	Frequency offset estimate
0x33	LQI	Demodulator estimate for link quality
0x34	RSSI	Received signal strength indication
0x35	MARCSTATE	Control state machine state
0x36	WORTIME1	High byte of WOR timer
0x37	WORTIME0	Low byte of WOR timer
0x38	PKTSTATUS	Current GDOx status and packet status
0x39	VCO_VC_DAC	Current setting from PLL calibration module
0x3A	TXBYTES	Underflow and number of bytes in the TX FIFO
0x3B	RXBYTES	Overflow and number of bytes in the RX FIFO

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.3.5.1 Radio Core Configuration Register Details

0x00: IOCFG2 – GDO2 Output Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6	GDO2_INV	0	R/W	Invert output; that is, select active low (1) or active high (0)
5:0	GDO2_CFG[5:0]	41 (0x29)	R/W	Signal selection according to Table 25-21. Default is RF_RDYn.

0x01: IOCFG1 – GDO1 Output Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6	GDO1_INV	0	R/W	Invert output; that is, select active low (1) or active high (0)
5:0	GDO1_CFG[5:0]	46 (0x2E)	R/W	Signal selection according to Table 25-21. Default is 3-state.

0x02: IOCFG0 – GDO0 Output Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6	GDO0_INV	0	R/W	Invert output; that is, select active low (1) or active high (0)
5:0	GDO0_CFG[5:0]	46 (0x2E)	R/W	Signal selection according to Table 25-21. Default is 3-state.

Table 25-21. GDOx Signal Selection (x = 0, 1, or 2)

GDOx_CFG[5:0]	DESCRIPTION
0 (0x00)	Associated to the RX FIFO: Asserts when RX FIFO is filled at or above the RX FIFO threshold. Deasserts when RX FIFO is drained below the same threshold.
1 (0x01)	Associated to the RX FIFO: Asserts when RX FIFO is filled at or above the RX FIFO threshold or the end of packet is reached. Deasserts when the RX FIFO is empty.
2 (0x02)	Associated to the TX FIFO: Asserts when the TX FIFO is filled at or above the TX FIFO threshold. Deasserts when the TX FIFO is below the same threshold.
3 (0x03)	Associated to the TX FIFO: Asserts when TX FIFO is full. Deasserts when the TX FIFO is drained below the TX FIFO threshold.
4 (0x04)	Asserts when the RX FIFO has overflowed. Deasserts when the FIFO has been flushed.
5 (0x05)	Asserts when the TX FIFO has underflowed. Deasserts when the FIFO is flushed.
6 (0x06)	Asserts when sync word has been sent or received, and deasserts at the end of the packet. In RX, the pin deassert when the optional address check fails or the RX FIFO overflows. In TX the pin deasserts if the TX FIFO underflows.
7 (0x07)	Asserts when a packet has been received with CRC OK. Deasserts when the first byte is read from the RX FIFO.
8 (0x08)	Preamble Quality Reached. Asserts when the PQI is above the programmed PQT value.
9 (0x09)	Clear channel assessment. High when RSSI level is below threshold (dependent on the current CCA_MODE setting).
10 (0x0A)	Lock detector output. The PLL is in lock if the lock detector output has a positive transition or is constantly logic high. To check for PLL lock the lock detector output should be used as an interrupt source.
11 (0x0B)	Serial Clock. Synchronous to the data in synchronous serial mode. In RX mode, data is set up on the falling edge when GDOx_INV=0. In TX mode, data is sampled on the rising edge of the serial clock when GDOx_INV=0.
12 (0x0C)	Serial Synchronous Data Output. Used for synchronous serial mode.
13 (0x0D)	Serial Data Output. Used for asynchronous serial mode.
14 (0x0E)	Carrier sense. High if RSSI level is above threshold.
15 (0x0F)	CRC_OK. The last CRC comparison matched. Cleared when entering/restarting RX mode.
16 (0x10)	Reserved – used for test.
17 (0x11)	Reserved – used for test.
18 (0x12)	Reserved – used for test.
19 (0x13)	Reserved – used for test.
20 (0x14)	Reserved – used for test.
21 (0x15)	Reserved – used for test.
22 (0x16)	RX_HARD_DATA[1]. Can be used together with RX_SYMBOL_TICK for alternative serial RX output.
23 (0x17)	RX_HARD_DATA[0]. Can be used together with RX_SYMBOL_TICK for alternative serial RX output.
24 (0x18)	Reserved – used for test.
25 (0x19)	Reserved – used for test.

Copyright © 2009–2013, Texas Instruments Incorporated

Table 25-21. GDOx Signal Selection (x = 0, 1, or 2) (continued)

26 (0x1A)	Reserved – used for test.
27 (0x1B)	PA power down signal to control an external PA and/or RX/TX switch.
28 (0x1C)	LNA power down signal to control an external LNA and/or RX/TX switch.
29 (0x1D)	RX_SYMBOL_TICK. Can be used together with RX_HARD_DATA for alternative serial RX output.
30 (0x1E)	RSSI_VALID.
31 (0x1F)	RX_TIMEOUT.
32 (0x20)	Reserved – used for test.
33 (0x21)	Reserved – used for test.
34 (0x22)	Reserved – used for test.
35 (0x23)	Reserved – used for test.
36 (0x24)	WOR_EVENT0.
37 (0x25)	WOR_EVENT1.
38 (0x26)	Reserved – used for test.
39 (0x27)	CLK_32k
40 (0x28)	Reserved – used for test.
41 (0x29)	RF_RDYn
42 (0x2A)	Reserved – used for test.
43 (0x2B)	XOSC_STABLE
44 (0x2C)	Reserved – used for test.
45 (0x2D)	On GDO1 and GDO2 this signal indicates, when 0, that GDO0 is configured as input (when GDO0_CFG = 0x2D) or that serial TX data is taken from the Timer_A (when GDO0_CFG ≠ 0x2D). If GDO0_CFG = 0x2D the serial input data is taken from GDO0 otherwise it is taken from Timer_A.
46 (0x2E)	Tri-State.
47 (0x2F)	Hard-wired to 0. (Hard-wired to 1 can be achieved by setting GDOx_INV=1.)
48 (0x30)	RFCLK/1 (1)
49 (0x31)	RFCLK/1.5 (1)
50 (0x32)	RFCLK/2 (1)
51 (0x33)	RFCLK/3 (1)
52 (0x34)	RFCLK/4 (1)
53 (0x35)	RFCLK/6 (1)
54 (0x36)	RFCLK/8 (1)
55 (0x37)	RFCLK/12 (1)
56 (0x38)	RFCLK/16 (1)
57 (0x39)	RFCLK/24 (1)
58 (0x3A)	RFCLK/32 (1)
59 (0x3B)	RFCLK/48 (1)
60 (0x3C)	RFCLK/64 (1)
61 (0x3D)	RFCLK/96 (1)
62 (0x3E)	RFCLK/128 (1)
63 (0x3F)	RFCLK/192 (1)

(1) There are 3 GDO signals, but only one RFCLK/n can be selected as an output at any time. If RFCLK/n is to be monitored on one of the GDO pins, the other two GDO pins must be configured to values less than 0x30.
The divided clock might also be used as a timer clock (for example, for Timer_A, refer to device-specific data sheet). If a RFCLK divider setting other than the default setting expected by the timer (/192) is selected this divider setting will be also used for the timer(s).
To optimize RF performance, these signal should not be used while the radio is in RX or TX mode.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x03: FIFOTHR – RX FIFO and TX FIFO Thresholds

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved	0	R/W	Write 0 for compatibility with possible future extensions
6	ADC_RETENTION	0	R/W	0: TEST1 = 0x31 and TEST2 = 0x88 when waking up from SLEEP 1: TEST1 = 0x35 and TEST2 = 0x81 when waking up from SLEEP
5:4	CLOSE_IN_RX [1:0]	0 (00)	R/W	For more details, see *Close-in Reception With CC1101* (SWRA147). **Setting** / **RX Attenuation, Typical Values** 0 (00) – 0 dB 1 (01) – 6 dB 2 (10) – 12 dB 3 (11) – 18 dB
3:0	FIFO_THR[3:0]	7 (0111)	R/W	Set the threshold for the TX FIFO and RX FIFO. The threshold is exceeded when the number of bytes in the FIFO is equal to or higher than the threshold value. **Setting** / **Bytes in TX FIFO** / **Bytes in RX FIFO** 0 (0000) – 61 – 4 1 (0001) – 57 – 8 2 (0010) – 53 – 12 3 (0011) – 49 – 16 4 (0100) – 45 – 20 5 (0101) – 41 – 24 6 (0110) – 37 – 28 7 (0111) – 33 – 32 8 (1000) – 29 – 36 9 (1001) – 25 – 40 10 (1010) – 21 – 44 11 (1011) – 17 – 48 12 (1100) – 13 – 52 13 (1101) – 9 – 56 14 (1110) – 5 – 60 15 (1111) – 1 – 64

0x04: SYNC1 – Sync Word, High Byte

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	SYNC[15:8]	211 (0xD3)	R/W	8 MSB of 16-bit sync word

0x05: SYNC0 – Sync Word, Low Byte

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	SYNC[7:0]	145 (0x91)	R/W	8 LSB of 16-bit sync word

0x06: PKTLEN – Packet Length

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	PACKET_LENGTH	255 (0xFF)	R/W	Indicates the packet length when fixed packet length mode is enabled. If variable packet length mode is used, this value indicates the maximum packet length allowed.

Copyright © 2009–2013, Texas Instruments Incorporated

0x07: PKTCTRL1 – Packet Automation Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:5	PQT[2:0]	0 (0x00)	R/W	Preamble quality estimator threshold. The preamble quality estimator increases an internal counter by one each time a bit is received that is different from the previous bit, and decreases the counter by 8 each time a bit is received that is the same as the last bit. A threshold of 4 × PQT for this counter is used to gate sync-word detection. When PQT = 0 a sync word is always accepted.
4	Reserved	0	R0	
3	CRC_AUTOFLUSH	0	R/W	Enable automatic flush of RX FIFO when CRC in not OK. This requires that only one packet is in the RX FIFO and that packet length is limited to the RX FIFO size.
2	APPEND_STATUS	1	R/W	When enabled, two status bytes are appended to the payload of the packet. The status bytes contain RSSI and LQI values, as well as CRC OK.
1:0	ADR_CHK[1:0]	0 (00)	R/W	Controls address check configuration of received packages.

Setting	Address Check Configuration
0 (00)	No address check
1 (01)	Address check, no broadcast
2 (10)	Address check and 0 (0x00) broadcast
3 (11)	Address check and 0 (0x00) and 255 (0xFF) broadcast

0x08: PKTCTRL0 – Packet Automation Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6	WHITE_DATA	1	R/W	Turn data whitening on/off 0: Whitening off 1: Whitening on
5:4	PKT_FORMAT[1:0]	0 (00)	R/W	Format of RX and TX data

Setting	Packet Format
0 (00)	Normal mode, use FIFOs for RX and TX
1 (01)	Synchronous serial mode. Used for backwards compatibility.
2 (10)	Random TX mode. Sends random data using PN9 generator. Used for test. Works as normal mode, setting 0 (00), in RX.
3 (11)	Asynchronous serial mode.

3	Reserved	0	R0	
2	CRC_EN	1	R/W	Enable CRC 1: CRC calculation in TX and CRC check in RX enabled 0: CRC disabled for TX and RX
1:0	LENGTH_CONFIG[1:0]	1 (01)	R/W	Configure the packet length

Setting	Packet Length Configuration
0 (00)	Fixed packet length mode. Length configured in PKTLEN register
1 (01)	Variable packet length mode. Packet length configured by the first byte after sync word
2 (10)	Infinite packet length mode
3 (11)	Reserved

0x09: ADDR – Device Address

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	DEVICE_ADDR[7:0]	0 (0x00)	R/W	Address used for packet filtration. Optional broadcast addresses are 0 (0x00) and 255 (0xFF).

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x0A: CHANNR – Channel Number

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	CHAN[7:0]	0 (0x00)	R/W	The 8-bit unsigned channel number, which is multiplied by the channel spacing setting and added to the base frequency.

0x0B: FSCTRL1 – Frequency Synthesizer Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:5	Reserved		R0	
4:0	FREQ_IF[4:0]	15 (0x0F)	R/W	The desired IF frequency to employ in RX. Subtracted from FS base frequency in RX and controls the digital complex mixer in the demodulator. $f_{IF} = (f_{RFXT2}/2^{10}) \times FREQ_IF$ The default value gives an IF frequency of 381 kHz, assuming a 26-MHz crystal.

0x0C: FSCTRL0 – Frequency Synthesizer Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	FREQOFF[7:0]	0 (0x00)	R/W	Frequency offset added to the base frequency before being used by the frequency synthesizer (2s complement). Resolution is $f_{RFXT2}/2^{14}$ (1.59 kHz to 1.65 kHz). Range is ±202 kHz to ±210 kHz, dependent on crystal frequency.

0x0D: FREQ2 – Frequency Control Word, High Byte

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	FREQ[23:22]	0 (00)	R	FREQ[23:22] is always 0 (the FREQ2 register is less than 36 with 26-MHz to 27-MHz crystal)
5:0	FREQ[21:16]	30 (0x1E)	R/W	FREQ[23:22] is the base frequency for the frequency synthesizer in increments of $f_{RFXT2}/2^{16}$. $f_{carrier} = (f_{RFXT2}/2^{16}) \times FREQ[23:0]$

0x0E: FREQ1 – Frequency Control Word, Middle Byte

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	FREQ[15:8]	196 (0xC4)	R/W	See description in FREQ2 register

0x0F: FREQ0 – Frequency Control Word, Low Byte

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	FREQ[7:0]	236 (0xEC)	R/W	See description in FREQ2 register

0x10: MDMCFG4 – Modem Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	CHANBW_E[1:0]	2 (0x02)	R/W	
5:4	CHANBW_M[1:0]	0 (0x00)	R/W	Sets the decimation ratio for the delta-sigma ADC input stream and thus the channel bandwidth. $BW_{channel} = \frac{f_{XOSC}}{8 \times (4 + CHANBW_M) \times 2^{CHANBW_E}}$ The default values give 203 kHz channel filter bandwidth, assuming a 26-MHz crystal.
3:0	DRATE_E[3:0]	12 (0x0C)	R/W	The exponent of the user specified symbol rate

Copyright © 2009–2013, Texas Instruments Incorporated

0x11: MDMCFG3 – Modem Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	DRATE_M[7:0]	34 (0x22)	R/W	The mantissa of the user specified symbol rate. The symbol rate is configured using an unsigned, floating-point number with 9-bit mantissa and 4-bit exponent. The 9th bit is a hidden 1. The resulting data rate is: $$R_{DATA} = \frac{(256+DRATE_M) \times 2^{DRATE_E}}{2^{28}} \times f_{XOSC}$$ The default values give a data rate of 115.051 kBaud (closest setting to 115.2 kBaud), assuming a 26-MHz crystal.

0x12: MDMCFG2 – Modem Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	DEM_DCFILT_OFF	0	R/W	Disable digital dc blocking filter before demodulator. 0 = Enable (better sensitivity) 1 = Disable (current optimized). Only for data rates ≤ 250 kBaud. The recommended IF frequency changes when the dc blocking is disabled. Use SmartRF Studio to calculate correct register setting.
6:4	MOD_FORMAT[2:0]	0 (000)	R/W	The modulation format of the radio signal **Setting** / **Modulation Format** 0 (000) 2-FSK 1 (001) 2-GFSK 2 (010) Reserved 3 (011) ASK/OOK 4 (100) Reserved 5 (101) Reserved 6 (110) Reserved 7 (111) MSK MSK is supported only for data rates above 26 kBaud
3	MANCHESTER_EN	0	R/W	Enables Manchester encoding/decoding. 0 = Disable 1 = Enable
2:0	SYNC_MODE[2:0]	2 (010)	R/W	Combined sync-word qualifier mode. The values 0 (000) and 4 (100) disables preamble and sync word transmission in TX and preamble and sync word detection in RX. The values 1 (001), 2 (010), 5 (101) and 6 (110) enables 16-bit sync word transmission in TX and 16-bits sync word detection in RX. Only 15 of 16 bits need to match in RX when using setting 1 (001) or 5 (101). The values 3 (011) and 7 (111) enables repeated sync word transmission in TX and 32-bits sync word detection in RX (only 30 of 32 bits need to match). **Setting** / **Sync-Word Qualifier Mode** 0 (000) No preamble/sync 1 (001) 15/16 sync word bits detected 2 (010) 16/16 sync word bits detected 3 (011) 30/32 sync word bits detected 4 (100) No preamble/sync, carrier-sense above threshold 5 (101) 15/16 + carrier-sense above threshold 6 (110) 16/16 + carrier-sense above threshold 7 (111) 30/32 + carrier-sense above threshold

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x13: MDMCFG1– Modem Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6:4	NUM_PREAMBLE[2:0]	2 (010)	R/W	Sets the minimum number of preamble bytes to be transmitted

Setting	Number of Preamble Bytes
0 (000)	2
1 (001)	3
2 (010)	4
3 (011)	6
4 (100)	8
5 (101)	12
6 (110)	16
7 (111)	24

3:2	Reserved		R0	
1:0	CHANSPC_E[1:0]	2 (10)	R/W	Two bit exponent of channel spacing

0x14: MDMCFG0 – Modem Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	CHANSPC_M[7:0]	248 (0xF8)	R/W	8-bit mantissa of channel spacing. The channel spacing is multiplied by the channel number CHAN and added to the base frequency. It is unsigned and has the format: $\Delta f_{CHANNEL} = \frac{f_{XOSC}}{2^{18}} \times (256 + CHANSPC_M) \times 2^{CHANSPC_E}$ The default values give 199.951 kHz channel spacing (the closest setting to 200 kHz), assuming 26-MHz crystal frequency.

0x15: DEVIATN – Modem Deviation Setting

BIT	FIELD NAME	RESET	R/W	DESCRIPTION	
7	Reserved		R0		
6:4	DEVIATION_E[2:0]	4 (0x04)	R/W	Deviation exponent	
3	Reserved		R0		
2:0	DEVIATION_M[2:0]	7 (111)	R/W	In TX:	
				2-FSK/2-GFSK	Specifies the nominal frequency deviation from the carrier for a '0' (-DEVIATN) and '1' (+DEVIATN) in a mantissa-exponent format, interpreted as a 4-bit value with MSB implicit 1. The resulting frequency deviation is given by: $f_{dev} = \frac{f_{xosc}}{2^{17}} \times (8 + DEVIATION_M) \times 2^{DEVIATION_E}$ The default values give ±47.607 kHz deviation, assuming 26-MHz crystal frequency.
				MSK	Specifies the fraction of symbol period (1/8-8/8) during which a phase change occurs (‘0’: +90deg, ‘1’:-90deg). Refer to the SmartRF Studio software for correct DEVIATN setting when using MSK.
				ASK/OOK	This setting has no effect.
				In RX:	
				2-FSK/2-GFSK	Specifies the expected frequency deviation of incoming signal, must be approximately right for demodulation to be performed reliably and robustly.
				MSK/ASK/OOK	This setting has no effect.

