# Si1000/1/2/3/4/5

**Ultra Low Power, 64/32 kB, 10-Bit ADC MCU with Integrated 240–960 MHz EZRadioPRO® Transceiver**

## Ultra Low Power: 0.9 to 3.6 V Operation

- Typical sleep mode current < 0.1 µA; retains state and RAM contents over full supply range; fast wakeup of < 2 µs
- Less than 600 nA with RTC running
- Less than 1 µA with RTC running and radio state retained
- On-chip dc-dc converter allows operation down to 0.9 V.
- Two built-in brown-out detectors cover sleep and active modes

## 10-Bit Analog to Digital Converter

- Up to 300 ksps
- Up to 18 external inputs
- External pin or internal VREF (no external capacitor required)
- Built-in temperature sensor
- External conversion start input option
- Autonomous burst mode with 16-bit automatic averaging accumulator

## Dual Comparators

- Programmable hysteresis and response time
- Configurable as interrupt or reset source
- Low current (< 0.5 µA)

## On-Chip Debug

- On-chip debug circuitry facilitates full-speed, non-intrusive in-system debug (No emulator required)
- Provides breakpoints, single stepping
- Inspect/modify memory and registers
- Complete development kit

## High-Speed 8051 µC Core

- Pipelined instruction architecture; executes 70% of instructions in 1 or 2 system clocks
- Up to **25 MIPS** throughput with 25 MHz clock
- Expanded interrupt handler

## Memory

- 4352 bytes internal data RAM (256 + 4096)
- 64 kB (Si1000/2/4) or 32 kB (Si1001/3/5) flash; In-system programmable in 1024-byte sectors—1024 bytes are reserved in the 64 kB devices

## EZRadioPRO® Transceiver

- Frequency range = 240–960 MHz
- Sensitivity = –121 dBm
- FSK, GFSK, and OOK modulation
- Max output power = +20 dBm (Si1000/1), +13 dBm (Si1002/3/4/5)
- RF power consumption
  - 18.5 mA receive
  - 18 mA @ +1 dBm transmit
  - 30 mA @ +13 dBm transmit
  - 85 mA @ +20 dBm transmit
- Data rate = 0.123 to 256 kbps
- Auto-frequency calibration (AFC)
- Antenna diversity and transmit/receive switch control
- Programmable packet handler
- TX and RX 64 byte FIFOs
- Frequency hopping capability
- On-chip crystal tuning

## Digital Peripherals

- 19 or 16 port I/O plus 3 GPIO pins; Hardware enhanced UART, SPI, and $I^2C$ serial ports available concurrently
- Low power 32-bit SmaRTClock
- Four general purpose 16-bit counter/timers; six channel programmable counter array (PCA)

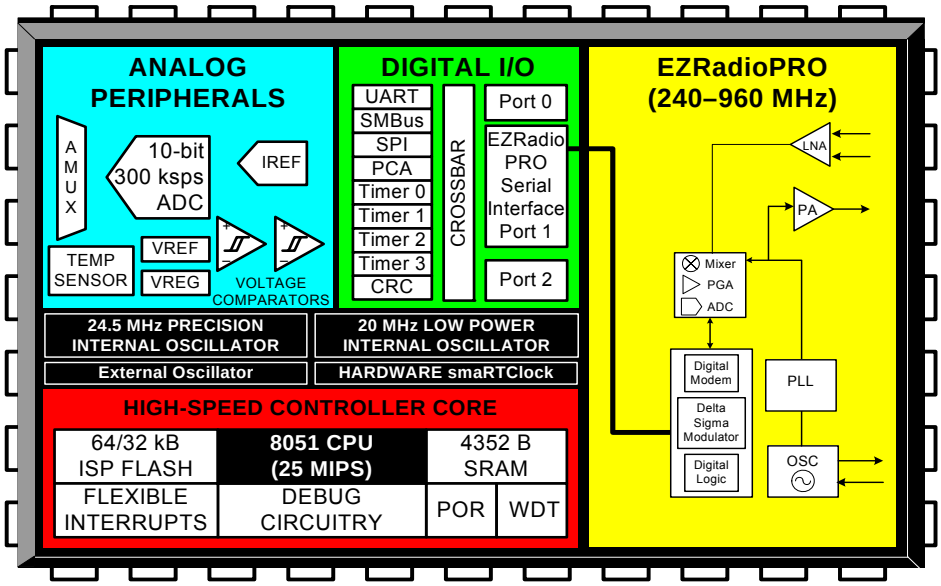
## Clock Sources

- Precision internal oscillators: 24.5 MHz with ±2% accuracy supports UART operation; spread-spectrum mode for reduced EMI; Low power 20 MHz internal oscillator
- External oscillator: Crystal, RC, C, CMOS clock
- SmaRTClock oscillator: 32.768 kHz crystal or self-oscillate
- Can switch between clock sources on-the-fly; useful in implementing various power saving modes

## Package

- 42-pin LGA (5 x 7 mm)

## Temperature Range: –40 to +85 °C





Copyright © 2013 by Silicon Laboratories

## Table of Contents

**1. System Overview ..... 17**
1.1. Typical Connection Diagram ..... 21
1.2. CIP-51™ Microcontroller Core ..... 22
1.3. Port Input/Output ..... 23
1.4. Serial Ports ..... 24
1.5. Programmable Counter Array ..... 24
1.6. 10-bit SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode ..... 25
1.7. Programmable Current Reference (IREF0) ..... 26
1.8. Comparators ..... 26
**2. Ordering Information ..... 28**
**3. Pinout and Package Definitions ..... 29**
**4. Electrical Characteristics ..... 40**
4.1. Absolute Maximum Specifications ..... 40
4.2. MCU Electrical Characteristics ..... 41
4.3. EZRadioPRO® Electrical Characteristics ..... 66
4.4. Definition of Test Conditions for the EZRadioPRO Peripheral ..... 73
**5. 10-Bit SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode ..... 74**
5.1. Output Code Formatting ..... 74
5.2. Modes of Operation ..... 76
5.3. 8-Bit Mode ..... 81
5.4. Programmable Window Detector ..... 88
5.5. ADC0 Analog Multiplexer ..... 91
5.6. Temperature Sensor ..... 93
5.7. Voltage and Ground Reference Options ..... 96
5.8. External Voltage References ..... 97
5.9. Internal Voltage References ..... 97
5.10. Analog Ground Reference ..... 97
5.11. Temperature Sensor Enable ..... 97
5.12. Voltage Reference Electrical Specifications ..... 98
**6. Programmable Current Reference (IREF0) ..... 99**
6.1. IREF0 Specifications ..... 99
**7. Comparators ..... 100**
7.1. Comparator Inputs ..... 100
7.2. Comparator Outputs ..... 101
7.3. Comparator Response Time ..... 102
7.4. Comparator Hysteresis ..... 102
7.5. Comparator Register Descriptions ..... 103
7.6. Comparator0 and Comparator1 Analog Multiplexers ..... 107
**8. CIP-51 Microcontroller ..... 110**
8.1. Performance ..... 110
8.2. Programming and Debugging Support ..... 111

8.3. Instruction Set ..... 111
8.4. CIP-51 Register Descriptions ..... 116
**9. Memory Organization ..... 119**
9.1. Program Memory ..... 120
9.2. Data Memory ..... 120
**10. On-Chip XRAM ..... 122**
10.1. Accessing XRAM ..... 122
10.2. Special Function Registers ..... 123
**11. Special Function Registers ..... 124**
11.1. SFR Paging ..... 125
**12. Interrupt Handler ..... 130**
12.1. Enabling Interrupt Sources ..... 130
12.2. MCU Interrupt Sources and Vectors ..... 130
12.3. Interrupt Priorities ..... 131
12.4. Interrupt Latency ..... 131
12.5. Interrupt Register Descriptions ..... 133
12.6. External Interrupts $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ ..... 140
**13. Flash Memory ..... 142**
13.1. Programming the Flash Memory ..... 142
13.2. Non-volatile Data Storage ..... 144
13.3. Security Options ..... 144
13.4. Determining the Device Part Number at Run Time ..... 146
13.5. Flash Write and Erase Guidelines ..... 146
13.6. Minimizing Flash Read Current ..... 148
**14. Power Management ..... 152**
14.1. Normal Mode ..... 153
14.2. Idle Mode ..... 154
14.3. Stop Mode ..... 154
14.4. Suspend Mode ..... 155
14.5. Sleep Mode ..... 155
14.6. Configuring Wakeup Sources ..... 156
14.7. Determining the Event that Caused the Last Wakeup ..... 156
14.8. Power Management Specifications ..... 158
**15. Cyclic Redundancy Check Unit (CRC0) ..... 159**
15.1. 16-bit CRC Algorithm ..... 159
15.2. 32-bit CRC Algorithm ..... 161
15.3. Preparing for a CRC Calculation ..... 163
15.4. Performing a CRC Calculation ..... 163
15.5. Accessing the CRC0 Result ..... 163
15.6. CRC0 Bit Reverse Feature ..... 167
**16. On-Chip DC-DC Converter (DC0) ..... 168**
16.1. Startup Behavior ..... 169
16.2. **High Power Applications ..... 170**
16.3. Pulse Skipping Mode ..... 170
16.4. Enabling the DC-DC Converter ..... 170

16.5. Minimizing Power Supply Noise ..... 172
16.6. Selecting the Optimum Switch Size ..... 172
16.7. DC-DC Converter Clocking Options ..... 172
16.8. DC-DC Converter Behavior in Sleep Mode ..... 173
16.9. DC-DC Converter Register Descriptions ..... 174
16.10. DC-DC Converter Specifications ..... 176
**17. Voltage Regulator (VREG0) ..... 177**
17.1. Voltage Regulator Electrical Specifications ..... 177
**18. Reset Sources ..... 178**
18.1. Power-On (VBAT Supply Monitor) Reset ..... 179
18.2. Power-Fail (VDD_MCU Supply Monitor) Reset ..... 180
18.3. External Reset ..... 182
18.4. Missing Clock Detector Reset ..... 182
18.5. Comparator0 Reset ..... 183
18.6. PCA Watchdog Timer Reset ..... 183
18.7. Flash Error Reset ..... 183
18.8. SmaRTClock (Real Time Clock) Reset ..... 183
18.9. Software Reset ..... 183
**19. Clocking Sources ..... 185**
19.1. Programmable Precision Internal Oscillator ..... 186
19.2. Low Power Internal Oscillator ..... 186
19.3. External Oscillator Drive Circuit ..... 186
19.4. Special Function Registers for Selecting and Configuring the System Clock ..... 190
**20. SmaRTClock (Real Time Clock) ..... 193**
20.1. SmaRTClock Interface ..... 193
20.2. SmaRTClock Clocking Sources ..... 200
20.3. SmaRTClock Timer and Alarm Function ..... 204
**21. Port Input/Output ..... 210**
21.1. Port I/O Modes of Operation ..... 211
21.2. Assigning Port I/O Pins to Analog and Digital Functions ..... 212
21.3. Priority Crossbar Decoder ..... 214
21.4. Port Match ..... 219
21.5. Special Function Registers for Accessing and Configuring Port I/O ..... 222
**22. EZRadioPRO® Serial Interface (SPI1) ..... 231**
22.1. Signal Descriptions ..... 232
22.2. SPI Master Operation on the MCU Core Side ..... 232
22.3. SPI Slave Operation on the EZRadioPRO Peripheral Side ..... 232
22.4. EZRadioPRO Serial Interface Interrupt Sources ..... 235
22.5. Serial Clock Phase and Polarity ..... 235
22.6. SPI Special Function Registers ..... 236
**23. EZRadioPRO® 240–960 MHz Transceiver ..... 242**
23.1. EZRadioPRO Operating Modes ..... 243
23.2. Interrupts ..... 246
23.3. System Timing ..... 247

23.4. Modulation Options ..... 254
23.5. Internal Functional Blocks ..... 259
23.6. Data Handling and Packet Handler ..... 264
23.7. RX Modem Configuration ..... 272
23.8. Auxiliary Functions ..... 272
23.9. Reference Design ..... 285
23.10. Application Notes and Reference Designs ..... 287
23.11. Customer Support ..... 287
23.12. Register Table and Descriptions ..... 288
23.13. Required Changes to Default Register Values ..... 290
**24. SMBus ..... 291**
24.1. Supporting Documents ..... 292
24.2. SMBus Configuration ..... 292
24.3. SMBus Operation ..... 292
24.4. Using the SMBus ..... 294
24.5. SMBus Transfer Modes ..... 306
24.6. SMBus Status Decoding ..... 309
**25. UART0 ..... 314**
25.1. Enhanced Baud Rate Generation ..... 315
25.2. Operational Modes ..... 316
25.3. Multiprocessor Communications ..... 317
**26. Enhanced Serial Peripheral Interface (SPI0) ..... 322**
26.1. Signal Descriptions ..... 323
26.2. SPI0 Master Mode Operation ..... 323
26.3. SPI0 Slave Mode Operation ..... 325
26.4. SPI0 Interrupt Sources ..... 326
26.5. Serial Clock Phase and Polarity ..... 327
26.6. SPI Special Function Registers ..... 328
**27. Timers ..... 335**
27.1. Timer 0 and Timer 1 ..... 337
27.2. Timer 2 ..... 345
27.3. Timer 3 ..... 351
**28. Programmable Counter Array ..... 357**
28.1. PCA Counter/Timer ..... 358
28.2. PCA0 Interrupt Sources ..... 359
28.3. Capture/Compare Modules ..... 360
28.4. Watchdog Timer Mode ..... 368
28.5. Register Descriptions for PCA0 ..... 370
**29. Device Specific Behavior ..... 376**
29.1. Device Identification ..... 376
**30. C2 Interface ..... 377**
30.1. C2 Interface Registers ..... 377
30.2. C2 Pin Sharing ..... 380
**Document Change List ..... 381**
**Contact Information ..... 382**

## List of Figures

Figure 1.1. Si1000 Block Diagram ..... 18
Figure 1.2. Si1001 Block Diagram ..... 18
Figure 1.3. Si1002 Block Diagram ..... 19
Figure 1.4. Si1003 Block Diagram ..... 19
Figure 1.5. Si1004 Block Diagram ..... 20
Figure 1.6. Si1005 Block Diagram ..... 20
Figure 1.7. Si1002/3 RX/TX Direct-tie Application Example ..... 21
Figure 1.8. Si1000/1 Antenna Diversity Application Example ..... 21
Figure 1.9. Port I/O Functional Block Diagram ..... 23
Figure 1.10. PCA Block Diagram ..... 24
Figure 1.11. ADC0 Functional Block Diagram ..... 25
Figure 1.12. ADC0 Multiplexer Block Diagram ..... 26
Figure 1.13. Comparator 0 Functional Block Diagram ..... 27
Figure 1.14. Comparator 1 Functional Block Diagram ..... 27
Figure 3.1. Si100/1/2/3-E-GM2 Pinout Diagram (Top View) ..... 33
Figure 3.2. Si1004/5-E-GM2 Pinout Diagram (Top View) ..... 34
Figure 3.3. LGA-42 Package Drawing (Si1000/1/2/3/4/5-E-GM2) ..... 35
Figure 3.4. LGA-42 PCB Land Pattern Dimensions (Si1000/1/2/3/4/5-E-GM2) ..... 37
Figure 3.5. LGA-42 PCB Stencil and Via Placement ..... 39
Figure 4.1. Active Mode Current (External CMOS Clock) ..... 44
Figure 4.2. Idle Mode Current (External CMOS Clock) ..... 45
Figure 4.3. Typical DC-DC Converter Efficiency (High Current, VDD/DC+ = 2 V ..... 46
Figure 4.4. Typical DC-DC Converter Efficiency (High Current, VDD/DC+ = 3 V) ..... 47
Figure 4.5. Typical DC-DC Converter Efficiency (Low Current, VDD/DC+ = 2 V) ..... 48
Figure 4.6. Typical One-Cell Suspend Mode Current ..... 49
Figure 4.7. Typical VOH Curves, 1.8–3.6 V ..... 51
Figure 4.8. Typical VOH Curves, 0.9–1.8 V ..... 52
Figure 4.9. Typical VOL Curves, 1.8–3.6 V ..... 53
Figure 4.10. Typical VOL Curves, 1.8–3.6 V ..... 54
Figure 4.11. Typical VOL Curves, 0.9–1.8 V ..... 55
Figure 5.1. ADC0 Functional Block Diagram ..... 74
Figure 5.2. 10-Bit ADC Track and Conversion Example Timing (BURSTEN = 0) ..... 77
Figure 5.3. Burst Mode Tracking Example with Repeat Count Set to 4 ..... 79
Figure 5.4. ADC0 Equivalent Input Circuits ..... 80
Figure 5.5. ADC Window Compare Example: Right-Justified Single-Ended Data ..... 90
Figure 5.6. ADC Window Compare Example: Left-Justified Single-Ended Data ..... 90
Figure 5.7. ADC0 Multiplexer Block Diagram ..... 91

Figure 5.8. Temperature Sensor Transfer Function ..... 93
Figure 5.9. Temperature Sensor Error with 1-Point Calibration ($V_{REF}$ = 1.68 V) ..... 94
Figure 5.10. Voltage Reference Functional Block Diagram ..... 96
Figure 7.1. Comparator 0 Functional Block Diagram ..... 100
Figure 7.2. Comparator 1 Functional Block Diagram ..... 101
Figure 7.3. Comparator Hysteresis Plot ..... 102
Figure 7.4. CPn Multiplexer Block Diagram ..... 107
Figure 8.1. CIP-51 Block Diagram ..... 110
Figure 9.1. Si1000/1/2/3/4/5 Memory Map ..... 119
Figure 9.2. Flash Program Memory Map ..... 120
Figure 13.1. Flash Program Memory Map ..... 144
Figure 14.1. Si1000/1/2/3/4/5 Power Distribution ..... 153
Figure 15.1. CRC0 Block Diagram ..... 159
Figure 15.2. Bit Reverse Register ..... 167
Figure 16.1. DC-DC Converter Block Diagram ..... 168
Figure 16.2. DC-DC Converter Configuration Options ..... 171
Figure 18.1. Reset Sources ..... 178
Figure 18.2. Power-Fail Reset Timing Diagram ..... 179
Figure 18.3. Power-Fail Reset Timing Diagram ..... 180
Figure 19.1. Clocking Sources Block Diagram ..... 185
Figure 19.2. 25 MHz External Crystal Example ..... 187
Figure 20.1. SmaRTClock Block Diagram ..... 193
Figure 20.2. Interpreting Oscillation Robustness (Duty Cycle) Test Results ..... 202
Figure 21.1. Port I/O Functional Block Diagram ..... 210
Figure 21.2. Port I/O Cell Block Diagram ..... 211
Figure 21.3. Crossbar Priority Decoder with No Pins Skipped ..... 215
Figure 21.4. Crossbar Priority Decoder with Crystal Pins Skipped ..... 216
Figure 22.1. EZRadioPRO Serial Interface Block Diagram ..... 231
Figure 22.2. SPI Timing ..... 233
Figure 22.3. SPI Timing—READ Mode ..... 233
Figure 22.4. SPI Timing—Burst Write Mode ..... 234
Figure 22.5. SPI Timing—Burst Read Mode ..... 234
Figure 22.6. Master Mode Data/Clock Timing ..... 235
Figure 22.7. SPI Master Timing ..... 241
Figure 23.1. State Machine Diagram ..... 244
Figure 23.2. TX Timing ..... 247
Figure 23.3. RX Timing ..... 248
Figure 23.4. Frequency Deviation ..... 251
Figure 23.5. Sensitivity at 1% PER vs. Carrier Frequency Offset ..... 253
Figure 23.6. FSK vs. GFSK Spectrums ..... 255
Figure 23.7. Direct Synchronous Mode Example ..... 258
Figure 23.8. Direct Asynchronous Mode Example ..... 258
Figure 23.9. Microcontroller Connections ..... 259
Figure 23.10. PLL Synthesizer Block Diagram ..... 261

Figure 23.11. FIFO Thresholds ..... 264
Figure 23.12. Packet Structure ..... 265
Figure 23.13. Multiple Packets in TX Packet Handler ..... 266
Figure 23.14. Required RX Packet Structure with Packet Handler Disabled ..... 266
Figure 23.15. Multiple Packets in RX Packet Handler ..... 267
Figure 23.16. Multiple Packets in RX with CRC or Header Error ..... 267
Figure 23.17. Operation of Data Whitening, Manchester Encoding, and CRC ..... 269
Figure 23.18. Manchester Coding Example ..... 269
Figure 23.19. Header ..... 271
Figure 23.20. POR Glitch Parameters ..... 272
Figure 23.21. General Purpose ADC Architecture ..... 275
Figure 23.22. Temperature Ranges using ADC8 ..... 277
Figure 23.23. WUT Interrupt and WUT Operation ..... 280
Figure 23.24. Low Duty Cycle Mode ..... 281
Figure 23.25. RSSI Value vs. Input Power ..... 284
Figure 23.26. Si1002 Split RF TX/RX Direct-Tie Reference Design—Schematic ..... 285
Figure 23.27. Si1000 Switch Matching Reference Design—Schematic ..... 286
Figure 24.1. SMBus Block Diagram ..... 291
Figure 24.2. Typical SMBus Configuration ..... 292
Figure 24.3. SMBus Transaction ..... 293
Figure 24.4. Typical SMBus SCL Generation ..... 295
Figure 24.5. Typical Master Write Sequence ..... 306
Figure 24.6. Typical Master Read Sequence ..... 307
Figure 24.7. Typical Slave Write Sequence ..... 308
Figure 24.8. Typical Slave Read Sequence ..... 309
Figure 25.1. UART0 Block Diagram ..... 314
Figure 25.2. UART0 Baud Rate Logic ..... 315
Figure 25.3. UART Interconnect Diagram ..... 316
Figure 25.4. 8-Bit UART Timing Diagram ..... 316
Figure 25.5. 9-Bit UART Timing Diagram ..... 317
Figure 25.6. UART Multi-Processor Mode Interconnect Diagram ..... 318
Figure 26.1. SPI Block Diagram ..... 322
Figure 26.2. Multiple-Master Mode Connection Diagram ..... 324
Figure 26.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram ..... 324
Figure 26.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram ..... 325
Figure 26.5. Master Mode Data/Clock Timing ..... 327
Figure 26.6. Slave Mode Data/Clock Timing (CKPHA = 0) ..... 328
Figure 26.7. Slave Mode Data/Clock Timing (CKPHA = 1) ..... 328
Figure 26.8. SPI Master Timing (CKPHA = 0) ..... 332
Figure 26.9. SPI Master Timing (CKPHA = 1) ..... 332
Figure 26.10. SPI Slave Timing (CKPHA = 0) ..... 333
Figure 26.11. SPI Slave Timing (CKPHA = 1) ..... 333

Figure 27.1. T0 Mode 0 Block Diagram ..... 338
Figure 27.2. T0 Mode 2 Block Diagram ..... 339
Figure 27.3. T0 Mode 3 Block Diagram ..... 340
Figure 27.4. Timer 2 16-Bit Mode Block Diagram ..... 345
Figure 27.5. Timer 2 8-Bit Mode Block Diagram ..... 346
Figure 27.6. Timer 2 Capture Mode Block Diagram ..... 347
Figure 27.7. Timer 3 16-Bit Mode Block Diagram ..... 351
Figure 27.8. Timer 3 8-Bit Mode Block Diagram. ..... 352
Figure 27.9. Timer 3 Capture Mode Block Diagram ..... 353
Figure 28.1. PCA Block Diagram ..... 357
Figure 28.2. PCA Counter/Timer Block Diagram ..... 358
Figure 28.3. PCA Interrupt Block Diagram ..... 359
Figure 28.4. PCA Capture Mode Diagram ..... 361
Figure 28.5. PCA Software Timer Mode Diagram ..... 362
Figure 28.6. PCA High-Speed Output Mode Diagram ..... 363
Figure 28.7. PCA Frequency Output Mode ..... 364
Figure 28.8. PCA 8-Bit PWM Mode Diagram ..... 365
Figure 28.9. PCA 9, 10 and 11-Bit PWM Mode Diagram ..... 366
Figure 28.10. PCA 16-Bit PWM Mode ..... 367
Figure 28.11. PCA Module 5 with Watchdog Timer Enabled ..... 368
Figure 29.1. Si100x Revision Information ..... 376
Figure 30.1. Typical C2 Pin Sharing ..... 380

## List of Tables

Table 2.1. Product Selection Guide ..... 28
Table 3.1. Pin Definitions for the Si1000/1/2/3/4/5 ..... 29
Table 3.2. LGA-42 Package Dimensions (Si1000/1/2/3/4/5-E-GM2) ..... 36
Table 3.3. LGA-42 PCB Land Pattern Dimensions (Si1000/1/2/3/4/5-E-GM2) ..... 38
Table 4.1. Absolute Maximum Ratings ..... 40
Table 4.2. Global Electrical Characteristics ..... 41
Table 4.3. Port I/O DC Electrical Characteristics ..... 50
Table 4.4. Reset Electrical Characteristics ..... 56
Table 4.5. Power Management Electrical Specifications ..... 57
Table 4.6. Flash Electrical Characteristics ..... 57
Table 4.7. Internal Precision Oscillator Electrical Characteristics ..... 58
Table 4.8. Internal Low-Power Oscillator Electrical Characteristics ..... 58
Table 4.9. ADC0 Electrical Characteristics ..... 59
Table 4.10. Temperature Sensor Electrical Characteristics ..... 60
Table 4.11. Voltage Reference Electrical Characteristics ..... 60
Table 4.12. IREF0 Electrical Characteristics ..... 61
Table 4.13. Comparator Electrical Characteristics ..... 62
Table 4.14. DC-DC Converter (DC0) Electrical Characteristics ..... 64
Table 4.15. VREG0 Electrical Characteristics ..... 65
Table 4.16. DC Characteristics ..... 66
Table 4.17. Synthesizer AC Electrical Characteristics ..... 67
Table 4.18. Receiver AC Electrical Characteristics ..... 68
Table 4.19. Transmitter AC Electrical Characteristics ..... 69
Table 4.20. Auxiliary Block Specifications ..... 70
Table 4.21. Digital IO Specifications (nIRQ) ..... 71
Table 4.22. GPIO Specifications (GPIO_0, GPIO_1, and GPIO_2) ..... 71
Table 4.23. Absolute Maximum Ratings ..... 72
Table 8.1. CIP-51 Instruction Set Summary ..... 112
Table 11.1. Special Function Register (SFR) Memory Map (Page 0x0) ..... 124
Table 11.2. Special Function Register (SFR) Memory Map (Page 0xF) ..... 125
Table 11.3. Special Function Registers ..... 126
Table 12.1. Interrupt Summary ..... 132
Table 13.1. Flash Security Summary ..... 145
Table 14.1. Power Modes ..... 152
Table 15.1. Example 16-bit CRC Outputs ..... 160
Table 15.2. Example 32-bit CRC Outputs ..... 162
Table 16.1. IPeak Inductor Current Limit Settings ..... 169
Table 19.1. Recommended XFCN Settings for Crystal Mode ..... 187
Table 19.2. Recommended XFCN Settings for RC and C modes ..... 188
Table 20.1. SmaRTClock Internal Registers ..... 194
Table 20.2. SmaRTClock Load Capacitance Settings ..... 201
Table 20.3. SmaRTClock Bias Settings ..... 203
Table 21.1. Port I/O Assignment for Analog Functions ..... 213

Table 21.2. Port I/O Assignment for Digital Functions ..... 213
Table 21.3. Port I/O Assignment for External Digital Event Capture Functions .... 214
Table 22.1. Serial Interface Timing Parameters ..... 233
Table 22.2. SPI Timing Parameters ..... 241
Table 23.1. EZRadioPRO Operating Modes ..... 243
Table 23.2. EZRadioPRO Operating Modes Response Time ..... 244
Table 23.3. Frequency Band Selection ..... 249
Table 23.4. Packet Handler Registers ..... 268
Table 23.5. Minimum Receiver Settling Time ..... 270
Table 23.6. POR Parameters ..... 273
Table 23.7. Temperature Sensor Range ..... 276
Table 23.8. Antenna Diversity Control ..... 283
Table 23.9. EZRadioPRO Internal Register Descriptions ..... 288
Table 24.1. SMBus Clock Source Selection ..... 295
Table 24.2. Minimum SDA Setup and Hold Times ..... 296
Table 24.3. Sources for Hardware Changes to SMB0CN ..... 300
Table 24.4. Hardware Address Recognition Examples (EHACK = 1) ..... 301
Table 24.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) ..... 310
Table 24.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) ..... 312
Table 25.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator ..... 321
Table 25.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator ..... 321
Table 26.1. SPI Slave Timing Parameters ..... 334
Table 27.1. Timer 0 Running Modes ..... 337
Table 28.1. PCA Timebase Input Options ..... 358
Table 28.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules ..... 360
Table 28.3. Watchdog Timer Timeout Intervals1 ..... 369