Copyright © 2009–2013, Texas Instruments Incorporated

0x16: MCSM2 – Main Radio Control State Machine Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:5	Reserved		R0	Reserved
4	RX_TIME_RSSI	0	R/W	Direct RX termination based on RSSI measurement (carrier sense). For ASK/OOK modulation, RX times out if there is no carrier sense in the first 8 symbol periods.
3	RX_TIME_QUAL	0	R/W	When the RX_TIME timer expires, the radio checks if sync word is found when RX_TIME_QUAL = 0, or either sync word is found or PQI is set when RX_TIME_QUAL = 1.
2:0	RX_TIME[2:0]	7 (111)	R/W	Timeout for sync word search in RX for both WOR mode and normal RX operation. With RX_TIME = 7 the timeout is disabled. For RX_TIME < 7 the RX timeout (t_{RX_time}) is a fraction of the programmed EVENT0 timeout (t_{Event0}): $t_{RX_time} = t_{Event0} / 2^{(RX_TIME+3+WOR_RES)} = 1/f_{ACLK} \times EVENT0 \times 2^{(4 \times WOR_RES-RX_TIME-3)}$

The RX timeout as a percentage of the EVENT0 timeout (which is equal to the duty cycle using WOR) is approximated by:

	WOR_RES			
RX_TIME	**0**	**1**	**2**	**3**
0 (000)	12.50%	6.25%	3.13%	1.56%
1 (001)	6.25%	3.13%	1.56%	0.78%
2 (010)	3.13%	1.56%	0.78%	0.39%
3 (011)	1.56%	0.78%	0.39%	0.20%
4 (100)	0.78%	0.39%	0.20%	0.10%
5 (101)	0.39%	0.20%	0.10%	0.05%
6 (110)	0.20%	0.10%	0.05%	0.024%
7 (111)	Timeout disabled			

Note: WORCTRL.ACLK_PD bit must be reset to 0 to use settings 0-6, because the timeout timer requires ACLK.

WOR_RES should be 0 or 1 when using WOR - but WOR_RES > 1 can be used to set longer RX timeouts when not using WOR.

The timeout counter resolution is limited: With RX_TIME = 0, the timeout count is given by the 13 MSBs of EVENT0, decreasing to the 7 MSBs of EVENT0 with RX_TIME = 6.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x17: MCSM1 – Main Radio Control State Machine Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	Reserved		R0	
5:4	CCA_MODE[1:0]	3 (11)	R/W	Selects CCA_MODE. Reflected in CCA signal.

Setting	Clear Channel Indication
0 (00)	Always
1 (01)	If RSSI below threshold
2 (10)	Unless currently receiving a packet
3 (11)	If RSSI below threshold unless currently receiving a packet

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
3:2	RXOFF_MODE[1:0]	0 (00)	R/W	Select what should happen when a packet has been received

Setting	Next State After Finishing Packet Reception
0 (00)	IDLE
1 (01)	FSTXON
2 (10)	TX
3 (11)	Stay in RX

It is not possible to set RXOFF_MODE to be TX or FSTXON and at the same time use CCA.

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
1:0	TXOFF_MODE[1:0]	0 (00)	R/W	Select what should happen when a packet has been sent (TX)

Setting	Next State After Finishing Packet Transmission
0 (00)	IDLE
1 (01)	FSTXON
2 (10)	Stay in TX (start sending preamble)
3 (11)	RX

0x18: MCSM0 - Main Radio Control State Machine Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	Reserved		R0	
5:4	FS_AUTOCAL[1:0]	0 (00)	R/W	Automatically calibrate when going to RX or TX, or back to IDLE

Setting	When To Perform Automatic Calibration
0 (00)	Never (manually calibrate using SCAL strobe)
1 (01)	When going from IDLE to RX or TX (or FSTXON)
2 (10)	When going from RX or TX back to IDLE automatically
3 (11)	Every fourth time when going from RX or TX to IDLE automatically

In some automatic wake-on-radio (WOR) applications, using setting 3 (11) can significantly reduce current consumption.

3:2	Reserved		R0	
1	PIN_CTRL_EN	0	R/W	Enables the pin radio control option
0	XOSC_FORCE_ON	0	R/W	Force the RF XT2 oscillator to stay on in the SLEEP state.

Copyright © 2009–2013, Texas Instruments Incorporated

0x19: FOCCFG - Frequency Offset Compensation Configuration

<table>
<tr><th>BIT</th><th>FIELD NAME</th><th>RESET</th><th>R/W</th><th>DESCRIPTION</th></tr>
<tr><td>7:6</td><td>Reserved</td><td></td><td>R0</td><td></td></tr>
<tr><td>5</td><td>FOC_BS_CS_GATE</td><td>1</td><td>R/W</td><td>If set, the demodulator freezes the frequency offset compensation and clock recovery feedback loops until the CS signal goes high.</td></tr>
<tr><td>4:3</td><td>FOC_PRE_K[1:0]</td><td>2 (10)</td><td>R/W</td><td>The frequency compensation loop gain to be used before a sync word is detected.
<table><tr><th>Setting</th><th>Frequency Compensation Loop Gain Before Sync Word</th></tr><tr><td>0 (00)</td><td>K</td></tr><tr><td>1 (01)</td><td>2K</td></tr><tr><td>2 (10)</td><td>3K</td></tr><tr><td>3 (11)</td><td>4K</td></tr></table></td></tr>
<tr><td>2</td><td>FOC_POST_K</td><td>1</td><td>R/W</td><td>The frequency compensation loop gain to be used after a sync word is detected.
<table><tr><th>Setting</th><th>Frequency Compensation Loop Gain After Sync Word</th></tr><tr><td>0</td><td>Same as FOC_PRE_K</td></tr><tr><td>1</td><td>K/2</td></tr></table></td></tr>
<tr><td>1:0</td><td>FOC_LIMIT[1:0]</td><td>2 (10)</td><td>R/W</td><td>The saturation point for the frequency offset compensation algorithm:
<table><tr><th>Setting</th><th>Saturation Point (Maximum Compensated Offset)</th></tr><tr><td>0 (00)</td><td>±0 (no frequency offset compensation)</td></tr><tr><td>1 (01)</td><td>$\pm BW_{CHAN}/8$</td></tr><tr><td>2 (10)</td><td>$\pm BW_{CHAN}/4$</td></tr><tr><td>3 (11)</td><td>$\pm BW_{CHAN}/2$</td></tr></table>Frequency offset compensation is not supported for ASK/OOK. Always use FOC_LIMIT = 0 with these modulation formats.</td></tr>
</table>

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x1A: BSCFG - Bit Synchronization Configuration

<table>
<tr><th>BIT</th><th>FIELD NAME</th><th>RESET</th><th>R/W</th><th>DESCRIPTION</th></tr>
<tr><td>7:6</td><td>BS_PRE_KI[1:0]</td><td>1 (01)</td><td>R/W</td><td>The clock recovery feedback loop integral gain to be used before a sync word is detected (used to correct offsets in data rate):
<table>
<tr><th>Setting</th><th>Clock Recovery Loop Integral Gain Before Sync Word</th></tr>
<tr><td>0 (00)</td><td>K_I</td></tr>
<tr><td>1 (01)</td><td>$2K_I$</td></tr>
<tr><td>2 (10)</td><td>$3K_I$</td></tr>
<tr><td>3 (11)</td><td>$4K_I$</td></tr>
</table></td></tr>
<tr><td>5:4</td><td>BS_PRE_KP[1:0]</td><td>2 (10)</td><td>R/W</td><td>The clock recovery feedback loop proportional gain to be used before a sync word is detected.
<table>
<tr><th>Setting</th><th>Clock Recovery Loop Proportional Gain Before Sync Word</th></tr>
<tr><td>0 (00)</td><td>K_P</td></tr>
<tr><td>1 (01)</td><td>$2K_P$</td></tr>
<tr><td>2 (10)</td><td>$3K_P$</td></tr>
<tr><td>3 (11)</td><td>$4K_P$</td></tr>
</table></td></tr>
<tr><td>3</td><td>BS_POST_KI</td><td>1</td><td>R/W</td><td>The clock recovery feedback loop integral gain to be used after a sync word is detected.
<table>
<tr><th>Setting</th><th>Clock Recovery Loop Integral Gain After Sync Word</th></tr>
<tr><td>0</td><td>Same as BS_PRE_KI</td></tr>
<tr><td>1</td><td>K_I /2</td></tr>
</table></td></tr>
<tr><td>2</td><td>BS_POST_KP</td><td>1</td><td>R/W</td><td>The clock recovery feedback loop proportional gain to be used after a sync word is detected.
<table>
<tr><th>Setting</th><th>Clock Recovery Loop Proportional Gain After Sync Word</th></tr>
<tr><td>0</td><td>Same as BS_PRE_KP</td></tr>
<tr><td>1</td><td>K_P</td></tr>
</table></td></tr>
<tr><td>1:0</td><td>BS_LIMIT[1:0]</td><td>0 (00)</td><td>R/W</td><td>The saturation point for the data rate offset compensation algorithm:
<table>
<tr><th>Setting</th><th>Data Rate Offset Saturation (Max Data Rate Difference)</th></tr>
<tr><td>0 (00)</td><td>±0 (No data rate offset compensation performed)</td></tr>
<tr><td>1 (01)</td><td>±3.125% data rate offset</td></tr>
<tr><td>2 (10)</td><td>±6.25% data rate offset</td></tr>
<tr><td>3 (11)</td><td>±12.5% data rate offset</td></tr>
</table></td></tr>
</table>

Copyright © 2009–2013, Texas Instruments Incorporated

0x1B: AGCCTRL2 - AGC Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION	
7:6	MAX_DVGA_GAIN[1:0]	0 (00)	R/W	Reduces the maximum allowable DVGA gain.	
				Setting	**Allowable DVGA Settings**
				0 (00)	All gain settings can be used
				1 (01)	The highest gain setting cannot be used
				2 (10)	The two highest gain settings cannot be used
				3 (11)	The three highest gain settings cannot be used
5:3	MAX_LNA_GAIN[2:0]	0 (000)	R/W	Sets the maximum allowable LNA + LNA 2 gain relative to the maximum possible gain.	
				Setting	**Maximum Allowable LNA + LNA 2 Gain**
				0 (000)	Maximum possible LNA + LNA 2 gain
				1 (001)	Approximately 2.6 dB below maximum possible gain
				2 (010)	Approximately 6.1 dB below maximum possible gain
				3 (011)	Approximately 7.4 dB below maximum possible gain
				4 (100)	Approximately 9.2 dB below maximum possible gain
				5 (101)	Approximately 11.5 dB below maximum possible gain
				6 (110)	Approximately 14.6 dB below maximum possible gain
				7 (111)	Approximately 17.1 dB below maximum possible gain
2:0	MAGN_TARGET[2:0]	3 (011)	R/W	These bits set the target value for the averaged amplitude from the digital channel filter (1 LSB = 0 dB).	
				Setting	**Target Amplitude From Channel Filter**
				0 (000)	24 dB
				1 (001)	27 dB
				2 (010)	30 dB
				3 (011)	33 dB
				4 (100)	36 dB
				5 (101)	38 dB
				6 (110)	40 dB
				7 (111)	42 dB

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x1C: AGCCTRL1 - AGC Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6	AGC_LNA_PRIORITY	1	R/W	Selects between two different strategies for LNA and LNA 2 gain adjustment. When 1, the LNA gain is decreased first. When 0, the LNA 2 gain is decreased to minimum before decreasing LNA gain.
5:4	CARRIER_SENSE_REL_THR[1:0]	0 (00)	R/W	Sets the relative change threshold for asserting carrier sense
				Setting / **Carrier Sense Relative Threshold**
				0 (00) Relative carrier sense threshold disabled
				1 (01) 6 dB increase in RSSI value
				2 (10) 10 dB increase in RSSI value
				3 (11) 14 dB increase in RSSI value
3:0	CARRIER_SENSE_ABS_THR[3:0]	0 (0000)	R/W	Sets the absolute RSSI threshold for asserting carrier sense. The 2s-complement signed threshold is programmed in steps of 1 dB and is relative to the MAGN_TARGET setting.
				Setting / **Carrier Sense Absolute Threshold** (Equal to channel filter amplitude when AGC has not decreased gain)
				-8 (1000) Absolute carrier sense threshold disabled
				-7 (1001) 7 dB below MAGN_TARGET setting
				⋮ ⋮
				-1 (1111) 1 dB below MAGN_TARGET setting
				0 (0000) At MAGN_TARGET setting
				1 (0001) 1 dB above MAGN_TARGET setting
				⋮ ⋮
				7 (0111) 7 dB above MAGN_TARGET setting

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x1D: AGCCTRL0 - AGC Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	HYST_LEVEL[1:0]	2 (10)	R/W	Sets the level of hysteresis on the magnitude deviation (internal AGC signal that determine gain changes).

Setting	Description
0 (00)	No hysteresis, small symmetric dead zone, high gain
1 (01)	Low hysteresis, small asymmetric dead zone, medium gain
2 (10)	Medium hysteresis, medium asymmetric dead zone, medium gain
3 (11)	Large hysteresis, large asymmetric dead zone, low gain

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
5:4	WAIT_TIME[1:0]	1 (01)	R/W	Sets the number of channel filter samples from a gain adjustment has been made until the AGC algorithm starts accumulating new samples.

Setting	Channel Filter Samples
0 (00)	8
1 (01)	16
2 (10)	24
3 (11)	32

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
3:2	AGC_FREEZE[1:0]	0 (00)	R/W	Controls when the AGC gain should be frozen.

Setting	Function
0 (00)	Normal operation. Always adjust gain when required.
1 (01)	The gain setting is frozen when a sync word has been found.
2 (10)	Manually freeze the analog gain setting and continue to adjust the digital gain.
3 (11)	Manually freezes both the analog and the digital gain setting. Used for manually overriding the gain.

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
1:0	FILTER_LENGTH[1:0]	1 (01)	R/W	Sets the averaging length for the amplitude from the channel filter. Sets the OOK/ASK decision boundary for OOK/ASK reception.

Setting	Channel Filter Samples	OOK Decision
0 (00)	8	4 dB
1 (01)	16	8 dB
2 (10)	32	12 dB
3 (11)	64	16 dB

0x1E: WOREVT1 - High Byte Event0 Timeout

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	EVENT0[15:8]	128 (0x80)	R/W	High byte of EVENT0 timeout register $t_{Event0} = 1/f_{ACLK} \times EVENT0 \times 2^{5 \times WOR_RES}$

0x1F: WOREVT0 - Low Byte Event0 Timeout

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	EVENT0[7:0]	0 (0x00)	R/W	Low byte of EVENT0 timeout register. The default EVENT0 value gives 1 second timeout, assuming f_{ACLK}=32kHz.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x20: WORCTRL - Wake On Radio Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	ACLK_PD	1	R/W	ACLK input power down signal. Set to 1 the ACLK input to the WOR timer is disabled, reset to 0 the ACLK input is enabled and ACLK is provided to the WOR and RX Timeout timers.
6:4	EVENT1[2:0]	7 (111)	R/W	Timeout setting from register block. Decoded to Event 1 timeout. The following table lists the number of clock periods after Event 0 before Event 1 times out asuming f_{ACLK} = 32 kHz.
3	Reserved		R1	
2	Reserved		R0	
1:0	WOR_RES	0 (00)	R/W	Controls the Event 0 resolution as well as maximum timeout of the WOR module and maximum timeout under normal RX operation.

Setting	t_{Event1}
0 (000)	4 (0.122 ms)
1 (001)	6 (0.183 ms)
2 (010)	8 (0.244 ms)
3 (011)	12 (0.366 ms)
4 (100)	16 (0.488 ms)
5 (101)	24 (0.732 ms)
6 (110)	32 (0.977 ms)
7 (111)	48 (1.465 ms)

Setting	Resolution (1 LSB)	Maximum Timeout
0 (00)	1 period (~30 µs, f_{ACLK} = 32 kHz)	2 seconds
1 (01)	2^5 periods (~977 µs, f_{ACLK} = 32 kHz)	64 seconds
2 (10)	2^{10} periods (~31 ms, f_{ACLK} = 32 kHz)	34 minutes
3 (11)	2^{15} periods (1 s, f_{ACLK} = 32 kHz)	18.2 hours

NOTE: WOR_RES should be 0 or 1 when using WOR, because WOR_RES > 1 results in a very low duty cycle.

In normal RX operation all settings of WOR_RES can be used.

0x21: FREND1 - Front End RX Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	LNA_CURRENT[1:0]	1 (01)	R/W	Adjusts front-end LNA PTAT current output
5:4	LNA2MIX_CURRENT[1:0]	1 (01)	R/W	Adjusts front-end PTAT outputs
3:2	LODIV_BUF_CURRENT_RX[1:0]	1 (01)	R/W	Adjusts current in RX LO buffer (LO input to mixer)
1:0	MIX_CURRENT[1:0]	2 (10)	R/W	Adjusts current in mixer

0x22: FREND0 - Front End TX Configuration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	Reserved		R0	
5:4	LODIV_BUF_CURRENT_TX[1:0]	1 (0x01)	R/W	Adjusts current TX LO buffer (input to PA). The value to use in this field is given by the SmartRF Studio software.
3	Reserved		R0	
2:0	PA_POWER[2:0]	0 (0x00)	R/W	Selects PA power setting. This value is an index to the PATABLE, which can be programmed with up to 8 different PA settings. In OOK/ASK mode, this selects the PATABLE index to use when transmitting a 1. PATABLE index zero is used in OOK/ASK when transmitting a 0. The PATABLE settings from index 0 to the PA_POWER value are used for ASK TX shaping, and for power ramp-up/ramp-down at the start/end of transmission in all TX modulation formats.

Copyright © 2009–2013, Texas Instruments Incorporated

0x23: FSCAL3 - Frequency Synthesizer Calibration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	FSCAL3[7:6]	2 (0x02)	R/W	Frequency synthesizer calibration configuration. The value to write in this field before calibration is given by the SmartRF Studio software.
5:4	CHP_CURR_CAL_EN[1:0]	2 (0x02)	R/W	Enable charge pump calibration stage when 1
3:0	FSCAL3[3:0]	9 (1001)	R/W	Frequency synthesizer calibration result register. Digital bit vector defining the charge pump output current, on an exponential scale: $I_{OUT} = I_0 \times 2^{FSCAL3[3:0]/4}$ Fast frequency hopping without calibration for each hop can be done by calibrating earlier for each frequency and saving the resulting FSCAL3, FSCAL2, and FSCAL1 register values. Between each frequency hop, calibration can be replaced by writing the FSCAL3, FSCAL2, and FSCAL1 register values corresponding to the next RF frequency.

0x24: FSCAL2 - Frequency Synthesizer Calibration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	Reserved		R0	
5	VCO_CORE_H_EN	0	R/W	Choose high (1) / low (0) VCO
4:0	FSCAL2[4:0]	10 (0x0A)	R/W	Frequency synthesizer calibration result register. VCO current calibration result and override value. Fast frequency hopping without calibration for each hop can be done by calibrating earlier for each frequency and saving the resulting FSCAL3, FSCAL2, and FSCAL1 register values. Between each frequency hop, calibration can be replaced by writing the FSCAL3, FSCAL2, and FSCAL1 register values corresponding to the next RF frequency.

0x25: FSCAL1 - Frequency Synthesizer Calibration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:6	Reserved		R0	
5:0	FSCAL1[5:0]	32 (0x20)	R/W	Frequency synthesizer calibration result register. Capacitor array setting for VCO coarse tuning. Fast frequency hopping without calibration for each hop can be done by calibrating earlier for each frequency and saving the resulting FSCAL3, FSCAL2, and FSCAL1 register values. Between each frequency hop, calibration can be replaced by writing the FSCAL3, FSCAL2, and FSCAL1 register values corresponding to the next RF frequency.

0x26: FSCAL0 - Frequency Synthesizer Calibration

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	Reserved		R0	
6:0	FSCAL0[6:0]	13 (0x0D)	R/W	Frequency synthesizer calibration control. The value to use in this register is given by the SmartRF Studio software.

0x29: FSTEST - Frequency Synthesizer Calibration Control

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	FSTEST[7:0]	89 (0x59)	R/W	For test only. Do not write to this register.

0x2A: PTEST - Production Test

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	PTEST[7:0]	127 (0x7F)	R/W	For test only. Do not write to this register.