## List of Registers

SFR Definition 5.1. ADC0CN: ADC0 Control ..... 82
SFR Definition 5.2. ADC0CF: ADC0 Configuration ..... 83
SFR Definition 5.3. ADC0AC: ADC0 Accumulator Configuration ..... 84
SFR Definition 5.4. ADC0PWR: ADC0 Burst Mode Power-Up Time ..... 85
SFR Definition 5.5. ADC0TK: ADC0 Burst Mode Track Time ..... 86
SFR Definition 5.6. ADC0H: ADC0 Data Word High Byte ..... 87
SFR Definition 5.7. ADC0L: ADC0 Data Word Low Byte ..... 87
SFR Definition 5.8. ADC0GTH: ADC0 Greater-Than High Byte ..... 88
SFR Definition 5.9. ADC0GTL: ADC0 Greater-Than Low Byte ..... 88
SFR Definition 5.10. ADC0LTH: ADC0 Less-Than High Byte ..... 89
SFR Definition 5.11. ADC0LTL: ADC0 Less-Than Low Byte ..... 89
SFR Definition 5.12. ADC0MX: ADC0 Input Channel Select ..... 92
SFR Definition 5.13. TOFFH: ADC0 Data Word High Byte ..... 95
SFR Definition 5.14. TOFFL: ADC0 Data Word Low Byte ..... 95
SFR Definition 5.15. REF0CN: Voltage Reference Control ..... 98
SFR Definition 6.1. IREF0CN: Current Reference Control ..... 99
SFR Definition 7.1. CPT0CN: Comparator 0 Control ..... 103
SFR Definition 7.2. CPT0MD: Comparator 0 Mode Selection ..... 104
SFR Definition 7.3. CPT1CN: Comparator 1 Control ..... 105
SFR Definition 7.4. CPT1MD: Comparator 1 Mode Selection ..... 106
SFR Definition 7.5. CPT0MX: Comparator0 Input Channel Select ..... 108
SFR Definition 7.6. CPT1MX: Comparator1 Input Channel Select ..... 109
SFR Definition 8.1. DPL: Data Pointer Low Byte ..... 116
SFR Definition 8.2. DPH: Data Pointer High Byte ..... 116
SFR Definition 8.3. SP: Stack Pointer ..... 117
SFR Definition 8.4. ACC: Accumulator ..... 117
SFR Definition 8.5. B: B Register ..... 117
SFR Definition 8.6. PSW: Program Status Word ..... 118
SFR Definition 10.1. EMI0CN: External Memory Interface Control ..... 123
SFR Definition 11.1. SFRPage: SFR Page ..... 126
SFR Definition 12.1. IE: Interrupt Enable ..... 134
SFR Definition 12.2. IP: Interrupt Priority ..... 135
SFR Definition 12.3. EIE1: Extended Interrupt Enable 1 ..... 136
SFR Definition 12.4. EIP1: Extended Interrupt Priority 1 ..... 137
SFR Definition 12.5. EIE2: Extended Interrupt Enable 2 ..... 138
SFR Definition 12.6. EIP2: Extended Interrupt Priority 2 ..... 139
SFR Definition 12.7. IT01CF: INT0/INT1 Configuration ..... 141
SFR Definition 13.1. PSCTL: Program Store R/W Control ..... 149
SFR Definition 13.2. FLKEY: Flash Lock and Key ..... 150
SFR Definition 13.3. FLSCL: Flash Scale ..... 151
SFR Definition 13.4. FLWR: Flash Write Only ..... 151
SFR Definition 14.1. PMU0CF: Power Management Unit Configuration ..... 157
SFR Definition 14.2. PCON: Power Management Control Register ..... 158
SFR Definition 15.1. CRC0CN: CRC0 Control ..... 164

SFR Definition 15.2. CRC0IN: CRC0 Data Input ..... 165
SFR Definition 15.3. CRC0DAT: CRC0 Data Output ..... 165
SFR Definition 15.4. CRC0AUTO: CRC0 Automatic Control ..... 166
SFR Definition 15.5. CRC0CNT: CRC0 Automatic Flash Sector Count ..... 166
SFR Definition 15.6. CRC0FLIP: CRC0 Bit Flip ..... 167
SFR Definition 16.1. DC0CN: DC-DC Converter Control ..... 174
SFR Definition 16.2. DC0CF: DC-DC Converter Configuration ..... 175
SFR Definition 17.1. REG0CN: Voltage Regulator Control ..... 177
SFR Definition 18.1. VDM0CN: VDD_MCU Supply Monitor Control ..... 182
SFR Definition 18.2. RSTSRC: Reset Source ..... 184
SFR Definition 19.1. CLKSEL: Clock Select ..... 190
SFR Definition 19.2. OSCICN: Internal Oscillator Control ..... 191
SFR Definition 19.3. OSCICL: Internal Oscillator Calibration ..... 191
SFR Definition 19.4. OSCXCN: External Oscillator Control ..... 192
SFR Definition 20.1. RTC0KEY: SmaRTClock Lock and Key ..... 197
SFR Definition 20.2. RTC0ADR: SmaRTClock Address ..... 198
SFR Definition 20.3. RTC0DAT: SmaRTClock Data ..... 199
Internal Register Definition 20.4. RTC0CN: SmaRTClock Control ..... 206
Internal Register Definition 20.5. RTC0XCN:
SmaRTClock Oscillator Control ..... 207
Internal Register Definition 20.6. RTC0XCF:
SmaRTClock Oscillator Configuration ..... 208
Internal Register Definition 20.7. RTC0PIN:
SmaRTClock Pin Configuration ..... 208
Internal Register Definition 20.8. CAPTUREn:
SmaRTClock Timer Capture ..... 209
Internal Register Definition 20.9. ALARMn: SmaRTClock Alarm
Programmed Value ..... 209
SFR Definition 21.1. XBR0: Port I/O Crossbar Register 0 ..... 217
SFR Definition 21.2. XBR1: Port I/O Crossbar Register 1 ..... 218
SFR Definition 21.3. XBR2: Port I/O Crossbar Register 2 ..... 219
SFR Definition 21.4. P0MASK: Port0 Mask Register ..... 220
SFR Definition 21.5. P0MAT: Port0 Match Register ..... 220
SFR Definition 21.6. P1MASK: Port1 Mask Register ..... 221
SFR Definition 21.7. P1MAT: Port1 Match Register ..... 221
SFR Definition 21.8. P0: Port0 ..... 223
SFR Definition 21.9. P0SKIP: Port0 Skip ..... 223
SFR Definition 21.10. P0MDIN: Port0 Input Mode ..... 224
SFR Definition 21.11. P0MDOUT: Port0 Output Mode ..... 224
SFR Definition 21.12. P0DRV: Port0 Drive Strength ..... 225
SFR Definition 21.13. P1: Port1 ..... 226
SFR Definition 21.14. P1SKIP: Port1 Skip ..... 226
SFR Definition 21.15. P1MDIN: Port1 Input Mode ..... 227
SFR Definition 21.16. P1MDOUT: Port1 Output Mode ..... 227
SFR Definition 21.17. P1DRV: Port1 Drive Strength ..... 228

SFR Definition 21.18. P2: Port2 ..... 228
SFR Definition 21.19. P2SKIP: Port2 Skip ..... 229
SFR Definition 21.20. P2MDIN: Port2 Input Mode ..... 229
SFR Definition 21.21. P2MDOUT: Port2 Output Mode ..... 230
SFR Definition 21.22. P2DRV: Port2 Drive Strength ..... 230
SFR Definition 22.1. SPI1CFG: SPI Configuration ..... 237
SFR Definition 22.2. SPI1CN: SPI Control ..... 238
SFR Definition 22.3. SPI1CKR: SPI Clock Rate ..... 239
SFR Definition 22.4. SPI1DAT: SPI Data ..... 240
SFR Definition 24.1. SMB0CF: SMBus Clock/Configuration ..... 297
SFR Definition 24.2. SMB0CN: SMBus Control ..... 299
SFR Definition 24.3. SMB0ADR: SMBus Slave Address ..... 302
SFR Definition 24.4. SMB0ADM: SMBus Slave Address Mask ..... 302
SFR Definition 24.5. SMB0DAT: SMBus Data ..... 305
SFR Definition 25.1. SCON0: Serial Port 0 Control ..... 319
SFR Definition 25.2. SBUF0: Serial (UART0) Port Data Buffer ..... 320
SFR Definition 26.7. SPI0CFG: SPI0 Configuration ..... 329
SFR Definition 26.8. SPI0CN: SPI0 Control ..... 330
SFR Definition 26.9. SPI0CKR: SPI0 Clock Rate ..... 331
SFR Definition 26.10. SPI0DAT: SPI0 Data ..... 331
SFR Definition 27.1. CKCON: Clock Control ..... 336
SFR Definition 27.2. TCON: Timer Control ..... 341
SFR Definition 27.3. TMOD: Timer Mode ..... 342
SFR Definition 27.4. TL0: Timer 0 Low Byte ..... 343
SFR Definition 27.5. TL1: Timer 1 Low Byte ..... 343
SFR Definition 27.6. TH0: Timer 0 High Byte ..... 344
SFR Definition 27.7. TH1: Timer 1 High Byte ..... 344
SFR Definition 27.8. TMR2CN: Timer 2 Control ..... 348
SFR Definition 27.9. TMR2RLL: Timer 2 Reload Register Low Byte ..... 349
SFR Definition 27.10. TMR2RLH: Timer 2 Reload Register High Byte ..... 349
SFR Definition 27.11. TMR2L: Timer 2 Low Byte ..... 350
SFR Definition 27.12. TMR2H Timer 2 High Byte ..... 350
SFR Definition 27.13. TMR3CN: Timer 3 Control ..... 354
SFR Definition 27.14. TMR3RLL: Timer 3 Reload Register Low Byte ..... 355
SFR Definition 27.15. TMR3RLH: Timer 3 Reload Register High Byte ..... 355
SFR Definition 27.16. TMR3L: Timer 3 Low Byte ..... 356
SFR Definition 27.17. TMR3H Timer 3 High Byte ..... 356
SFR Definition 28.1. PCA0CN: PCA Control ..... 370
SFR Definition 28.2. PCA0MD: PCA Mode ..... 371
SFR Definition 28.3. PCA0PWM: PCA PWM Configuration ..... 372
SFR Definition 28.4. PCA0CPMn: PCA Capture/Compare Mode ..... 373
SFR Definition 28.5. PCA0L: PCA Counter/Timer Low Byte ..... 374
SFR Definition 28.6. PCA0H: PCA Counter/Timer High Byte ..... 374
SFR Definition 28.7. PCA0CPLn: PCA Capture Module Low Byte ..... 375
SFR Definition 28.8. PCA0CPHn: PCA Capture Module High Byte ..... 375

C2 Register Definition 30.1. C2ADD: C2 Address ..... 377
C2 Register Definition 30.2. DEVICEID: C2 Device ID ..... 378
C2 Register Definition 30.3. REVID: C2 Revision ID ..... 378
C2 Register Definition 30.4. FPCTL: C2 Flash Programming Control ..... 379
C2 Register Definition 30.5. FPDAT: C2 Flash Programming Data ..... 379

# 1. System Overview

Si1000/1/2/3/4/5 devices are fully integrated mixed-signal system-on-a-chip MCUs. Highlighted features are listed below. Refer to Table 2.1 for specific product feature selection and part ordering numbers.

- 240–960 MHz EZRadioPRO® transceiver
- Single/Dual battery operation with on-chip dc-dc boost converter
- High-speed pipelined 8051-compatible microcontroller core (up to 25 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- True 10-bit 300 ksps 23-channel single-ended ADC with analog multiplexer
- 6-bit programmable current reference
- Precision programmable 24.5 MHz internal oscillator with spread spectrum technology
- 64 kB or 32 kB of on-chip flash memory
- 4352 bytes of on-chip RAM
- SMBus/$I^2C$, Enhanced UART, and two Enhanced SPI serial interfaces implemented in hardware (SPI1 is dedicated for communication with the EZRadioPRO peripheral)
- Four general-purpose 16-bit timers
- Programmable counter/timer array (PCA) with six capture/compare modules and watchdog timer (WDT) function
- On-chip power-on reset, $V_{DD}$ monitor, and temperature sensor
- Two on-chip voltage comparators with 18 touch sense inputs
- 19 or 22 port I/O (5 V tolerant except for GPIO_0, GPIO_1, and GPIO_2)

With on-chip power-on reset, $V_{DD}$ monitor, watchdog timer, and clock oscillator, the Si1000/1/2/3/4/5 devices are truly standalone system-on-a-chip solutions. The flash memory can be reprogrammed even in-circuit, providing non-volatile data storage, and also allowing field upgrades of the 8051 firmware. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

The on-chip Silicon Labs 2-Wire (C2) Development Interface allows non-intrusive (uses no on-chip resources), full speed, in-circuit debugging using the production MCU installed in the final application. This debug logic supports inspection and modification of memory and registers, setting breakpoints, single stepping, and run and halt commands. All analog and digital peripherals are fully functional while debugging using C2. The two C2 interface pins can be shared with user functions, allowing in-system debugging without occupying package pins.

Each device is specified for 1.8 to 3.6 V operation over the industrial temperature range (–40 to +85 °C). The Port I/O and $\overline{\text{RST}}$ pins are tolerant of input signals up to 5 V. The Si1000/1/2/3/4/5 are available in a 42-pin LGA package (lead-free and RoHS compliant). See Table 2.1 for ordering information. Block diagrams are included in Figure 1.1 through Figure 1.6.

The transceiver's extremely low receive sensitivity (–121 dBm) coupled with industry leading +20 dBm output power ensures extended range and improved link performance. Built-in antenna diversity and support for frequency hopping can be used to further extend range and enhance performance. The advanced radio features including continuous frequency coverage from 240–960 MHz in 156 Hz or 312 Hz steps allow precise tuning control. Additional system features such as an automatic wake-up timer, low battery detector, 64 byte TX/RX FIFOs, automatic packet handling, and preamble detection reduce overall current consumption. The transceivers digital receive architecture features a high-performance ADC and DSP-based modem which performs demodulation, filtering, and packet handling for increased flexibility and performance. The direct digital transmit modulation and automatic PA power ramping ensure precise transmit modulation and reduced spectral spreading, ensuring compliance with global regulations including FCC, ETSI, ARIB, and 802.15.4d regulations.

An easy-to-use calculator is provided to quickly configure the radio settings, simplifying customer's system design and reducing time to market.

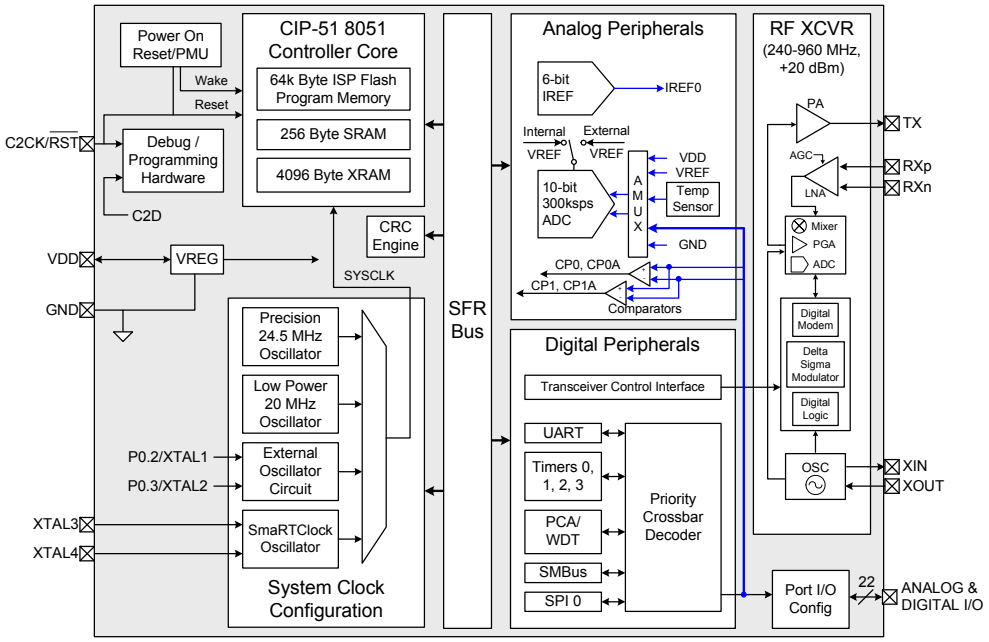



**Figure 1.1. Si1000 Block Diagram**

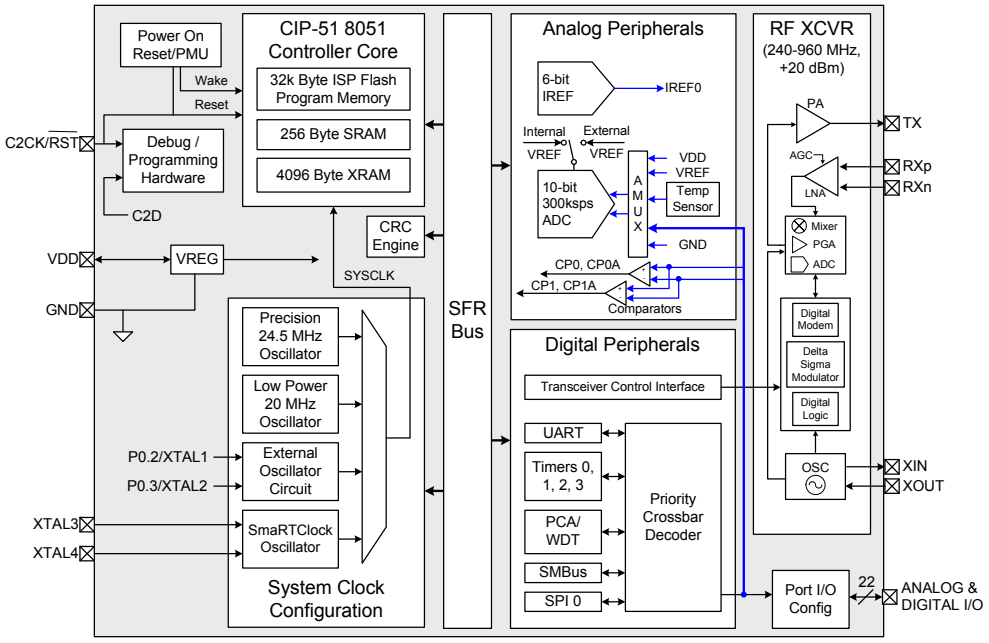



**Figure 1.2. Si1001 Block Diagram**

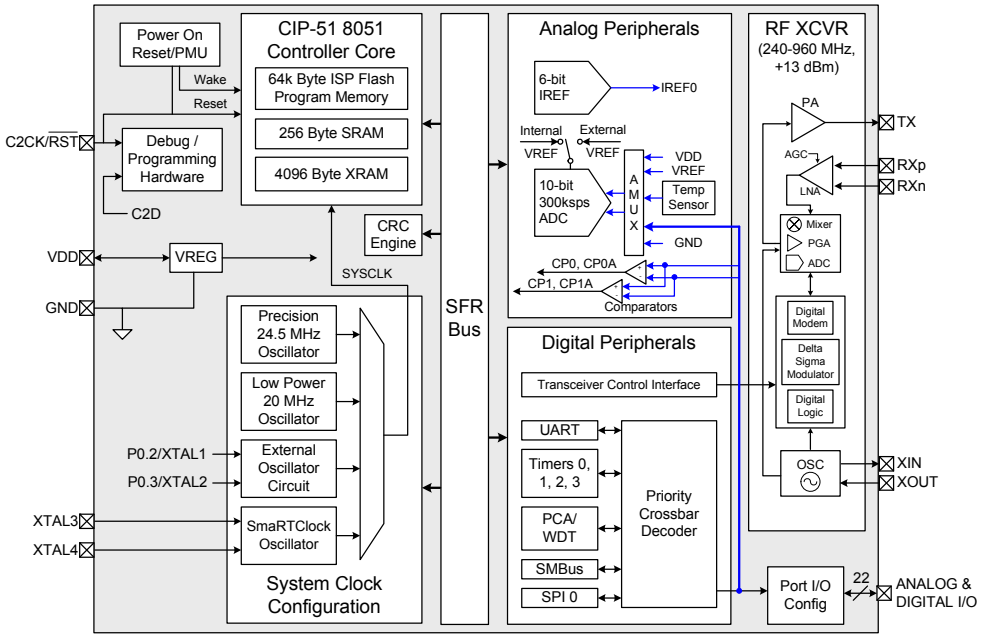



**Figure 1.3. Si1002 Block Diagram**

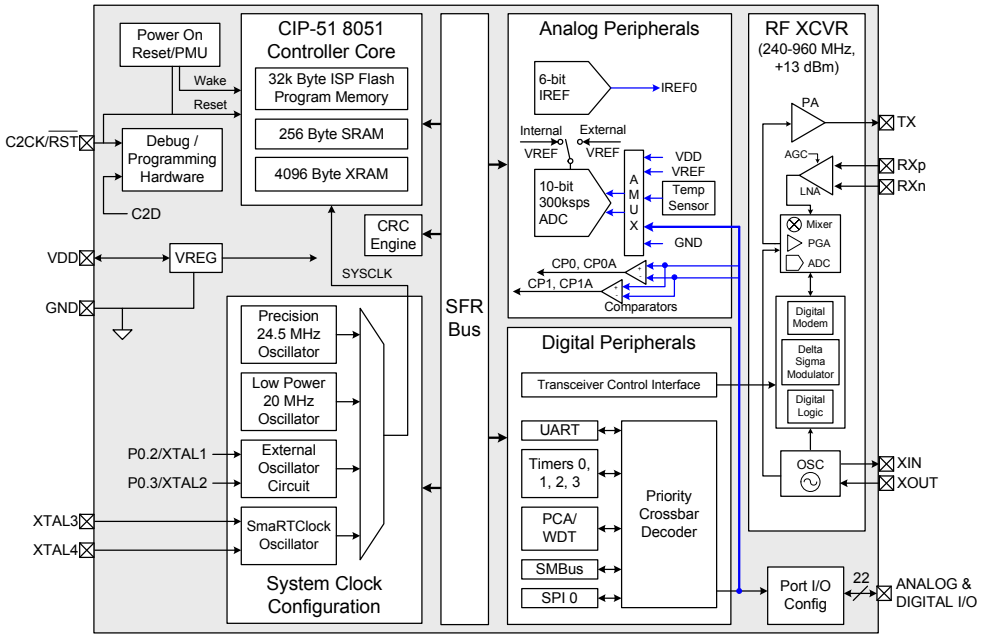



**Figure 1.4. Si1003 Block Diagram**

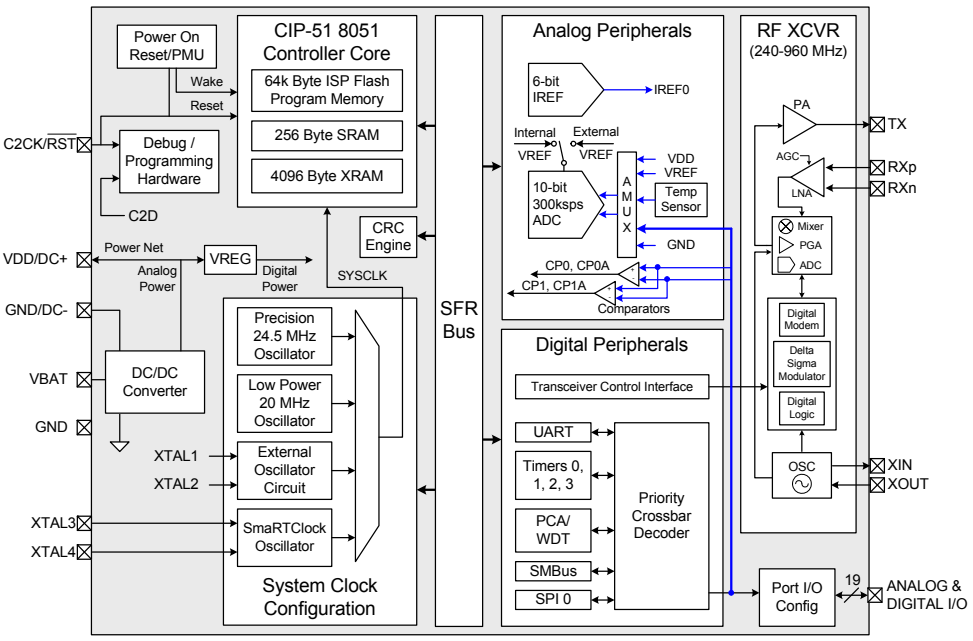



**Figure 1.5. Si1004 Block Diagram**

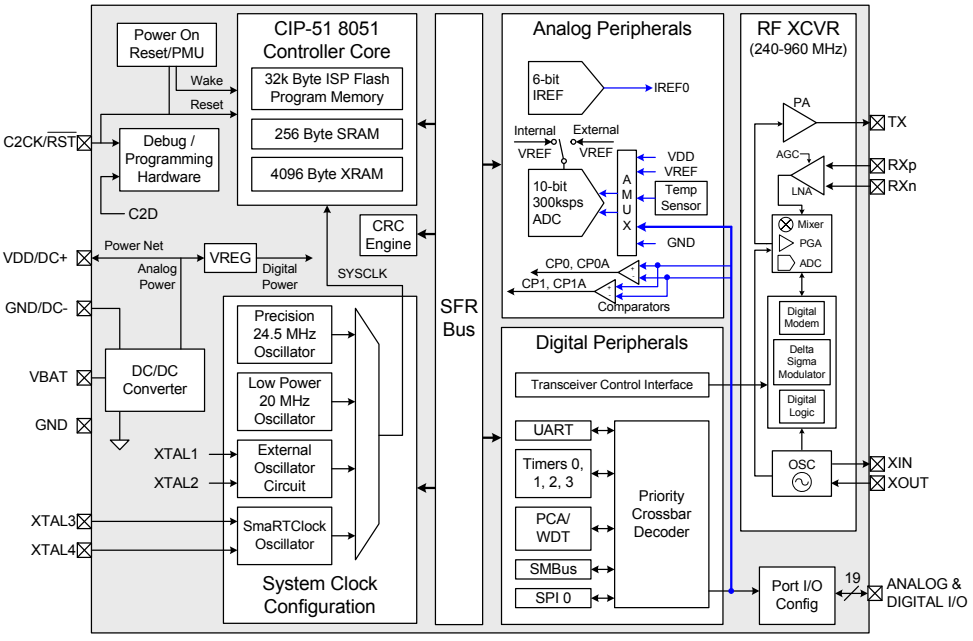



**Figure 1.6. Si1005 Block Diagram**

## 1.1. Typical Connection Diagram

The application shown in Figure 1.7 is designed for a system with a TX/RX direct-tie configuration without the use of a TX/RX switch. Most lower power applications will use this configuration. A complete direct-tie reference design is available from Silicon Laboratories applications support.

For applications seeking improved performance in the presence of multipath fading, antenna diversity can be used. Antenna diversity support is integrated into the EZRadioPRO transceiver and can improve the system link budget by 8–10 dB in the presence of these fading conditions, resulting in substantial range increases. A complete Antenna Diversity reference design is available from Silicon Laboratories applications support.

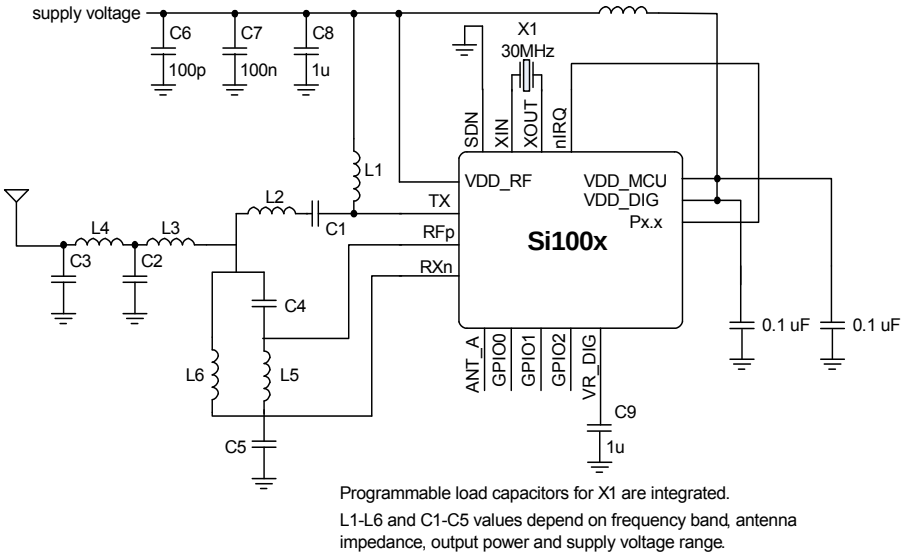



**Figure 1.7. Si1002/3 RX/TX Direct-tie Application Example**

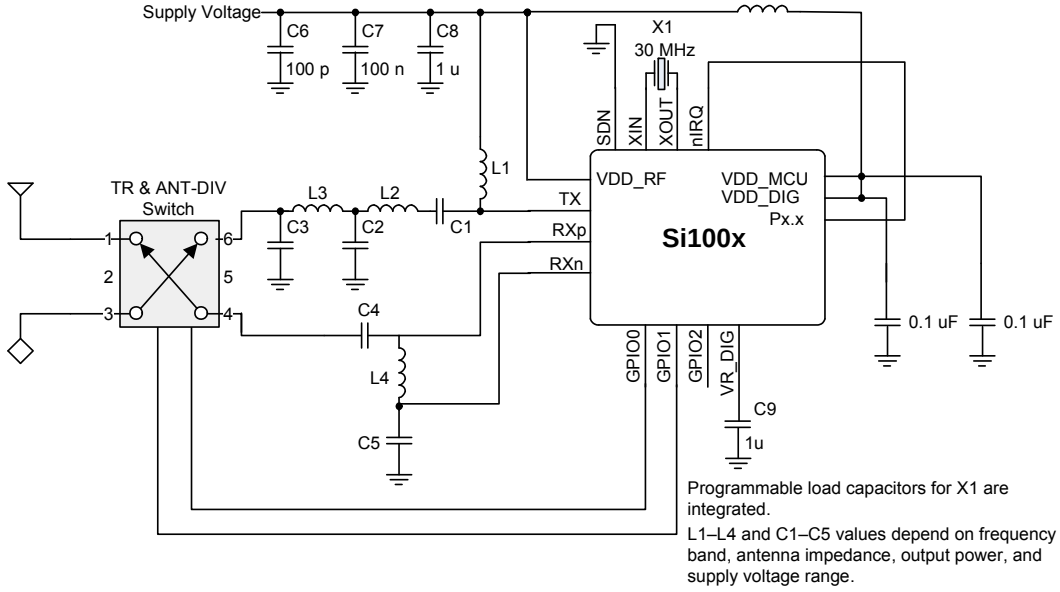



**Figure 1.8. Si1000/1 Antenna Diversity Application Example**

## 1.2. CIP-51™ Microcontroller Core

### 1.2.1. Fully 8051 Compatible

The Si1000/1/2/3/4/5 family utilizes Silicon Labs' proprietary CIP-51 microcontroller core. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The CIP-51 core offers all the peripherals included with a standard 8052.

### 1.2.2. Improved Throughput

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. In a standard 8051, all instructions except for MUL and DIV take 12 or 24 system clock cycles to execute with a maximum system clock of 12-to-24 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with only four instructions taking more than four system clock cycles.

The CIP-51 has a total of 109 instructions. The table below shows the total number of instructions that require each execution time.

| Clocks to Execute | 1 | 2 | 2/3 | 3 | 3/4 | 4 | 4/5 | 5 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| Number of Instructions | 26 | 50 | 5 | 14 | 7 | 3 | 1 | 2 | 1 |

With the CIP-51's maximum system clock at 25 MHz, it has a peak throughput of 25 MIPS.

### 1.2.3. Additional Features

The Si1000/1/2/3/4/5 SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

The extended interrupt handler provides multiple interrupt sources into the CIP-51, allowing numerous analog and digital peripherals to interrupt the controller. An interrupt driven system requires less intervention by the MCU, giving it more effective throughput. The extra interrupt sources are very useful when building multi-tasking, real-time systems.

Eight reset sources are available: power-on reset circuitry (POR), an on-chip $V_{DD}$ monitor (forces reset when power supply voltage drops below safe levels), a watchdog timer, a Missing Clock Detector, SmaRT-Clock oscillator fail or alarm, a voltage level detection from Comparator0, a forced software reset, an external reset pin, and an illegal flash access protection circuit. Each reset source except for the POR, Reset Input Pin, or flash error may be disabled by the user in software. The WDT may be permanently disabled in software after a power-on reset during MCU initialization.

The internal oscillator factory is calibrated to 24.5 MHz and is accurate to ±2% over the full temperature and supply range. The internal oscillator period can also be adjusted by user firmware. An additional 20 MHz low power oscillator is also available which facilitates low-power operation. An external oscillator drive circuit is included, allowing an external crystal, ceramic resonator, capacitor, RC, or CMOS clock source to generate the system clock. If desired, the system clock source may be switched between both internal and external oscillator circuits. An external oscillator can also be extremely useful in low power applications, allowing the MCU to run from a slow (power saving) source, while periodically switching to the fast (up to 25 MHz) internal oscillator as needed.

## 1.3. Port Input/Output

Digital and analog resources are available through 19 (Si1000/1/2/3) or 16 (Si1004/5) I/O pins. Three additional GPIO pins are available through the EZRadioPRO peripheral. Port pins are organized as three byte-wide ports. Port pins P0.0–P2.6 can be defined as digital or analog I/O. Digital I/O pins can be assigned to one of the internal digital resources or used as general purpose I/O (GPIO). Analog I/O pins are used by the internal analog resources. P1.0, P1.1, P1.2, and P1.4 are dedicated for communication with the EZRadioPRO peripheral. P1.3 is not available. P2.7 can be used as GPIO and is shared with the C2 Interface Data signal (C2D). See Section “29. Device Specific Behavior” on page 376 for more details.

The designer has complete control over which digital and analog functions are assigned to individual port pins and is limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. See Section “21.3. Priority Crossbar Decoder” on page 214 for more information on the crossbar.

All Px.x Port I/Os are 5 V tolerant when used as digital inputs or open-drain outputs. For Port I/Os configured as push-pull outputs, current is sourced from the VDD_MCU supply. Port I/Os used for analog functions can operate up to the VDD_MCU supply voltage. See Section “21.1. Port I/O Modes of Operation” on page 211 for more information on Port I/O operating modes and the electrical specifications chapter for detailed electrical specifications.

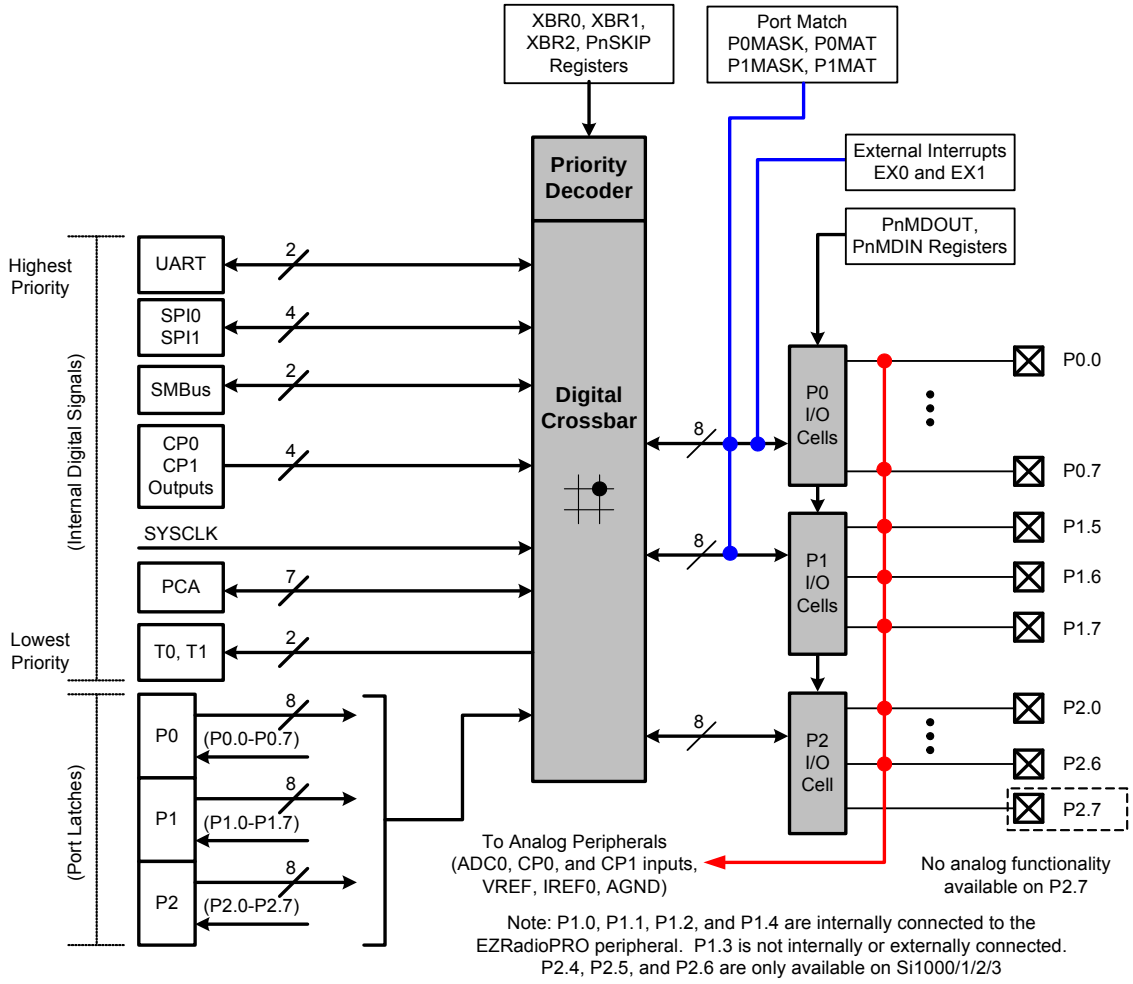



**Figure 1.9. Port I/O Functional Block Diagram**

## 1.4. Serial Ports

The Si1000/1/2/3/4/5 family includes an SMBus/$I^2C$ interface, a full-duplex UART with enhanced baud rate configuration, and an Enhanced SPI interface. Each of the serial buses is fully implemented in hardware and makes extensive use of the CIP-51's interrupts, thus requiring very little CPU intervention. There is also a dedicated EZRadioPRO Serial Interface (SPI1) to allow communication with the EZRadioPRO peripheral.

## 1.5. Programmable Counter Array

An on-chip Programmable Counter/Timer Array (PCA) is included in addition to the four 16-bit general purpose counter/timers. The PCA consists of a dedicated 16-bit counter/timer time base with six programmable capture/compare modules. The PCA clock is derived from one of six sources: the system clock divided by 12, the system clock divided by 4, Timer 0 overflows, an External Clock Input (ECI), the system clock, or the external oscillator clock source divided by 8.

Each capture/compare module can be configured to operate in a variety of modes: edge-triggered capture, software timer, high-speed output, pulse width modulator (8, 9, 10, 11, or 16-bit), or frequency output. Additionally, Capture/Compare Module 5 offers watchdog timer capabilities. Following a system reset, Module 5 is configured and enabled in WDT mode. The PCA Capture/Compare Module I/O and External Clock Input may be routed to Port I/O via the Digital Crossbar.

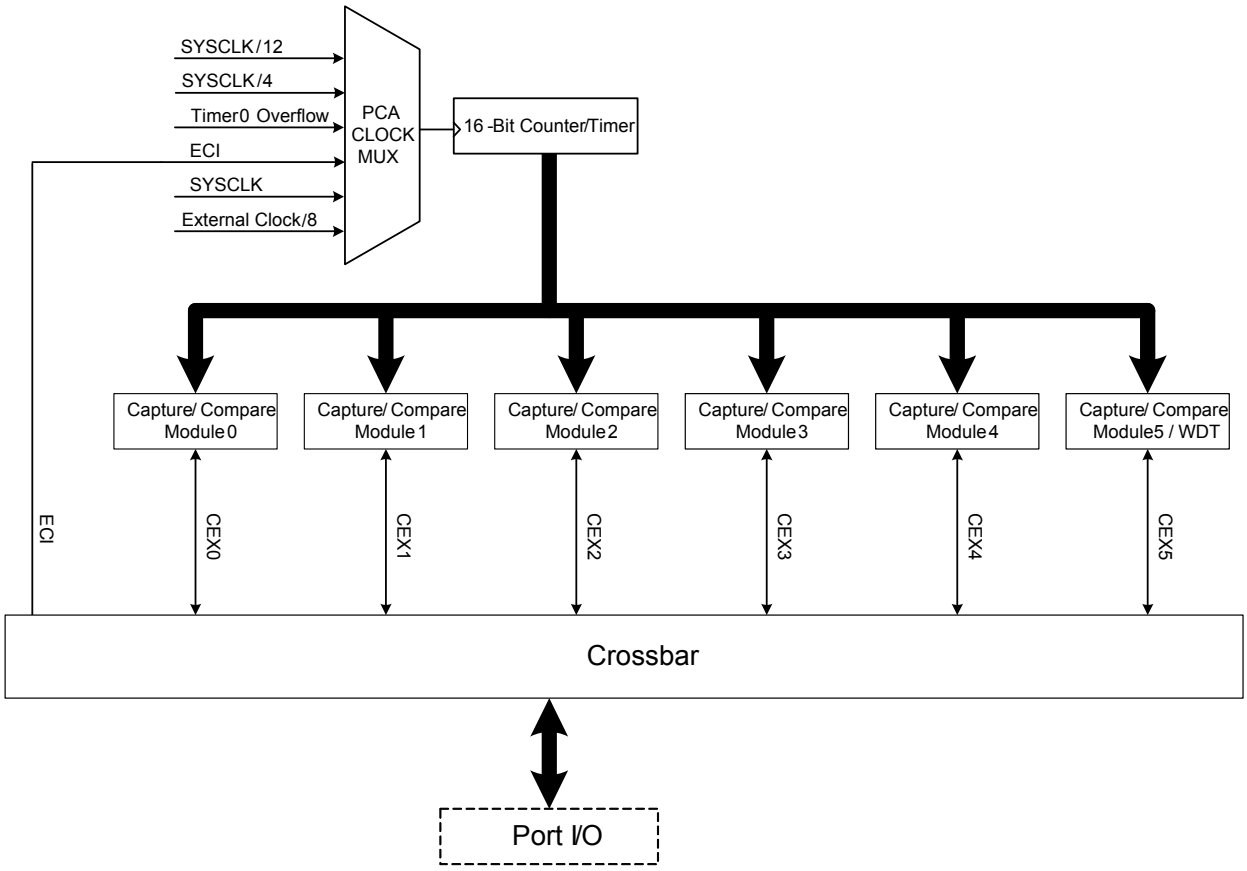



**Figure 1.10. PCA Block Diagram**

## 1.6. 10-bit SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode

Si1000/1/2/3/4/5 devices have a 300 ksps, 10-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and programmable window detector. ADC0 also has an autonomous low power Burst Mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. It also has a 16-bit accumulator that can automatically average the ADC results, providing an effective 11, 12, or 13-bit ADC result without any additional CPU intervention.

The ADC can sample the voltage at any of the GPIO pins (with the exception of P2.7) and has an on-chip attenuator that allows it to measure voltages up to twice the voltage reference. Additional ADC inputs include an on-chip temperature sensor, the VDD_MCU supply voltage, the VBAT supply voltage, and the internal digital supply voltage.

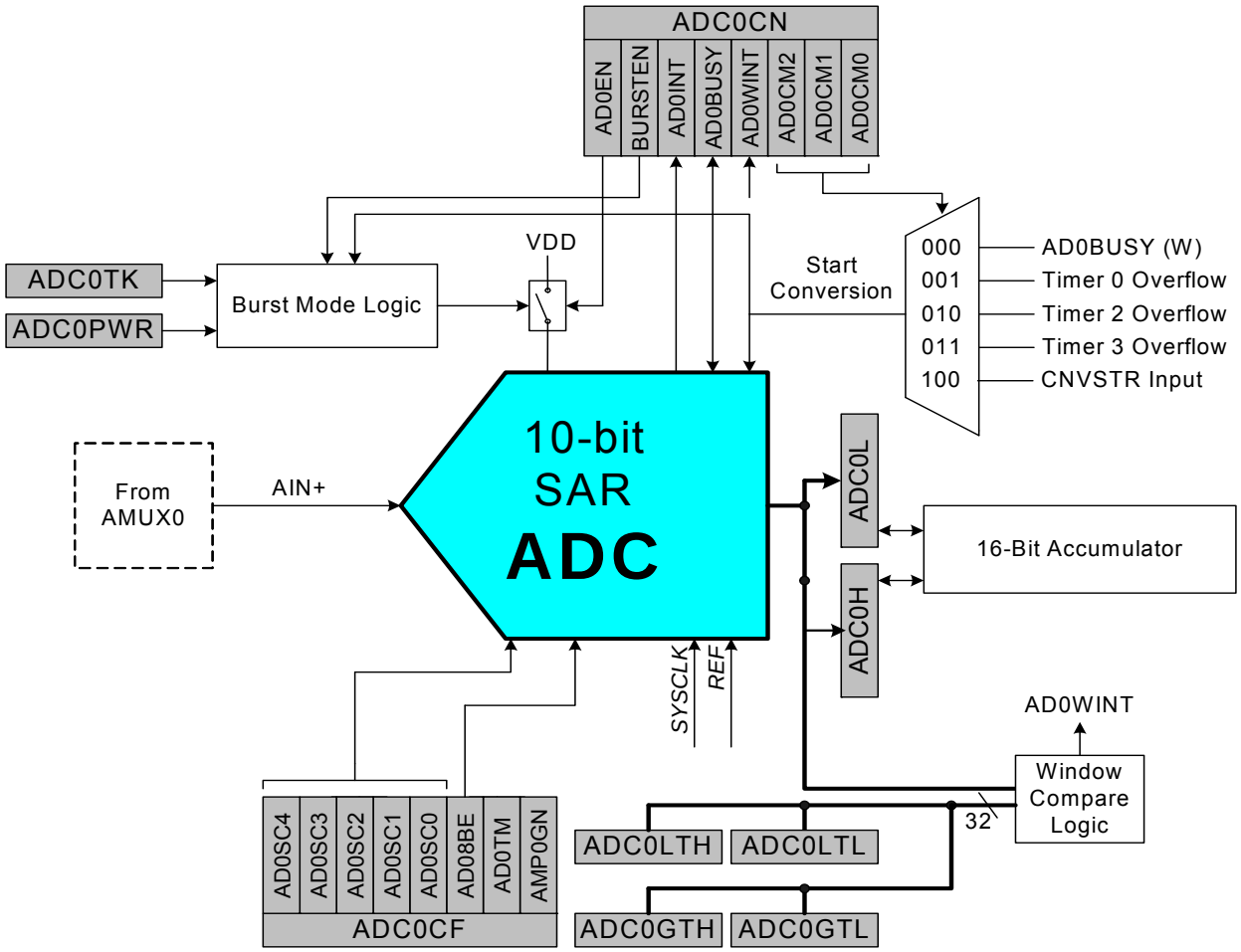



**Figure 1.11. ADC0 Functional Block Diagram**

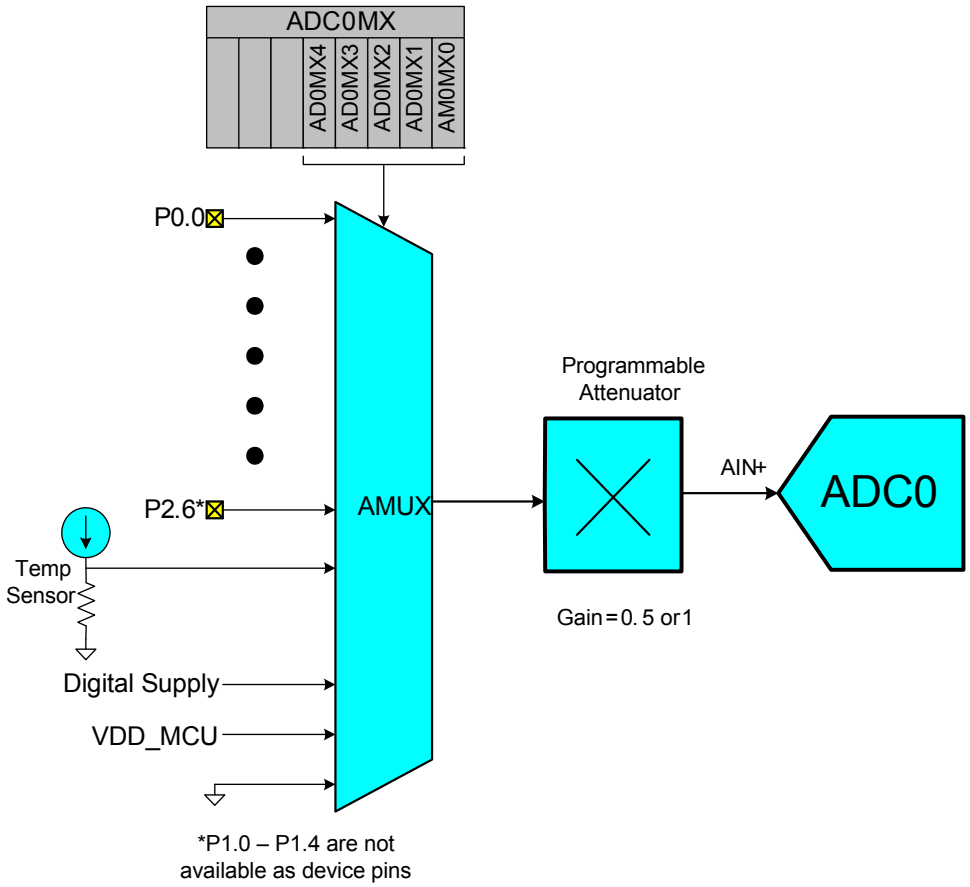



**Figure 1.12. ADC0 Multiplexer Block Diagram**

## 1.7. Programmable Current Reference (IREF0)

Si1000/1/2/3/4/5 devices include an on-chip programmable current reference (source or sink) with two output current settings: low power mode and high current mode. The maximum current output in low power mode is 63 µA (1 µA steps) and the maximum current output in high current mode is 504 µA (8 µA steps).

## 1.8. Comparators

Si1000/1/2/3/4/5 devices include two on-chip programmable voltage comparators: Comparator 0 (CPT0), which is shown in Figure 1.13, and Comparator 1 (CPT1), which is shown in Figure 1.14. The two comparators operate identically but may differ in their ability to be used as reset or wake-up sources. See Section “18. Reset Sources” on page 178 and Section “14. Power Management” on page 152 for details on reset sources and low power mode wake-up sources, respectively.

The comparators offer programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0, CP1), or an asynchronous “raw” output (CP0A, CP1A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the comparator to operate and generate an output when the device is in some low power modes.

The comparator inputs may be connected to Port I/O pins or to other internal signals. Port pins may also be used to directly sense capacitive touch switches. See Application Note “AN338: Capacitive Touch Sense Solution” for details on Capacitive Touch Switch sensing.

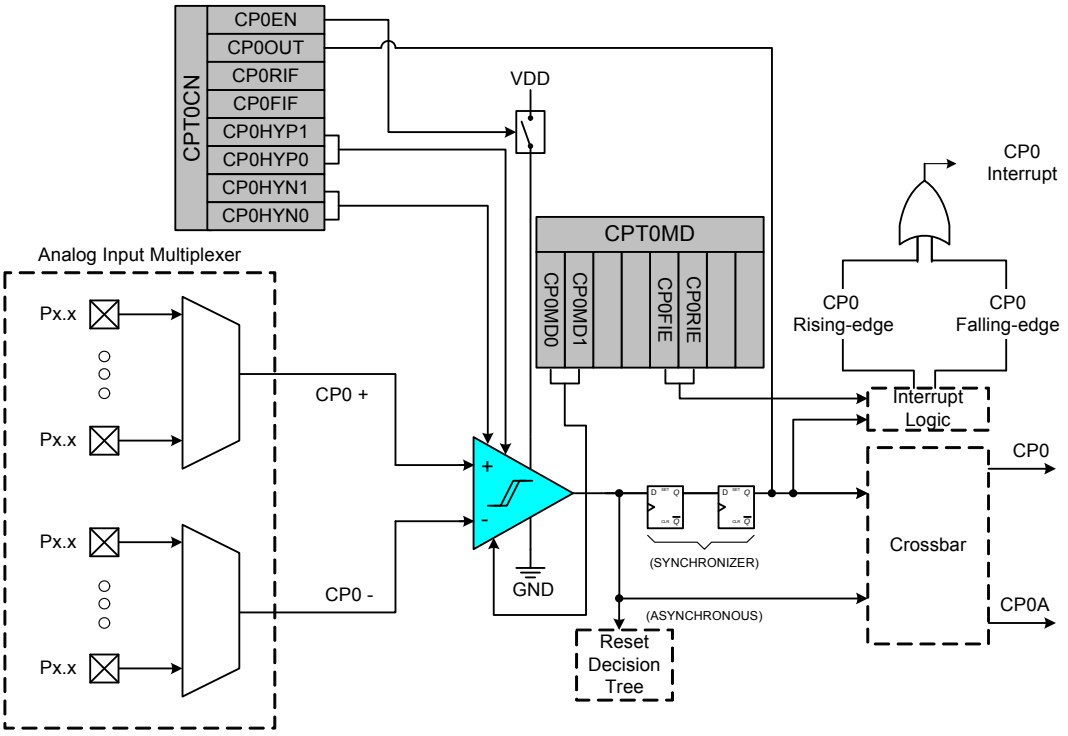



**Figure 1.13. Comparator 0 Functional Block Diagram**

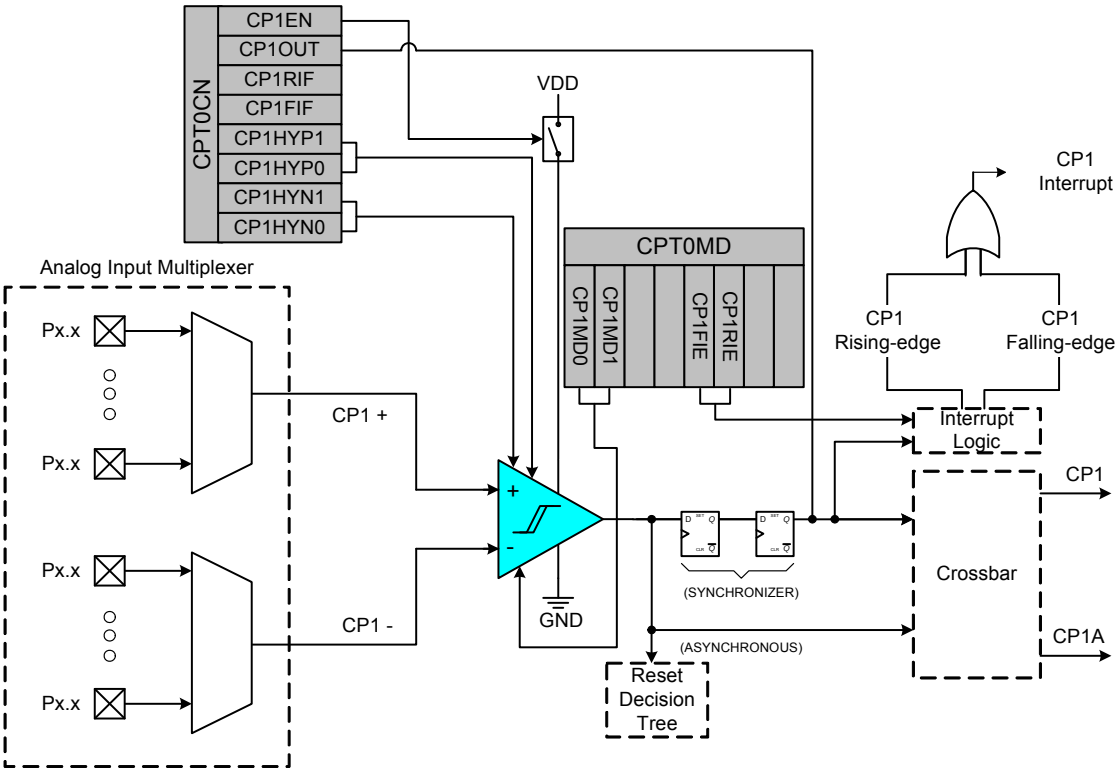



**Figure 1.14. Comparator 1 Functional Block Diagram**

## 2. Ordering Information

**Table 2.1. Product Selection Guide**

| Ordering Part Number | MIPS (Peak) | Flash Memory (kB) | RAM (bytes) | SmaRTClock Real Time Clock | SMBus/$I^2C$ | UART | Enhanced SPI (available for external communication) | Timers (16-bit) | Programmable Counter Array | Digital Port I/Os (includes EZRadioPRO GPIOs) | 10-bit 300ksps ADC | Internal Voltage Reference | Temperature Sensor | Maximum Transmit Power | Minimum Operating Voltage (Volts) | Lead-free (RoHS Compliant) | Package |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Si1000-E-GM2 | 25 | 64 | 4352 | P | 1 | 1 | 1 | 4 | P | 22 | P | P | P | +20 dBm | 1.8 | P | LGA-42 |
| Si1001-E-GM2 | 25 | 32 | 4352 | P | 1 | 1 | 1 | 4 | P | 22 | P | P | P | +20 dBm | 1.8 | P | LGA-42 |
| Si1002-E-GM2 | 25 | 64 | 4352 | P | 1 | 1 | 1 | 4 | P | 22 | P | P | P | +13 dBm | 1.8 | P | LGA-42 |
| Si1003-E-GM2 | 25 | 32 | 4352 | P | 1 | 1 | 1 | 4 | P | 22 | P | P | P | +13 dBm | 1.8 | P | LGA-42 |
| Si1004-E-GM2 | 25 | 64 | 4352 | P | 1 | 1 | 1 | 4 | P | 19 | P | P | P | +13 dBm | 0.9 | P | LGA-42 |
| Si1005-E-GM2 | 25 | 32 | 4352 | P | 1 | 1 | 1 | 4 | P | 19 | P | P | P | +13 dBm | 0.9 | P | LGA-42 |