0x2B: AGCTEST - AGC Test

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	AGCTEST[7:0]	63 (0x3F)	R/W	For test only. Do not write to this register.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x2C: TEST2 - Various Test Settings

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	TEST2[7:0]	136 (0x88)	R/W	The value to use in this register is given by the SmartRF Studio software. This register is forced to 0x88 or 0x81 when it wakes up from SLEEP mode, depending on the configuration of FIFOTHR. ADC_RETENTION. Note that the value read from this register when waking up from SLEEP always is the reset value (0x88) regardless of the ADC_RETENTION setting. The inverting of some of the bits due to the ADC_RETENTION setting is only seen INTERNALLY in the analog part.

0x2D: TEST1 - Various Test Settings

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	TEST1[7:0]	49 (0x31)	R/W	The value to use in this register is given by the SmartRF Studio software. This register is forced to 0x31 or 0x35 when it wakes up from SLEEP mode, depending on the configuration of FIFOTHR. ADC_RETENTION. Note that the value read from this register when waking up from SLEEP always is the reset value (0x31) regardless of the ADC_RETENTION setting. The inverting of some of the bits due to the ADC_RETENTION setting is only seen INTERNALLY in the analog part.

0x2E: TEST0 - Various Test Settings

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:2	TEST0[7:2]	2 (0x02)	R/W	The value to use in this register is given by the SmartRF Studio software.
1	VCO_SEL_CAL_EN	1	R/W	Enable VCO selection calibration stage when 1
0	TEST0[0]	1	R/W	The value to use in this register is given by the SmartRF Studio software.

Copyright © 2009–2013, Texas Instruments Incorporated

25.3.5.2 Radio Core Status Register Details

0x30 (0xF0): PARTNUM - Chip ID

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	PARTNUM[7:0]	0 (0x00)	R	Chip part number

0x31 (0xF1): VERSION - Chip ID

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	VERSION[7:0]	6 (0x06)	R	Chip version number

0x32 (0xF2): FREQEST - Frequency Offset Estimate From Demodulator

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	FREQOFF_EST		R	The estimated frequency offset (2s complement) of the carrier. Resolution is $f_{RFXT2}/2^{14}$ (1.59 to 1.65 kHz). Range is ±202 kHz to ±210 kHz, dependent on RF XT2 crystal frequency. Frequency offset compensation is only supported for 2-FSK, 2-GFSK, and MSK modulation. This register reads 0 when using ASK or OOK modulation.

0x33 (0xF3): LQI - Demodulator Estimate for Link Quality

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	CRC OK		R	The last CRC comparison matched. Cleared when entering/restarting RX mode.
6:0	LQI_EST[6:0]		R	The Link Quality Indicator estimates how easily a received signal can be demodulated. Calculated over the 64 symbols following the sync word.

0x34 (0xF4): RSSI - Received Signal Strength Indication

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	RSSI		R	Received signal strength indicator

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x35 (0xF5): MARCSTATE - Main Radio Control State Machine State

BIT	FIELD NAME	RESET	R/W	DESCRIPTION		
7:5	Reserved		R0			
4:0	MARC_STATE[4:0]		R	Main radio control FSM state		
				Value	**State Name**	**State** (see Figure 25-7)
				0 (0x00)	SLEEP	SLEEP
				1 (0x01)	IDLE	IDLE
				2 (0x02)	Reserved. (CC1101: XOFF)	Reserved. (CC1101: XOFF)
				3 (0x03)	VCOON_MC	MANCAL
				4 (0x04)	REGON_MC	MANCAL
				5 (0x05)	MANCAL	MANCAL
				6 (0x06)	VCOON	FS_WAKEUP
				7 (0x07)	REGON	FS_WAKEUP
				8 (0x08)	STARTCAL	CALIBRATE
				9 (0x09)	BWBOOST	SETTLING
				10 (0x0A)	FS_LOCK	SETTLING
				11 (0x0B)	IFADCON	SETTLING
				12 (0x0C)	ENDCAL	CALIBRATE
				13 (0x0D)	RX	RX
				14 (0x0E)	RX_END	RX
				15 (0x0F)	RX_RST	RX
				16 (0x10)	TXRX_SWITCH	TXRX_SETTLING
				17 (0x11)	RX_OVERFLOW	RX_OVERFLOW
				18 (0x12)	FSTXON	FSTXON
				19 (0x13)	TX	TX
				20 (0x14)	TX_END	TX
				21 (0x15)	RXTX_SWITCH	RXTX_SETTLING
				22 (0x16)	TX_UNDERFLOW	TX_UNDERFLOW

0x36 (0xF6): WORTIME1 - High Byte of WOR Time

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	TIME[15:8]		R	High byte of timer value in WOR module

0x37 (0xF7): WORTIME0 - Low Byte of WOR Time

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	TIME[7:0]		R	Low byte of timer value in WOR module

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

0x38 (0xF8): PKTSTATUS - Current GDOx Status and Packet Status

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	CRC_OK		R	The last CRC comparison matched. Cleared when entering/restarting RX mode.
6	CS		R	Carrier sense
5	PQT_REACHED		R	Preamble Quality reached
4	CCA		R	Channel is clear
3	SFD		R	Sync word found. Asserted when sync word has been sent or received, and deasserted at the end of the packet. In RX, this bit deasserts when the optional address check fails or the radio enter RX_OVERFLOW state. In TX this bit deasserts if the radio enters TX_UNDERFLOW state.
2	GDO2		R	Current GDO2 value. Note: the reading gives the noninverted value irrespective of what IOCFG2.GDO2_INV is programmed to. It is not recommended to check for PLL lock by reading PKTSTATUS[2] with GDO2_CFG = 0x0A.
1	Reserved		R0	
0	GDO0		R	Current GDO0 value. Note: Gives the noninverted value, regardless of the IOCFG0.GDO0_INV setting. It is not recommended to check for PLL lock by reading PKTSTATUS[0] with GDO0_CFG = 0x0A.

0x39 (0xF9): VCO_VC_DAC - Current Setting from PLL Calibration Module

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7:0	VCO_VC_DAC[7:0]		R	Status register for test only

0x3A (0xFA): TXBYTES - Underflow and Number of Bytes

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	TX_UNDERFLOW		R	
6:0	NUM_TXBYTES		R	Number of bytes in TX FIFO

0x3B (0xFB): RXBYTES - Overflow and Number of Bytes

BIT	FIELD NAME	RESET	R/W	DESCRIPTION
7	RX_OVERFLOW		R	
6:0	NUM_RXBYTES		R	Number of bytes in RX FIFO

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.4 RF1A Registers

The radio module (RF1A) registers are listed in Table 25-22.

Table 25-22. RF1A Registers

Offset	Acronym	Register Name	Type	Access	Reset
000h	RF1AIFCTL0	Radio interface control register 0	Read/write	Word	Reset with PUC
002h	RF1AIFCTL1	Radio interface control register 1	Read/write	Word	Reset with PUC
002h	RF1AIFIFG	Radio interface interrupt flag register	Read/write	Byte	Reset with PUC
003h	RF1AIFIE	Radio interface interrupt enable register	Read/write	Byte	Reset with PUC
004h		Reserved			
006h	RF1AIFERR	Radio interface error flag register	Read/write-Word	Word	Reset with PUC
008h		Reserved			
00Ah		Reserved			
00Ch	RF1AIFERRV	Radio interface error vector word register	Read/write	Word	Reset with PUC
00Eh	RF1AIFIV	Radio interface interrupt vector word register	Read/write	Word	Reset with PUC
010h	RF1AINSTRW	Radio instruction word register	Read/write	Word	Reset with PUC
010h	RF1ADINB	Radio byte data in register	Read/write	Byte	Reset with PUC
011h	RF1AINSTRB	Radio instruction byte register	Read/write	Byte	Reset with PUC
012h	RF1AINSTR1W	Radio instruction word register with 1-byte auto-read (low-byte ignored)	Read/write	Word	Reset with PUC
012h		Any write is ignored. Always reads 0.	Read/write	Byte	Reset with PUC
013h	RF1AINSTR1B	Radio instruction byte register with 1-byte auto-read	Read/write	Byte	Reset with PUC
014h	RF1AINSTR2W	Radio instruction word register with 2-byte auto-read (low-byte ignored)	Read/write	Word	Reset with PUC
014h		Any write is ignored. Always reads 0.	Read/write	Byte	Reset with PUC
015h	RF1AINSTR2B	Radio instruction byte register with 2-byte auto-read	Read/write	Byte	Reset with PUC
016h	RF1ADINW	Radio word data in register	Read/write	Word	Reset with PUC
018h		Reserved			
01Ah		Reserved			
01Ch		Reserved			
01Eh		Reserved			
020h	RF1ASTATW (alias: RF1ASTAT0W)	Radio status word register without auto-read	Read/write	Word	Reset with PUC
020h	RF1ADOUTB (alias: RF1ADOUT0B)	Radio byte data out register without auto-read	Read/write	Byte	Reset with PUC
021h	RF1ASTATB (alias: RF1ASTAT0B)	Radio status byte register without auto-read	Read/write	Byte	Reset with PUC
022h	RF1ASTAT1W	Radio status word register with 1-byte auto-read	Read/write	Word	Reset with PUC
022h	RF1ADOUT1B	Radio byte data out register with 1-byte auto-read	Read/write	Byte	Reset with PUC
023h	RF1ASTAT1B	Radio status byte register with 1-byte auto-read	Read/write	Byte	Reset with PUC
024h	RF1ASTAT2W	Radio status word register with 2-byte auto-read	Read/write	Word	Reset with PUC
024h	RF1ADOUT2B	Radio byte data out register with 2-byte auto-read	Read/write	Byte	Reset with PUC
025h	RF1ASTAT2B	Radio status byte register with 2-byte auto-read	Read/write	Byte	Reset with PUC
026h		Reserved			
028h	RF1ADOUTW (alias: RF1ADOUT0W)	Radio core word data out register without auto-read	Read/write	Word	Reset with PUC
02Ah	RF1ADOUT1W	Radio core word data out register with 1-byte auto-read	Read/write	Word	Reset with PUC
02Ch	RF1ADOUT2W	Radio core word data out register with 2-byte auto-read	Read/write	Word	Reset with PUC
02Eh		Reserved			
030h	RF1AIN	Radio core signal input register	Read/write	Word	Reset with PUC

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 25-22. RF1A Registers (continued)

Offset	Acronym	Register Name	Type	Access	Reset
032h	RF1AIFG	Radio core interrupt flag register	Read/write	Word	Reset with PUC
034h	RF1AIES	Radio core interrupt edge select register	Read/write	Word	Reset with PUC
036h	RF1AIE	Radio core interrupt enable register	Read/write	Word	Reset with PUC
038h	RF1AIV	Radio core interrupt vector word register	Read/write	Word	Reset with PUC
03Ah		Reserved			
03Ch		Reserved			
03Eh		Reserved			

Copyright © 2009–2013, Texas Instruments Incorporated

25.4.1 RF1AIFCTL0 Register

Radio Interface Control Register 0

Figure 25-14. RF1AIFCTL0 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved						RFENDIAN	Reserved
r0	r0	r0	r0	r0	r0	rw-0	r0

Table 25-23. RF1AIFCTL0 Register Description

Bit	Field	Type	Reset	Description
15-2	Reserved	R	0h	Reserved. Always reads as 0.
1	RFENDIAN	RW	0h	Disable endianness conversion 0b = MSP430 little-endian words and double-words are converted to the endianness of the radio core (CC1101: big-endian). 1b = Word and double-words are not converted.
0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

25.4.2 RF1AIFCTL1 Register

Radio Interface Control Register 1

Figure 25-15. RF1AIFCTL1 Register

15	14	13	12	11	10	9	8
RFDOUTIE	RFSTATIE	RFDINIE	RFINSTRIE	Reserved	RFERRIE	Reserved	Reserved
rw-0	rw-0	rw-0	rw-0	r0	rw-0	r0	r0
7	6	5	4	3	2	1	0
RFDOUTIFG	RFSTATIFG	RFDINIFG	RFINSTRIFG	Reserved	RFERRIFG	Reserved	Reserved
rw-0	rw-0	rw-0	rw-1	r0	r-0	r0	r0

Table 25-24. RF1AIFCTL1 Register Description

Bit	Field	Type	Reset	Description
15	RFDOUTIE	RW	0h	Radio interface data out interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
14	RFSTATIE	RW	0h	Radio interface status interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
13	RFDINIE	RW	0h	Radio interface data in interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
12	RFINSTRIE	RW	0h	Radio interface instruction interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
11	Reserved	R	0h	Reserved. Always reads as 0.
10	RFERRIE	RW	0h	Radio interface error interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
9-8	Reserved	R	0h	Reserved. Always reads as 0.
7	RFDOUTIFG	RW	0h	Radio interface data out interrupt flag 0b = No interrupt pending 1b = Interrupt pending
6	RFSTATIFG	RW	0h	Radio interface status interrupt flag 0b = No interrupt pending 1b = Interrupt pending
5	RFDINIFG	RW	0h	Radio interface data in interrupt flag 0b = No interrupt pending 1b = Interrupt pending
4	RFINSTRIFG	RW	0h	Radio interface instruction interrupt flag 0b = No interrupt pending 1b = Interrupt pending
3	Reserved	R	0h	Reserved. Always reads as 0.
2	RFERRIFG	RW	0h	Radio interface error interrupt flag. The error interrupt flag is set as long as one of the error flags is set. It is automatically cleared when all error flags are cleared. 0b = No interrupt pending 1b = Interrupt pending
1-0	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.4.3 RF1AIFERR Register

Radio Interface Error Flag Register

Figure 25-16. RF1AIFERR Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved				OPOVERR	OUTERR	OPERR	LVERR
r0	r0	r0	r0	rw-0	rw-0	rw-0	rw-0

Table 25-25. RF1AIFERR Register Description

Bit	Field	Type	Reset	Description
15-4	Reserved	R	0h	Reserved. Always reads as 0.
3	OPOVERR	RW	0h	Operand overwrite error flag 0b = No error occurred 1b = Error occurred
2	OUTERR	RW	0h	Output data not available error flag 0b = No error occurred 1b = Error occurred
1	OPERR	RW	0h	Operand error flag 0b = No error occurred 1b = Error occurred
0	LVERR	RW	0h	Low core voltage error flag 0b = No error occurred 1b = Error occurred

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.4.4 RF1AIFERRV Register

Radio Interface Error Vector Register

Figure 25-17. RF1AIFERRV Register

15	14	13	12	11	10	9	8
RF1AIFERRVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
RF1AIFERRVx							
r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 25-26. RF1AIFERRV Register Description

Bit	Field	Type	Reset	Description
15-0	RF1AIFERRVx	R	0h	Radio interface error vector value 00h = No error 02h = Error Source: Low core voltage error; Error Flag: LVERR; Error Priority: Highest 04h = Error Source: Operand error; Error Flag: OPERR 06h = Error Source: Output data not available error; Error Flag: OUTERR 08h = Error Source: Operand overwrite error; Error Flag: OPOVERR; Error Priority: Lowest

25.4.5 RF1AIFIV Register

Radio Interface Interrupt Vector Register

Figure 25-18. RF1AIFIV Register

15	14	13	12	11	10	9	8
RF1AIFIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
RF1AIFIVx							
r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 25-27. RF1AIFIV Register Description

Bit	Field	Type	Reset	Description
15-0	RF1AIFIVx	R	0h	Radio interface interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Radio interface error; Interrupt Flag: RFERRIFG; Interrupt Priority: Highest 04h = Interrupt Source: Radio interface data out; Interrupt Flag: RFDOUTIFG 06h = Interrupt Source: Radio interface status out; Interrupt Flag: RFSTATIFG 08h = Interrupt Source: Radio interface data in; Interrupt Flag: RFDINIFG 0Ah = Interrupt Source: Radio interface instruction in; Interrupt Flag: RFINSTRIFG 0Ch = Reserved 0Eh = Reserved; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.4.6 RF1AIN Register

Radio Core Signal Input Register

Figure 25-19. RF1AIN Register

15	14	13	12	11	10	9	8
RFINx							
r	r	r	r	r	r	r	r
7	6	5	4	3	2	1	0
RFINx							
r	r	r	r	r	r	r	r

Table 25-28. RF1AIN Register Description

Bit	Field	Type	Reset	Description
15-0	RFINx	RW	undefined	Radio core signal input 0b = Current state of signal is low. 1b = Current state of signal is high.

25.4.7 RF1AIFG Register

Radio Core Interrupt Flag Register

Figure 25-20. RF1AIFG Register

15	14	13	12	11	10	9	8
RFIFGx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
RFIFGx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 25-29. RF1AIFG Register Description

Bit	Field	Type	Reset	Description
15-0	RFIFGx	RW	0h	Radio core interrupt flags 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

25.4.8 RF1AIES Register

Radio Core Interrupt Edge Select Register

Figure 25-21. RF1AIES Register

15	14	13	12	11	10	9	8
RFIESx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
RFIESx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 25-30. RF1AIES Register Description

Bit	Field	Type	Reset	Description
15-0	RFIESx	RW	0h	Radio core interrupt edge select 0b = Low-to-high transition sets interrupt flag 1b = High-to-low transition sets interrupt flag

25.4.9 RF1AIE Register

Radio Core Interrupt Enable Register

Figure 25-22. RF1AIE Register

15	14	13	12	11	10	9	8
RFIEx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
RFIEx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 25-31. RF1AIE Register Description

Bit	Field	Type	Reset	Description
15-0	RFIEx	RW	0h	Radio core interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

Copyright © 2009–2013, Texas Instruments Incorporated

25.4.10 RF1AIV Register

Radio Core Interrupt Vector Register

Figure 25-23. RF1AIV Register

15	14	13	12	11	10	9	8
RF1AIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
RF1AIVx							
r0	r0	r-(0)	r-(0)	r-(0)	r-(0)	r-(0)	r0

Table 25-32. RF1AIV Register Description

Bit	Field	Type	Reset	Description
15-0	RF1AIVx	R	0h	Radio core interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: Radio core signal 0; Interrupt Flag: RFIFG0; Interrupt Priority: Highest 04h = Interrupt Source: Radio core signal 1; Interrupt Flag: RFIFG1 06h = Interrupt Source: Radio core signal 2; Interrupt Flag: RFIFG2 08h = Interrupt Source: Radio core signal 3; Interrupt Flag: RFIFG3 0Ah = Interrupt Source: Radio core signal 4; Interrupt Flag: RFIFG4 0Ch = Interrupt Source: Radio core signal 5; Interrupt Flag: RFIFG5 0Eh = Interrupt Source: Radio core signal 6; Interrupt Flag: RFIFG6 10h = Interrupt Source: Radio core signal 7; Interrupt Flag: RFIFG7 12h = Interrupt Source: Radio core signal 8; Interrupt Flag: RFIFG8 14h = Interrupt Source: Radio core signal 9; Interrupt Flag: RFIFG9 16h = Interrupt Source: Radio core signal 10; Interrupt Flag: RFIFG10 18h = Interrupt Source: Radio core signal 11; Interrupt Flag: RFIFG11 1Ah = Interrupt Source: Radio core signal 12; Interrupt Flag: RFIFG12 1Ch = Interrupt Source: Radio core signal 13; Interrupt Flag: RFIFG13 1Eh = Interrupt Source: Radio core signal 14; Interrupt Flag: RFIFG14 20h = Interrupt Source: Radio core signal 15; Interrupt Flag: RFIFG15; Lowest Priority: Highest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

LCD_B Controller

The LCD_B controller drives static, 2-mux, 3-mux, or 4-mux LCDs. This chapter describes the LCD_B controller.

Topic **Page**

26.1 LCD_B Controller Introduction **742**
26.2 LCD_B Controller Operation **744**
26.3 LCD_B Registers **762**

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.1 LCD_B Controller Introduction

The LCD_B controller directly drives LCD displays by creating the ac segment and common voltage signals automatically. The LCD_B controller can support static, 2-mux, 3-mux, and 4-mux LCD glasses.