# 3. Pinout and Package Definitions

**Table 3.1. Pin Definitions for the Si1000/1/2/3/4/5**

| Name | Pin Number | | Type | Description |
|---|---|---|---|---|
| | Si1000/1 Si1002/3 | Si1004/5 | | |
| VDD_MCU | 38 | — | P In | Power Supply Voltage for the entire MCU except for the EZRadioPRO peripheral. Must be 1.8 to 3.6 V. |
| GND_MCU | 37 | — | G | Required Ground for the entire MCU except for the EZRadioPRO peripheral. |
| VBAT | — | 41 | P In | Battery Supply Voltage. Must be 0.9 to 1.8 V in single-cell battery mode and 1.8 to 3.6 V in dual-cell battery mode. |
| GND<br><br>VBAT- | — | 38 | P In<br><br>G | In dual-cell battery mode, this pin must be connected directly to ground.<br>In one-cell applications, this pin should be connected directly to the negative battery terminal, which is not connected to the ground plane. |
| DCEN | — | 40 | P In<br><br>G | DC-DC Enable Pin. In single-cell battery mode, this pin must be connected to VBAT through a 0.68 µH inductor.<br>In dual-cell battery mode, this pin must be connected directly to ground. |
| VDD_MCU /<br><br>DC+ | — | 39 | P In<br><br>P Out | Power Supply Voltage for the entire MCU except for the EZRadioPRO peripheral. Must be 1.8 to 3.6 V. This supply voltage is not required in low power sleep mode. This voltage must always be ≥ VBAT.<br>Positive output of the dc-dc converter. In single-cell battery mode, a 1uF ceramic capacitor is required between dc+ and dc–. This pin can supply power to external devices when operating in single-cell battery mode. |
| GND_MCU<br><br>DC– | — | 37 | G<br><br>G | In dual-cell battery mode, this pin must be connected directly to ground.<br>DC-DC converter return current path. In one-cell mode, this pin must be connected to the ground plane. |
| VDD_RF | 16 | 16 | P In | Power Supply Voltage for the analog portion of the EZRadioPRO peripheral. Must be 1.8 to 3.6 V. |
| VDD_DIG | 28 | 28 | P In | Power Supply Voltage for the digital portion of the EZRadioPRO peripheral. Must be 1.8 to 3.6 V. |
| VR_DIG | 27 | 27 | P Out | Regulated Output Voltage of the digital 1.7 V regulator for the EZRadioPRO peripheral. A 1 µF decoupling capacitor is required. |
| GND_RF | 23 | 23 | G | Required Ground for the digital and analog portions of the EZRadioPRO peripheral. |

**Table 3.1. Pin Definitions for the Si1000/1/2/3/4/5 (Continued)**

| Name | Pin Number | | Type | Description |
|---|---|---|---|---|
| | Si1000/1 Si1002/3 | Si1004/5 | | |
| $\overline{RST}$/ | 39 | 42 | D I/O | Device Reset. Open-drain output of internal POR or $V_{DD}$ monitor. An external source can initiate a system reset by driving this pin low for at least 15 µs. A 1–5 kΩ pullup to VDD_MCU is recommended. See Reset Sources section for a complete description. |
| C2CK | | | D I/O | Clock signal for the C2 Debug Interface. |
| P2.7/ | 40 | 1 | D I/O | Port 2.7. This pin can only be used as GPIO. The Crossbar cannot route signals to this pin and it cannot be configured as an analog input. See Port I/O section for a complete description. |
| C2D | | | D I/O | Bi-directional data signal for the C2 Debug Interface. |
| XTAL3 | 1 | 3 | A In | SmaRTClock Oscillator Crystal Input.<br>See Section 20 for a complete description. |
| XTAL4 | 42 | 2 | A Out | SmaRTClock Oscillator Crystal Output.<br>See Section 20 for a complete description. |
| P0.0 | 36 | 36 | D I/O or A In | Port 0.0. See Port I/O section for a complete description. |
| $V_{REF}$ | | | A In<br>A Out | External $V_{REF}$ Input.<br>Internal $V_{REF}$ Output. External $V_{REF}$ decoupling capacitors are recommended. See Voltage Reference section. |
| P0.1 | 35 | 35 | D I/O or A In | Port 0.1. See Port I/O Section for a complete description. |
| AGND | | | G | Optional Analog Ground. See VREF chapter. |
| P0.2 | 34 | 34 | D I/O or A In | Port 0.2. See Port I/O Section for a complete description. |
| XTAL1 | | | A In | External Clock Input. This pin is the external oscillator return for a crystal or resonator. See Oscillator section. |
| P0.3 | 33 | 33 | D I/O or A In | Port 0.3. See Port I/O Section for a complete description. |
| XTAL2 | | | A Out | External Clock Output. This pin is the excitation driver for an external crystal or resonator. |
| | | | D In | External Clock Input. This pin is the external clock input in external CMOS clock mode. |
| | | | A In | External Clock Input. This pin is the external clock input in capacitor or RC oscillator configurations.<br>See Oscillator section for complete details. |

**Table 3.1. Pin Definitions for the Si1000/1/2/3/4/5 (Continued)**

| Name | Pin Number | | Type | Description |
|---|---|---|---|---|
| | Si1000/1 Si1002/3 | Si1004/5 | | |
| P0.4 | 32 | 32 | D I/O or A In | Port 0.4. See Port I/O section for a complete description. |
| TX | | | D Out | UART TX Pin. See Port I/O section. |
| P0.5 | 31 | 31 | D I/O or A In | Port 0.5. See Port I/O section for a complete description. |
| RX | | | D In | UART RX Pin. See Port I/O section. |
| P0.6 | 30 | 30 | D I/O or A In | Port 0.6. See Port I/O section for a complete description. |
| CNVSTR | | | D In | External Convert Start Input for ADC0. See ADC0 section for a complete description. |
| P0.7 | 29 | 29 | D I/O or A In | Port 0.7. See Port I/O section for a complete description. |
| IREF0 | | | A Out | IREF0 Output. See IREF section for complete description. |
| P1.5 | 10 | 10 | D I/O or A In | Port 1.5. See Port I/O section for a complete description. |
| P1.6 | 9 | 9 | D I/O or A In | Port 1.6. See Port I/O section for a complete description. |
| P1.7 | 8 | 8 | D I/O or A In | Port 1.7. See Port I/O section for a complete description. |
| P2.0 | 7 | 7 | D I/O or A In | Port 2.0. See Port I/O section for a complete description. |
| P2.1 | 6 | 6 | D I/O or A In | Port 2.1. See Port I/O section for a complete description. |
| P2.2 | 5 | 5 | D I/O or A In | Port 2.2. See Port I/O section for a complete description. |
| P2.3 | 4 | 4 | D I/O or A In | Port 2.3. See Port I/O section for a complete description. |
| P2.4 | 3 | — | D I/O or A In | Port 2.4. See Port I/O section for a complete description. |
| P2.5 | 2 | — | D I/O or A In | Port 2.5. See Port I/O section for a complete description. |
| P2.6 | 41 | — | D I/O or A In | Port 2.6. See Port I/O section for a complete description. |

**Table 3.1. Pin Definitions for the Si1000/1/2/3/4/5 (Continued)**

| Name | Pin Number | | Type | Description |
|---|---|---|---|---|
| | Si1000/1 Si1002/3 | Si1004/5 | | |
| GPIO_0 | 24 | 24 | D I/O or A I/O | General Purpose I/O controlled by the EZRadioPRO peripheral. May be configured through the EZRadioPRO registers to perform various functions including: Clock Output, FIFO status, POR, Wake-Up Timer, Low Battery Detect, TRSW, AntDiversity control, etc. See the EZRadioPRO GPIO Configuration Registers for more information. |
| GPIO_1 | 25 | 25 | D I/O or A I/O | |
| GPIO_2 | 26 | 26 | D I/O or A I/O | |
| nIRQ | 11 | 11 | D O | EZRadioPRO peripheral interrupt status pin. Will be set low to indicate a pending EZRadioPRO interrupt event. See the EZRadioPRO Control Logic Registers for more details. This pin is an open-drain output with a 220 kΩ internal pullup resistor. An external pull-up resistor is recommended. |
| XOUT | 12 | 12 | A O | EZRadioPRO peripheral crystal oscillator output. Connect to an external 30 MHz crystal or to an external clock source. If using an external clock source with no crystal, dc coupling with a nominal 0.8 VDC level is recommended with a minimum ac amplitude of 700 mVpp. Refer to AN417 for more details about using an external clock source. |
| XIN | 13 | 13 | A I | EZRadioPRO peripheral crystal oscillator input. Connect to an external 30 MHz crystal or leave floating if driving the XOUT pin with an external signal source. |
| NC | 14, 20, 22 | 14, 20, 22 | | No Connect. May be left floating or tied to power or ground. |
| SDN | 15 | 15 | D I | EZRadioPRO peripheral shutdown pin. When driven to logic HIGH, the EZRadioPRO peripheral will be completely shut down and the contents of the EZRadioPRO registers will be lost. This pin should be driven to logic LOW during all other times; this pin should never be left floating. |
| TX | 17 | 17 | A O | EZRadioPRO peripheral transmit RF output pin. The PA output is an open-drain connection so the L-C match must supply (1.8 to 3.6 VDC) to this pin. |
| RXp | 18 | 18 | A I | EZRadioPRO peripheral differential RF input pins of the LNA. See application schematic for example matching network. |
| RXn | 19 | 19 | A I | |
| ANT_A | 21 | 21 | D O | EZRadioPRO peripheral TR switch control signal. |

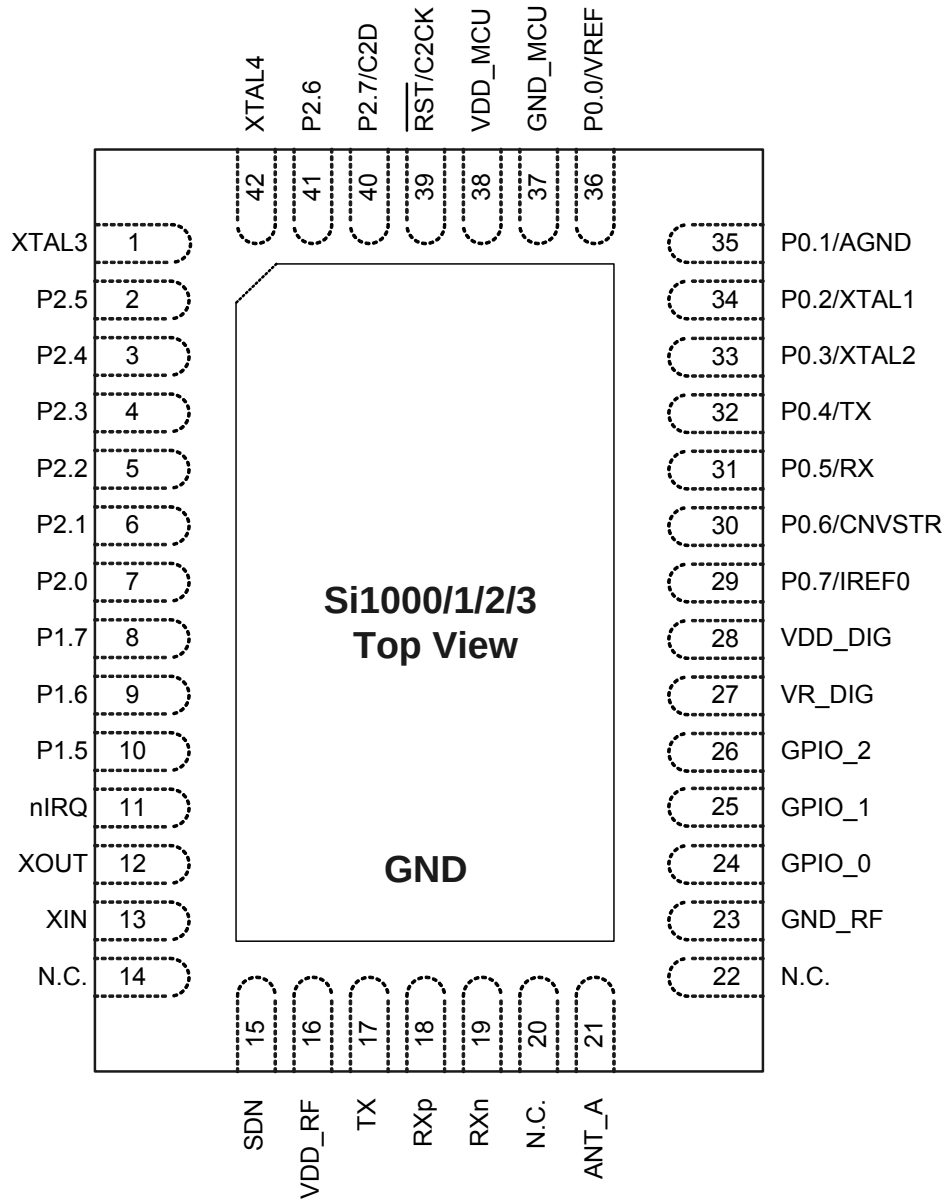



**Figure 3.1. Si100/1/2/3-E-GM2 Pinout Diagram (Top View)**

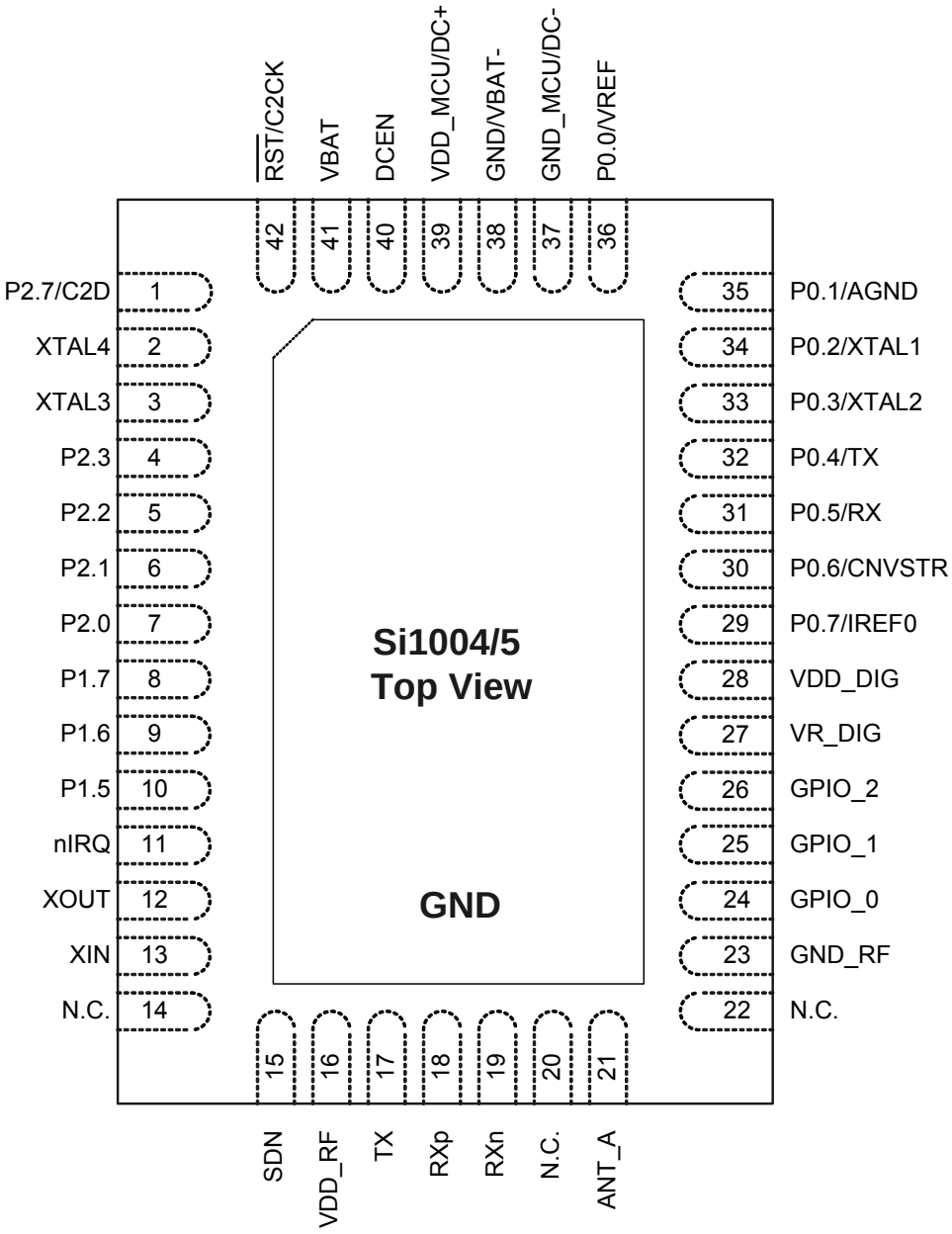



**Figure 3.2. Si1004/5-E-GM2 Pinout Diagram (Top View)**

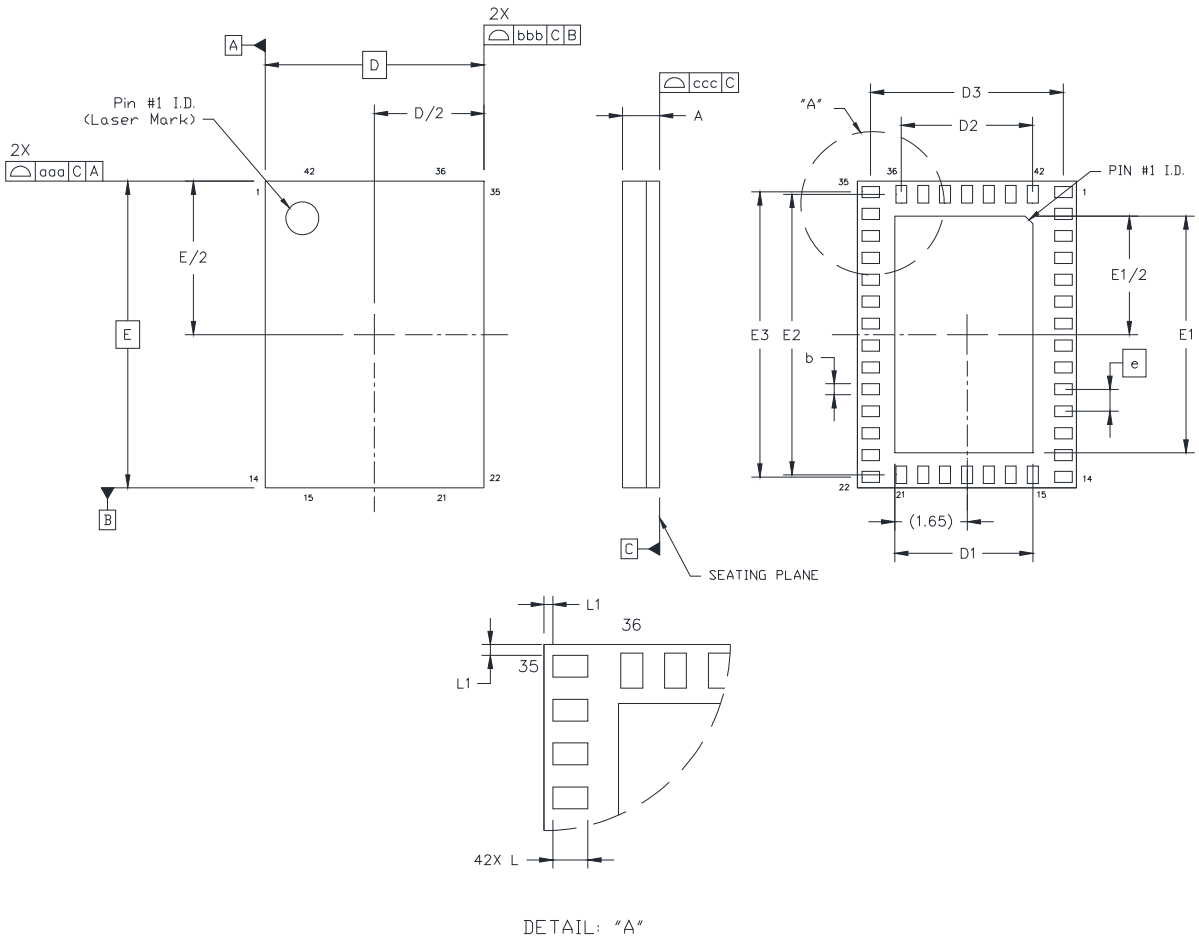



**Figure 3.3. LGA-42 Package Drawing (Si1000/1/2/3/4/5-E-GM2)**

**Table 3.2. LGA-42 Package Dimensions (Si1000/1/2/3/4/5-E-GM2)**

| Dimension | Min | Nom | Max |
|---|---|---|---|
| A | 0.85 | 0.90 | 0.95 |
| b | 0.20 | 0.25 | 0.30 |
| D | 5.00 BSC. | | |
| D1 | 3.15 | | |
| D2 | 3.00 | | |
| D3 | 4.40 | | |
| e | 0.50 BSC. | | |
| E | 7.00 BSC. | | |
| E1 | 5.40 | | |
| E2 | 6.40 | | |
| E3 | 6.50 | | |
| L | 0.35 | 0.40 | 0.45 |
| L1 | 0.05 | 0.10 | 0.15 |
| aaa | — | — | 0.10 |
| bbb | — | — | 0.10 |
| ccc | — | — | 0.08 |

**Notes:**
1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

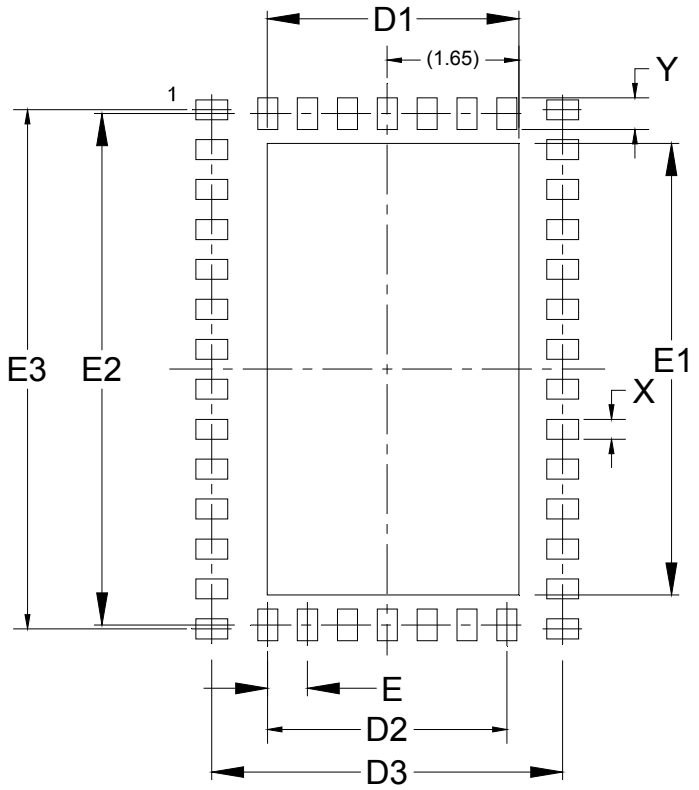



**Figure 3.4. LGA-42 PCB Land Pattern Dimensions (Si1000/1/2/3/4/5-E-GM2)**

**Table 3.3. LGA-42 PCB Land Pattern Dimensions (Si1000/1/2/3/4/5-E-GM2)**

| Dimension | mm |
|---|---|
| D1 | 3.15 |
| D2 | 3.00 |
| D3 | 4.40 |
| E | 0.50 |
| E1 | 5.45 |
| E2 | 6.40 |
| E3 | 6.50 |
| X | 0.25 |
| Y | 0.50 |

**Notes:**

**General**

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on the IPC-7351 guidelines.
3. All dimensions shown are at Maximum Material Condition (MMC). Least Material Condition (LMC) is calculated based on a Fabrication Allowance of 0.05 mm.

**PCB Design**

4. PCB design must ensure sufficient thermal relief for operation of the device.
5. Place vias in E-pad as shown in Figure 3.5.

**Solder Mask Design**

6. All metal pads are to be non-solder mask defined (NSMD). Clearance between the solder mask and the metal pad is to be 60 µm minimum, all the way around the pad.

**Stencil Design**

7. Place ground pad openings as shown in Figure 3.5.
8. The stencil thickness should be 0.125 mm (5 mils).
9. The ratio of stencil aperture to land pad size should be 1:1 for the perimeter pads.
10. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.

**Card Assembly**

11. A No-Clean, Type-3 solder paste is recommended.
12. The recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

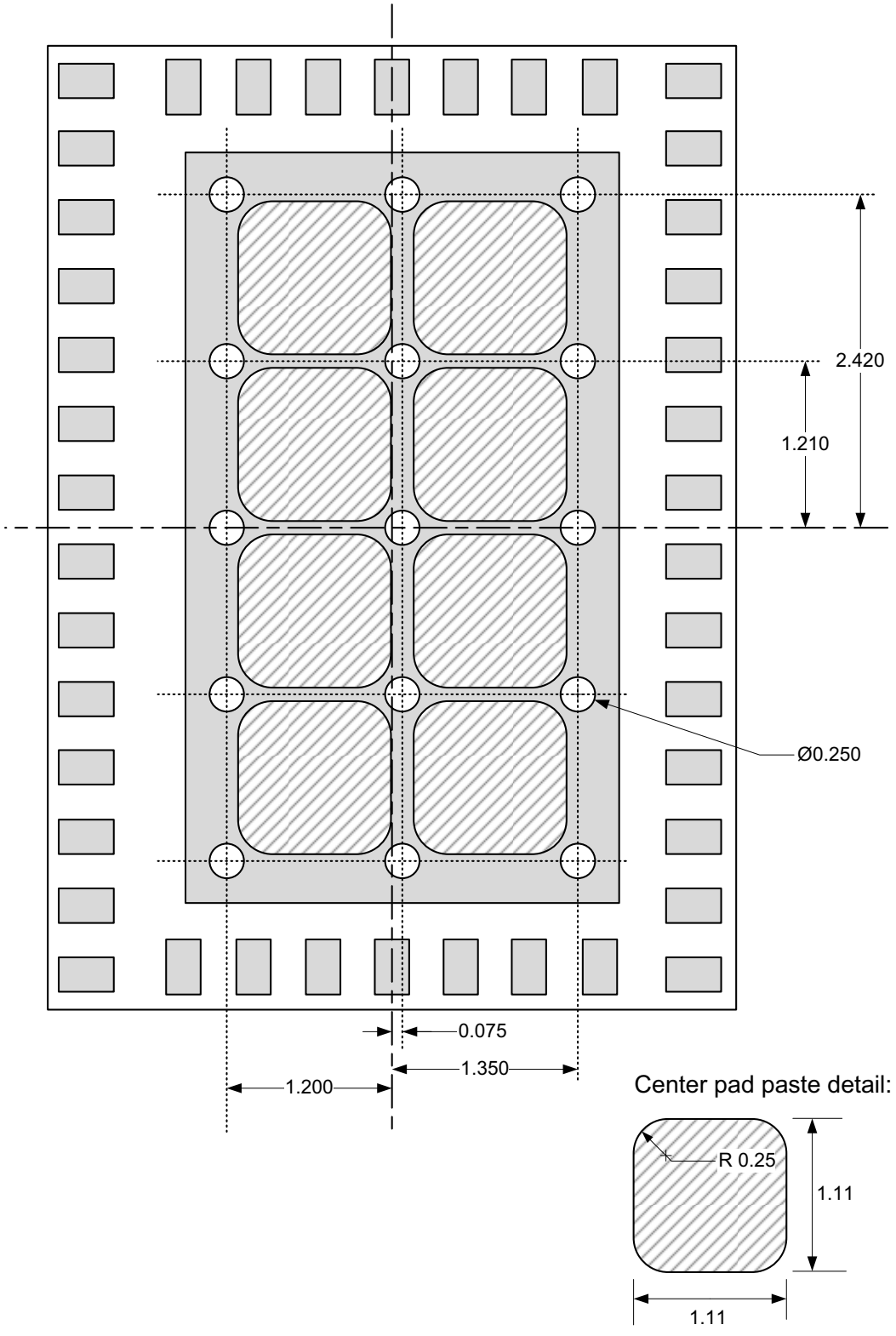



**Figure 3.5. LGA-42 PCB Stencil and Via Placement**

# 4. Electrical Characteristics

In sections 4.1 and 4.2, "$V_{DD}$" refers to the VDD_MCU supply voltage on Si1000/1/2/3 devices and to the VDD_MCU/DC+ supply voltage on Si1004/5 devices. The ADC, Comparator, and Port I/O specifications in these two sections do not apply to the EZRadioPRO peripheral.

In sections 4.3 and 4.4, "$V_{DD}$" refers to the VDD_RF and VDD_DIG Supply Voltage. All specifications in these sections pertain to the EZRadioPRO peripheral.

## 4.1. Absolute Maximum Specifications

**Table 4.1. Absolute Maximum Ratings**

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Ambient Temperature under Bias | | –55 | — | 125 | °C |
| Storage Temperature | | –65 | — | 150 | °C |
| Voltage on any Px.x I/O Pin or RST with Respect to GND | $V_{DD}$ > 2.2 V<br>$V_{DD}$ < 2.2 V | –0.3<br>–0.3 | —<br>— | 5.8<br>$V_{DD}$ + 3.6 | V |
| Voltage on VBAT with respect to GND | One-Cell Mode<br>Two-Cell Mode | –0.3<br>–0.3 | —<br>— | 2.0<br>4.0 | V |
| Voltage on VDD_MCU or VDD_MCU/DC+ with respect to GND | | –0.3 | — | 4.0 | V |
| Maximum Total Current through VBAT, DCEN, VDD_MCU/DC+ or GND | | — | — | 500 | mA |
| Maximum Output Current Sunk by RST or any Px.x Pin | | — | — | 100 | mA |
| Maximum Total Current through all Px.x Pins | | — | — | 200 | mA |
| DC-DC Converter Output Power | | — | — | 110 | mW |
| ESD (Human Body Model) | All pins except TX, RXp, and RXn | — | — | 2 | kV |
| | TX, RXp, and RXn | — | — | 1 | kV |
| ESD (Machine Model) | All pins except TX, RXp, and RXn | — | — | 150 | V |
| | TX, RXp, and RXn | — | — | 45 | V |

**Note:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the devices at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

## 4.2. MCU Electrical Characteristics

### Table 4.2. Global Electrical Characteristics

–40 to +85 °C, 25 MHz system clock unless otherwise specified. See "AN358: Optimizing Low Power Operation of the ‘F9xx" for details on how to achieve the supply current specifications listed in this table.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Battery Supply Voltage (VBAT) | One-Cell Mode<br>Two-Cell Mode | 0.9<br>1.8 | 1.2<br>2.4 | 1.8<br>3.6 | V |
| Supply Voltage (VDD_MCU/DC+) | One-Cell Mode<br>Two-Cell Mode | 1.8<br>1.8 | 1.9<br>2.4 | 3.6<br>3.6 | V |
| Minimum RAM Data Retention Voltage[1] | VDD (not in Sleep Mode)<br>VBAT (in Sleep Mode) | —<br>— | 1.4<br>0.3 | —<br>0.5 | V |
| SYSCLK (System Clock)[2] | | 0 | — | 25 | MHz |
| $T_{SYSH}$ (SYSCLK High Time) | | 18 | — | — | ns |
| $T_{SYSL}$ (SYSCLK Low Time) | | 18 | — | — | ns |
| Specified Operating Temperature Range | | –40 | — | +85 | °C |
| **Digital Supply Current—CPU Active (Normal Mode, fetching instructions from flash)** | | | | | |
| $I_{DD}$ [3, 4, 5, 6, 7, 8] | $V_{DD}$ = 1.8–3.6 V, F = 24.5 MHz (includes precision oscillator current) | — | 4.1 | 5.0 | mA |
| | $V_{DD}$ = 1.8–3.6 V, F = 20 MHz (includes low power oscillator current) | — | 3.5 | — | mA |
| | $V_{DD}$ = 1.8 V, F = 1 MHz | — | 295 | — | µA |
| | $V_{DD}$ = 3.6 V, F = 1 MHz (includes external oscillator/GPIO current) | — | 365 | — | µA |
| | $V_{DD}$ = 1.8–3.6 V, F = 32.768 kHz (includes SmaRTClock oscillator current) | — | 90 | — | µA |
| $I_{DD}$ Frequency Sensitivity[3, 5, 6, 7, 8] | $V_{DD}$ = 1.8–3.6 V, T = 25 °C, F < 10 MHz (flash oneshot active, see 13.6) | — | 226 | — | µA/MHz |
| | $V_{DD}$ = 1.8–3.6 V, T = 25 °C, F > 10 MHz (flash oneshot bypassed, see 13.6) | — | 120 | — | µA/MHz |

**Table 4.2. Global Electrical Characteristics (Continued)**

–40 to +85 °C, 25 MHz system clock unless otherwise specified. See "AN358: Optimizing Low Power Operation of the 'F9xx" for details on how to achieve the supply current specifications listed in this table.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **Digital Supply Current—CPU Inactive (Idle Mode, not fetching instructions from flash)** | | | | | |
| $I_{DD}$[4, 6,7,8] | $V_{DD}$ = 1.8–3.6 V, F = 24.5 MHz (includes precision oscillator current) | — | 2.5 | 3.0 | mA |
| | $V_{DD}$ = 1.8–3.6 V, F = 20 MHz (includes low power oscillator current) | — | 1.8 | — | mA |
| | $V_{DD}$ = 1.8 V, F = 1 MHz | — | 165 | — | µA |
| | $V_{DD}$ = 3.6 V, F = 1 MHz (includes external oscillator/GPIO current) | — | 235 | — | µA |
| | $V_{DD}$ = 1.8–3.6 V, F = 32.768 kHz (includes SmaRTClock oscillator current) | — | 84 | — | µA |
| $I_{DD}$ Frequency Sensitivity[1,6,8] | $V_{DD}$ = 1.8–3.6 V, T = 25 °C | — | 95 | — | µA/MHz |
| **Digital Supply Current—Suspend and Sleep Mode** | | | | | |
| Digital Supply Current[6,7,8] (Suspend Mode) | $V_{DD}$ = 1.8–3.6 V, two-cell mode | — | 77 | — | µA |
| Digital Supply Current[8] (Sleep Mode, SmaRTClock running) | 1.8 V, T = 25 °C | — | 0.61 | — | µA |
| | 3.0 V, T = 25 °C | — | 0.76 | — | |
| | 3.6 V, T = 25 °C | — | 0.87 | — | |
| | 1.8 V, T = 85 °C | — | 1.32 | — | |
| | 3.0 V, T = 85 °C | — | 1.62 | — | |
| | 3.6 V, T = 85 °C (includes SmaRTClock oscillator and brownout detector) | — | 1.93 | — | |
| Digital Supply Current[8] (Sleep Mode) | 1.8 V, T = 25 °C | — | 0.06 | — | µA |
| | 3.0 V, T = 25 °C | — | 0.09 | — | |
| | 3.6 V, T = 25 °C | — | 0.14 | — | |
| | 1.8 V, T = 85 °C | — | 0.77 | — | |
| | 3.0 V, T = 85 °C | — | 0.92 | — | |
| | 3.6 V, T = 85 °C (includes brownout detector) | — | 1.23 | — | |

**Table 4.2. Global Electrical Characteristics (Continued)**

–40 to +85 °C, 25 MHz system clock unless otherwise specified. See "AN358: Optimizing Low Power Operation of the ‘F9xx" for details on how to achieve the supply current specifications listed in this table.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|

**Notes:**

1. Based on device characterization data; Not production tested.
2. SYSCLK must be at least 32 kHz to enable debugging.
3. Digital Supply Current depends upon the particular code being executed. The values in this table are obtained with the CPU executing an “sjmp $” loop, which is the compiled form of a while(1) loop in C. One iteration requires 3 CPU clock cycles, and the flash memory is read on each cycle. The supply current will vary slightly based on the physical location of the sjmp instruction and the number of flash address lines that toggle as a result. In the worst case, current can increase by up to 30% if the sjmp loop straddles a 128-byte flash address boundary (e.g., 0x007F to 0x0080). Real-world code with larger loops and longer linear sequences will have few transitions across the 128-byte address boundaries.
4. Includes oscillator and regulator supply current.
5. IDD can be estimated for frequencies ≤10 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range, then adding an offset of 90 µA. When using these numbers to estimate $I_{DD}$ for >10 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 3.0 V; F = 20 MHz, $I_{DD}$ = 4.1 mA – (25 MHz – 20 MHz) x 0.120 mA/MHz = 3.5 mA.
6. The Supply Voltage is the voltage at the VDD_MCU pin, typically 1.8 to 3.6 V (default = 1.9 V).
   Idle IDD can be estimated by taking the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 3.0 V; F = 5 MHz, Idle $I_{DD}$ = 2.5 mA – (25 MHz – 5 MHz) x 0.095 mA/MHz = 0.6 mA.
7. The supply current specifications in Table 4.2 are for two cell mode. The VBAT current in one-cell mode can be estimated using the following equation:

$$\text{VBAT Current (one-cell mode)} = \frac{\text{Supply Voltage} \times \text{Supply Current (two-cell mode)}}{\text{DC-DC Converter Efficiency} \times \text{VBAT Voltage}}$$

   The VBAT Voltage is the voltage at the VBAT pin, typically 0.9 to 1.8 V.
   The Supply Current (two-cell mode) is the data sheet specification for supply current.
   The Supply Voltage is the voltage at the VDD/DC+ pin, typically 1.8 to 3.3 V (default = 1.9 V).
   The DC-DC Converter Efficiency can be estimated using Figure 4.3–Figure 4.5.
8. The EZRadioPRO peripheral is placed in Shutdown mode.