The LCD_B controller features are:

- Display memory
- Automatic signal generation
- Configurable frame frequency
- Blinking of individual segments with separate blinking memory
- Regulated charge pump
- Contrast control by software
- Support for four types of LCDs
  - Static
  - 2-mux, 1/2 bias or 1/3 bias
  - 3-mux, 1/2 bias or 1/3 bias
  - 4-mux, 1/2 bias or 1/3 bias

The LCD_B controller block diagram for a configuration with a maximum of 160 segments is shown in Figure 26-1.

NOTE: Maximum LCD Segment Control

The maximum number of segment lines and memory registers available differs with device. See the device-specific data sheet for available segment pins and the maximum number of segments supported.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

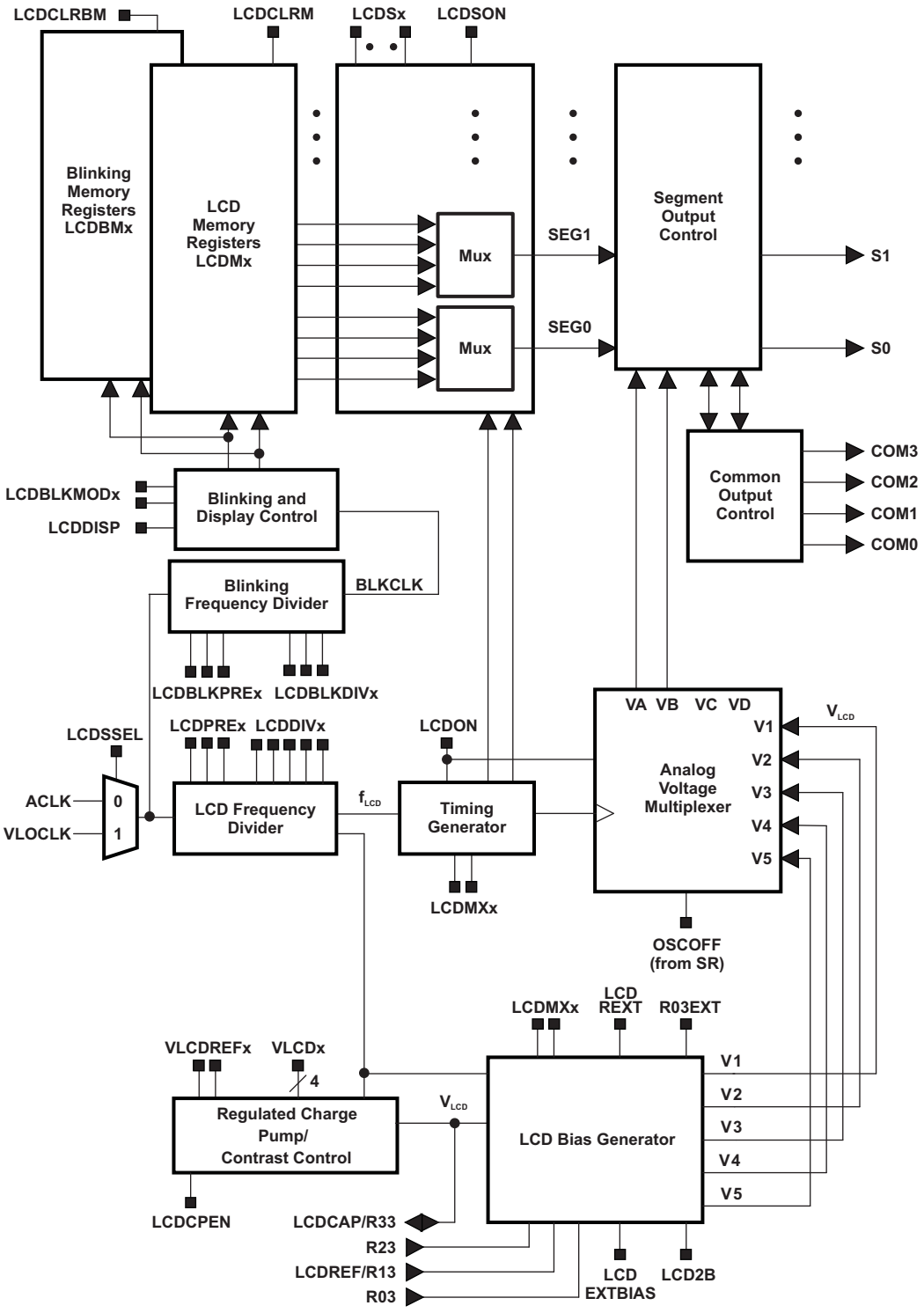


Figure 26-1. LCD_B Controller Block Diagram

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.2 LCD_B Controller Operation

The LCD_B controller is configured with user software. The setup and operation of the LCD_B controller is discussed in the following sections.

26.2.1 LCD Memory

The LCD memory map for a device with a 160-segment maximum is shown in Figure 26-2. Each memory bit corresponds to one LCD segment or is not used, depending on the mode. To turn on an LCD segment, its corresponding memory bit is set.

The memory can also be accessed word-wise using the even addresses starting at LCDM1, LCDM3, etc.

Setting the bit LCDCLRM clears all LCD memory registers at the next frame boundary. It is reset automatically after the registers are cleared.

Associated Common Pins	3	2	1	0	3	2	1	0		
Register	7							0	n	Associated Segment Pins
LCDM20	--	--	--	--	--	--	--	--	38	39, 38
LCDM19	--	--	--	--	--	--	--	--	36	37, 36
LCDM18	--	--	--	--	--	--	--	--	34	35, 34
LCDM17	--	--	--	--	--	--	--	--	32	33, 32
LCDM16	--	--	--	--	--	--	--	--	30	31, 30
LCDM15	--	--	--	--	--	--	--	--	28	29, 28
LCDM14	--	--	--	--	--	--	--	--	26	27, 26
LCDM13	--	--	--	--	--	--	--	--	24	25, 24
LCDM12	--	--	--	--	--	--	--	--	22	23, 22
LCDM11	--	--	--	--	--	--	--	--	20	21, 20
LCDM10	--	--	--	--	--	--	--	--	18	19, 18
LCDM9	--	--	--	--	--	--	--	--	16	17, 16
LCDM8	--	--	--	--	--	--	--	--	14	15, 14
LCDM7	--	--	--	--	--	--	--	--	12	13, 12
LCDM6	--	--	--	--	--	--	--	--	10	1, 10
LCDM5	--	--	--	--	--	--	--	--	8	9, 8
LCDM4	--	--	--	--	--	--	--	--	6	7, 6
LCDM3	--	--	--	--	--	--	--	--	4	5, 4
LCDM2	--	--	--	--	--	--	--	--	2	3, 2
LCDM1	--	--	--	--	--	--	--	--	0	1, 0
	Sn+1				Sn					

Figure 26-2. LCD Memory - Example for 160 Segments Maximum

26.2.2 LCD Timing Generation

The LCD_B controller uses the f_{LCD} signal from the integrated clock divider to generate the timing for common and segment lines. With the LCDSSEL bit ACLK with a frequency between 30 kHz and 40 kHz or VLOCLK can be selected as clock source into the divider. The f_{LCD} frequency is selected with the LCDPREx and LCDDIVx bits. The resulting f_{LCD} frequency is calculated by:

$$f_{LCD} = \frac{f_{ACLK/VLOCLK}}{(LCDDIVx + 1) \times 2^{LCDPRE}}$$

The proper f_{LCD} frequency depends on the LCD's requirement for framing frequency and the LCD multiplex rate and is calculated by:

$$f_{LCD} = 2 \times mux \times f_{Frame}$$

Copyright © 2009–2013, Texas Instruments Incorporated

For example, to calculate f_{LCD} for a 3-mux LCD, with a frame frequency of 30 Hz to 100 Hz:

f_{Frame} (from LCD data sheet) = 30 Hz to 100 Hz
$f_{LCD} = 2 \times 3 \times f_{Frame}$
f_{LCD}(min) = 180 Hz
f_{LCD}(max) = 600 Hz

With $f_{ACLK/VLOCLK}$ = 32768 Hz, LCDPREx = 011, and LCDDIVx = 10101:

f_{LCD} = 32768 Hz / ((21+1) × 2^3) = 32768 Hz / 176 = 186 Hz

With LCDPREx = 001 and LCDDIVx = 11011:

f_{LCD} = 32768 Hz / ((27+1) × 2^1) = 32768 Hz / 56 = 585 Hz

The lowest frequency has the lowest current consumption. The highest frequency has the least flicker.

26.2.3 Blanking the LCD

The LCD controller allows to blank the complete LCD. The LCDSON bit is ANDed with each segment's memory bit. When LCDSON = 1, each segment is on or off according to its bit value. When LCDSON = 0, each LCD segment is off.

26.2.4 LCD Blinking

The LCD_B controller also supports blinking. The blinking mode LCDBLKMODx = 01 allows to blink individual segments, with LCDBLKMODx = 10 all segments are blinking, and with LCDBLKMODx = 00 blinking is disabled.

26.2.4.1 Blinking Memory

To enable individual segments for blinking the corresponding bit in the blinking memory LCDBMx registers needs to be set. The memory uses the same structure as the LCD memory shown in Figure 26-2. Each memory bit corresponds to one LCD segment, or is not used, depending on the multiplexing mode LCDMXx. To enable blinking for a LCD segment, its corresponding memory bit is set.

The blinking memory can also be accessed word-wise using the even addresses starting at LCDBM1, LCDBM3, etc.

Setting the bit LCDCLRBM clears all blinking memory registers at the next frame boundary. It is automatically reset after the registers are cleared.

26.2.4.2 Blinking Frequency

The blinking frequency f_{BLINK} is selected with the LCDBLKPREx and LCDBLKDIVx bits. The same clock is used as selected for the LCD frequency f_{LCD}. The resulting f_{BLINK} frequency is calculated by:

$$f_{Blink} = \frac{f_{ACLK/VLO}}{(LCDBLKDIVx + 1) \times 2^{9+LCDBLKPREx}}$$

The divider generating the blinking frequency f_{BLINK} is reset while LCDBLKMODx = 00. After a blinking mode LCDBLKMODx = 01 or 10 is selected, the enabled segments or all segments go blank at the next frame boundary and stay off for half a BLKCLK period. Then they go active at the next frame boundary and stay on for another half BLKCLK period before they go blank again at a frame boundary.

NOTE: Blinking Frequency Restrictions

The blinking frequency must be smaller than the frame frequency, f_{Frame}.

The blinking frequency should be changed only when LCDBLKMODx = 00.

26.2.4.3 Dual Display Memory

The blinking memory can also be used as a secondary display memory when no blinking mode LCDBLKMODx = 01 or 10 is selected. The memory to be displayed can be selected either manually using the LCDDISP bit or automatically with LCDBLKMODx = 11.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

With LCDDISP = 0 the LCD memory is selected, with LCDDISP = 1 the blinking memory is selected as display memory. Switching between the memories is synchronized to the frame boundaries.

With LCDBLKMODx = 11 the LCD controller switches automatically between the memories using the divider to generate the blinking frequency. After LCDBLKMODx = 11 is selected the memory to be displayed for the first half a BLKCLK period is the LCD memory. In the second half the blinking memory is used as display memory. Switching between the memories is synchronized to the frame boundaries.

26.2.5 LCD_B Voltage And Bias Generation

The LCD_B module allows selectable sources for the peak output waveform voltage, V1, as well as the fractional LCD biasing voltages V2 to V5. V_{LCD} may be sourced from V_{CC}, an internal charge pump, or externally.

All internal voltage generation is disabled if the selected clock source (ACLK or VLOCLK) is turned off (OSCOFF = 1) or the LCD_B module is disabled (LCDON = 0).

26.2.5.1 LCD Voltage Selection

V_{LCD} is sourced from V_{CC} when VLCDEXT = 0, VLCDx = 0, and VREFx = 0. V_{LCD} is sourced from the internal charge pump when VLCDEXT = 0, VLCDCPEN = 1, and VLCDx > 0. The charge pump is always sourced from DV_{CC}. The VLCDx bits provide a software selectable LCD voltage from 2.6 V to 3.44 V (typical) independent of DV_{CC}. See the device-specific data sheet for specifications.

When the internal charge pump is used, a 4.7-µF or larger capacitor must be connected between pin LCDCAP and ground. If no capacitor is connected and the charge pump is enabled, the LCDNOCAPIFG interrupt flag is set, and the charge pump is disabled to prevent damage to the device. The charge pump may be temporarily disabled by setting LCDCPEN = 0 with VLCDx > 0 to reduce system noise, or it can be automatically disabled during certain periods by setting the corresponding bits in the LCDBCPCTL register. In this case, the voltage present at the external capacitor is used for the LCD voltages until the charge pump is re-enabled.

NOTE: Capacitor Required For Internal Charge Pump

A 4.7-µF or larger capacitor must be connected from pin LCDCAP to ground when the internal charge pump is enabled. If no capacitor is connected, the LCDNOCAPIFG interrupt flag is set and the charge pump is disabled.

The internal charge pump may use an external reference voltage when VLCDREFx = 01 (and LCDREXT = 0 and LCDEXTBIAS = 0). In this case, the charge pump voltage is set to a multiply of the external reference voltage according to the VLCDx bits setting.

When VLCDEXT = 1, V_{LCD} is sourced externally from the LCDCAP, pin and the internal charge pump is disabled.

26.2.5.2 LCD Bias Generation

The fractional LCD biasing voltages, V2 to V5 can be generated internally or externally, independent of the source for V_{LCD}. The LCD bias generation block diagram is shown in Figure 26-3.

The internally generated bias voltages V2 to V4 are switched to external pins with LCDREXT = 1.

To source the bias voltages V2 to V4 externally, LCDEXTBIAS is set. This also disables the internal bias generation. Typically, an equally weighted resistor divider is used with resistors ranging from a few kΩ to 1 MΩ, depending on the size of the display. When using an external resistor divider, the V_{LCD} voltage may be sourced from the internal charge pump when VLCDEXT = 0 taking the maximum charge pump load current into account. V5 can also be sourced externally when R03EXT is set to control the contrast of the connected display by changing the voltage at the low end of the external resistor divider as shown in the left part of Figure 26-3.

When using an external resistor divider R33 may serve as a switched V_{LCD} output when VLCDEXT = 0. This allows the power to the resistor ladder to be turned off, eliminating current consumption when the LCD is not used. When VLCDEXT = 1, R33 serves as a V_{LCD} input.

Copyright © 2009–2013, Texas Instruments Incorporated

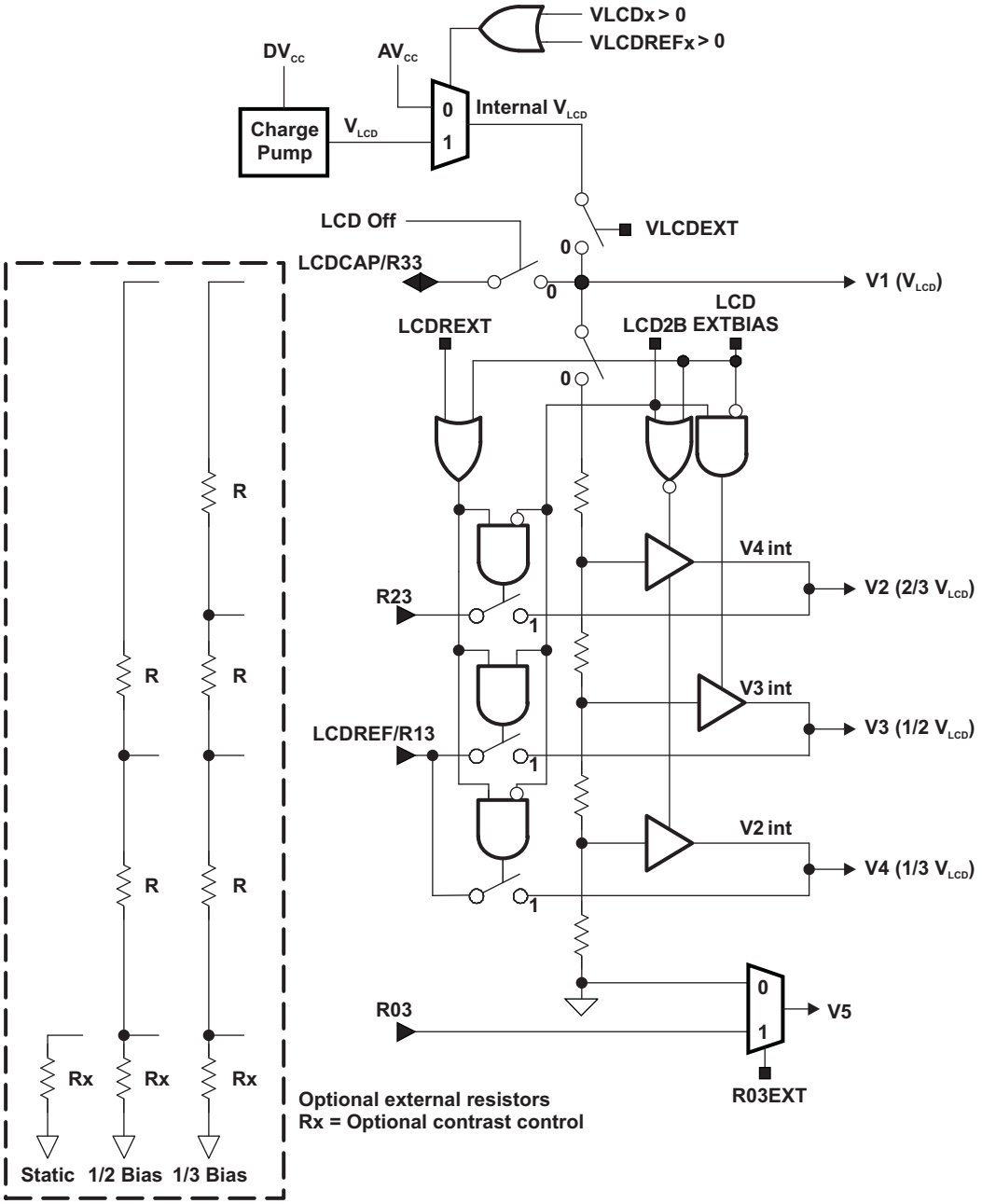


Figure 26-3. Bias Generation

The internal bias generator supports 1/2 bias LCDs when LCD2B = 1, and 1/3 bias LCDs when LCD2B = 0 in 2-mux, 3-mux, and 4-mux modes. In static mode, the internal divider is disabled.

Some devices share the LCDCAP, R33, and R23 functions. In this case, the charge pump cannot be used together with an external resistor divider with 1/3 biasing. When R03 is not available externally, V5 is always V_{SS}.

26.2.5.3 LCD Contrast Control

The peak voltage of the output waveforms together with the selected mode and biasing determine the contrast and the contrast ratio of the LCD. The LCD contrast can be controlled in software by adjusting the LCD voltage generated by the integrated charge pump using the VLCDx settings.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The contrast ratio depends on the used LCD display and the selected biasing scheme. Table 26-1 shows the biasing configurations that apply to the different modes together with the RMS voltages for the segments turned on ($V_{RMS,ON}$) and turned off ($V_{RMS,OFF}$) as functions of V_{LCD}. It also shows the resulting contrast ratios between the on and off states.