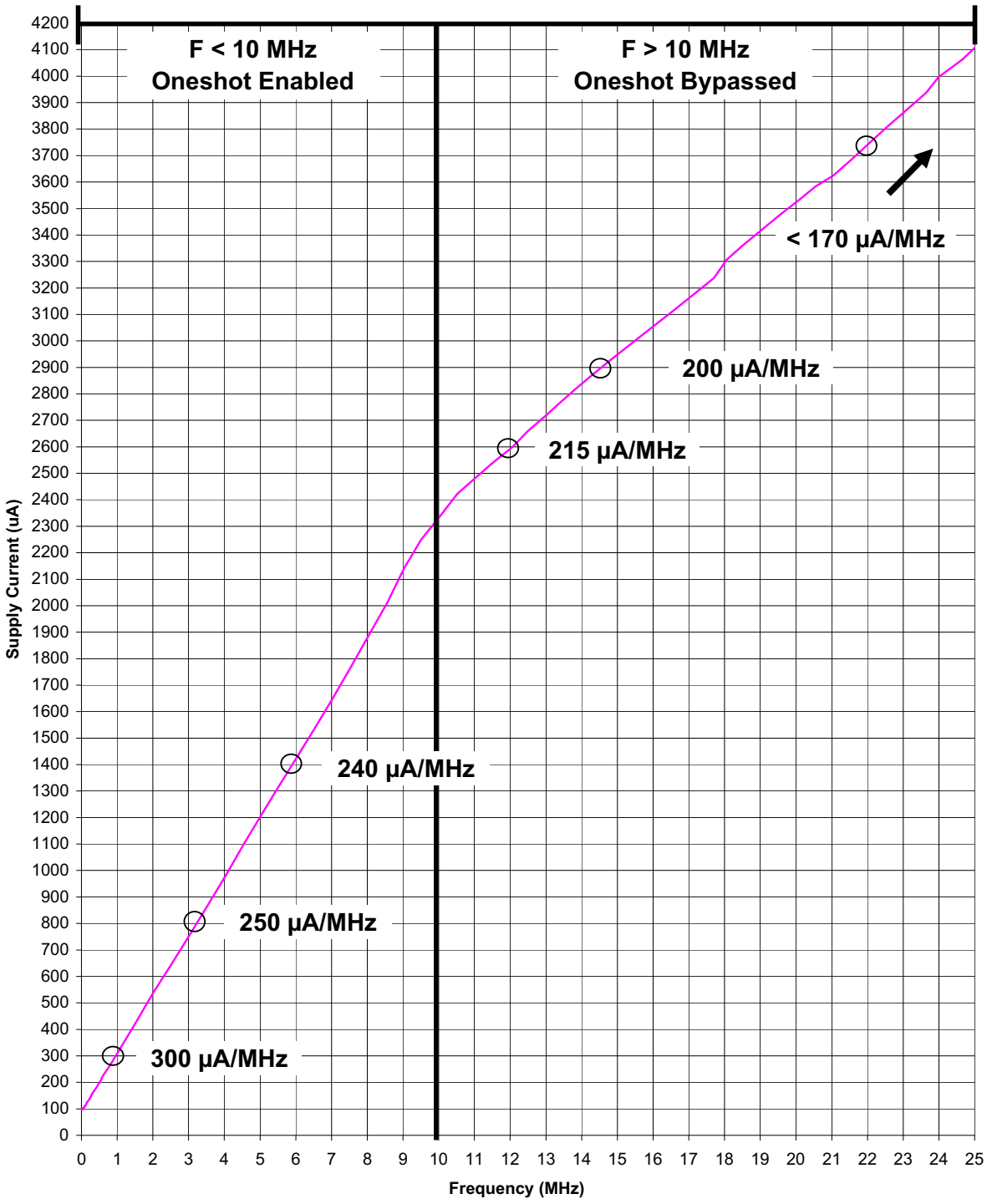



**Figure 4.1. Active Mode Current (External CMOS Clock)**

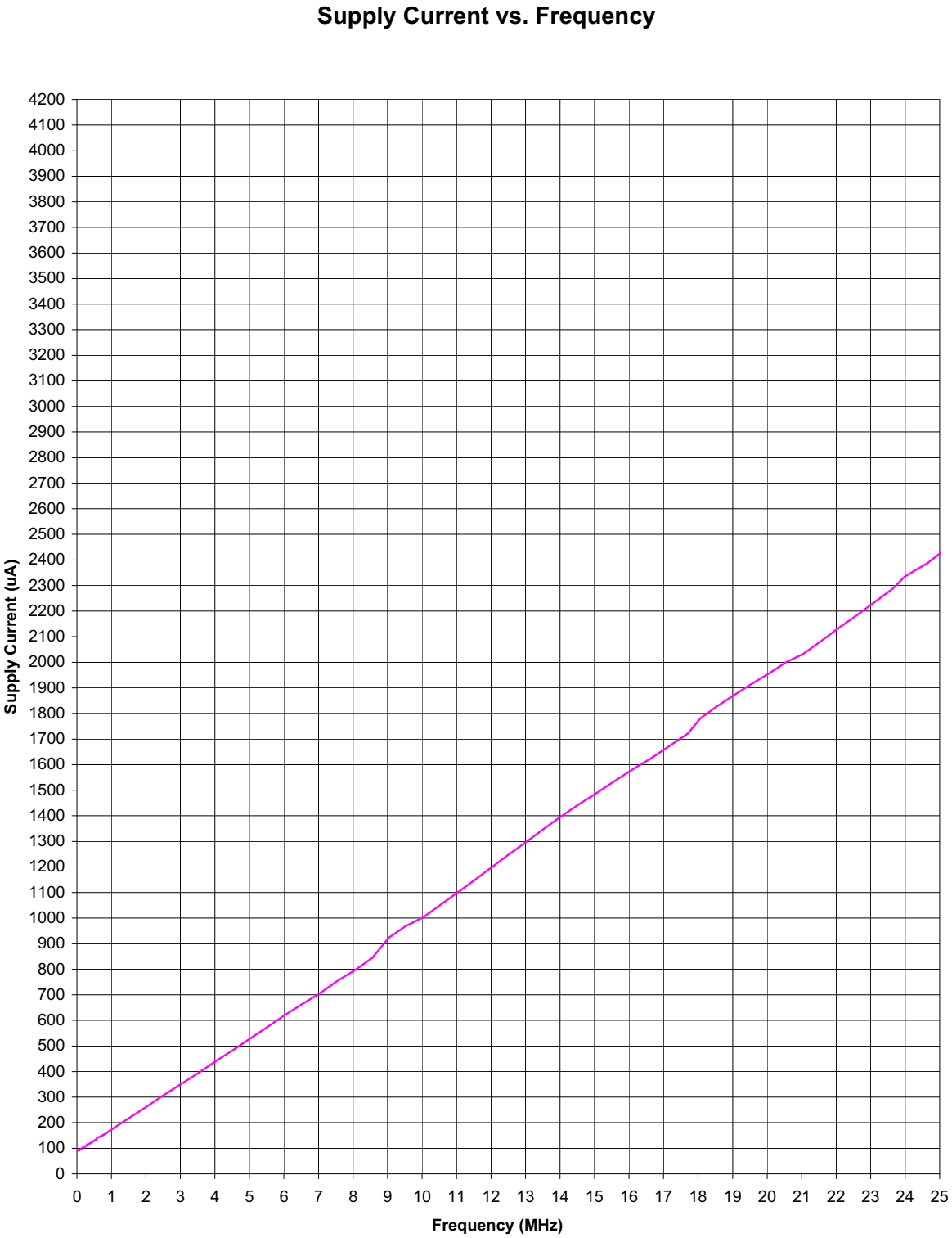



**Figure 4.2. Idle Mode Current (External CMOS Clock)**

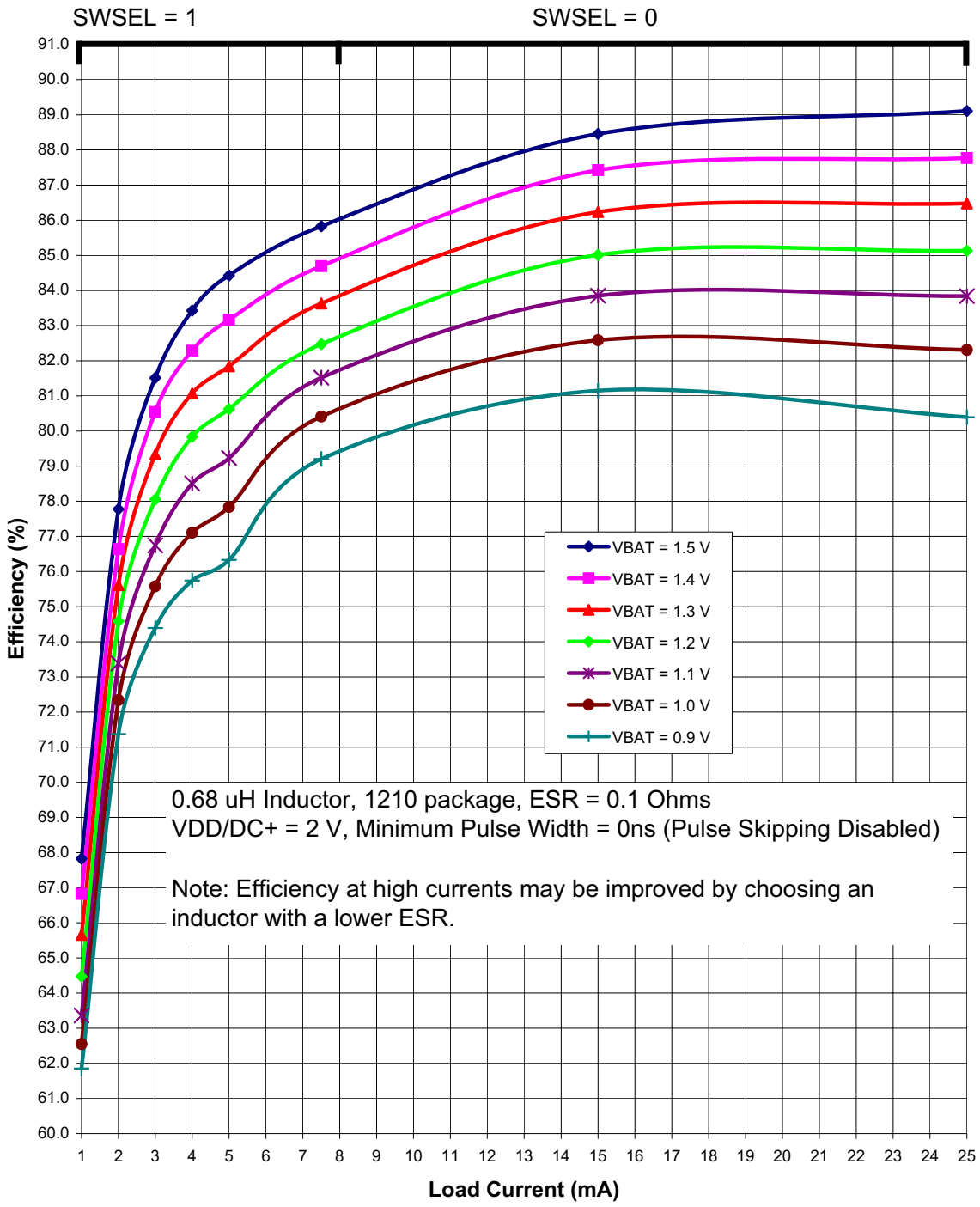



**Figure 4.3. Typical DC-DC Converter Efficiency (High Current, VDD/DC+ = 2 V**

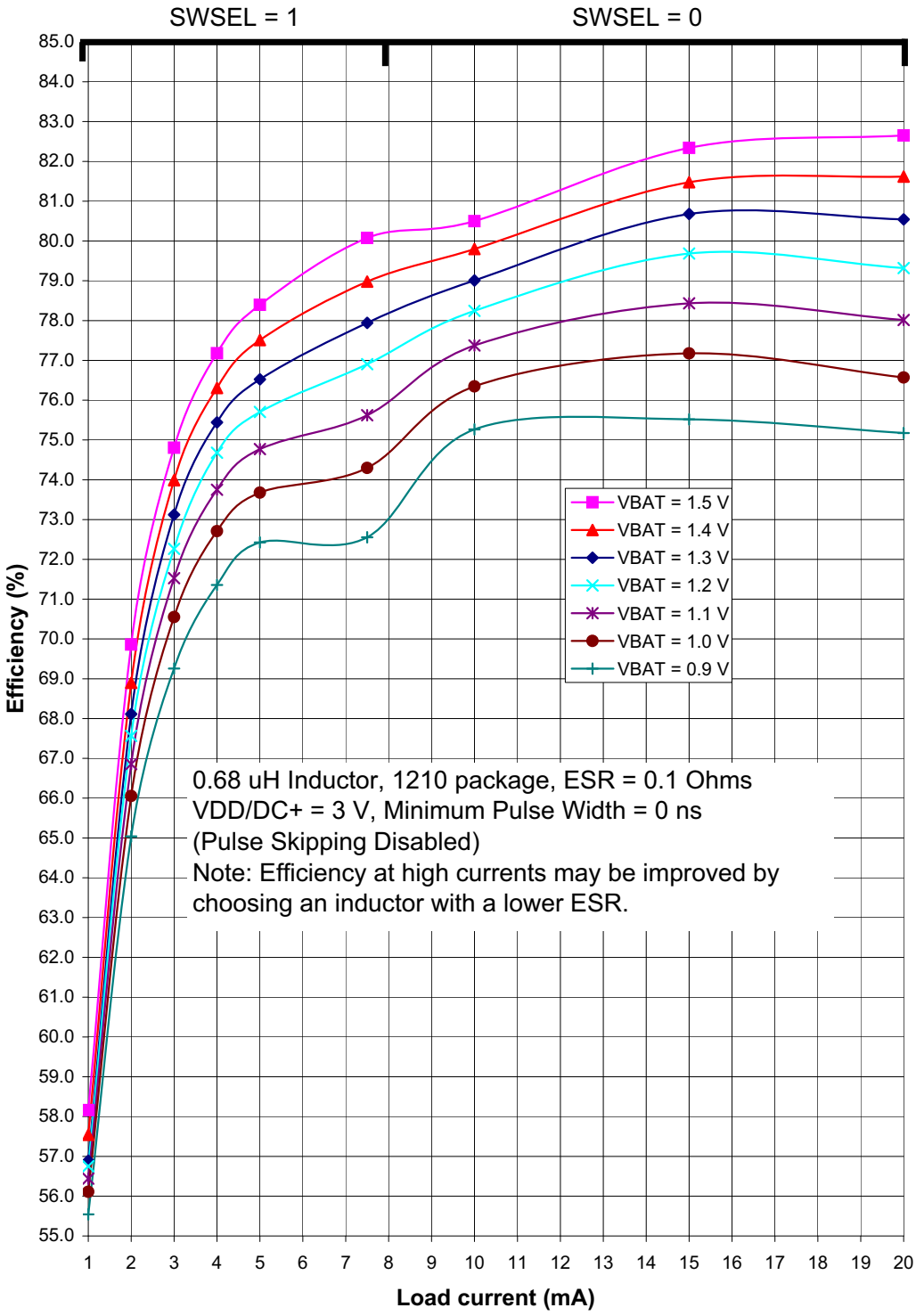



**Figure 4.4. Typical DC-DC Converter Efficiency (High Current, VDD/DC+ = 3 V)**

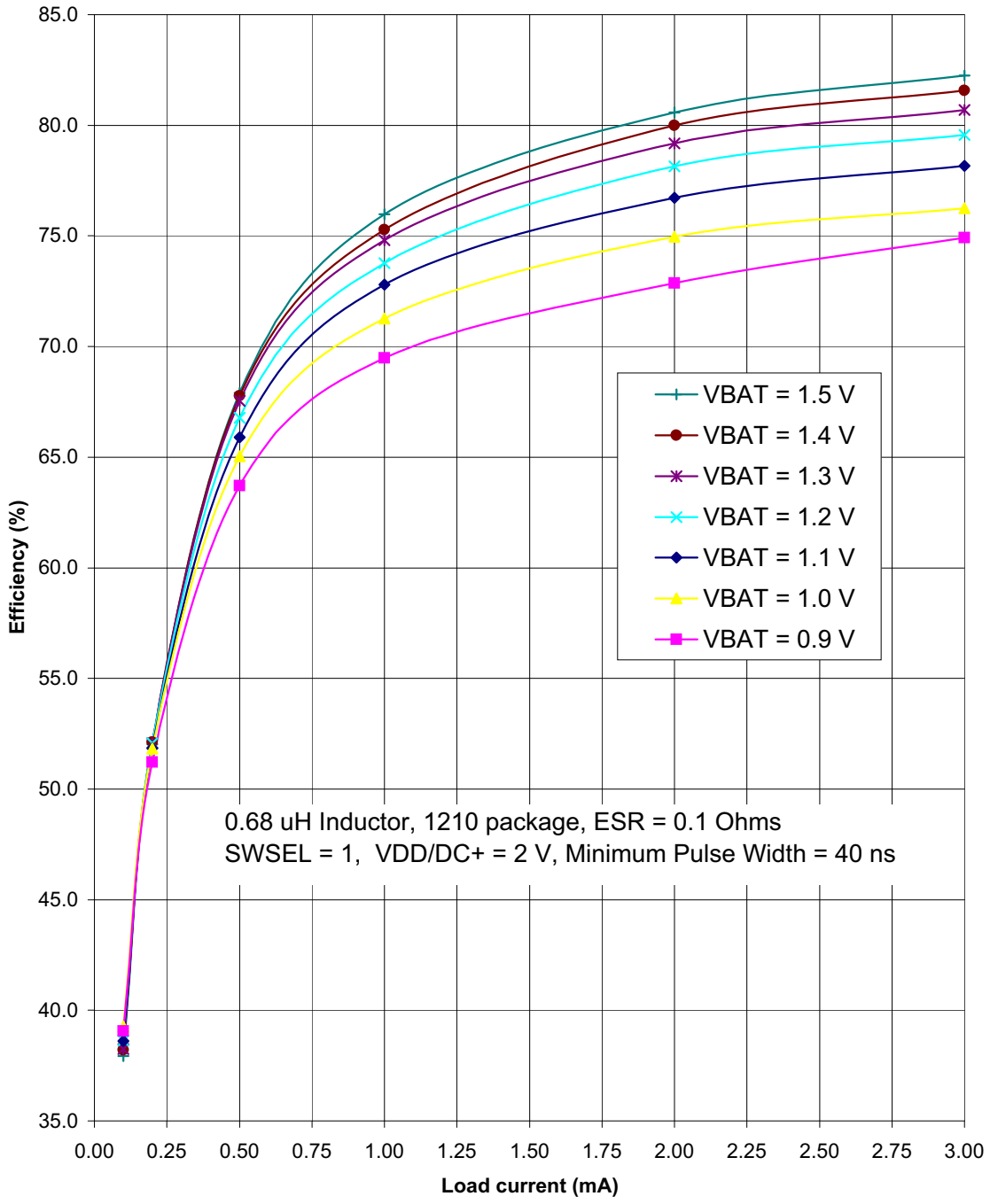



**Figure 4.5. Typical DC-DC Converter Efficiency (Low Current, VDD/DC+ = 2 V)**

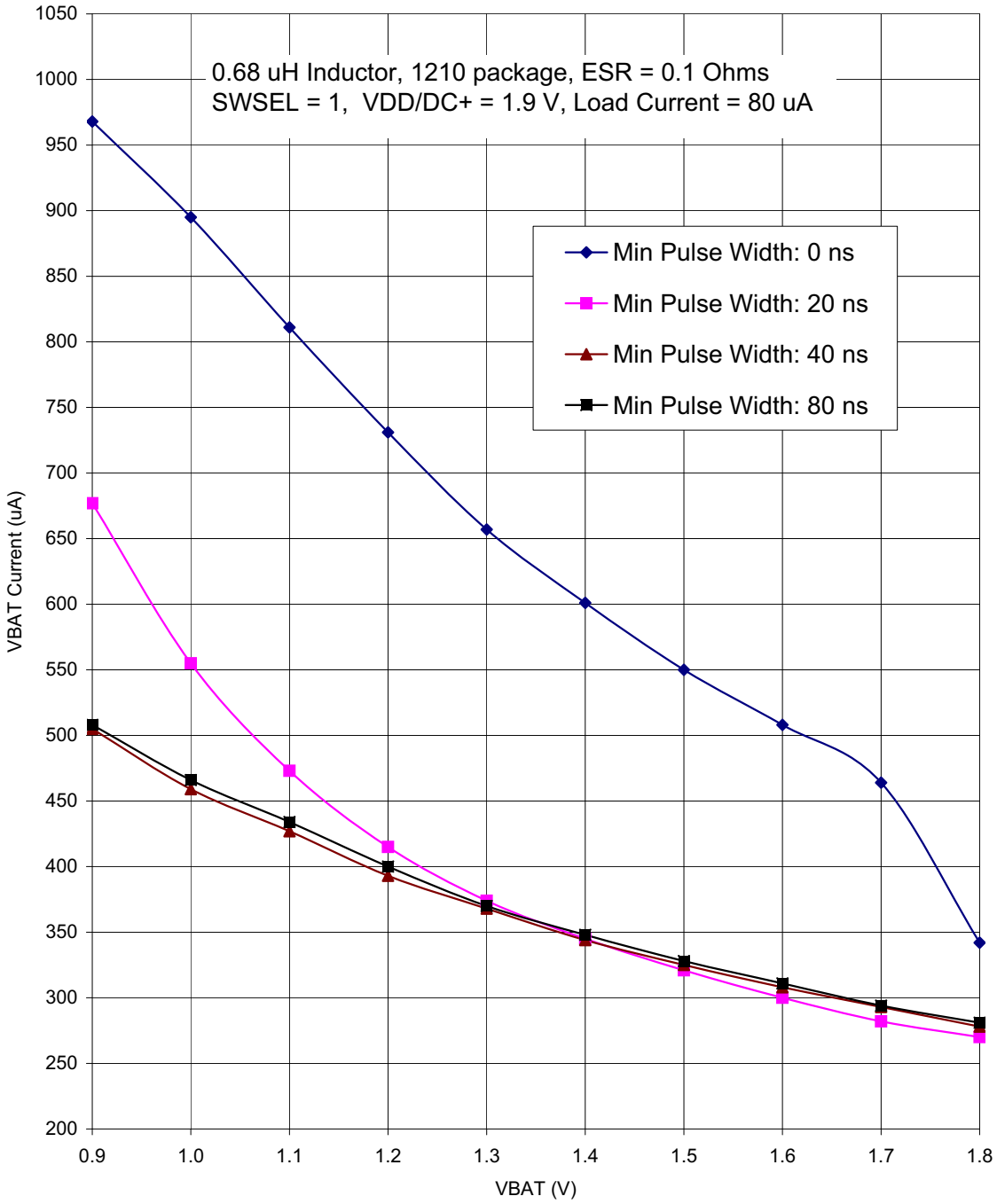



**Figure 4.6. Typical One-Cell Suspend Mode Current**

**Table 4.3. Port I/O DC Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameters | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Output High Voltage | High Drive Strength, PnDRV.n = 1 | | | | V |
| | IOH = –3 mA, Port I/O push-pull | $V_{DD}$ – 0.7 | — | — | |
| | IOH = –10 µA, Port I/O push-pull | $V_{DD}$ – 0.1 | — | — | |
| | IOH = –10 mA, Port I/O push-pull | | See Chart | | |
| | Low Drive Strength, PnDRV.n = 0 | | | | |
| | IOH = –1 mA, Port I/O push-pull | $V_{DD}$ – 0.7 | — | — | |
| | IOH = –10 µA, Port I/O push-pull | $V_{DD}$ – 0.1 | — | — | |
| | IOH = –3 mA, Port I/O push-pull | — | See Chart | — | |
| Output Low Voltage | High Drive Strength, PnDRV.n = 1 | | | | V |
| | $I_{OL}$ = 8.5 mA | — | — | 0.6 | |
| | $I_{OL}$ = 10 µA | — | — | 0.1 | |
| | $I_{OL}$ = 25 mA | — | See Chart | — | |
| | Low Drive Strength, PnDRV.n = 0 | | | | |
| | $I_{OL}$ = 1.4 mA | — | — | 0.6 | |
| | $I_{OL}$ = 10 µA | — | — | 0.1 | |
| | $I_{OL}$ = 4 mA | — | See Chart | — | |
| Input High Voltage | $V_{DD}$ = 2.0 to 3.6 V | $V_{DD}$ – 0.6 | — | — | V |
| | $V_{DD}$ = 0.9 to 2.0 V | 0.7 x VDD | — | — | V |
| Input Low Voltage | $V_{DD}$ = 2.0 to 3.6 V | — | — | 0.6 | V |
| | $V_{DD}$ = 0.9 to 2.0 V | — | — | 0.3 x VDD | V |
| Input Leakage Current | Weak Pullup On, $V_{IN}$ = 0 V, $V_{DD}$ = 1.8 V | — | 4 | — | µA |
| | Weak Pullup On, Vin = 0 V, $V_{DD}$ = 3.6 V | — | 20 | 30 | |

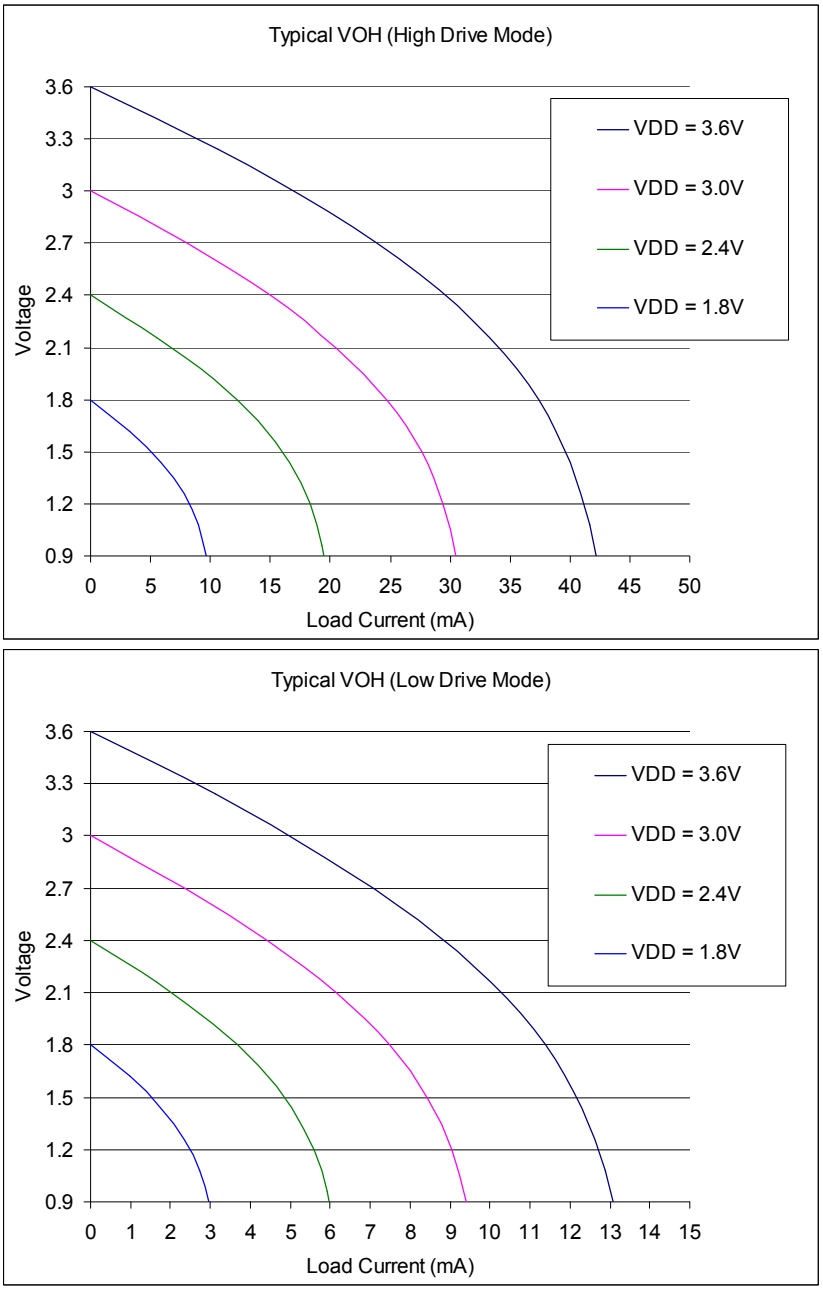



**Figure 4.7. Typical VOH Curves, 1.8–3.6 V**

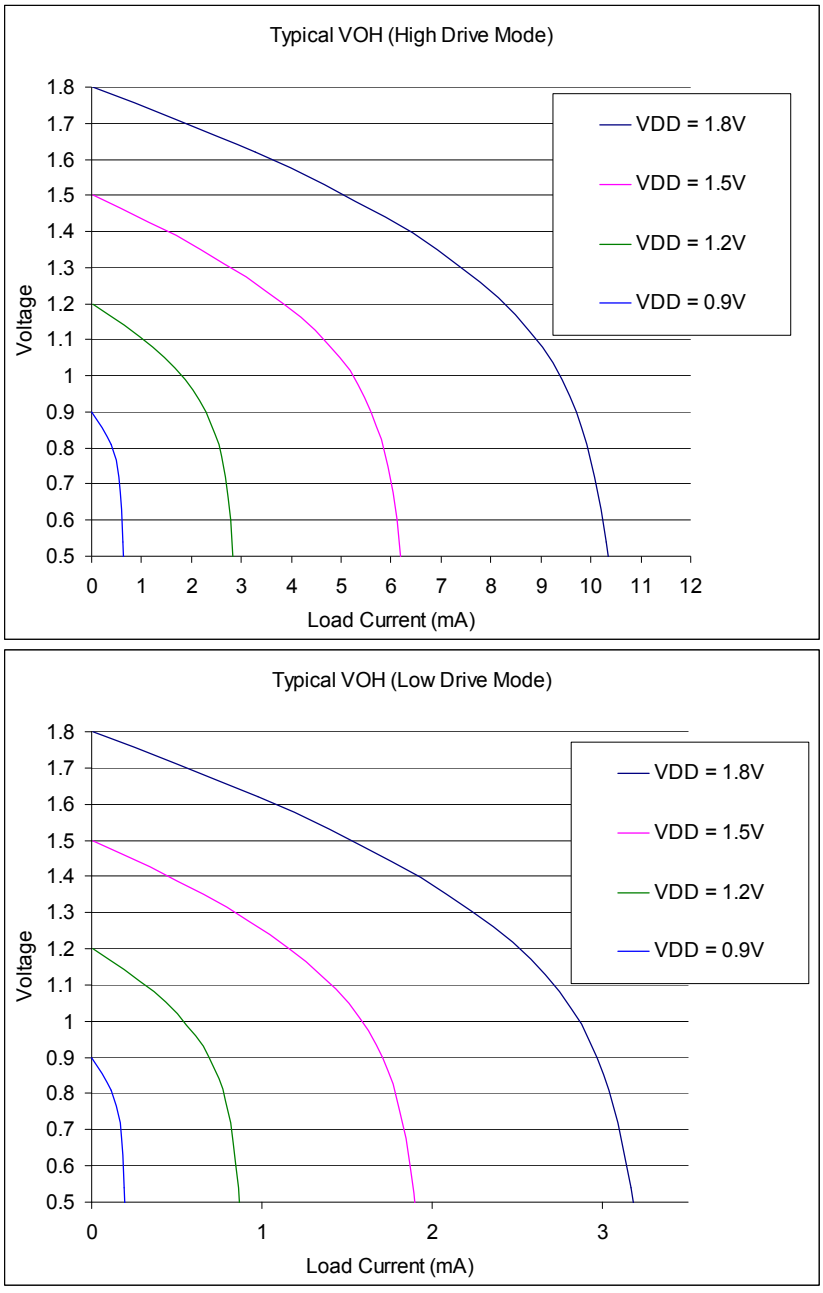



**Figure 4.8. Typical VOH Curves, 0.9–1.8 V**

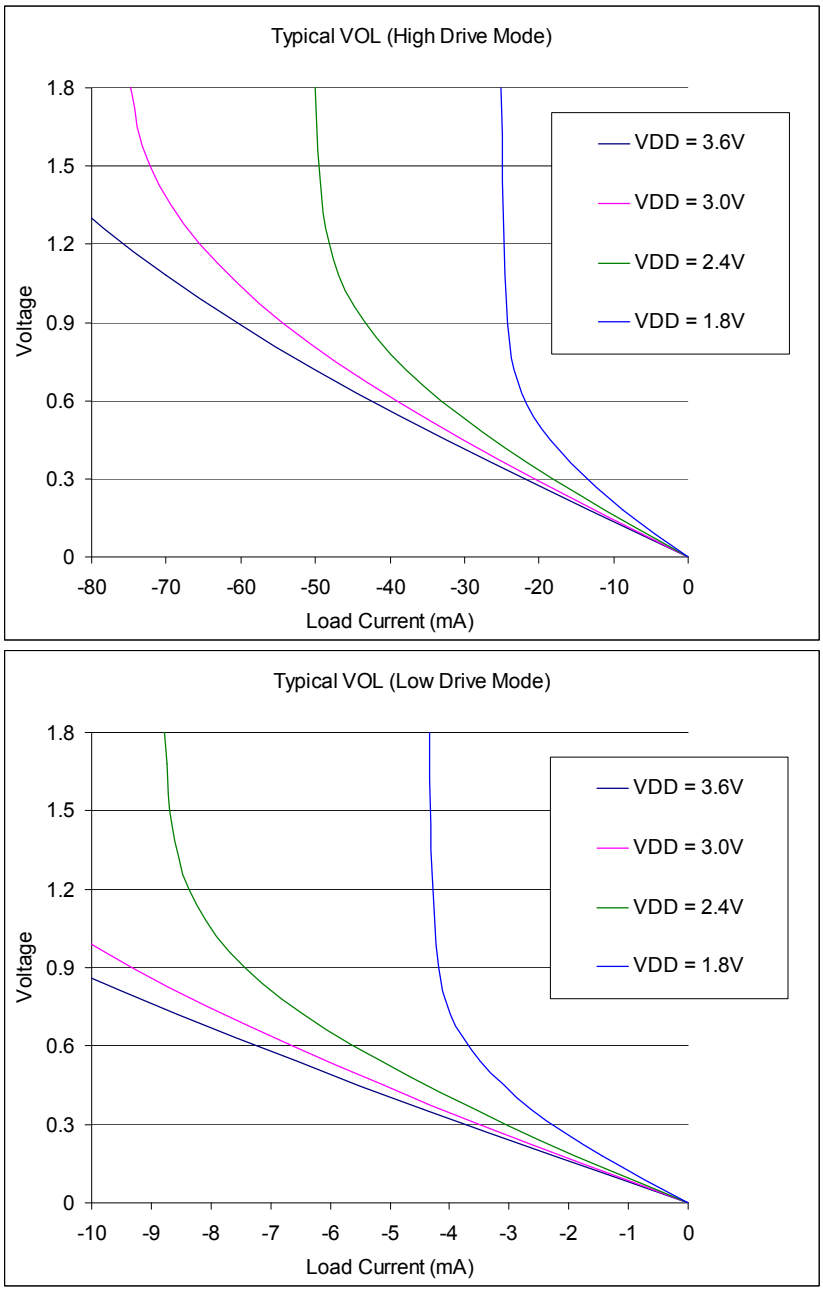



**Figure 4.9. Typical VOL Curves, 1.8–3.6 V**

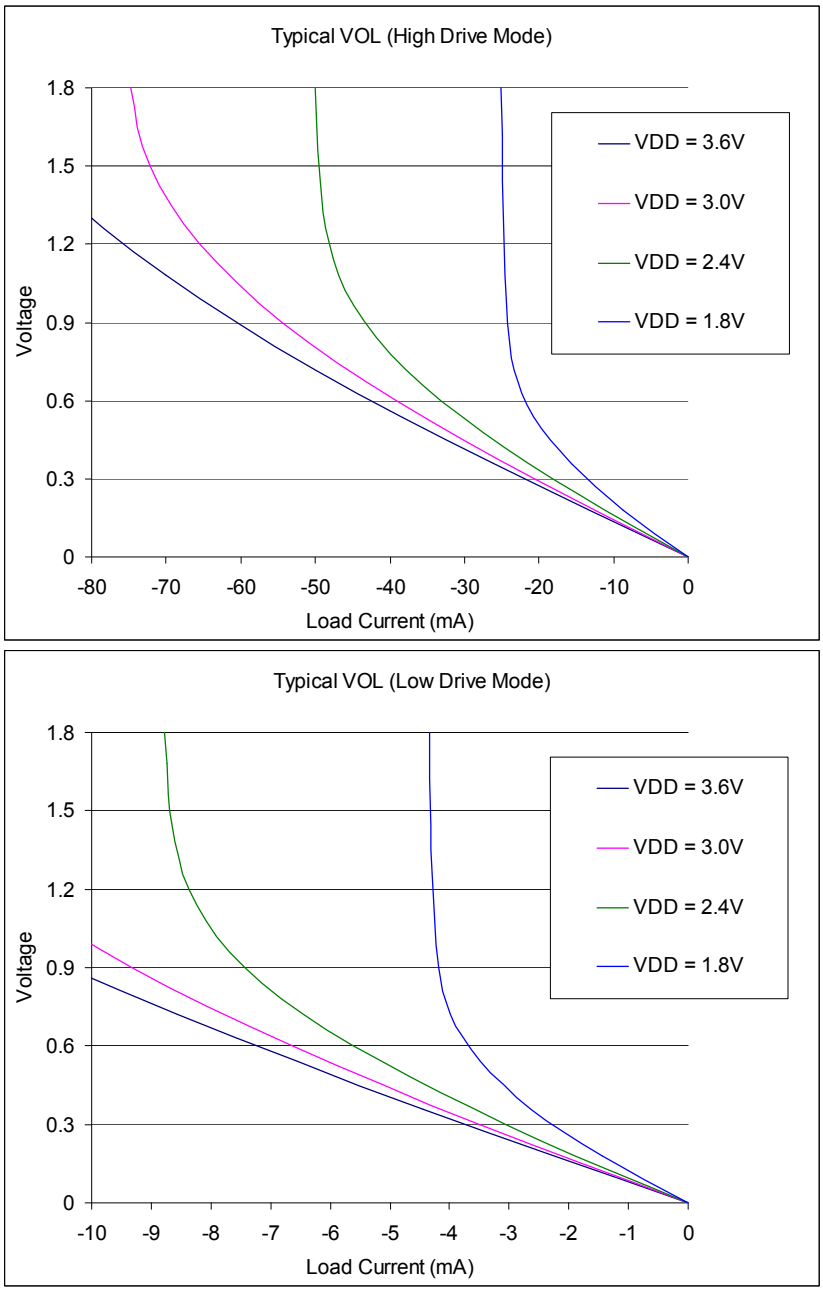



**Figure 4.10. Typical VOL Curves, 1.8–3.6 V**

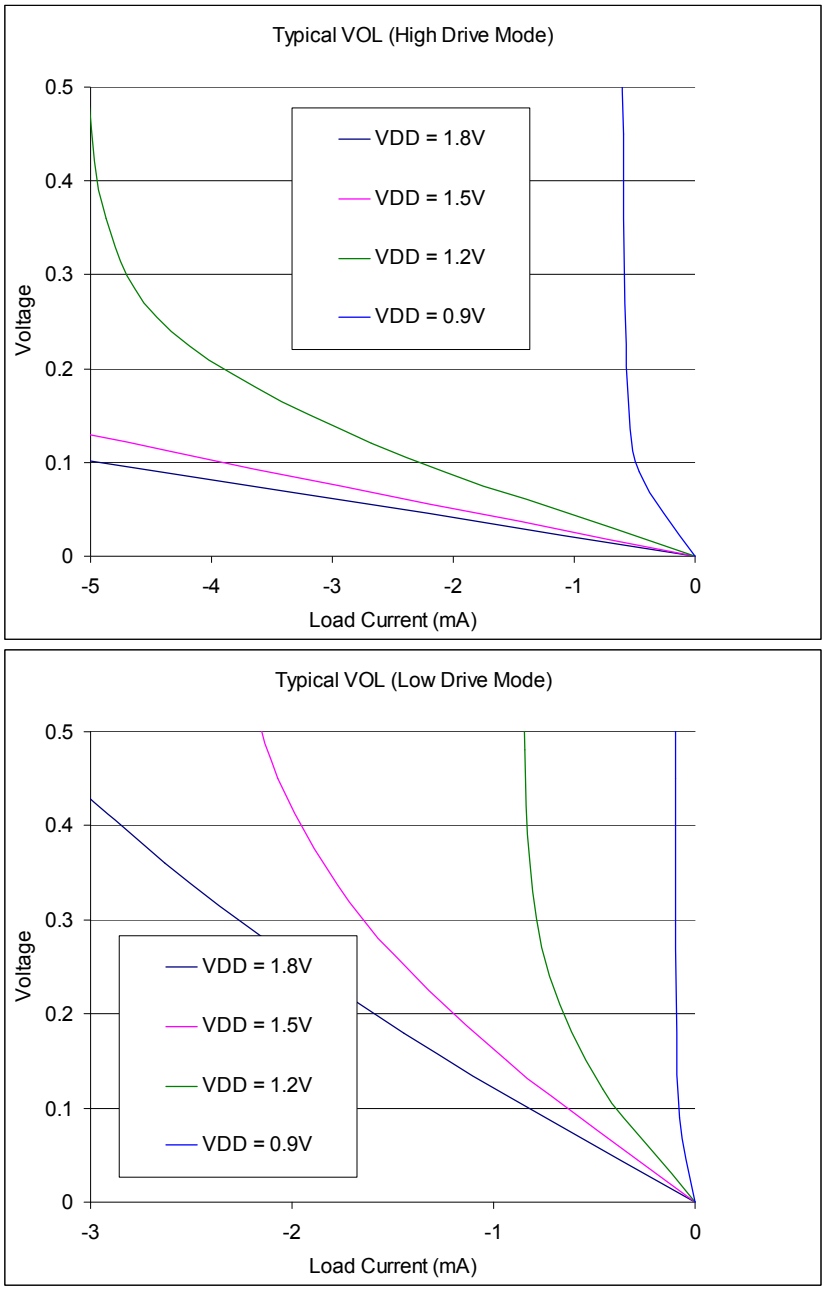



**Figure 4.11. Typical VOL Curves, 0.9–1.8 V**

**Table 4.4. Reset Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $\overline{RST}$ Output Low Voltage | $I_{OL}$ = 1.4 mA, | — | — | 0.6 | V |
| $\overline{RST}$ Input High Voltage | $V_{DD}$ = 2.0 to 3.6 V | $V_{DD}$ – 0.6 | — | — | V |
| | $V_{DD}$ = 0.9 to 2.0 V | 0.7 x $V_{DD}$ | — | — | V |
| $\overline{RST}$ Input Low Voltage | $V_{DD}$ = 2.0 to 3.6 V | — | — | 0.6 | V |
| | $V_{DD}$ = 0.9 to 2.0 V | — | — | 0.3 x $V_{DD}$ | V |
| $\overline{RST}$ Input Pullup Current | $\overline{RST}$ = 0.0 V, VDD = 1.8 V<br>$\overline{RST}$ = 0.0 V, VDD = 3.6 V | —<br>— | 4<br>20 | —<br>30 | µA |
| VDD_MCU Monitor Threshold ($V_{RST}$) | Early Warning<br>Reset Trigger<br>(all power modes except Sleep) | 1.8<br>1.7 | 1.85<br>1.75 | 1.9<br>1.8 | V |
| $V_{DD}$ Ramp Time for Power On | One-cell Mode: VBAT Ramp 0–0.9 V<br>Two-cell Mode: VBAT Ramp 0–1.8 V | — | — | 3 | ms |
| VDD Monitor Threshold ($V_{POR}$) | Initial Power-On ($V_{DD}$ Rising)<br>Brownout Condition ($V_{DD}$ Falling)<br>Recovery from Brownout ($V_{DD}$ Rising) | —<br>0.7<br>— | 0.75<br>0.8<br>0.95 | —<br>0.9<br>— | V |
| Missing Clock Detector Timeout | Time from last system clock rising edge to reset initiation | 100 | 650 | 1000 | µs |
| Minimum System Clock w/ Missing Clock Detector Enabled | System clock frequency which triggers a missing clock detector timeout | — | 7 | 10 | kHz |
| Reset Time Delay | Delay between release of any reset source and code execution at location 0x0000 | — | 10 | — | µs |
| Minimum $\overline{RST}$ Low Time to Generate a System Reset | | 15 | — | — | µs |
| $V_{DD}$ Monitor Turn-on Time | | — | 300 | — | ns |
| $V_{DD}$ Monitor Supply Current | | — | 7 | — | µA |

**Table 4.5. Power Management Electrical Specifications**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Idle Mode Wake-up Time | | 2 | — | 3 | SYSCLKs |
| Suspend Mode Wake-up Time | Low power oscillator | — | 400 | — | ns |
| | Precision oscillator | — | 1.3 | — | µs |
| Sleep Mode Wake-up Time | Two-cell mode | — | 2 | — | µs |
| | One-cell mode | — | 10 | — | µs |

**Table 4.6. Flash Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Flash Size | Si1000/2/4 | 65536* | — | — | bytes |
| | Si1001/3/5 | 32768 | — | — | bytes |
| Scratchpad Size | | 1024 | — | 1024 | bytes |
| Endurance | | 1k | 30k | — | Erase/Write Cycles |
| Erase Cycle Time | | 28 | 32 | 36 | ms |
| Write Cycle Time | | 57 | 64 | 71 | µs |
| **Note:** 1024 bytes at addresses 0xFC00 to 0xFFFF are reserved. | | | | | |

### Table 4.7. Internal Precision Oscillator Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V; $T_A$ = –40 to +85 °C unless otherwise specified; Using factory-calibrated settings.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Oscillator Frequency | –40 to +85 °C, $V_{DD}$ = 1.8–3.6 V | 24 | 24.5 | 25 | MHz |
| Oscillator Supply Current (from $V_{DD}$) | 25 °C; includes bias current of 90–100 µA | — | 300* | — | µA |
| ***Note:** Does not include clock divider or clock tree supply current. | | | | | |

### Table 4.8. Internal Low-Power Oscillator Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V; $T_A$ = –40 to +85 °C unless otherwise specified; Using factory-calibrated settings.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Oscillator Frequency | –40 to +85 °C, $V_{DD}$ = 1.8–3.6 V | 18 | 20 | 22 | MHz |
| Oscillator Supply Current (from $V_{DD}$) | 25 °C No separate bias current required. | — | 100* | — | µA |
| ***Note:** Does not include clock divider or clock tree supply current. | | | | | |

**Table 4.9. ADC0 Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6V V, VREF = 1.65 V (REFSL[1:0] = 11), –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **DC Accuracy** | | | | | |
| Resolution | | | 10 | | bits |
| Integral Nonlinearity | | — | ±0.5 | ±1 | LSB |
| Differential Nonlinearity | Guaranteed Monotonic | — | ±0.5 | ±1 | LSB |
| Offset Error | | — | ±<1 | ±2 | LSB |
| Full Scale Error | | — | ±1 | ±2.5 | LSB |
| **Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, 300 ksps)** | | | | | |
| Signal-to-Noise Plus Distortion | | 54 | 58 | — | dB |
| Signal-to-Distortion | | — | 73 | — | dB |
| Spurious-Free Dynamic Range | | — | 75 | — | dB |
| **Conversion Rate** | | | | | |
| SAR Conversion Clock | | — | — | 7.33 | MHz |
| Conversion Time in SAR Clocks | 10-bit Mode<br>8-bit Mode | 13<br>11 | —<br>— | —<br>— | clocks |
| Track/Hold Acquisition Time | | 1.5 | — | — | us |
| Throughput Rate | | — | — | 300 | ksps |
| **Analog Inputs** | | | | | |
| ADC Input Voltage Range | Single Ended (AIN+ – GND) | 0 | — | VREF | V |
| Absolute Pin Voltage with respect to GND | Single Ended | 0 | — | $V_{DD}$ | V |
| Sampling Capacitance | 1x Gain<br>0.5x Gain | — | 30<br>28 | — | pF |
| Input Multiplexer Impedance | | — | 5 | — | kΩ |
| **Power Specifications** | | | | | |
| Power Supply Current ($V_{DD}$ supplied to ADC0) | Conversion Mode (300 ksps)<br>Tracking Mode (0 ksps) | —<br>— | 800<br>680 | —<br>— | µA |
| Power Supply Rejection | Internal High Speed VREF<br>External VREF | —<br>— | 67<br>74 | —<br>— | dB |

### Table 4.10. Temperature Sensor Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6V V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Linearity | | — | ±1 | — | °C |
| Slope | | — | 3.40 | — | mV/°C |
| Slope Error[1] | | — | 40 | — | µV/°C |
| Offset | Temp = 25 °C | — | 1025 | — | mV |
| Offset Error[1] | Temp = 25 °C | — | 18 | — | mV |
| Temperature Sensor Settling Time[2] | Initial Voltage=0 V<br>Initial Voltage=3.6 V | — | — | 3.0<br>6.5 | µs |
| Supply Current | | — | 35 | — | µA |

**Notes:**
1. Represents one standard deviation from the mean.
2. The temperature sensor settling time, resulting from an ADC mux change or enabling of the temperature sensor, varies with the voltage of the previously sampled channel and can be up to 6.5 µs if the previously sampled channel voltage was greater than 3 V. To minimize the temperature sensor settling time, the ADC mux can be momentarily set to ground before being set to the temperature sensor output. This ensures that the temperature sensor output will settle in 3 µs or less.

### Table 4.11. Voltage Reference Electrical Characteristics

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **Internal High-Speed Reference (REFSL[1:0] = 11)** | | | | | |
| Output Voltage | –40 to +85 °C,<br>$V_{DD}$ = 1.8–3.6 V | 1.60 | 1.65 | 1.70 | V |
| VREF Turn-on Time | | — | — | 1.7 | µs |
| Supply Current | | — | 200 | — | µA |
| **Internal Precision Reference (REFSL[1:0] = 00, REFOE = 1)** | | | | | |
| Output Voltage | –40 to +85 °C,<br>$V_{DD}$ = 1.8–3.6 V | 1.645 | 1.680 | 1.715 | V |
| VREF Short-Circuit Current | | — | 3.5 | — | mA |
| Load Regulation | Load = 0 to 200 µA to AGND | — | 400 | — | µV/µA |
| VREF Turn-on Time 1 | 4.7 µF tantalum, 0.1 µF ceramic bypass, settling to 0.5 LSB | — | 15 | — | ms |
| VREF Turn-on Time 2 | 0.1 µF ceramic bypass, settling to 0.5 LSB | — | 300 | — | µs |
| VREF Turn-on Time 3 | no bypass cap, settling to 0.5 LSB | — | 25 | — | µs |
| Supply Current | | — | 15 | — | µA |
| **External Reference (REFSL[1:0] = 00, REFOE = 0)** | | | | | |
| Input Voltage Range | | 0 | — | $V_{DD}$ | V |
| Input Current | Sample Rate = 300 ksps; VREF = 3.0 V | — | 5.25 | — | µA |

**Table 4.12. IREF0 Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C, unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **Static Performance** | | | | | |
| Resolution | | | 6 | | bits |
| Output Compliance Range | Low Power Mode, Source | 0 | — | $V_{DD}$ – 0.4 | V |
| | High Current Mode, Source | 0 | — | $V_{DD}$ – 0.8 | V |
| | Low Power Mode, Sink | 0.3 | — | $V_{DD}$ | V |
| | High Current Mode, Sink | 0.8 | — | $V_{DD}$ | V |
| Integral Nonlinearity | | — | <±0.2 | ±1.0 | LSB |
| Differential Nonlinearity | | — | <±0.2 | ±1.0 | LSB |
| Offset Error | | — | <±0.1 | ±0.5 | LSB |
| Full Scale Error* | Low Power Mode, Source | — | — | ±5 | % |
| | High Current Mode, Source | — | — | ±6 | % |
| | Low Power Mode, Sink | — | — | ±8 | % |
| | High Current Mode, Sink | — | — | ±8 | % |
| Absolute Current Error | Low Power Mode Sourcing 20 µA | — | <±1 | ±3 | % |
| **Dynamic Performance** | | | | | |
| Output Settling Time to 1/2 LSB | | — | 300 | — | ns |
| Startup Time | | — | 1 | — | µs |
| **Power Consumption** | | | | | |
| Net Power Supply Current ($V_{DD}$ supplied to IREF0 minus any output source current) | Low Power Mode, Source | | | | |
| | IREF0DAT = 000001 | — | 10 | — | µA |
| | IREF0DAT = 111111 | — | 10 | — | µA |
| | High Current Mode, Source | | | | |
| | IREF0DAT = 000001 | — | 10 | — | µA |
| | IREF0DAT = 111111 | — | 10 | — | µA |
| | Low Power Mode, Sink | | | | |
| | IREF0DAT = 000001 | — | 1 | — | µA |
| | IREF0DAT = 111111 | — | 11 | — | µA |
| | High Current Mode, Sink | | | | |
| | IREF0DAT = 000001 | — | 12 | — | µA |
| | IREF0DAT = 111111 | — | 81 | — | µA |

***Note:** Full scale is 63 µA in low power mode and 504 µA in high power mode.

**Table 4.13. Comparator Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise noted.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Response Time: Mode 0, $V_{DD}$ = 2.4 V, $V_{CM}^*$ = 1.2 V | CP0+ – CP0– = 100 mV | — | 130 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 200 | — | ns |
| Response Time: Mode 1, $V_{DD}$ = 2.4 V, $V_{CM}^*$ = 1.2 V | CP0+ – CP0– = 100 mV | — | 210 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 410 | — | ns |
| Response Time: Mode 2, $V_{DD}$ = 2.4 V, $V_{CM}^*$ = 1.2 V | CP0+ – CP0– = 100 mV | — | 420 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 1200 | — | ns |
| Response Time: Mode 3, $V_{DD}$ = 2.4 V, $V_{CM}^*$ = 1.2 V | CP0+ – CP0– = 100 mV | — | 1750 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 6200 | — | ns |
| Common-Mode Rejection Ratio | | — | 1.5 | 4 | mV/V |
| Inverting or Non-Inverting Input Voltage Range | | –0.25 | — | $V_{DD}$ + 0.25 | V |
| Input Capacitance | | — | 12 | — | pF |
| Input Bias Current | | — | 1 | — | nA |
| Input Offset Voltage | | –7 | — | +7 | mV |
| **Power Supply** | | | | | |
| Power Supply Rejection | | — | 0.1 | — | mV/V |
| Power-up Time | VDD = 3.6 V | — | 0.6 | — | µs |
| | VDD = 3.0 V | — | 1.0 | — | µs |
| | VDD = 2.4 V | — | 1.8 | — | µs |
| | VDD = 1.8 V | — | 10 | — | µs |
| Supply Current at DC | Mode 0 | — | 23 | — | µA |
| | Mode 1 | — | 8.8 | — | µA |
| | Mode 2 | — | 2.6 | — | µA |
| | Mode 3 | — | 0.4 | — | µA |

**Note:** Vcm is the common-mode voltage on CP0+ and CP0–.

**Table 4.13. Comparator Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise noted.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| **Hysteresis** | | | | | |
| Mode 0 | | | | | |
| Hysteresis 1 | (CPnHYP/N1–0 = 00) | — | 0 | — | mV |
| Hysteresis 2 | (CPnHYP/N1–0 = 01) | — | 8.5 | — | mV |
| Hysteresis 3 | (CPnHYP/N1–0 = 10) | — | 17 | — | mV |
| Hysteresis 4 | (CPnHYP/N1–0 = 11) | — | 34 | — | mV |
| Mode 1 | | | | | |
| Hysteresis 1 | (CPnHYP/N1–0 = 00) | — | 0 | — | mV |
| Hysteresis 2 | (CPnHYP/N1–0 = 01) | — | 6.5 | — | mV |
| Hysteresis 3 | (CPnHYP/N1–0 = 10) | — | 13 | — | mV |
| Hysteresis 4 | (CPnHYP/N1–0 = 11) | — | 26 | — | mV |
| Mode 2 | | | | | |
| Hysteresis 1 | (CPnHYP/N1–0 = 00) | — | 0 | 1 | mV |
| Hysteresis 2 | (CPnHYP/N1–0 = 01) | 2 | 5 | 10 | mV |
| Hysteresis 3 | (CPnHYP/N1–0 = 10) | 5 | 10 | 20 | mV |
| Hysteresis 4 | (CPnHYP/N1–0 = 11) | 12 | 20 | 30 | mV |
| Mode 3 | | | | | |
| Hysteresis 1 | (CPnHYP/N1–0 = 00) | — | 0 | — | mV |
| Hysteresis 2 | (CPnHYP/N1–0 = 01) | — | 4.5 | — | mV |
| Hysteresis 3 | (CPnHYP/N1–0 = 10) | — | 9 | — | mV |
| Hysteresis 4 | (CPnHYP/N1–0 = 11) | — | 17 | — | mV |
| **Note:** Vcm is the common-mode voltage on CP0+ and CP0–. | | | | | |

**Table 4.14. DC-DC Converter (DC0) Electrical Characteristics**

VBAT = 0.9 to 1.8 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Input Voltage Range | | 0.9 | — | 1.8 | V |
| Input Inductor Value | | 500 | 680 | 900 | nH |
| Input Inductor Current Rating | | 250 | — | — | mA |
| Inductor DC Resistance | | — | — | 0.5 | Ω |
| Input Capacitor Value | <br>Source Impedance < 2 Ω | —<br>— | 4.7<br>1.0 | —<br>— | µF |
| Output Voltage Range | Target Output = 1.8 V | 1.73 | 1.80 | 1.87 | V |
| | Target Output = 1.9 V | 1.83 | 1.90 | 1.97 | V |
| | Target Output = 2.0 V | 1.93 | 2.00 | 2.07 | V |
| | Target Output = 2.1 V | 2.03 | 2.10 | 2.17 | V |
| | Target Output = 2.1 V | 2.30 | 2.40 | 2.50 | V |
| | Target Output = 2.7 V | 2.60 | 2.70 | 2.80 | V |
| | Target Output = 3.0 V | 2.90 | 3.00 | 3.10 | V |
| | Target Output = 3.3 V | 3.18 | 3.30 | 3.42 | V |
| Output Load Regulation | Target Output = 2.0 V, 1 to 30 mA | — | ±0.3 | — | % |
| | Target Output = 3.0 V, 1 to 20 mA | — | ±1 | — | % |
| Output Current (based on output power spec) | Target Output = 1.8 V | — | — | 36 | mA |
| | Target Output = 1.9 V | — | — | 34 | mA |
| | Target Output = 2.0 V | — | — | 32 | mA |
| | Target Output = 2.1 V | — | — | 30 | mA |
| | Target Output = 2.4 V | — | — | 27 | mA |
| | Target Output = 2.7 V | — | — | 24 | mA |
| | Target Output = 3.0 V | — | — | 21 | mA |
| | Target Output = 3.3 V | — | — | 19 | mA |
| Output Power | | — | — | 65 | mW |
| Bias Current | from VBAT supply<br>from VDD/DC+ supply | —<br>— | 80<br>100 | —<br>— | µA |
| Clocking Frequency | | 1.6 | 2.4 | 3.2 | MHz |
| Maximum DC Load Current During Startup | | — | — | 1 | mA |
| Capacitance Connected to Output | | 0.8 | 1.0 | 2.0 | µF |

**Table 4.15. VREG0 Electrical Characteristics**

$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Input Voltage Range | | 1.8 | — | 3.6 | V |
| Bias Current | Normal, Idle, Suspend, or Stop Mode | — | 20 | — | µA |

## 4.3. EZRadioPRO® Electrical Characteristics

### Table 4.16. DC Characteristics[1]

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Supply Voltage Range | $V_{DD}$ | | 1.8 | 3.0 | 3.6 | V |
| Power Saving Modes | $I_{Shutdown}$ | RC Oscillator, Main Digital Regulator, and Low Power Digital Regulator OFF | — | 15 | 50 | nA |
| | $I_{Standby}$ | Low Power Digital Regulator ON (Register values retained) and Main Digital Regulator, and RC Oscillator OFF | — | 450 | 800 | nA |
| | $I_{Sleep}$ | RC Oscillator and Low Power Digital Regulator ON (Register values retained) and Main Digital Regulator OFF | — | 1 | — | µA |
| | $I_{Sensor-LBD}$ | Main Digital Regulator and Low Battery Detector ON, Crystal Oscillator and all other blocks OFF[2] | — | 1 | — | µA |
| | $I_{Sensor-TS}$ | Main Digital Regulator and Temperature Sensor ON, Crystal Oscillator and all other blocks OFF[2] | — | 1 | — | µA |
| | $I_{Ready}$ | Crystal Oscillator and Main Digital Regulator ON, all other blocks OFF. Crystal Oscillator buffer disabled | — | 800 | — | µA |
| TUNE Mode Current | $I_{Tune}$ | Synthesizer and regulators enabled | — | 8.5 | — | mA |
| RX Mode Current | $I_{RX}$ | | — | 18.5 | — | mA |
| TX Mode Current —Si1000/1 | $I_{TX_+20}$ | txpow[2:0] = 111 (+20 dBm) Using Silicon Labs' Reference Design. TX current consumption is dependent on match and board layout. | — | 85 | — | mA |
| TX Mode Current —Si1002/3 | $I_{TX_+13}$ | txpow[2:0] = 110 (+13 dBm) Using Silicon Labs' Reference Design. TX current consumption is dependent on match and board layout. | — | 30 | — | mA |
| | $I_{TX_+1}$ | txpow[2:0] = 010 (+1 dBm) Using Silicon Labs' Reference Design. TX current consumption is dependent on match and board layout. | — | 17 | — | mA |

**Notes:**
1. All specification guaranteed by production test unless otherwise noted. Production test conditions and max limits are listed in the "Production Test Conditions" section on page 73.
2. Guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.17. Synthesizer AC Electrical Characteristics[1]**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Synthesizer Frequency Range | $F_{SYN}$ | | 240 | — | 960 | MHz |
| Synthesizer Frequency Resolution[2] | $F_{RES-LB}$ | Low Band, 240–480 MHz | — | 156.25 | — | Hz |
| | $F_{RES-HB}$ | High Band, 480–960 MHz | — | 312.5 | — | Hz |
| Reference Frequency Input Level[2] | $f_{REF_LV}$ | When using external reference signal driving XOUT pin, instead of using crystal. Measured peak-to-peak ($V_{PP}$) | 0.7 | — | 1.6 | V |
| Synthesizer Settling Time[2] | $t_{LOCK}$ | Measured from exiting Ready mode with XOSC running to any frequency. Including VCO Calibration. | — | 200 | — | µs |
| Residual FM[2] | $\Delta F_{RMS}$ | Integrated over ±250 kHz bandwidth (500 Hz lower bound of integration) | — | 2 | 4 | $kHz_{RMS}$ |
| Phase Noise[2] | $L\phi(f_M)$ | $\Delta F$ = 10 kHz | — | –80 | — | dBc/Hz |
| | | $\Delta F$ = 100 kHz | — | –90 | — | dBc/Hz |
| | | $\Delta F$ = 1 MHz | — | –115 | — | dBc/Hz |
| | | $\Delta F$ = 10 MHz | — | –130 | — | dBc/Hz |

**Notes:**
1. All specification guaranteed by production test unless otherwise noted. Production test conditions and max limits are listed in the "Production Test Conditions" section on page 73.
2. Guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.18. Receiver AC Electrical Characteristics[1]**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| RX Frequency Range | $F_{RX}$ | | 240 | — | 960 | MHz |
| RX Sensitivity[2] | $P_{RX_2}$ | (BER < 0.1%) (2 kbps, GFSK, BT = 0.5, Δf = ±5 kHz)[3] | — | –121 | — | dBm |
| | $P_{RX_40}$ | (BER < 0.1%) (40 kbps, GFSK, BT = 0.5, Δf = ±20 kHz)[3] | — | –108 | — | dBm |
| | $P_{RX_100}$ | (BER < 0.1%) (100 kbps, GFSK, BT = 0.5, Δf = ±50 kHz)[3] | — | –104 | — | dBm |
| | $P_{RX_125}$ | (BER < 0.1%) (125 kbps, GFSK, BT = 0.5, Δf = ±62.5 kHz) | — | –101 | — | dBm |
| | $P_{RX_OOK}$ | (BER < 0.1%) (4.8 kbps, 350 kHz BW, OOK)[3] | — | –110 | — | dBm |
| | | (BER < 0.1%) (40 kbps, 400 kHz BW, OOK)[3] | — | –102 | — | dBm |
| RX Channel Bandwidth[3] | BW | | 2.6 | — | 620 | kHz |
| BER Variation vs Power Level[3] | $P_{RX_RES}$ | Up to +5 dBm Input Level | — | 0 | 0.1 | ppm |
| LNA Input Impedance[3] (Unmatched—measured differentially across RX input pins) | $R_{IN-RX}$ | 915 MHz | — | 51–60j | — | Ω |
| | | 868 MHz | — | 54–63j | — | |
| | | 433 MHz | — | 89–110j | — | |
| | | 315 MHz | — | 107–137j | — | |
| RSSI Resolution | $RES_{RSSI}$ | | — | ±0.5 | — | dB |
| ±1-Ch Offset Selectivity[3] | $C/I_{1-CH}$ | Desired Ref Signal 3 dB above sensitivity, BER < 0.1%. Interferer and desired modulated with 40 kbps ΔF = 20 kHz GFSK with BT = 0.5, channel spacing = 150 kHz | — | –31 | — | dB |
| ±2-Ch Offset Selectivity[3] | $C/I_{2-CH}$ | | — | –35 | — | dB |
| ≥ ±3-Ch Offset Selectivity[3] | $C/I_{3-CH}$ | | — | –40 | — | dB |
| Blocking at 1 MHz Offset[3] | $1M_{BLOCK}$ | Desired Ref Signal 3 dB above sensitivity. Interferer and desired modulated with 40 kbps ΔF = 20 kHz GFSK with BT = 0.5 | — | –52 | — | dB |
| Blocking at 4 MHz Offset[3] | $4M_{BLOCK}$ | | — | –56 | — | dB |
| Blocking at 8 MHz Offset[3] | $8M_{BLOCK}$ | | — | –63 | — | dB |
| Image Rejection[3] | $Im_{REJ}$ | Rejection at the image frequency. IF=937 kHz | — | –30 | — | dB |
| Spurious Emissions[3] | $P_{OB_RX1}$ | Measured at RX pins | — | — | –54 | dBm |

**Notes:**

1. All specification guaranteed by production test unless otherwise noted. Production test conditions and max limits are listed in the "Production Test Conditions" section on page 73.
2. Receive sensitivity at multiples of 30 MHz may be degraded. If channels with a multiple of 30 MHz are required it is recommended to shift the crystal frequency. Contact Silicon Labs Applications Support for recommendations.
3. Guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.19. Transmitter AC Electrical Characteristics[1]**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| TX Frequency Range | $F_{TX}$ | | 240 | — | 960 | MHz |
| FSK Data Rate[2] | $DR_{FSK}$ | | 0.123 | — | 256 | kbps |
| OOK Data Rate[2] | $DR_{OOK}$ | | 0.123 | — | 40 | kbps |
| Modulation Deviation | Δf1 | 860–960 MHz | ±0.625 | | ±320 | kHz |
| | Δf2 | 240–860 MHz | ±0.625 | | ±160 | kHz |
| Modulation Deviation Resolution[2] | $\Delta f_{RES}$ | | — | 0.625 | — | kHz |
| Output Power Range—Si1000/1[3] | $P_{TX}$ | | +1 | — | +20 | dBm |
| Output Power Range—Si1002/3[3] | $P_{TX}$ | | –4 | — | +13 | dBm |
| TX RF Output Steps[2] | $\Delta P_{RF_OUT}$ | controlled by txpow[2:0] | — | 3 | — | dB |
| TX RF Output Level[2] Variation vs. Temperature | $\Delta P_{RF_TEMP}$ | –40 to +85 °C | — | 2 | — | dB |
| TX RF Output Level Variation vs. Frequency[2] | $\Delta P_{RF_FREQ}$ | Measured across any one frequency band | — | 1 | — | dB |
| Transmit Modulation Filtering[2] | B*T | Gaussian Filtering Bandwith Time Product | — | 0.5 | — | |
| Spurious Emissions[2] | $P_{OB\text{-}TX1}$ | $P_{OUT}$ = 11 dBm, Frequencies <1 GHz | — | — | –54 | dBm |
| | $P_{OB\text{-}TX2}$ | 1–12.75 GHz, excluding harmonics | — | — | –54 | dBm |
| Harmonics[2] | $P_{2HARM}$ | Using reference design TX matching network and filter with max output power. Harmonics reduce linearly with output power. | — | — | –42 | dBm |
| | $P_{3HARM}$ | | — | — | –42 | dBm |

**Notes:**

1. All specification guaranteed by production test unless otherwise noted. Production test conditions and max limits are listed in the "Production Test Conditions" section on page 73.
2. Guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.
3. Output power is dependent on matching components, board layout, and is measured at the pin.

**Table 4.20. Auxiliary Block Specifications[1]**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Temperature Sensor Accuracy[2] | $TS_A$ | After calibrated via sensor offset register tvoffs[7:0] | — | 0.5 | — | °C |
| Temperature Sensor Sensitivity[2] | $TS_S$ | | — | 5 | — | mV/°C |
| Low Battery Detector Resolution[2] | $LBD_{RES}$ | | — | 50 | — | mV |
| Low Battery Detector Conversion Time[2] | $LBD_{CT}$ | | — | 250 | — | µs |
| Microcontroller Clock Output Frequency | $F_{MC}$ | Configurable to 30 MHz, 15 MHz, 10 MHz, 4 MHz, 3 MHz, 2 MHz, 1 MHz, or 32.768 kHz | 32.768K | — | 30M | Hz |
| General Purpose ADC Resolution[2] | $ADC_{ENB}$ | | — | 8 | — | bit |
| General Purpose ADC Bit Resolution[2] | $ADC_{RES}$ | | — | 4 | — | mV/bit |
| Temp Sensor & General Purpose ADC Conversion Time[2] | $ADC_{CT}$ | | — | 305 | — | µs |
| 30 MHz XTAL Start-Up Time | $t_{30M}$ | Using XTAL and board layout in reference design. Start-up time will vary with XTAL type and board layout. | — | 600 | — | µs |
| 30 MHz XTAL Cap Resolution[2] | $30M_{RES}$ | See Section “23.5.8. Crystal Oscillator” on page 263 for total load capacitance calculation | — | 97 | — | fF |
| 32 kHz XTAL Start-Up Time[2] | $t_{32k}$ | | — | 6 | — | sec |
| 32 kHz Accuracy using Internal RC Oscillator[2] | $32KRC_{RES}$ | | — | 1000 | — | ppm |
| 32 kHz RC Oscillator Start-Up | $t_{32kRC}$ | | — | 500 | — | µs |
| POR Reset Time | $t_{POR}$ | | — | 9.5 | — | ms |
| Software Reset Time[2] | $t_{soft}$ | | — | 250 | — | µs |

**Notes:**

1. All specification guaranteed by production test unless otherwise noted. Production test conditions and max limits are listed in the "Production Test Conditions" section on page 73.
2. Guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.21. Digital IO Specifications (nIRQ)**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Rise Time | $T_{RISE}$ | 0.1 x $V_{DD}$ to 0.9 x $V_{DD}$, $C_L$= 5 pF | — | — | 8 | ns |
| Fall Time | $T_{FALL}$ | 0.9 x $V_{DD}$ to 0.1 x $V_{DD}$, $C_L$= 5 pF | — | — | 8 | ns |
| Input Capacitance | $C_{IN}$ | | — | — | 1 | pF |
| Logic High Level Input Voltage | $V_{IH}$ | | $V_{DD}$ – 0.6 | — | — | V |
| Logic Low Level Input Voltage | $V_{IL}$ | | | — | 0.6 | V |
| Input Current | $I_{IN}$ | 0<$V_{IN}$< $V_{DD}$ | –100 | — | 100 | nA |
| Logic High Level Output Voltage | $V_{OH}$ | $I_{OH}$<1 mA source, $V_{DD}$=1.8 V | $V_{DD}$ – 0.6 | — | — | V |
| Logic Low Level Output Voltage | $V_{OL}$ | $I_{OL}$<1 mA sink, $V_{DD}$=1.8 V | — | — | 0.6 | V |

**Note:** All specifications guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.22. GPIO Specifications (GPIO_0, GPIO_1, and GPIO_2)**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Rise Time | $T_{RISE}$ | 0.1 x $V_{DD}$ to 0.9 x $V_{DD}$, $C_L$= 10 pF, DRV<1:0>=HH | — | — | 8 | ns |
| Fall Time | $T_{FALL}$ | 0.9 x $V_{DD}$ to 0.1 x $V_{DD}$, $C_L$= 10 pF, DRV<1:0>=HH | — | — | 8 | ns |
| Input Capacitance | $C_{IN}$ | | — | — | 1 | pF |
| Logic High Level Input Voltage | $V_{IH}$ | | $V_{DD}$ – 0.6 | — | | V |
| Logic Low Level Input Voltage | $V_{IL}$ | | — | — | 0.6 | V |
| Input Current | $I_{IN}$ | 0<$V_{IN}$< $V_{DD}$ | –100 | — | 100 | nA |
| Input Current If Pullup is Activated | $I_{INP}$ | $V_{IL}$=0 V | 5 | — | 25 | µA |
| Maximum Output Current | $I_{OmaxLL}$ | DRV<1:0>=LL | 0.1 | 0.5 | 0.8 | mA |
| | $I_{OmaxLH}$ | DRV<1:0>=LH | 0.9 | 2.3 | 3.5 | mA |
| | $I_{OmaxHL}$ | DRV<1:0>=HL | 1.5 | 3.1 | 4.8 | mA |
| | $I_{OmaxHH}$ | DRV<1:0>=HH | 1.8 | 3.6 | 5.4 | mA |
| Logic High Level Output Voltage | $V_{OH}$ | $I_{OH}$< $I_{Omax}$ source, $V_{DD}$=1.8 V | $V_{DD}$ – 0.6 | — | — | V |
| Logic Low Level Output Voltage | $V_{OL}$ | $I_{OL}$< $I_{Omax}$ sink, $V_{DD}$=1.8 V | — | — | 0.6 | V |

**Note:** All specifications guaranteed by qualification. Qualification test conditions are listed in the "Production Test Conditions" section on page 73.

**Table 4.23. Absolute Maximum Ratings**

| Parameter | Value | Unit |
|---|---|---|
| $V_{DD}$ to GND | –0.3, +3.6 | V |
| Instantaneous $V_{RF\text{-}peak}$ to GND on TX Output Pin | –0.3, +8.0 | V |
| Sustained $V_{RF\text{-}peak}$ to GND on TX Output Pin | –0.3, +6.5 | V |
| Voltage on Digital Control Inputs | –0.3, $V_{DD}$ + 0.3 | V |
| Voltage on Analog Inputs | –0.3, $V_{DD}$ + 0.3 | V |
| RX Input Power | +10 | dBm |
| Operating Ambient Temperature Range $T_A$ | –40 to +85 | °C |
| Thermal Impedance $\theta_{JA}$ | 30 | °C/W |
| Junction Temperature $T_J$ | +125 | °C |
| Storage Temperature Range $T_{STG}$ | –55 to +125 | °C |

**Note:** Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. These are stress ratings only and functional operation of the device at or beyond these ratings in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. Power Amplifier may be damaged if switched on without proper load or termination connected. TX matching network design will influence TX VRF-peak on TX output pin. Caution: ESD sensitive device.

## 4.4. Definition of Test Conditions for the EZRadioPRO Peripheral

**Production Test Conditions:**

- $T_A$ = +25 °C
- $V_{DD}$ = +3.3 VDC
- Sensitivity measured at 919 MHz
- TX output power measured at 915 MHz
- External reference signal (XOUT) = 1.0 $V_{PP}$ at 30 MHz, centered around 0.8 VDC
- Production test schematic (unless noted otherwise)
- All RF input and output levels referred to the pins of the Si100x (not the RF module)

**Qualification Test Conditions:**

- $T_A$ = –40 to +85 °C
- $V_{DD}$ = +1.8 to +3.6 VDC
- Using 4432, 4431, or 4430 DKDB1 reference design or production test schematic
- All RF input and output levels referred to the pins of the Si100x (not the RF module)

# 5. 10-Bit SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode

The ADC0 on the Si1000/1/2/3/4/5 is a 300 ksps, 10-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and programmable window detector. ADC0 also has an autonomous low power Burst Mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. It also has a 16-bit accumulator that can automatically oversample and average the ADC results.

The ADC is fully configurable under software control via Special Function Registers. The ADC0 operates in Single-ended mode and may be configured to measure various different signals using the analog multiplexer described in “5.5. ADC0 Analog Multiplexer” on page 91. The voltage reference for the ADC is selected as described in “5.7. Voltage and Ground Reference Options” on page 96.