Table 26-1. LCD Voltage and Biasing Characteristics

Mode	Bias Config	LCDMx	LCD2B	COM Lines	Voltage Levels	$V_{RMS,OFF}/ V_{LCD}$	$V_{RMS,ON}/ V_{LCD}$	Contrast Ratio $V_{RMS,ON}/ V_{RMS,OFF}$
Static	Static	0	X	1	V1, V5	0	1	1/0
2-mux	1/2	1	1	2	V1, V3, V5	0.354	0.791	2.236
2-mux	1/3	1	0	2	V1, V2, V4, V5	0.333	0.745	2.236
3-mux	1/2	10	1	3	V1, V3, V5	0.408	0.707	1.732
3-mux	1/3	10	0	3	V1, V2, V4, V5	0.333	0.638	1.915
4-mux	1/2	11	1	4	V1, V3, V5	0.433	0.661	1.528
4-mux	1/3	11	0	4	V1, V2, V4, V5	0.333	0.577	1.732

A typical approach to determine the required V_{LCD} is by equating $V_{RMS,OFF}$ with a defined LCD threshold voltage, typically when the LCD exhibits approximately 10% contrast ($V_{th,10\%}$): $V_{RMS,OFF} = V_{th,10\%}$. Using the values for $V_{RMS,OFF}/V_{LCD}$ provided in the table results in $V_{LCD} = V_{th,10\%}/(V_{RMS,OFF}/V_{LCD})$. In the static mode, a suitable choice is V_{LCD} greater or equal than 3 times $V_{th,10\%}$.

In 3-mux and 4-mux mode typically a 1/3 biasing is used but a 1/2 biasing scheme is also possible. The 1/2 bias reduces the contrast ratio but the advantage is a reduction of the required full-scale LCD voltage V_{LCD}.

26.2.6 LCD Outputs

Some LCD segment, common, and Rxx functions are multiplexed with digital I/O functions. These pins can function either as digital I/O or as LCD functions.

The LCD segment functions, when multiplexed with digital I/O, are selected using the LCDSx bits in the LCDBPCTLx registers. The LCDSx bits select the LCD function for each segment line. When LCDSx = 0, a multiplexed pin is set to digital I/O function. When LCDSx = 1, a multiplexed pin is selected as LCD function.

The pin functions for COMx and Rxx, when multiplexed with digital I/O, are selected as described in the port schematic section of the device-specific datasheet. The COM1 to COM3 pins are shared with segment lines. If these pins are required as COM pins due to the selected LCD multiplexing mode the COM functionality takes precedence over the segment function that can be selected for those pins with the LCDSx bits as for all other segment pins.

See the port schematic section of the device-specific data sheet for details on controlling the pin functionality.

NOTE: LCDSx Bits Do Not Affect Dedicated LCD Segment Pins

The LCDSx bits only affect pins with multiplexed LCD segment functions and digital I/O functions. Dedicated LCD segment pins are not affected by the LCDSx bits.

26.2.7 LCD_B Interrupts

The LCD_B module has four interrupt sources available, each with independent enables and flags.

The four interrupt flags, namely LCDFRMIFG, LCDBLKOFFIFG, LCDBLKONIFG, and LCDNOCAPIFG, are prioritized and combined to source a single interrupt vector. The interrupt vector register LCDBIV is used to determine which flag requested an interrupt.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

The highest priority enabled interrupt generates a number in the LCDBIV register (see register description). This number can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled LCD_B interrupts do not affect the LCDBIV value.

Any read access of the LCDBIV register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. A write access to the LCDBIV register automatically resets all pending interrupt flags. In addition, all flags can be cleared via software.

The LCDNOCAPIFG indicates that no capacitor is present at the LCDCAP pin when the charge pump is enabled. Setting the LCDNOCAPIE bit enables the interrupt.

The LCDBLKONIFG is set at the BLKCLK edge synchronized to the frame boundaries that turns on the segments when blinking is enabled with LCDBLKMODx = 01 or 10. It is also set at the BLKCLK edge synchronized to the frame boundaries that selects the blinking memory as display memory when LCDBLKMODx = 11. It is automatically cleared when a LCD or blinking memory register is written. Setting the LCDBLKONIE bit enables the interrupt.

The LCDBLKOFFIFG is set at the BLKCLK edge synchronized to the frame boundaries that blanks the segments when blinking is enabled with LCDBLKMODx = 01 or 10. It is also set at the BLKCLK edge synchronized to the frame boundaries that selects the LCD memory as display memory when LCDBLKMODx = 11.It is automatically cleared when a LCD or blinking memory register is written. Setting the LCDBLKOFFIE bit enables the interrupt.

The LCDFRMIFG is set at a frame boundary. It is automatically cleared when a LCD or blinking memory register is written. Setting the LCDFRMIFGIE bit enables the interrupt.

26.2.7.1 LCDBIV Software Example

The following software example shows the recommended use of LCDBIV and the handling overhead. The LCDBIV value is added to the PC to automatically jump to the appropriate routine.

The numbers at the right margin show the necessary CPU cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles but not the task handling itself.

```
; Interrupt handler for LCD_B interrupt flags.
LCDB_HND             ; Interrupt latency          6
   ADD &LCDBIV,PC    ; Add offset to Jump table   3
   RETI              ; Vector 0: No interrupt     5
   JMP LCDNOCAP_HND  ; Vector 2: LCDNOCAPIFG      2
   JMP LCDBLKON_HND  ; Vector 4: LCDBLKONIFG      2
   JMP LCDBLKOFF_HND ; Vector 6: LCDBLKOFFIFG     2
LCDFRM_HND           ; Vector 8: LCDFRMIFG
   ...    ; Task starts here
   RETI                                          5
LCDNOCAP_HND ; Vector 2: LCDNOCAPIFG
   ... ; Task starts here
   RETI                                          5
LCDBLKON_HND ; Vector 4: LCDBLKONIFG
   ... ; Task starts here
   RETI ; Back to main program                   5
LCDBLKOFF_HND ; Vector 6: LCDBLKOFFIFG
   ... ; Task starts here
   RETI ; Back to main program                   5
```


Copyright © 2009–2013, Texas Instruments Incorporated

26.2.8 Static Mode

In static mode, each MSP430 segment pin drives one LCD segment and one common line, COM0, is used. Figure 26-4 shows some example static waveforms.

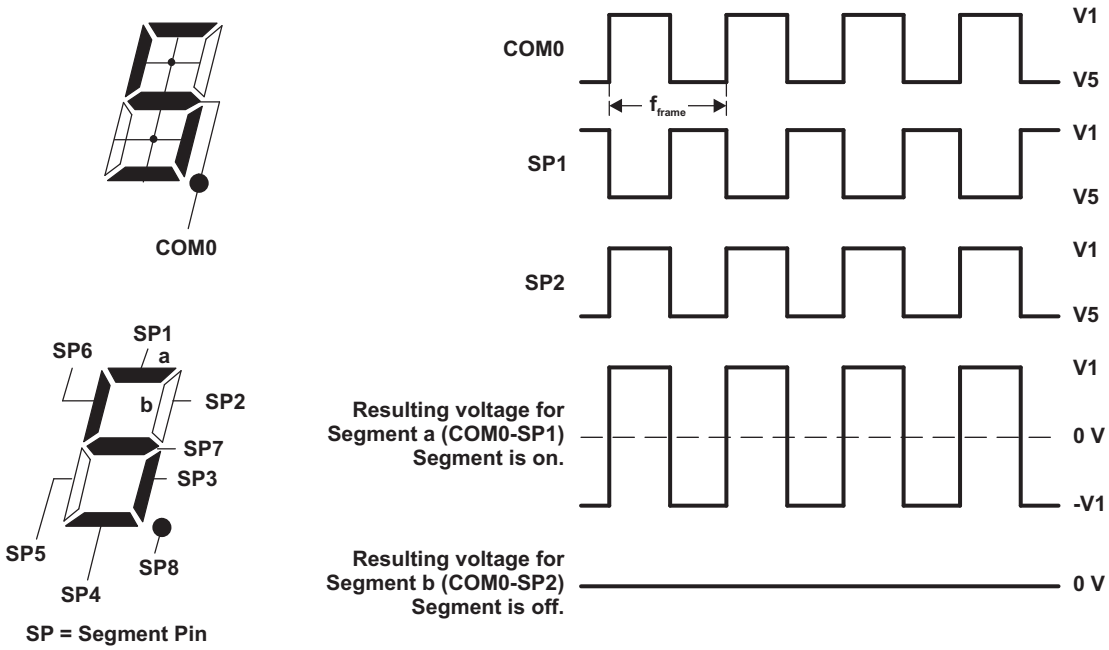


Figure 26-4. Example Static Waveforms

Copyright © 2009–2013, Texas Instruments Incorporated

Figure 26-5 shows an example static LCD, pinout, LCD-to-MSP430 connections, and the resulting segment mapping. This is only an example. Segment mapping in a user's application depends on the LCD pinout and on the MSP430-to-LCD connections.

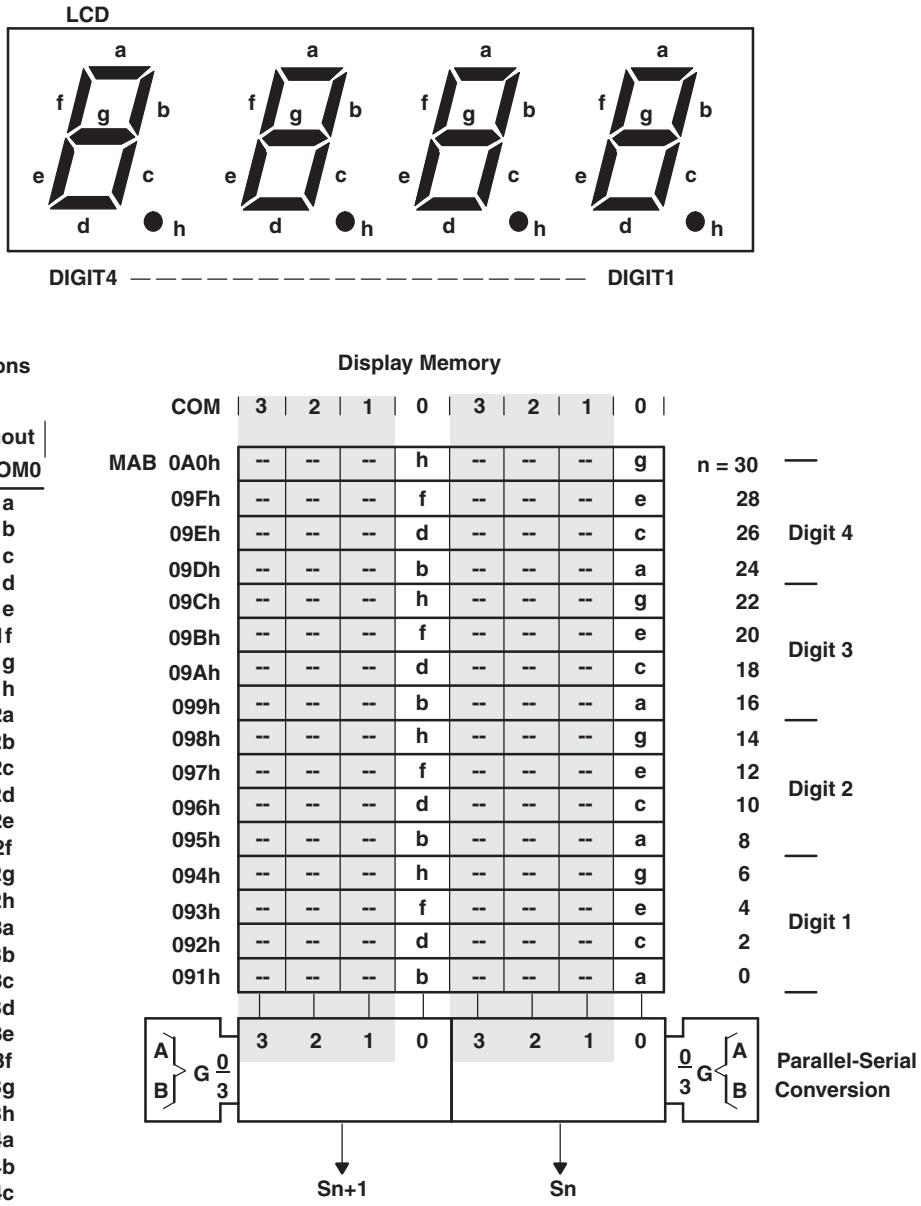


Figure 26-5. Static LCD Example (MAB addresses need to be replaced with LCDMx)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.2.8.1 Static Mode Software Example

```
; All eight segments of a digit are often located in four
; display memory bytes with the static display method.
;
a EQU 001h
b EQU 010h
c EQU 002h
d EQU 020h
e EQU 004h
f EQU 040h
g EQU 008h
h EQU 080h
; The register content of Rx should be displayed.
; The Table represents the 'on'-segments according to the
 ; content of Rx.
   MOV.B Table (Rx),RY ; Load segment information
                       ; into temporary memory.
                       ; (Ry) = 0000 0000 hfdb geca
   MOV.B Ry,&LCDn  ; Note:
                   ; All bits of an LCD memory
                   ; byte are written
   RRA Ry ; (Ry) = 0000 0000 0hfd bgec
   MOV.B Ry,&LCDn+1 ; Note:
                    ; All bits of an LCD memory
                    ; byte are written
   RRA Ry ; (Ry) = 0000 0000 00hf dbge
   MOV.B Ry,&LCDn+2 ; Note:
                    ; All bits of an LCD memory
                    ; byte are written
   RRA Ry ; (Ry) = 0000 0000 000h fdbg
   MOV.B Ry,&LCDn+3 ; Note:
                    ; All bits of an LCD memory
                    ; byte are written
   .......... .......... ; Table
   DB a+b+c+d+e+f ; displays "0"
   DB b+c; ; displays "1"
   .......... ..........
   DB ..........
```


Copyright © 2009–2013, Texas Instruments Incorporated

26.2.9 2-Mux Mode

In 2-mux mode, each MSP430 segment pin drives two LCD segments and two common lines, COM0 and COM1, are used. Figure 26-6 shows some example 2-mux, 1/2 bias waveforms.

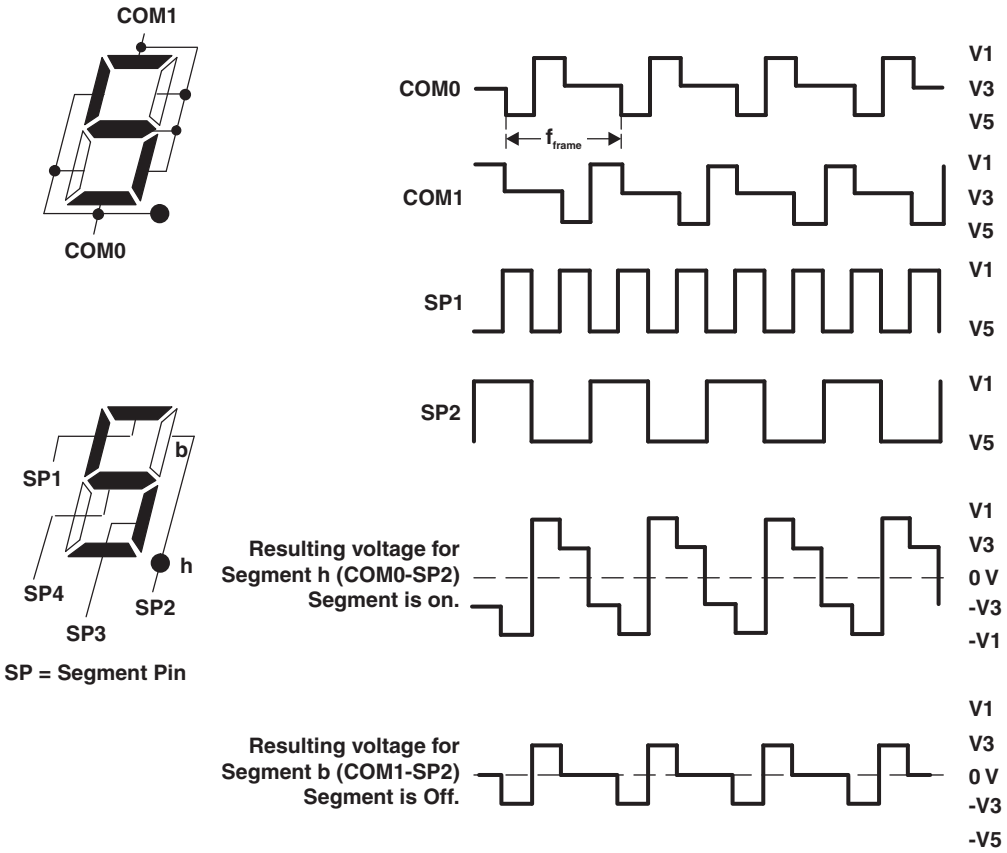


Figure 26-6. Example 2-Mux Waveforms

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Figure 26-7 shows an example 2-mux LCD, pinout, LCD-to-MSP430 connections, and the resulting segment mapping. This is only an example. Segment mapping in a user's application completely depends on the LCD pinout and on the MSP430-to-LCD connections.

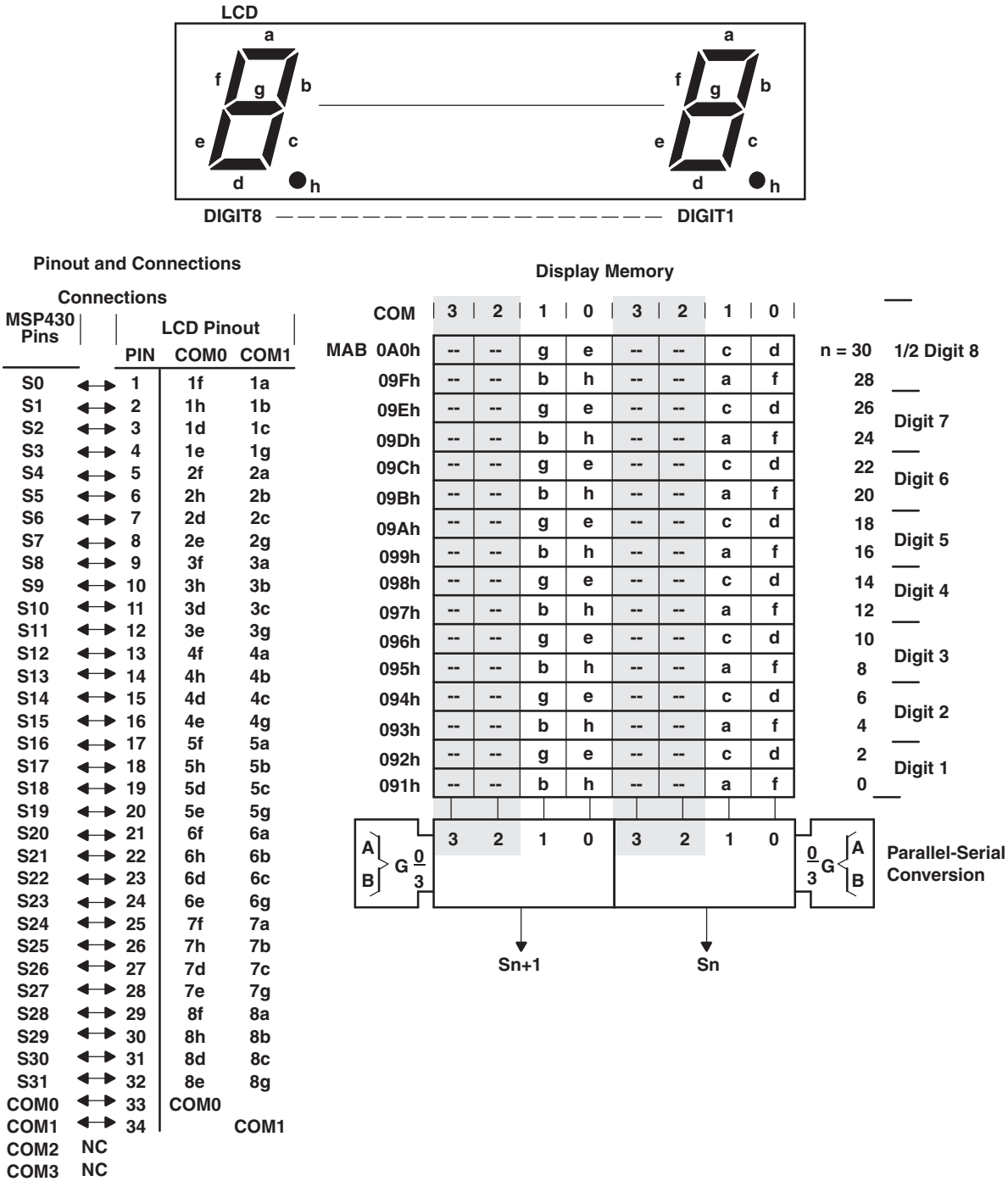


Figure 26-7. 2-Mux LCD Example (MAB addresses need to be replaced with LCDMx)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.2.9.1 2-Mux Mode Software Example

```
; All eight segments of a digit are often located in two
; display memory bytes with the 2-mux display rate ;
a EQU 002h
b EQU 020h
c EQU 008h
d EQU 004h
e EQU 040h
f EQU 001h
g EQU 080h
h EQU 010h
; The register content of Rx should be displayed.
; The Table represents the 'on'-segments according to the
; content of Rx. ;
.......... ..........
   MOV.B Table(Rx),Ry ; Load segment information into
                      ; temporary memory.
   MOV.B Ry,&LCDn ; (Ry) = 0000 0000 gebh cdaf
                      ; Note:
                      ; All bits of an LCD memory byte
                      ; are written
   RRA Ry ; (Ry) = 0000 0000 0geb hcda
   RRA Ry ; (Ry) = 0000 0000 00ge bhcd
   MOV.B Ry,&LCDn+1 ; Note:
                    ; All bits of an LCD memory byte
                    ; are written
   .......... .......... ..........
Table
   DB a+b+c+d+e+f   ; displays "0"
   ..........
   DB a+b+c+d+e+f+g ; displays "8"
   .......... ..........
   DB .......... ;
```


Copyright © 2009–2013, Texas Instruments Incorporated

26.2.10 3-Mux Mode

In 3-mux mode, each MSP430 segment pin drives three LCD segments and three common lines (COM0, COM1, and COM2) are used.Figure 26-8 shows some example 3-mux, 1/3 bias waveforms.

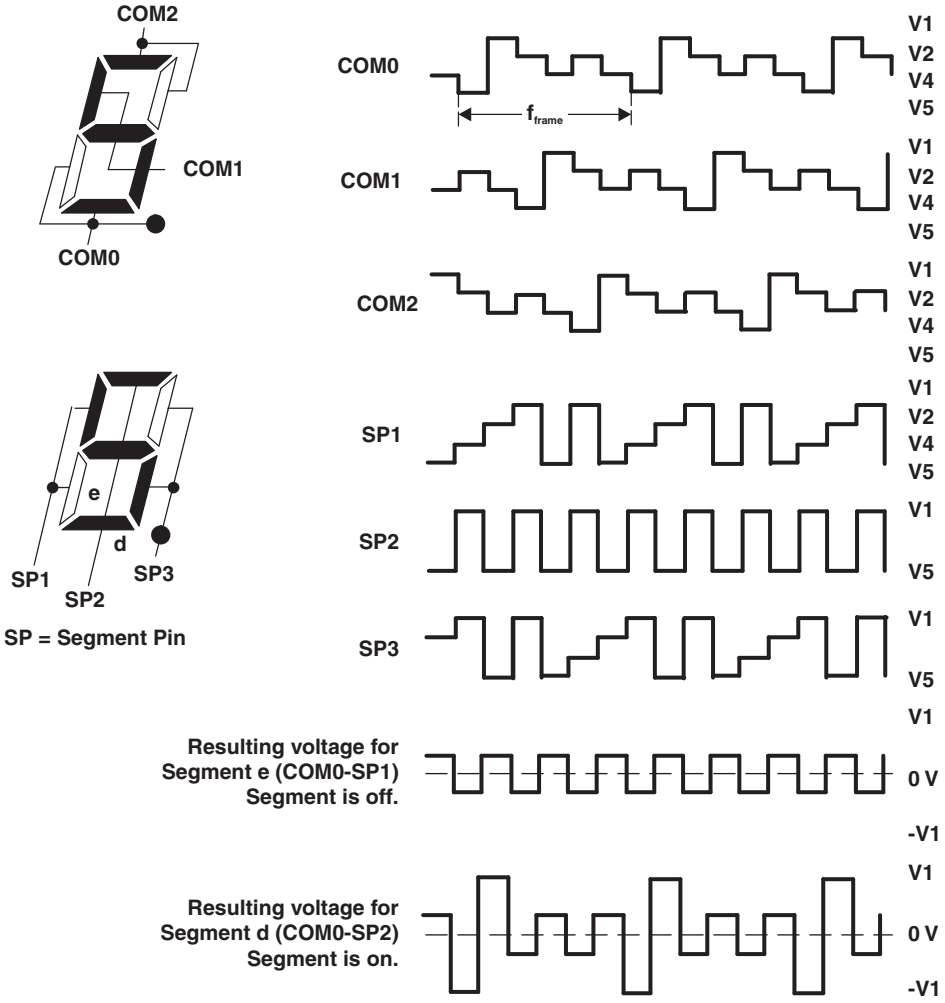


Figure 26-8. Example 3-Mux Waveforms

Copyright © 2009–2013, Texas Instruments Incorporated

Figure 26-9 shows an example 3-mux LCD, pinout, LCD-to-MSP430 connections, and the resulting segment mapping. This is only an example. Segment mapping in a user's application depends on the LCD pinout and on the MSP430-to-LCD connections.

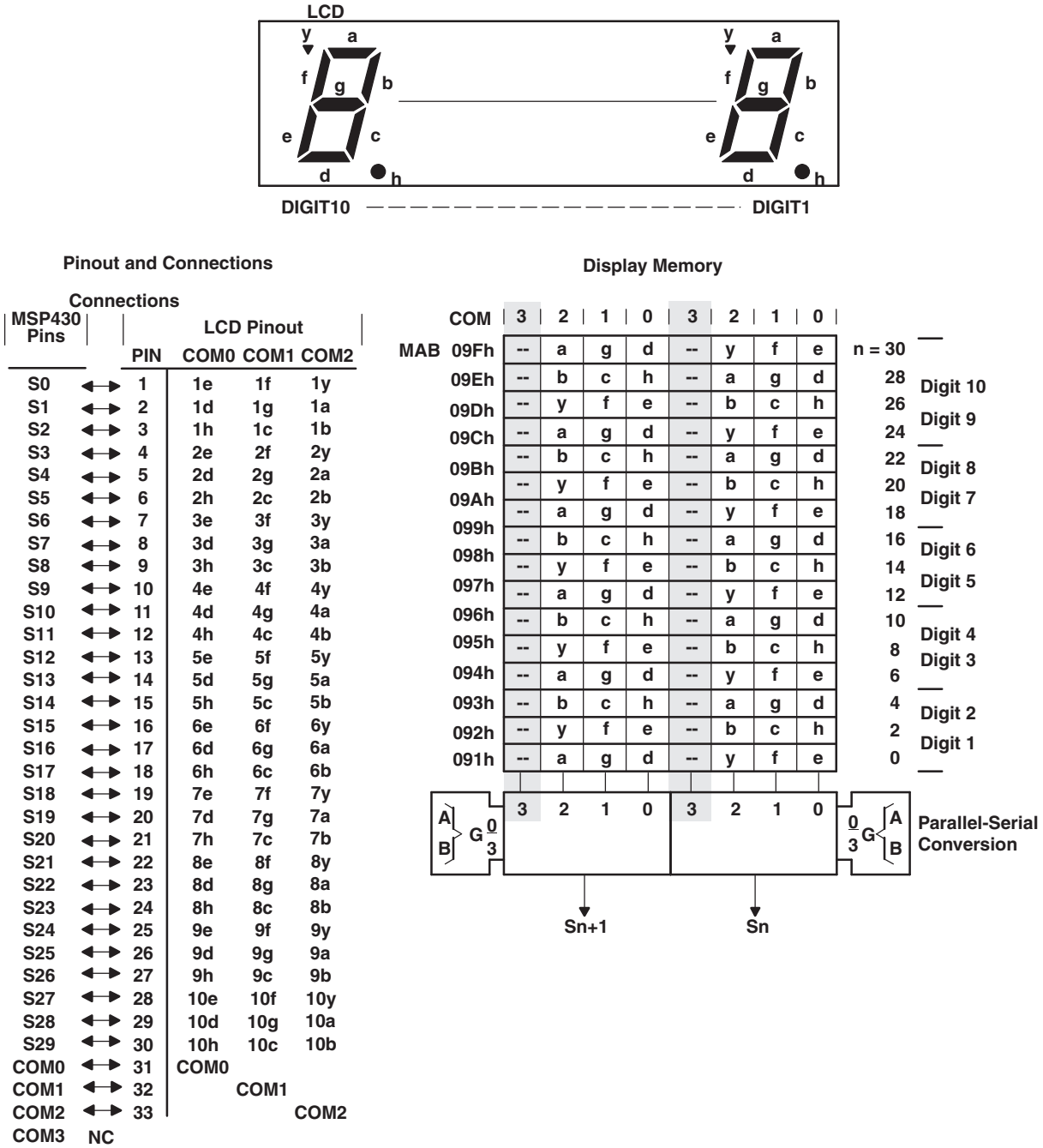


Figure 26-9. 3-Mux LCD Example (MAB addresses need to be replaced with LCDMx)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.2.10.1 3-Mux Mode Software Example

```
; The 3-mux rate can support nine segments for each
; digit. The nine segments of a digit are located in
; 1 1/2 display memory bytes.
;
a EQU 0040h
b EQU 0400h
c EQU 0200h
d EQU 0010h
e EQU 0001h
f EQU 0002h
g EQU 0020h
h EQU 0100h
Y EQU 0004h
; The LSDigit of register Rx should be displayed.
; The Table represents the 'on'-segments according to the
; LSDigit of register of Rx.
; The register Ry is used for temporary memory
;
ODDDIG
   RLA Rx ; LCD in 3-mux has 9 segments per
            ; digit
            ; word table required for
            ; displayed characters.
   MOV Table(Rx),Ry ; Load segment information to
                    ; temporary mem.
                    ; (Ry) = 0000 0bch 0agd 0yfe
   MOV.B Ry,&LCDn ; write 'a, g, d, y, f, e' of
                  ; Digit n (LowByte)
   SWPB Ry ; (Ry) = 0agd 0yfe 0000 0bch
   BIC.B #07h,&LCDn+1 ; write 'b, c, h' of Digit n
                      ; (HighByte)
   BIS.B Ry,&LCDn+1
   .....
EVNDIG
   RLA Rx ; LCD in 3-mux has 9 segments per
          ; digit
          ; word table required for
          ; displayed characters.
   MOV Table(Rx),Ry ; Load segment information to
                    ; temporary mem.
                    ; (Ry) = 0000 0bch 0agd 0yfe
   RLA Ry ; (Ry) = 0000 bch0 agd0 yfe0
   RLA Ry ; (Ry) = 000b ch0a gd0y fe00
   RLA Ry ; (Ry) = 00bc h0ag d0yf e000
   RLA Ry ; (Ry) = 0bch 0agd 0yfe 0000
   BIC.B #070h,&LCDn+1
   BIS.B Ry,&LCDn+1 ; write 'y, f, e' of Digit n+1
                    ; (LowByte)
   SWPB Ry ; (Ry) = 0yfe 0000 0bch 0agd
   MOV.B Ry,&LCDn+2 ; write 'b, c, h, a, g, d' of
                    ; Digit n+1 (HighByte)
   ...........
Table
   DW a+b+c+d+e+f  ; displays "0"
   DW b+c          ; displays "1"
.......... ..........
   DW a+e+f+g ; displays "F"
```


Copyright © 2009–2013, Texas Instruments Incorporated

26.2.11 4-Mux Mode

In 4-mux mode, each MSP430 segment pin drives four LCD segments and all four common lines (COM0, COM1, COM2, and COM3) are used. Figure 26-10 shows some example 4-mux, 1/3 bias waveforms.

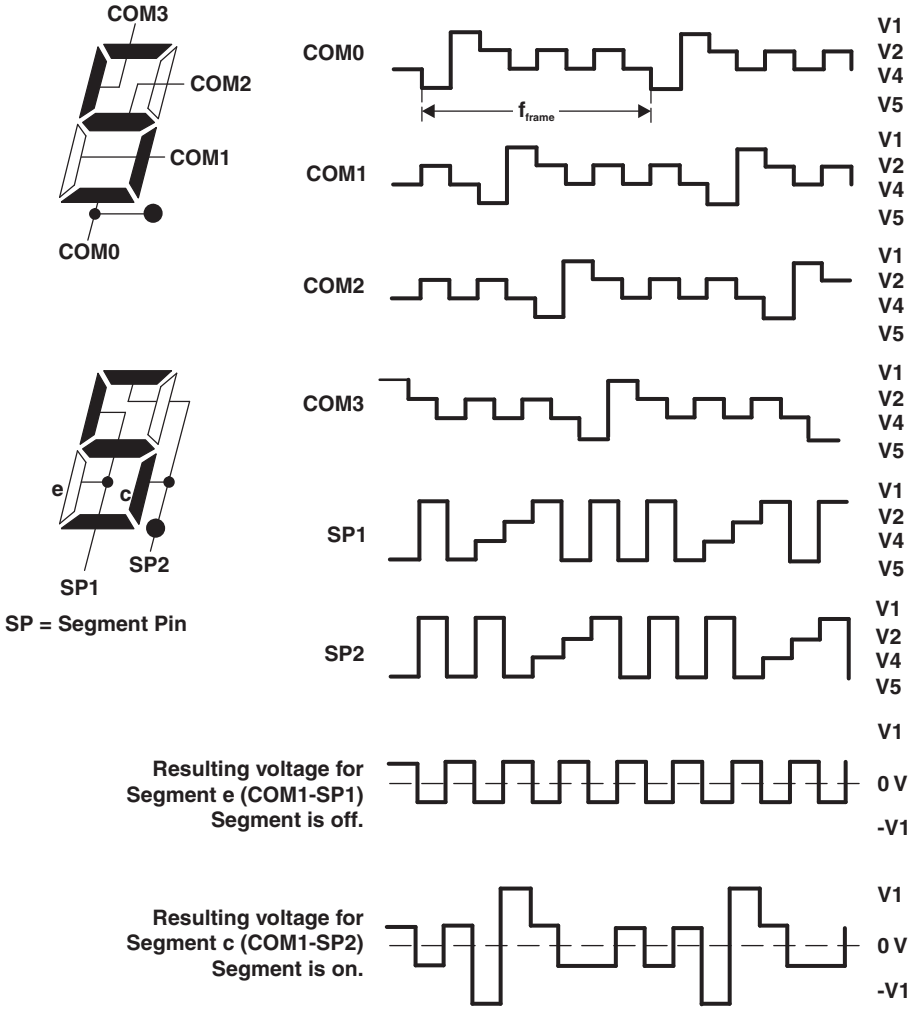


Figure 26-10. Example 4-Mux Waveforms

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Figure 26-11 shows an example 4-mux LCD, pinout, LCD-to-MSP430 connections, and the resulting segment mapping. This is only an example. Segment mapping in a user's application depends on the LCD pinout and on the MSP430-to-LCD connections.

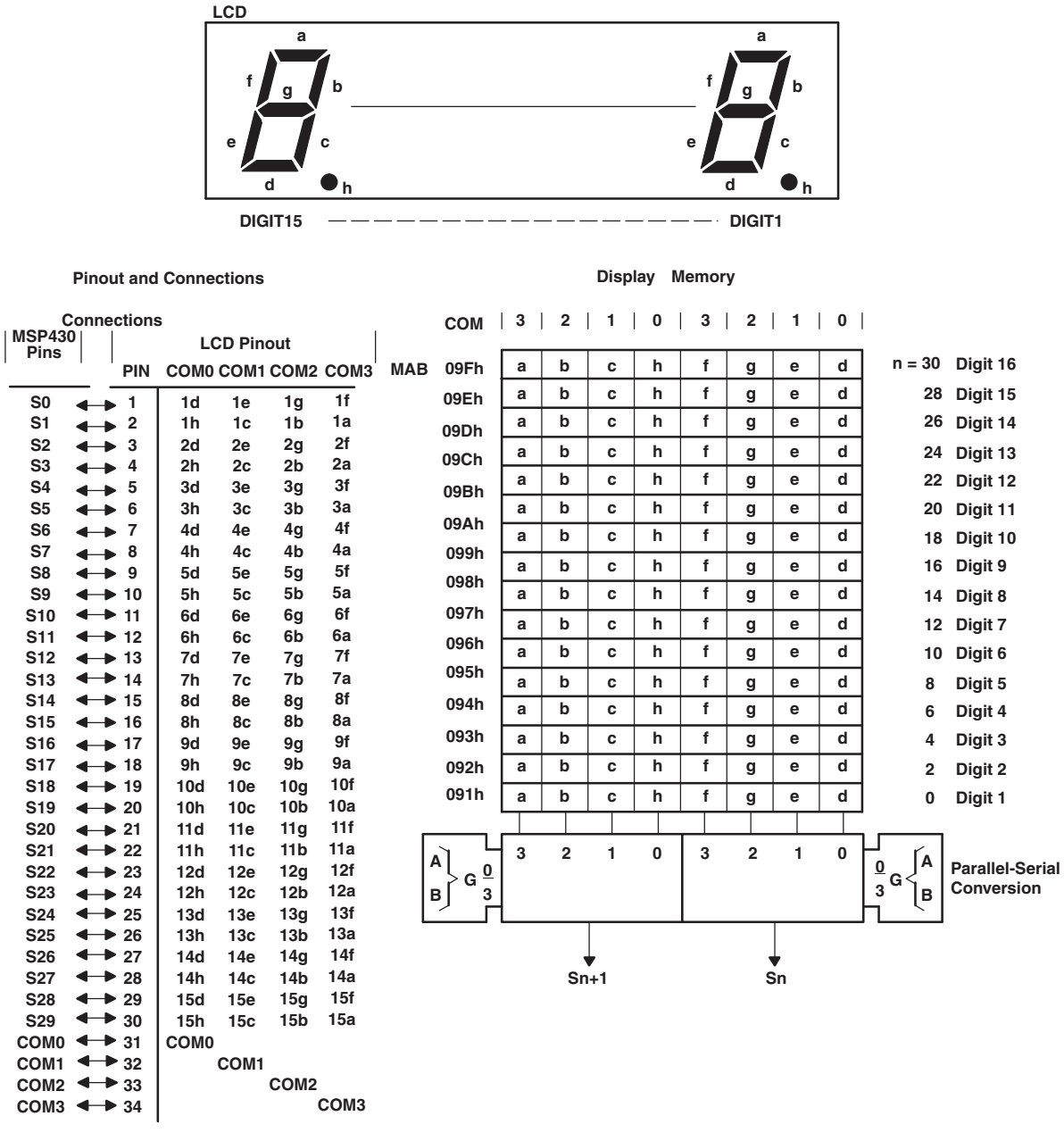


Figure 26-11. 4-Mux LCD Example (MAB addresses need to be replaced with LCDMx)

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.2.11.1 4-Mux Mode Software Example

```
; The 4-mux rate supports eight segments for each digit.
; All eight segments of a digit can often be located in
; one display memory byte
a EQU 080h
b EQU 040h
c EQU 020h
d EQU 001h
e EQU 002h
f EQU 008h
g EQU 004h
h EQU 010h
;
; The LSDigit of register Rx should be displayed.
; The Table represents the 'on'-segments according to the
; content of Rx.
;
   MOV.B Table(Rx),&LCDn ; n = 1 ..... 15
                         ; all eight segments are
                         ; written to the display
                         ; memory
.......... ..........
Table
   DB a+b+c+d+e+f ; displays "0"
   DB b+c         ; displays "1"
.......... ..........
   DB b+c+d+e+g   ; displays "d"
   DB a+d+e+f+g   ; displays "E"
   DB a+e+f+g     ; displays "F"
```


Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3 LCD_B Registers

The LCD_B registers are listed in Table 26-2 to Table 26-4. The LCD memory and blinking memory registers can also be accessed as word.

Table 26-2. LCD_B Registers

Offset	Acronym	Register Name	Type	Reset	Section
000h	LCDBCTL0	LCD_B control register 0	Read/write	0000h	Section 26.3.1
002h	LCDBCTL1	LCD_B control register 1	Read/write	0000h	Section 26.3.2
004h	LCDBBLKCTL	LCD_B blinking control register	Read/write	0000h	Section 26.3.3
006h	LCDBMEMCTL	LCD_B memory control register	Read/write	0000h	Section 26.3.4
008h	LCDBVCTL	LCD_B voltage control register	Read/write	0000h	Section 26.3.5
00Ah	LCDBPCTL0	LCD_B port control 0	Read/write	0000h	Section 26.3.6
00Ch	LCDBPCTL1	LCD_B port control 1	Read/write	0000h	Section 26.3.7
00Eh	LCDBPCTL2	LCD_B port control 2 (≥128 segments)	Read/write	0000h	Section 26.3.8
010h	LCDBPCTL3	LCD_B port control 3 (192 segments)	Read/write	0000h	Section 26.3.9
012h	LCDBCPCTL	LCD_B charge pump control	Read/write	0000h	Section 26.3.10
014h		Reserved			
016h		Reserved			
018h		Reserved			
01Ah		Reserved			
01Ch		Reserved			
01Eh	LCDBIV	LCD_B interrupt vector	Read/write	0000h	Section 26.3.11

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 26-3. LCD_B Memory Registers(1)

Offset	Acronym	Register Name	Type	Reset
020h	LCDM1	LCD memory 1 (S1/S0)	Read/write	Unchanged
021h	LCDM2	LCD memory 2 (S3/S2)	Read/write	Unchanged
022h	LCDM3	LCD memory 3 (S5/S4)	Read/write	Unchanged
023h	LCDM4	LCD memory 4 (S7/S6)	Read/write	Unchanged
024h	LCDM5	LCD memory 5 (S9/S8)	Read/write	Unchanged
025h	LCDM6	LCD memory 6 (S11/S10)	Read/write	Unchanged
026h	LCDM7	LCD memory 7 (S13/S12)	Read/write	Unchanged
027h	LCDM8	LCD memory 8 (S15/S14)	Read/write	Unchanged
028h	LCDM9	LCD memory 9 (S17/S16)	Read/write	Unchanged
029h	LCDM10	LCD memory 10 (S19/S18)	Read/write	Unchanged
02Ah	LCDM11	LCD memory 11 (S21/S20)	Read/write	Unchanged
02Bh	LCDM12	LCD memory 12 (S23/S22)	Read/write	Unchanged
02Ch	LCDM13	LCD memory 13 (S25/S24)	Read/write	Unchanged
02Dh	LCDM14	LCD memory 14 (S27/S26)	Read/write	Unchanged
02Eh	LCDM15	LCD memory 15 (S29/S28, ≥128 segments)	Read/write	Unchanged
02Fh	LCDM16	LCD memory 16 (S31/S30, ≥128 segments)	Read/write	Unchanged
030h	LCDM17	LCD memory 17 (S33/S32, ≥128 segments)	Read/write	Unchanged
031h	LCDM18	LCD memory 18 (S35/S34, ≥128 segments)	Read/write	Unchanged
032h	LCDM19	LCD memory 19 (S37/S36, ≥160 segments)	Read/write	Unchanged
033h	LCDM20	LCD memory 20 (S39/S38, ≥160 segments)	Read/write	Unchanged
034h	LCDM21	LCD memory 21 (S41/S40, ≥160 segments)	Read/write	Unchanged
035h	LCDM22	LCD memory 22 (S43/S42, ≥160 segments)	Read/write	Unchanged
036h	LCDM23	LCD memory 23 (S45/S44, 192 segments)	Read/write	Unchanged
037h	LCDM24	LCD memory 24 (S47/S46, 192 segments)	Read/write	Unchanged
038h	LCDM25	LCD memory 25 (S49/S48, 192 segments)	Read/write	Unchanged
039h	LCDM26	LCD memory 26 (S50, 192 segments)	Read/write	Unchanged
03Ah		Reserved	Read/write	Unchanged
03Bh		Reserved	Read/write	Unchanged
03Ch		Reserved	Read/write	Unchanged
03Dh		Reserved	Read/write	Unchanged
03Eh		Reserved	Read/write	Unchanged
03Fh		Reserved	Read/write	Unchanged

(1) The LCD memory registers can also be accessed as word.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Table 26-4. LCD_B Blinking Memory Registers(1)

Offset	Acronym	Register Name	Type	Reset
040h	LCDBM1	LCD blinking memory 1	Read/write	Unchanged
041h	LCDBM2	LCD blinking memory 2	Read/write	Unchanged
042h	LCDBM3	LCD blinking memory 3	Read/write	Unchanged
043h	LCDBM4	LCD blinking memory 4	Read/write	Unchanged
044h	LCDBM5	LCD blinking memory 5	Read/write	Unchanged
045h	LCDBM6	LCD blinking memory 6	Read/write	Unchanged
046h	LCDBM7	LCD blinking memory 7	Read/write	Unchanged
047h	LCDBM8	LCD blinking memory 8	Read/write	Unchanged
048h	LCDBM9	LCD blinking memory 9	Read/write	Unchanged
049h	LCDBM10	LCD blinking memory 10	Read/write	Unchanged
04Ah	LCDBM11	LCD blinking memory 11	Read/write	Unchanged
04Bh	LCDBM12	LCD blinking memory 12	Read/write	Unchanged
04Ch	LCDBM13	LCD blinking memory 13	Read/write	Unchanged
04Dh	LCDBM14	LCD blinking memory 14	Read/write	Unchanged
04Eh	LCDBM15	LCD blinking memory 15 (≥128 segments)	Read/write	Unchanged
04Fh	LCDBM16	LCD blinking memory 16 (≥128 segments)	Read/write	Unchanged
050h	LCDBM17	LCD blinking memory 17 (≥128 segments)	Read/write	Unchanged
051h	LCDBM18	LCD blinking memory 18 (≥128 segments)	Read/write	Unchanged
052h	LCDBM19	LCD blinking memory 19 (≥160 segments)	Read/write	Unchanged
053h	LCDBM20	LCD blinking memory 20 (≥160 segments)	Read/write	Unchanged
054h	LCDBM21	LCD blinking memory 21 (≥160 segments)	Read/write	Unchanged
055h	LCDBM22	LCD blinking memory 22 (≥160 segments)	Read/write	Unchanged
056h	LCDBM23	LCD blinking memory 23 (190 segments)	Read/write	Unchanged
057h	LCDBM24	LCD blinking memory 24 (190 segments)	Read/write	Unchanged
058h	LCDBM25	LCD blinking memory 25 (190 segments)	Read/write	Unchanged
059h	LCDBM26	LCD blinking memory 26 (190 segments)	Read/write	Unchanged
05Ah		Reserved	Read/write	Unchanged
05Bh		Reserved	Read/write	Unchanged
05Ch		Reserved	Read/write	Unchanged
05Dh		Reserved	Read/write	Unchanged
05Eh		Reserved	Read/write	Unchanged
05Fh		Reserved	Read/write	Unchanged

(1) The LCD blinking memory registers can also be accessed as word.

Copyright © 2009–2013, Texas Instruments Incorporated

26.3.1 LCDBCTL0 Register

LCD_B Control Register 0

Figure 26-12. LCDBCTL0 Register

15	14	13	12	11	10	9	8
LCDDIVx					LCDPREx		
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
LCDSSEL	Reserved		LCDMXx		LCDSON	Reserved	LCDON
rw-0	r0	r0	rw-0	rw-0	rw-0	r0	rw-0

Table 26-5. LCDBCTL0 Register Description

Bit	Field	Type	Reset	Description
15-11	LCDDIVx	RW	0h	LCD frequency divider. Together with LCDPREx the LCD frequency f(LCD) is calculated as f(LCD) = f(ACLK/VLO) / [(LCDDIVx + 1) × 2^(LCDPREx)]. Settings for this bit should be changed only while LCDON = 0. 00000b = Divide by 1 00001b = Divide by 2 ⋮ 11110b = Divide by 31 11111b = Divide by 32
10-8	LCDPREx	RW	0h	LCD frequency pre-scaler. Together with LCDDIVx the LCD frequency f(LCD) is calculated as f(LCD) = f(ACLK/VLO) / [(LCDDIVx + 1) × 2^(LCDPREx)]. Settings for this bit should be changed only while LCDON = 0. 000b = Divide by 1 001b = Divide by 2 010b = Divide by 4 011b = Divide by 8 100b = Divide by 16 101b = Divide by 32 110b = Reserved - Defaults to divide by 32 111b = Reserved - Defaults to divide by 32
7	LCDSSEL	RW	0h	Clock source select for LCD and blinking frequency Settings for this bit should be changed only while LCDON = 0. 0b = ACLK (30 kHz to 40 kHz) 1b = VLOCLK
6-5	Reserved	R	0h	Reserved. Always reads as 0.
4-3	LCDMXx	RW	0h	LCD mux rate. These bits select the LCD mode. Settings for this bit should be changed only while LCDON = 0. 00b = Static 01b = 2-mux 10b = 3-mux 11b = 4-mux
2	LCDSON	RW	0h	LCD segments on. This bit supports flashing LCD applications by turning off all segment lines, while leaving the LCD timing generator and R33 enabled. 0b = All LCD segments are off. 1b = All LCD segments are enabled and on or off according to their corresponding memory location.
1	Reserved	R	0h	Reserved. Always reads as 0.
0	LCDON	RW	0h	LCD on. This bit turns the LCD_B module on or off. 0b = LCD_B module off 1b = LCD_B module on

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.2 LCDBCTL1 Register

LCD_B Control Register 1

Figure 26-13. LCDBCTL1 Register

15	14	13	12	11	10	9	8
Reserved				LCDNOCAPIE	LCDBLKONIE	LCDBLKOFFIE	LCDFRMIE
r0	r0	r0	r0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
Reserved				LCDNOCAPIFG	LCDBLKONIFG	LCDBLKOFFIFG	LCDFRMIFG
r0	r0	r0	r0	rw-0	rw-0	rw-0	rw-0

Table 26-6. LCDBCTL1 Register Description

Bit	Field	Type	Reset	Description
15-12	Reserved	R	0h	Reserved. Always reads as 0.
11	LCDNOCAPIE	RW	0h	No capacitance connected interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
10	LCDBLKONIE	RW	0h	LCD blinking interrupt enable, segments switched on 0b = Interrupt disabled 1b = Interrupt enabled
9	LCDBLKOFFIE	RW	0h	LCD blinking interrupt enable, segments switched off 0b = Interrupt disabled 1b = Interrupt enabled
8	LCDFRMIE	RW	0h	LCD frame interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
7-4	Reserved	R	0h	Reserved. Always reads as 0.
3	LCDNOCAPIFG	RW	0h	No capacitance connected interrupt flag. Set when charge pump is enabled but no capacitance is connected to LCDCAP pin. 0b = No interrupt pending 1b = Interrupt pending
2	LCDBLKONIFG	RW	0h	LCD blinking interrupt flag, segments switched on. Automatically cleared when data is written into a memory register. 0b = No interrupt pending 1b = Interrupt pending
1	LCDBLKOFFIFG	RW	0h	LCD blinking interrupt flag, segments switched off. Automatically cleared when data is written into a memory register. 0b = No interrupt pending 1b = Interrupt pending
0	LCDFRMIFG	RW	0h	LCD frame interrupt flag. Automatically cleared when data is written into a memory register. 0b = No interrupt pending 1b = Interrupt pending

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.3 LCDBBLKCTL Register

LCD_B Blink Control Register

Figure 26-14. LCDBBLKCTL Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
LCDBLKDIVx			LCDBLKPREx			LCDBLKMODx	
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-7. LCDBBLKCTL Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved. Always reads as 0.
7-5	LCDBLKDIVx	RW	0h	Clock divider for blinking frequency. Together with LCDBLKPREx, the blinking frequency f(BLINK) is calculated as f(BLINK) = f(ACLK/VLO) / [(LCDBLKDIVx + 1) × 2^(9+LCDBLKPREx)]. Settings for this bit should be changed only while LCDBLKMODx = 00. 000b = Divide by 1 001b = Divide by 2 010b = Divide by 3 011b = Divide by 4 100b = Divide by 5 101b = Divide by 6 110b = Divide by 7 111b = Divide by 8
4-2	LCDBLKPREx	RW	0h	Clock pre-scaler for blinking frequency. Together with LCDBLKDIVx, the blinking frequency f(BLINK) is calculated as f(BLINK) = f(ACLK/VLO) / ((LCDBLKDIVx + 1) × 2^(9+LCDBLKPREx)). Settings for this bit should be changed only while LCDBLKMODx = 00. 000b = Divide by 512 001b = Divide by 1024 010b = Divide by 2048 011b = Divide by 4096 100b = Divide by 8162 101b = Divide by 16384 110b = Divide by 32768 111b = Divide by 65536
1-0	LCDBLKMODx	RW	0h	Blinking mode 00b = Blinking disabled 01b = Blinking of individual segments as enabled in blinking memory register LCDBMx 10b = Blinking of all segments 11b = Switching between display contents as stored in LCDMx and LCDBMx memory registers.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.4 LCDBMEMCTL Register

LCD_B Memory Control Register

Figure 26-15. LCDBMEMCTL Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved					LCDCLRBM	LCDCLRM	LCDDISP
r0	r0	r0	r0	r0	rw-0	rw-0	rw-0

Table 26-8. LCDBMEMCTL Register Description

Bit	Field	Type	Reset	Description
15-3	Reserved	R	0h	Reserved. Always reads as 0.
2	LCDCLRBM	RW	0h	Clear LCD blinking memory Clears all blinking memory registers LCDBMx. The bit is automatically reset when the blinking memory is cleared. 0b = Contents of blinking memory registers LCDBMx remain unchanged 1b = Clear content of all blinking memory registers LCDBMx
1	LCDCLRM	RW	0h	Clear LCD memory Clears all LCD memory registers LCDMx. The bit is automatically reset when the LCD memory is cleared. 0b = Contents of LCD memory registers LCDMx remain unchanged 1b = Clear content of all LCD memory registers LCDMx
0	LCDDISP	RW	0h	Select LCD memory registers for display The bit is cleared in LCDBLKMODx = 01 and LCDBLKMODx = 10 and cannot be changed by software. When LCDBLKMODx = 11, this bit reflects the currently displayed memory but cannot be changed by software. When returning to LCDBLKMODx = 00 the bit is cleared. 0b = Display content of LCD memory registers LCDMx 1b = Display content of LCD blinking memory registers LCDBMx

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.5 LCDBVCTL Register

LCD_B Voltage Control Register

Figure 26-16. LCDBVCTL Register

15	14	13	12	11	10	9	8
Reserved			VLCDx				Reserved
r0	r0	r0	rw-0	rw-0	rw-0	rw-0	r0
7	6	5	4	3	2	1	0
LCDREXT	R03EXT	LCDEXTBIAS	VLCDEXT	LCDCPEN	VLCDREFx		LCD2B
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-9. LCDBVCTL Register Description

Bit	Field	Type	Reset	Description
15-13	Reserved	R	0h	Reserved. Always reads as 0.
12-9	VLCDx	RW	0h	Charge pump voltage select. LCDCPEN must be 1 for the charge pump to be enabled. V(CC) is used for V(LCD) when VLCDx = 0000 and VLCDREFx = 00 and VLCDEXT = 0. If VLCDREFx = 00 or 10: 0000b = Charge pump disabled 0001b = V(LCD) = 2.60 V 0010b = V(LCD) = 2.66 V 0011b = V(LCD) = 2.72 V 0100b = V(LCD) = 2.78 V 0101b = V(LCD) = 2.84 V 0110b = V(LCD) = 2.90 V 0111b = V(LCD) = 2.96 V 1000b = V(LCD) = 3.02 V 1001b = V(LCD) = 3.08 V 1010b = V(LCD) = 3.14 V 1011b = V(LCD) = 3.20 V 1100b = V(LCD) = 3.26 V 1101b = V(LCD) = 3.32 V 1110b = V(LCD) = 3.38 V 1111b = V(LCD) = 3.44 V If VLCDREFx = 01 or 11: 0000b = Charge pump disabled 0001b = V(LCD) = 2.17 × V(REF) 0010b = V(LCD) = 2.22 × V(REF) 0011b = V(LCD) = 2.27 × V(REF) 0100b = V(LCD) = 2.32 × V(REF) 0101b = V(LCD) = 2.37 × V(REF) 0110b = V(LCD) = 2.42 × V(REF) 0111b = V(LCD) = 2.47 × V(REF) 1000b = V(LCD) = 2.52 × V(REF) 1001b = V(LCD) = 2.57 × V(REF) 1010b = V(LCD) = 2.62 × V(REF) 1011b = V(LCD) = 2.67 × V(REF) 1100b = V(LCD) = 2.72 × V(REF) 1101b = V(LCD) = 2.77 × V(REF) 1110b = V(LCD) = 2.82 × V(REF) 1111b = V(LCD) = 2.87 × V(REF)
8	Reserved	R	0h	Reserved. Always reads as 0.

Copyright © 2009–2013, Texas Instruments Incorporated

Table 26-9. LCDBVCTL Register Description (continued)

Bit	Field	Type	Reset	Description
7	LCDREXT	RW	0h	V2 to V4 voltage on external Rx3 pins. This bit selects the external connections for voltages V2 to V4 with internal bias generation (LCDEXTBIAS = 0). The bit is don't care if external biasing is selected (LCDEXTBIAS = 1). Settings for this bit should be changed only while LCDON = 0. 0b = Internally generated V2 to V4 are not switched to pins (LCDEXTBIAS = 0). 1b = Internally generated V2 to V4 are switched to pins (LCDEXTBIAS = 0).
6	R03EXT	RW	0h	V5 voltage select. This bit selects the external connection for the lowest LCD voltage. R03EXT is ignored if there is no R03 pin available. Settings for this bit should be changed only while LCDON = 0. 0b = V5 is V(SS) 1b = V5 is sourced from the R03 pin
5	LCDEXTBIAS	RW	0h	V2 to V4 voltage select. This bit selects the generation for voltages V2 to V4. Settings for this bit should be changed only while LCDON = 0. 0b = V2 to V4 are generated internally. 1b = V2 to V4 are sourced externally and the internal bias generator is switched off.
4	VLCDEXT	RW	0h	V(LCD) source select Settings for this bit should be changed only while LCDON = 0. 0b = V(LCD) is generated internally. 1b = V(LCD) is sourced externally.
3	LCDCPEN	RW	0h	Charge pump enable 0b = Charge pump disabled 1b = Charge pump enabled when V(LCD) is generated internally (VLCDEXT = 0) and VLCDx > 0 or VLCDREFx > 0.
2-1	VLCDREFx	RW	0h	Charge pump reference select If LCDEXTBIAS = 1 or LCDREXT = 1 settings 01, 10 and 11 are not supported. Internal reference voltage used instead. Settings for this bit should be changed only while LCDON = 0. 00b = Internal reference voltage 01b = External reference voltage 10b = Internal reference voltage switched to external pin LCDREF/R13. 11b = Reserved. Defaults to external reference voltage.
0	LCD2B	RW	0h	Bias select. LCD2B is ignored when LCDMx = 00. 0b = 1/3 bias 1b = 1/2 bias

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.6 LCDBPCTL0 Register

LCD_B Port Control Register 0

Figure 26-17. LCDBPCTL0 Register

15	14	13	12	11	10	9	8
LCDS15	LCDS14	LCDS13	LCDS12	LCDS11	LCDS10	LCDS9	LCDS8
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
LCDS7	LCDS6	LCDS5	LCDS4	LCDS3	LCDS2	LCDS1	LCDS0
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-10. LCDBPCTL0 Register Description

Bit	Field	Type	Reset	Description
15-0	LCDSx	RW	0h	LCD segment line x enable This bit affects only pins with multiplexed functions. Dedicated LCD pins are always LCD function. 0b = Multiplexed pins are port functions. 1b = Pins are LCD functions.

26.3.7 LCDBPCTL1 Register

LCD_B Port Control Register 1

Figure 26-18. LCDBPCTL1 Register

15	14	13	12	11	10	9	8
LCDS31	LCDS30	LCDS29	LCDS28	LCDS27	LCDS26	LCDS25	LCDS24
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
LCDS23	LCDS22	LCDS21	LCDS20	LCDS19	LCDS18	LCDS17	LCDS16
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-11. LCDBPCTL1 Register Description

Bit	Field	Type	Reset	Description
15-0	LCDSx	RW	0h	LCD segment line x enable LCDS27 to LCDS31 are reserved on devices supporting a maximum of 96 segments. This bit affects only pins with multiplexed functions. Dedicated LCD pins are always LCD function. 0b = Multiplexed pins are port functions. 1b = Pins are LCD functions.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.8 LCDBPCTL2 Register

LCD_B Port Control Register 2 (≥ 128 Segments)

Figure 26-19. LCDBPCTL2 Register

15	14	13	12	11	10	9	8
LCDS47	LCDS46	LCDS45	LCDS44	LCDS43	LCDS42	LCDS41	LCDS40
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0
7	6	5	4	3	2	1	0
LCDS39	LCDS38	LCDS37	LCDS36	LCDS35	LCDS34	LCDS33	LCDS32
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-12. LCDBPCTL2 Register Description

Bit	Field	Type	Reset	Description
15-0	LCDSx	RW	0h	LCD segment line x enable LCDS35 to LCDS47 are reserved on devices supporting a maximum of 128 segments. LCDS43 to LCDS47 are reserved on devices supporting a maximum of 160 segments. This bit affects only pins with multiplexed functions. Dedicated LCD pins are always LCD function. 0b = Multiplexed pins are port functions. 1b = Pins are LCD functions.

26.3.9 LCDBPCTL3 Register

LCD_B Port Control Register 2 (192 Segments)

Figure 26-20. LCDBPCTL3 Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
Reserved					LCDS50	LCDS49	LCDS48
r0	r0	r0	r0	r0	rw-0	rw-0	rw-0

Table 26-13. LCDBPCTL3 Register Description

Bit	Field	Type	Reset	Description
15-3	Reserved	R	0h	Reserved. Always reads as 0.
2-0	LCDSx	RW	0h	LCD segment line x enable This bit affects only pins with multiplexed functions. Dedicated LCD pins are always LCD function. NOTE: Settings for LCDSx should be changed only while LCDON = 0. 0b = Multiplexed pins are port functions. 1b = Pins are LCD functions.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

26.3.10 LCDBCPCTL Register

LCD_B Charge Pump Control Register

Figure 26-21. LCDBCPCTL Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
LCDCPDIS7	LCDCPDIS6	LCDCPDIS5	LCDCPDIS4	LCDCPDIS3	LCDCPDIS2	LCDCPDIS1	LCDCPDIS0
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-14. LCDBCPCTL Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved. Always reads as 0.
7	LCDCPDIS7	RW	0h	Reserved
6	LCDCPDIS6	RW	0h	Reserved
5	LCDCPDIS5	RW	0h	Reserved
4	LCDCPDIS4	RW	0h	Reserved
3	LCDCPDIS3	RW	0h	Reserved
2	LCDCPDIS2	RW	0h	LCD charge pump disable during ADC12 conversion 0b = LCD charge pump not automatically disabled during conversion. 1b = LCD charge pump automatically disabled during conversion.
1	LCDCPDIS1	RW	0h	LCD charge pump disable during radio transmit 0b = LCD charge pump not automatically disabled during radio transmit. 1b = LCD charge pump automatically disabled during radio transmit.
0	LCDCPDIS0	RW	0h	LCD charge pump disable during radio receive 0b = LCD charge pump not automatically disabled during radio receive. 1b = LCD charge pump automatically disabled during radio receive.

Copyright © 2009–2013, Texas Instruments Incorporated

26.3.11 LCDBIV Register

LCD_B Interrupt Vector Register

Figure 26-22. LCDBIV Register

15	14	13	12	11	10	9	8
LCDBIVx							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
LCDBIVx							
r0	r0	r0	r0	r0	r0	r0	r0

Table 26-15. LCDBIV Register Description

Bit	Field	Type	Reset	Description
15-0	LCDBIVx	R	0h	LCD_B interrupt vector value 00h = No interrupt pending 02h = Interrupt Source: No capacitor connected; Interrupt Flag: LCDNOCAPIFG; Interrupt Priority: Highest 04h = Interrupt Source: Blink, segments off; Interrupt Flag: LCDBLKOFFIFG 06h = Interrupt Source: Blink, segments on; Interrupt Flag: LCDBLKONIFG 08h = Interrupt Source: Frame interrupt; Interrupt Flag: LCDFRMIFG; Interrupt Priority: Lowest

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Embedded Emulation Module (EEM)

This chapter describes the embedded emulation module (EEM) that is implemented in all flash devices.

Topic **Page**

27.1 Embedded Emulation Module (EEM) Introduction 776
27.2 EEM Building Blocks 778
27.3 EEM Configurations 779

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

27.1 Embedded Emulation Module (EEM) Introduction

Every MSP430 flash-based microcontroller implements an EEM. It is accessed and controlled through either 4-wire JTAG mode or Spy-Bi-Wire mode. Each implementation is device dependent and is described in Section 27.3, the EEM Configurations section, and the device-specific data sheet.

In general, the following features are available:

- Nonintrusive code execution with real-time breakpoint control
- Single-step, step-into, and step-over functionality
- Full support of all low-power modes
- Support for all system frequencies, for all clock sources
- Up to eight (device-dependent) hardware triggers or breakpoints on memory address bus (MAB) or memory data bus (MDB)
- Up to two (device-dependent) hardware triggers or breakpoints on CPU register write accesses
- MAB, MDB, and CPU register access triggers can be combined to form up to ten (device dependent) complex triggers or breakpoints
- Up to two (device dependent) cycle counters
- Trigger sequencing (device dependent)
- Storage of internal bus and control signals using an integrated trace buffer (device dependent)
- Clock control for timers, communication peripherals, and other modules on a global device level or on a per-module basis during an emulation stop

Figure 27-1 shows a simplified block diagram of the largest currently-available EEM implementation.

For more details on how the features of the EEM can be used together with the IAR Embedded Workbench ™ debugger or with Code Composer Studio (CCS), see *Advanced Debugging Using the Enhanced Emulation Module* (SLAA393) at www.msp430.com. Most other debuggers supporting the MSP430 devices have the same or a similar feature set. For details, see the user's guide of the applicable debugger.

Copyright © 2009–2013, Texas Instruments Incorporated

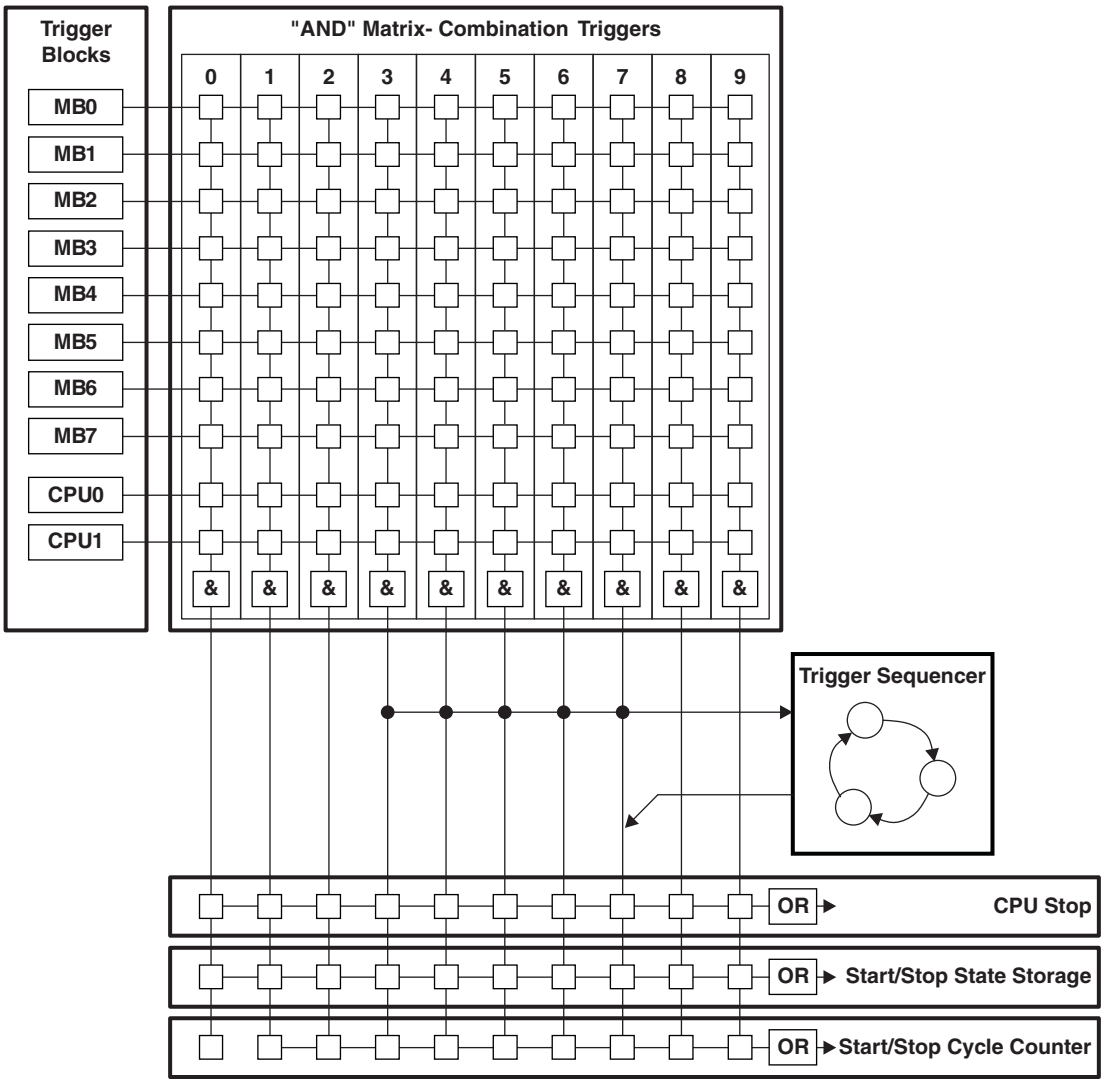


Figure 27-1. Large Implementation of EEM

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

27.2 EEM Building Blocks

27.2.1 Triggers

The event control in the EEM of the MSP430 system consists of triggers, which are internal signals indicating that a certain event has happened. These triggers may be used as simple breakpoints, but it is also possible to combine two or more triggers to allow detection of complex events and cause various reactions other than stopping the CPU.

In general, the triggers can be used to control the following functional blocks of the EEM:

- Breakpoints (CPU stop)
- State storage
- Sequencer
- Cycle counter

There are two different types of triggers – the memory trigger and the CPU register write trigger.

Each memory trigger block can be independently selected to compare either the MAB or the MDB with a given value. Depending on the implemented EEM, the comparison can be =, ≠, ≥, or ≤. The comparison can also be limited to certain bits with the use of a mask. The mask is either bit-wise or byte-wise, depending upon the device. In addition to selecting the bus and the comparison, the condition under which the trigger is active can be selected. The conditions include read access, write access, DMA access, and instruction fetch.

Each CPU register write trigger block can be independently selected to compare what is written into a selected register with a given value. The observed register can be selected for each trigger independently. The comparison can be =, ≠, ≥, or ≤. The comparison can also be limited to certain bits with the use of a bit mask.

Both types of triggers can be combined to form more complex triggers. For example, a complex trigger can signal when a particular value is written into a user-specified address.

27.2.2 Trigger Sequencer

The trigger sequencer allows the definition of a certain sequence of trigger signals before an event is accepted for a break or state storage event. Within the trigger sequencer, it is possible to use the following features:

- Four states (State 0 to State 3)
- Two transitions per state to any other state
- Reset trigger that resets the sequencer to State 0.

The trigger sequencer always starts at State 0 and must execute to State 3 to generate an action. If State 1 or State 2 are not required, they can be bypassed.

27.2.3 State Storage (Internal Trace Buffer)

The state storage function uses a built-in buffer to store MAB, MDB, and CPU control signal information (that is, read, write, or instruction fetch) in a nonintrusive manner. The built-in buffer can hold up to eight entries. The flexible configuration allows the user to record the information of interest very efficiently.

27.2.4 Cycle Counter

The cycle counter provides one or two 40-bit counters to measure the cycles used by the CPU to execute certain tasks. On some devices, the cycle counter operation can be controlled using triggers. This allows, for example, conditional profiling, such as profiling a specific section of code.

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

27.2.5 Clock Control

The EEM provides device-dependent flexible clock control. This is useful in applications where a running clock is needed for peripherals after the CPU is stopped (for example, to allow a UART module to complete its transfer of a character or to allow a timer to continue generating a PWM signal).

The clock control is flexible and supports both modules that need a running clock and modules that must be stopped when the CPU is stopped due to a breakpoint.

27.3 EEM Configurations

Table 27-1 gives an overview of the EEM configurations. The implemented configuration is device dependent, and details can be found in the device-specific data sheet and these documents:

Advanced Debugging Using the Enhanced Emulation Module (EEM) With CCS Version 4 (SLAA393)

IAR Embedded Workbench Version 3+ for MSP430 User's Guide (SLAU138)

Code Composer Studio v4.2 for MSP430 User's Guide (SLAU157)

Table 27-1. EEM Configurations

Feature	XS	S	M	L
Memory bus triggers	2 (=, ≠ only)	3	5	8
Memory bus trigger mask for	1) Low byte 2) High byte 3) Four upper addr bits	1) Low byte 2) High byte 3) Four upper addr bits	1) Low byte 2) High byte 3) Four upper addr bits	All 16 or 20 bits
CPU register write triggers	0	1	1	2
Combination triggers	2	4	6	10
Sequencer	No	No	Yes	Yes
State storage	No	No	No	Yes
Cycle counter	1	1	1	2 (including triggered start or stop)

In general, the following features can be found on any device:

- At least two MAB or MDB triggers supporting:
  - Distinction between CPU, DMA, read, and write accesses
  - =, ≠, ≥, or ≤ comparison (in XS, only =, ≠)
- At least two trigger combination registers
- Hardware breakpoints using the CPU stop reaction
- At least one 40-bit cycle counter
- Enhanced clock control with individual control of module clocks

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Revision History

Changes from revision D (July 2012) to revision E (January 2013)

Location	Description
Section 1.3.4.1, Section 4.6.2.19, Section 4.6.2.20	Added note "Enable and Disable Interrupt"
Section 13.2.1.1	Added INCLK as a clock source option.
Section 13.3.1	Changed TASSEL bit option 11b to INCLK.
Section 20.2.7.2	Changed description of ADC12SC.
Section 20.3.5	Changed description of ADC12INCHx.

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from revision C (July 2012) to revision D (July 2012)

Location	Description
Section 8.4	Replaced Digital I/O Registers section with correct registers

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from revision B (July 2010) to revision C (July 2012)

Location	Description
Throughout	Changed format of all register descriptions
Section 1.2.1	Added note that a blank device enters LPM4 on system reset
Section 1.3.4.1	Changed step 7
Table 1-2	Added note regarding automatic reset of bits
Section 1.4.2	Changed code example to show LPMx.5 naming
Section 1.4.2	Updated description regarding proper clock system settings
Section 1.5	Updated description regarding powering down unused RAM
Section 1.6	Added pins and notes
Section 1.9	Changed description
Section 1.10	Changed description of USB BSL
Section 1.11.2	Added note regarding customer return devices
Section 1.14.2	Changed WDTIFG description
Section 1.15.8	Added BUSIFG
Section 2.2.6	Added section
Section 2.2.2	Added typical application scenarios
Figure 2-3	Added figure
Section 2.2.2.2	Changed description of overvoltage detection
Section 2.2.4	Changed description and code example
Section 2.2.5	Changed description
Section 2.2.6	Added section
Section 2.2.9	Changed description
Section 2.2.9.1	Added section
Section 2.2.9	Changed description
Table 2-15	Corrected address offset of PMMIFG
Section 2.3.1	Changed PMM password description
Section 3.2	Added note regarding enabling XT1
Section 3.2.12	Changed description
Section 3.2.12 and following	Corrected DCOOFFG to DCOFFG in several places

Copyright © 2009–2013, Texas Instruments Incorporated

ECCN 5E002 TSPA - Technology / Software Publicly Available.

Changes from revision B (July 2010) to revision C (July 2012) (continued)

Location	Description
Table 3-1	Changed note (1); added notes (2) and (3)
Section 3.4.2	Updated description
Section 3.4.7	Updated description regarding LPM3.5 data retention
Section 3.4.10	Added UCSCTL9 register
Section 4.5.1	Changed Tables 4-4, 4-5, 4-13, and 4-14 status flags
Section 4.6.2.9	Changed instruction flow description; changed offset to ±32KB
Section 4.6.2.13	Changed offset in description to ±32KB
Section 4.6.2.17	Changed code example
Section 4.6.4.5	Changed code example
Section 5.2.1	Changed code example
Section 5.3.4	Added section
Section 5.4.2	Added EMEX bit
Section 6.3.1	Changed descriptions to be more generic on number of sectors available
Section 8.3	Added comment about clearing pending port interrupt flags prior to LPMx.5 entry
Section 8.4.14	Added register description
Section 11.2	Added note regarding operation during debug
Section 12.3.1	Updated WDTIS description
Section 13.2.1.1	Changed description
Section 13.3.4	Added section
Section 13.3.6	Changed bit name from IDEX to TAIDEX Added note to TAIDEX
Section 15.2.4	Updated description Added note regarding minimal calibration
Chapter 14	Added chapter
Chapter 16	Added chapter
Section 17.2.4	Updated description to clarify radix point rather than decimal point
Table 18-1	Changed REFON description
Chapter 19	Added chapter
Figure 20-1	Updated figure
Figure 20-2	Added figure
Section 20.2.3	The internal AVCC can also be used as the reference
Section 20.2.8	Changed description
Section 20.3.1	Updated ADC12REFON and ADC12_REF2_5V descriptions
Section 20.3.3	Updated ADC12TCOFF and ADC12REFOUT descriptions
Section 20.3.6	Fixed typo
Table 21-1	Changed CBIV reset
Section 22.4	Indicated registers fields that should be changed only when UCSWRST = 1
Section 23.4	Indicated registers fields that should be changed only when UCSWRST = 1
Section 23.5	Indicated registers fields that should be changed only when UCSWRST = 1
Section 24.3.4.2.2	Added note regarding repeated start
Section 24.4	Indicated registers fields that should be changed only when UCSWRST = 1

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Copyright © 2009–2013, Texas Instruments Incorporated

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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 significant portions 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. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2013, Texas Instruments Incorporated