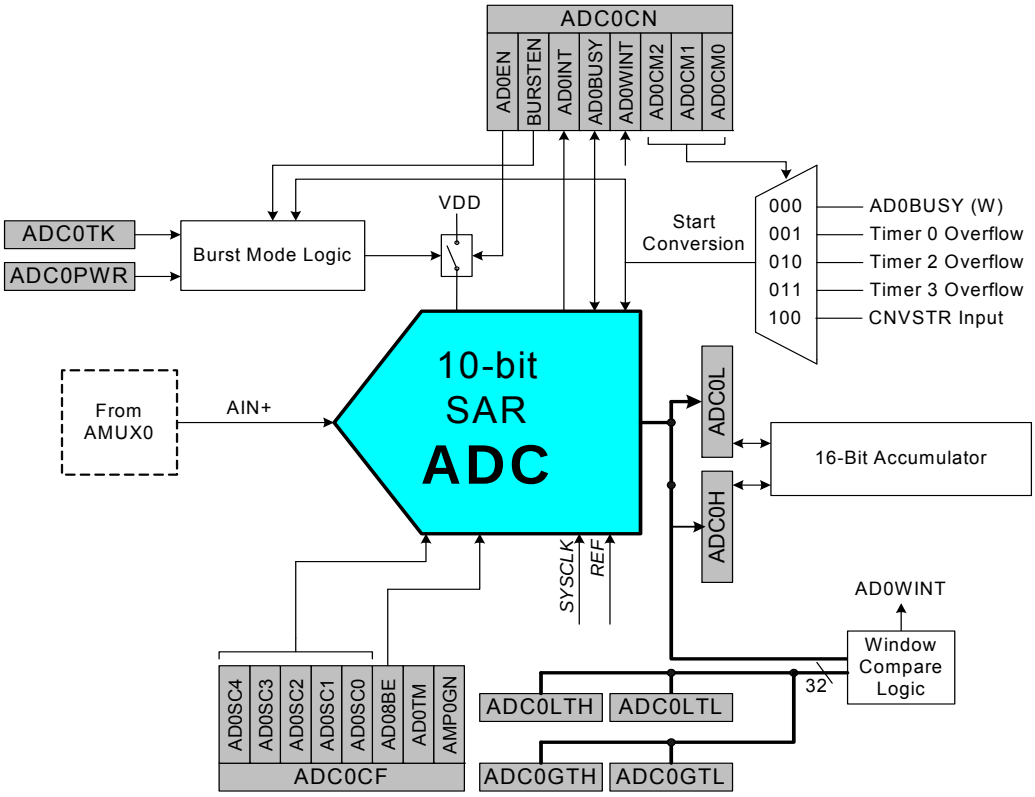



**Figure 5.1. ADC0 Functional Block Diagram**

## 5.1. Output Code Formatting

The registers ADC0H and ADC0L contain the high and low bytes of the output conversion code from the ADC at the completion of each conversion. Data can be right-justified or left-justified, depending on the setting of the AD0SJST[2:0]. When the repeat count is set to 1, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to 0.

| Input Voltage | Right-Justified ADC0H:ADC0L (AD0SJST = 000) | Left-Justified ADC0H:ADC0L (AD0SJST = 100) |
|---|---|---|
| VREF x 1023/1024 | 0x03FF | 0xFFC0 |
| VREF x 512/1024 | 0x0200 | 0x8000 |
| VREF x 256/1024 | 0x0100 | 0x4000 |
| 0 | 0x0000 | 0x0000 |

When the repeat count is greater than 1, the output conversion code represents the accumulated result of the conversions performed and is updated after the last conversion in the series is finished. Sets of 4, 8, 16, 32, or 64 consecutive samples can be accumulated and represented in unsigned integer format. The repeat count can be selected using the AD0RPT bits in the ADC0AC register. When a repeat count higher than 1, the ADC output must be right-justified (AD0SJST = 0xx); unused bits in the ADC0H and ADC0L registers are set to 0. The example below shows the right-justified result for various input voltages and repeat counts. Notice that accumulating $2^n$ samples is equivalent to left-shifting by *n* bit positions when all samples returned from the ADC have the same value.

| Input Voltage | Repeat Count = 4 | Repeat Count = 16 | Repeat Count = 64 |
|---|---|---|---|
| $V_{REF}$ x 1023/1024 | 0x0FFC | 0x3FF0 | 0xFFC0 |
| $V_{REF}$ x 512/1024 | 0x0800 | 0x2000 | 0x8000 |
| $V_{REF}$ x 511/1024 | 0x07FC | 0x1FF0 | 0x7FC0 |
| 0 | 0x0000 | 0x0000 | 0x0000 |

The AD0SJST bits can be used to format the contents of the 16-bit accumulator. The accumulated result can be shifted right by 1, 2, or 3 bit positions. Based on the principles of oversampling and averaging, the effective ADC resolution increases by 1 bit each time the oversampling rate is increased by a factor of 4. The example below shows how to increase the effective ADC resolution by 1, 2, and 3 bits to obtain an effective ADC resolution of 11-bit, 12-bit, or 13-bit respectively without CPU intervention.

| Input Voltage | Repeat Count = 4<br>Shift Right = 1<br>11-Bit Result | Repeat Count = 16<br>Shift Right = 2<br>12-Bit Result | Repeat Count = 64<br>Shift Right = 3<br>13-Bit Result |
|---|---|---|---|
| $V_{REF}$ x 1023/1024 | 0x07F7 | 0x0FFC | 0x1FF8 |
| $V_{REF}$ x 512/1024 | 0x0400 | 0x0800 | 0x1000 |
| $V_{REF}$ x 511/1024 | 0x03FE | 0x04FC | 0x0FF8 |
| 0 | 0x0000 | 0x0000 | 0x0000 |

## 5.2. Modes of Operation

ADC0 has a maximum conversion speed of 300 ksps. The ADC0 conversion clock (SARCLK) is a divided version of the system clock when Burst Mode is disabled (BURSTEN = 0), or a divided version of the low power oscillator when Burst Mode is enabled (BURSEN = 1). The clock divide value is determined by the AD0SC bits in the ADC0CF register.

### 5.2.1. Starting a Conversion

A conversion can be initiated in one of five ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM2–0) in register ADC0CN. Conversions may be initiated by one of the following:

1. Writing a 1 to the AD0BUSY bit of register ADC0CN
2. A Timer 0 overflow (i.e., timed continuous conversions)
3. A Timer 2 overflow
4. A Timer 3 overflow
5. A rising edge on the CNVSTR input signal (pin P0.6)

Writing a 1 to AD0BUSY provides software control of ADC0 whereby conversions are performed "on-demand". During conversion, the AD0BUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the ADC0 interrupt flag (AD0INT). Note: When polling for ADC conversion completions, the ADC0 interrupt flag (AD0INT) should be used. Converted data is available in the ADC0 data registers, ADC0H:ADC0L, when bit AD0INT is logic 1. Note that when Timer 2 or Timer 3 overflows are used as the conversion source, Low Byte overflows are used if Timer 2/3 is in 8-bit mode; High byte overflows are used if Timer 2/3 is in 16-bit mode. See “27. Timers” on page 335 for timer configuration.

**Important Note About Using CNVSTR:** The CNVSTR input pin also functions as Port pin P0.6. When the CNVSTR input is used as the ADC0 conversion source, Port pin P0.6 should be skipped by the Digital Crossbar. To configure the Crossbar to skip P0.6, set to 1 Bit 6 in register P0SKIP. See “21. Port Input/Output” on page 210 for details on Port I/O configuration.

**Important Note:** When operating the device in one-cell mode, there is an option available to automatically synchronize the start of conversion with the quietest portion of the dc-dc converter switching cycle. Activating this option may help to reduce interference from internal or external power supply noise generated by the dc-dc converter. Asserting this bit will hold off the start of an ADC conversion initiated by any of the methods described above until the ADC receives a synchronizing signal from the dc-dc converter. The delay in initiation of the conversion can be as much as one cycle of the dc-dc converter clock, which is 625 ns at the minimum dc-dc clock frequency of 1.6 MHz. The synchronization feature also causes the dc-dc converter clock to be used as the ADC0 conversion clock. The maximum conversion rate will be limited to approximately 170 ksps at the maximum dc-dc converter clock rate of 3.2 MHz. In this mode, the ADC0 SAR Conversion Clock Divider must be set to 1 by setting AD0SC[4:0] = 00000b in SFR register ADC0CF. To provide additional flexibility in minimizing noise, the ADC0 conversion clock provided by the dc-dc converter can be inverted by setting the AD0CKINV bit in the DC0CF register. For additional information on the synchronization feature, see the description of the SYNC bit in “SFR Definition 16.1. DC0CN: DC-DC Converter Control” on page 174 and the description of the AD0CKINV bit in “SFR Definition 16.2. DC0CF: DC-DC Converter Configuration” on page 175. This bit must be set to 0 in two-cell mode for the ADC to operate.

### 5.2.2. Tracking Modes

Each ADC0 conversion must be preceded by a minimum tracking time in order for the converted result to be accurate. The minimum tracking time is given in Table 4.9. The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state when Burst Mode is disabled, the ADC0 input is continuously tracked, except when a conversion is in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power track-and-hold mode. In this mode, each conversion is preceded by a tracking period of 3 SAR clocks (after the start-of-conversion signal). When the CNVSTR signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR (see Figure 5.2). Tracking can also be disabled (shutdown) when the device is in low power standby or sleep modes. Low-power track-and-hold mode is also useful when AMUX settings are frequently changed, due to the settling time requirements described in “5.2.4. Settling Time Requirements” on page 80.

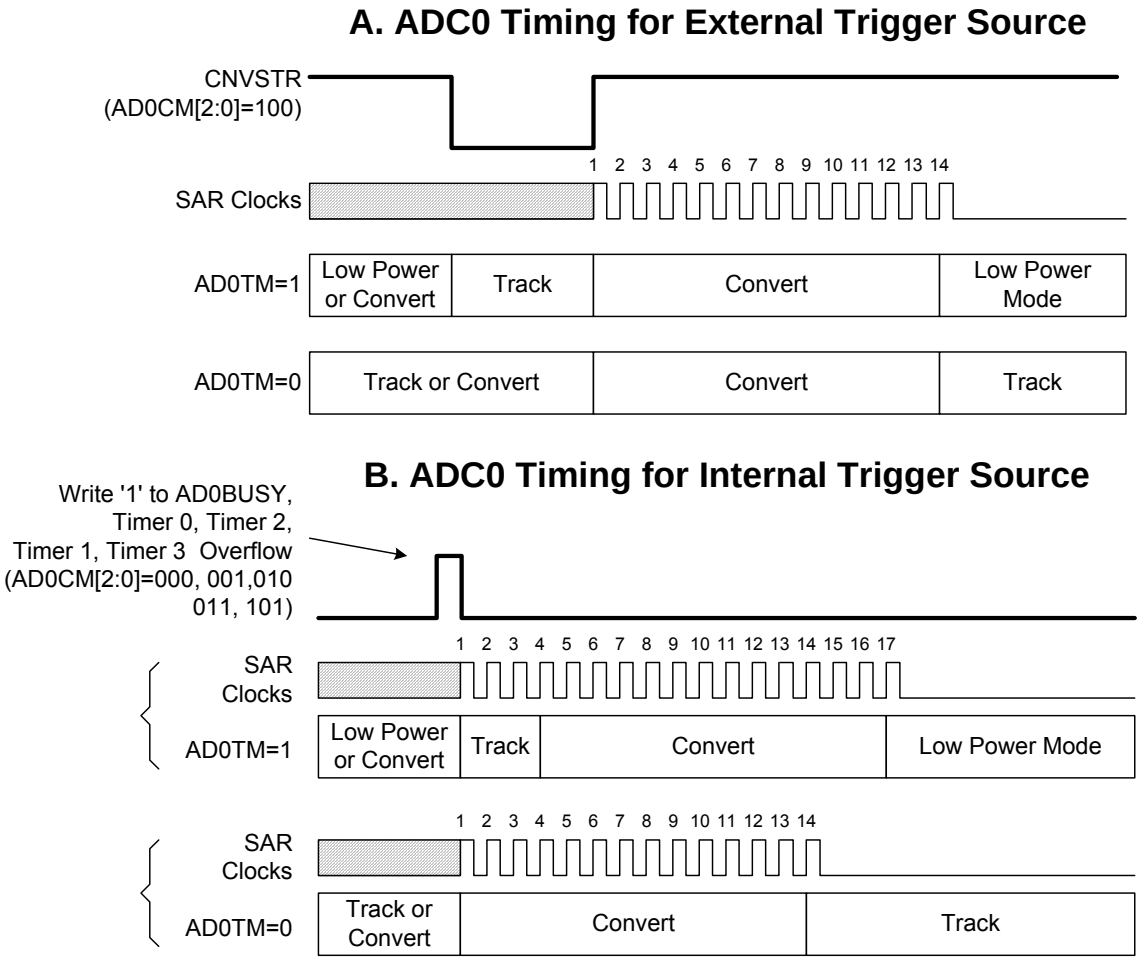



**Figure 5.2. 10-Bit ADC Track and Conversion Example Timing (BURSTEN = 0)**

### 5.2.3. Burst Mode

Burst Mode is a power saving feature that allows ADC0 to remain in a low power state between conversions. When Burst Mode is enabled, ADC0 wakes from a low power state, accumulates 1, 4, 8, 16, 32, or 64 using an internal Burst Mode clock (approximately 20 MHz), then re-enters a low power state. Since the Burst Mode clock is independent of the system clock, ADC0 can perform multiple conversions then enter a low power state within a single system clock cycle, even if the system clock is slow (e.g. 32.768 kHz), or suspended.

Burst Mode is enabled by setting BURSTEN to logic 1. When in Burst Mode, AD0EN controls the ADC0 idle power state (i.e., the state ADC0 enters when not tracking or performing conversions). If AD0EN is set to logic 0, ADC0 is powered down after each burst. If AD0EN is set to logic 1, ADC0 remains enabled after each burst. On each convert start signal, ADC0 is awakened from its Idle Power State. If ADC0 is powered down, it will automatically power up and wait the programmable Power-Up Time controlled by the AD0PWR bits. Otherwise, ADC0 will start tracking and converting immediately. Figure 5.3 shows an example of Burst Mode Operation with a slow system clock and a repeat count of 4.

When Burst Mode is enabled, a single convert start will initiate a number of conversions equal to the repeat count. When Burst Mode is disabled, a convert start is required to initiate each conversion. In both modes, the ADC0 End of Conversion Interrupt Flag (AD0INT) will be set after “repeat count” conversions have been accumulated. Similarly, the Window Comparator will not compare the result to the greater-than and less-than registers until “repeat count” conversions have been accumulated.

In Burst Mode, tracking is determined by the settings in AD0PWR and AD0TK. The default settings for these registers will work in most applications without modification; however, settling time requirements may need adjustment in some applications. Refer to “5.2.4. Settling Time Requirements” on page 80 for more details.

**Notes:**

- Setting AD0TM to 1 will insert an additional 3 SAR clocks of tracking before each conversion, regardless of the settings of AD0PWR and AD0TK.
- When using Burst Mode, care must be taken to issue a convert start signal no faster than once every four SYSCLK periods. This includes external convert start signals.
- A rising edge of external start-of-conversion (CNVSTR) will cause only one ADC conversion in Burst Mode, regardless of the value of the Repeat Count field. The end-of-conversion interrupt will occur after the number of conversions specified in Repeat Count have completed. In other words, if Repeat Count is set to 4, four pulses on CNVSTR will cause an ADC end-of-conversion interrupt. Refer to the bottom portion of Figure 5.3, “Burst Mode Tracking Example with Repeat Count Set to 4,” on page 79 for an example.
- To start multiple conversions in Burst Mode with one external start-of-conversion signal, the external interrupts (/INT0 or /INT1) or Port Match can be used to trigger an ISR that writes to AD0BUSY. External interrupts are configurable to be active low or active high, edge or level sensitive, but is only avail-able on a limited number of pins. Port Match is only level sensitive, but is available on more port pins than the external interrupts. Refer to section “12.6. External Interrupts INT0 and INT1” on page 140 for details on external interrupts and section “21.4. Port Match” on page 219 for details on Port Match.

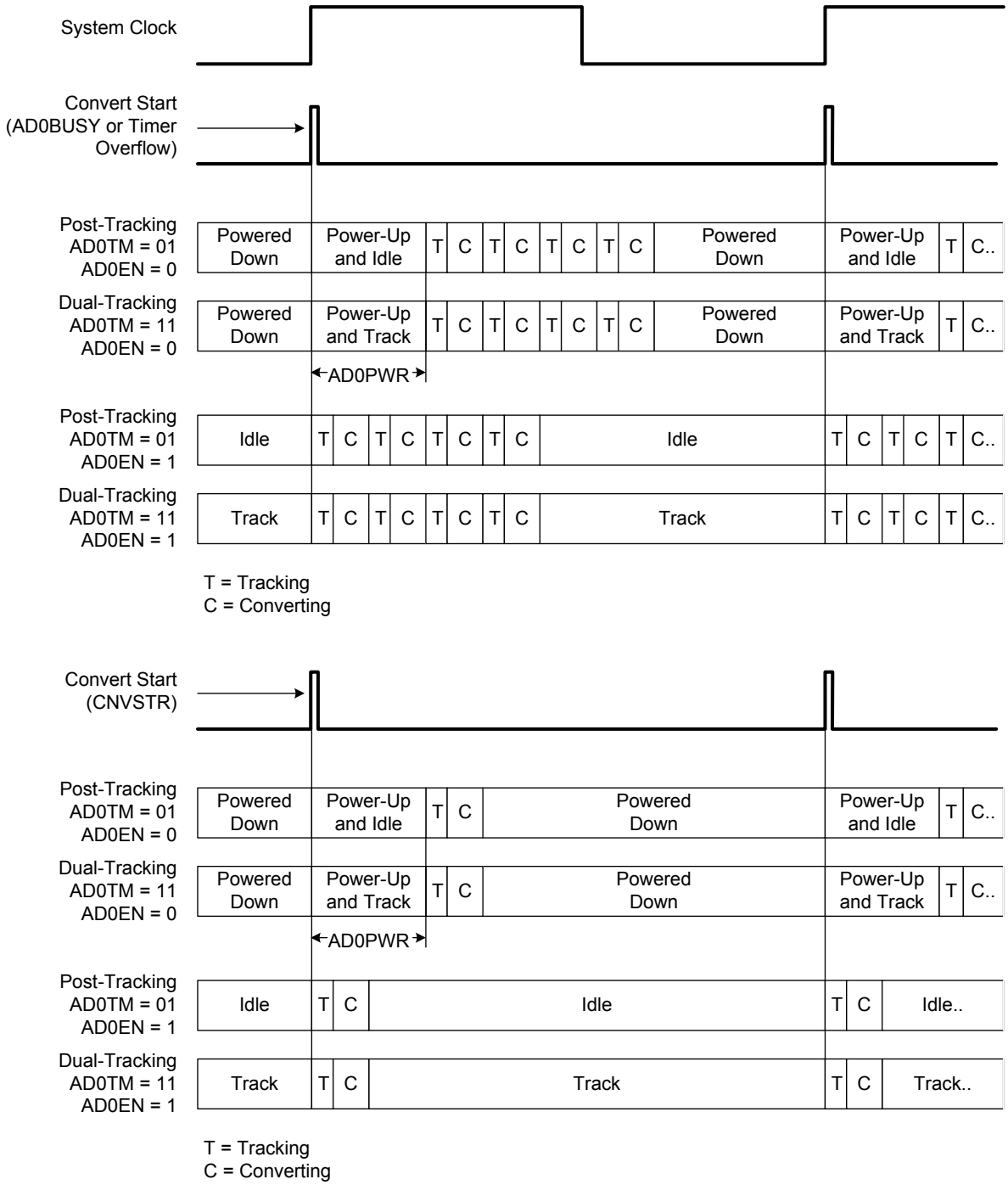



**Figure 5.3. Burst Mode Tracking Example with Repeat Count Set to 4**

### 5.2.4. Settling Time Requirements

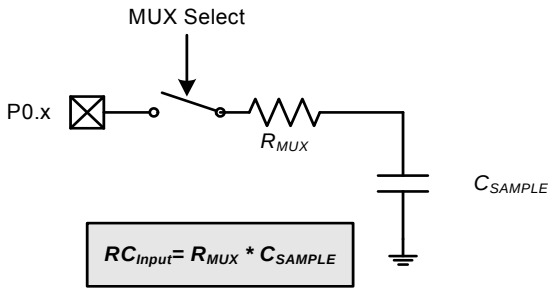
A minimum amount of tracking time is required before each conversion can be performed, to allow the sampling capacitor voltage to settle. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion. Note that in low-power tracking mode, three SAR clocks are used for tracking at the start of every conversion. For many applications, these three SAR clocks will meet the minimum tracking time requirements, and higher values for the external source impedance will increase the required tracking time.

Figure 5.4 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 5.1. When measuring the Temperature Sensor output or $V_{DD}$ with respect to GND, $R_{TOTAL}$ reduces to $R_{MUX}$. See Table 4.9 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

$$t = \ln\left(\frac{2^n}{SA}\right) \times R_{TOTAL} C_{SAMPLE}$$

**Equation 5.1. ADC0 Settling Time Requirements**

Where:
*SA* is the settling accuracy, given as a fraction of an LSB (for example, 0.25 to settle within 1/4 LSB)
*t* is the required settling time in seconds
$R_{TOTAL}$ is the sum of the AMUX0 resistance and any external source resistance.
*n* is the ADC resolution in bits (10).




**Note:** The value of CSAMPLE depends on the PGA Gain. See Table 4.9 for details.

**Figure 5.4. ADC0 Equivalent Input Circuits**

#### 5.2.5. Gain Setting

The ADC has gain settings of 1x and 0.5x. In 1x mode, the full scale reading of the ADC is determined directly by $V_{REF}$. In 0.5x mode, the full-scale reading of the ADC occurs when the input voltage is $V_{REF}$ x 2. The 0.5x gain setting can be useful to obtain a higher input Voltage range when using a small $V_{REF}$ voltage, or to measure input voltages that are between $V_{REF}$ and $V_{DD}$. Gain settings for the ADC are controlled by the AMP0GN bit in register ADC0CF.

## 5.3. 8-Bit Mode

Setting the ADC08BE bit in register ADC0CF to 1 will put the ADC in 8-bit mode.In 8-bit mode, only the 8 MSBs of data are converted, allowing the conversion to be completed in two fewer SAR clock cycles than a 10-bit conversion. This can result in an overall lower power consumption since the system can spend more time in a low power mode. The two LSBs of a conversion are always 00 in this mode, and the ADC0L register will always read back 0x00.

## SFR Definition 5.1. ADC0CN: ADC0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0EN | BURSTEN | AD0INT | AD0BUSY | AD0WINT | ADC0CM | | |
| Type | R/W | R/W | R/W | W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE8; bit-addressable;

| Bit | Name | Function |
|---|---|---|
| 7 | AD0EN | **ADC0 Enable.**<br>0: ADC0 Disabled (low-power shutdown).<br>1: ADC0 Enabled (active and ready for data conversions). |
| 6 | BURSTEN | **ADC0 Burst Mode Enable.**<br>0: ADC0 Burst Mode Disabled.<br>1: ADC0 Burst Mode Enabled. |
| 5 | AD0INT | **ADC0 Conversion Complete Interrupt Flag.**<br>Set by hardware upon completion of a data conversion (BURSTEN=0), or a burst of conversions (BURSTEN=1). Can trigger an interrupt. Must be cleared by software. |
| 4 | AD0BUSY | **ADC0 Busy.**<br>Writing 1 to this bit initiates an ADC conversion when ADC0CM[2:0] = 000. |
| 3 | AD0WINT | **ADC0 Window Compare Interrupt Flag.**<br>Set by hardware when the contents of ADC0H:ADC0L fall within the window specified by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL. Can trigger an interrupt. Must be cleared by software. |
| 2:0 | ADC0CM[2:0] | **ADC0 Start of Conversion Mode Select.**<br>Specifies the ADC0 start of conversion source.<br>000: ADC0 conversion initiated on write of 1 to AD0BUSY.<br>001: ADC0 conversion initiated on overflow of Timer 0.<br>010: ADC0 conversion initiated on overflow of Timer 2.<br>011: ADC0 conversion initiated on overflow of Timer 3.<br>1xx: ADC0 conversion initiated on rising edge of CNVSTR. |

## SFR Definition 5.2. ADC0CF: ADC0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0SC[4:0] | | | | | AD08BE | AD0TM | AMP0GN |
| Type | R/W | | | | | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xBC

| Bit | Name | Function |
|---|---|---|
| 7:3 | AD0SC[4:0] | **ADC0 SAR Conversion Clock Divider.**<br>SAR Conversion clock is derived from FCLK by the following equation, where *AD0SC* refers to the 5-bit value held in bits AD0SC[4:0]. SAR Conversion clock requirements are given in Table 4.9.<br>BURSTEN = 0: FCLK is the current system clock.<br>BURSTEN = 1: FCLK is the 20 MHz low power oscillator, independent of the system clock.<br>$AD0SC = \frac{FCLK}{CLK_{SAR}} - 1$ *<br>*Round the result up.<br>or<br>$CLK_{SAR} = \frac{FCLK}{AD0SC + 1}$ |
| 2 | AD08BE | **ADC0 8-Bit Mode Enable.**<br>0: ADC0 operates in 10-bit mode (normal operation).<br>1: ADC0 operates in 8-bit mode. |
| 1 | AD0TM | **ADC0 Track Mode.**<br>Selects between Normal or Delayed Tracking Modes.<br>0: Normal Track Mode: When ADC0 is enabled, conversion begins immediately following the start-of-conversion signal.<br>1: Delayed Track Mode: When ADC0 is enabled, conversion begins 3 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time. |
| 0 | AMP0GN | **ADC0 Gain Control.**<br>0: The on-chip PGA gain is 0.5.<br>1: The on-chip PGA gain is 1. |

## SFR Definition 5.3. ADC0AC: ADC0 Accumulator Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | AD0AE | AD0SJST | | | AD0RPT | | |
| Type | R/W | W | R/W | | | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xBA

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Read = 0b. |
| 6 | AD0AE | **ADC0 Accumulate Enable.**<br>Enables multiple conversions to be accumulated when burst mode is disabled.<br>0: ADC0H:ADC0L contain the result of the latest conversion when Burst Mode is disabled.<br>1: ADC0H:ADC0L contain the accumulated conversion results when Burst Mode is disabled. Software must write 0x0000 to ADC0H:ADC0L to clear the accumulated result.<br>This bit is write-only. Always reads 0b. |
| 5:3 | AD0SJST[2:0] | **ADC0 Accumulator Shift and Justify.**<br>Specifies the format of data read from ADC0H:ADC0L.<br>000: Right justified. No shifting applied.<br>001: Right justified. Shifted right by 1 bit.<br>010: Right justified. Shifted right by 2 bits.<br>011: Right justified. Shifted right by 3 bits.<br>100: Left justified. No shifting applied.<br>All remaining bit combinations are reserved. |
| 2:0 | AD0RPT[2:0] | **ADC0 Repeat Count.**<br>Selects the number of conversions to perform and accumulate in Burst Mode. This bit field must be set to 000 if Burst Mode is disabled.<br>000: Perform and Accumulate 1 conversion.<br>001: Perform and Accumulate 4 conversions.<br>010: Perform and Accumulate 8 conversions.<br>011: Perform and Accumulate 16 conversions.<br>100: Perform and Accumulate 32 conversions.<br>101: Perform and Accumulate 64 conversions.<br>All remaining bit combinations are reserved. |

## SFR Definition 5.4. ADC0PWR: ADC0 Burst Mode Power-Up Time

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | AD0PWR[3:0] | | | |
| Type | R | R | R | R | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

SFR Page = 0xF; SFR Address = 0xBA

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Read = 0b; Must write 0b. |
| 6:4 | Unused | Read = 0000b; Write = Don't Care. |
| 3:0 | AD0PWR[3:0] | **ADC0 Burst Mode Power-Up Time.**<br>Sets the time delay required for ADC0 to power up from a low power state.<br>For BURSTEN = 0:<br>ADC0 power state controlled by AD0EN.<br>For BURSTEN = 1 and AD0EN = 1:<br>ADC0 remains enabled and does not enter a low power state after all conversions are complete.<br>Conversions can begin immediately following the start-of-conversion signal.<br>For BURSTEN = 1 and AD0EN = 0:<br>ADC0 enters a low power state (as specified in Table 5.1) after all conversions are complete.<br>Conversions can begin a programmed delay after the start-of-conversion signal.<br><br>The ADC0 Burst Mode Power-Up time is programmed according to the following equation:<br><br>$AD0PWR = \frac{Tstartup}{400ns} - 1$<br><br>or<br><br>$Tstartup = (AD0PWR + 1)400ns$ |

## SFR Definition 5.5. ADC0TK: ADC0 Burst Mode Track Time

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | AD0TK[5:0] | | | | | |
| Type | R | R | R/W | | | | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

SFR Page = 0xF; SFR Address = 0xBD

| Bit | Name | Function |
|---|---|---|
| 7:6 | Unused | Read = 00b; Write = Don't Care. |
| 5:0 | AD0TK[5:0] | **ADC0 Burst Mode Track Time.**<br>Sets the time delay between consecutive conversions performed in Burst Mode.<br><br>The ADC0 Burst Mode Track time is programmed according to the following equation:<br><br>$AD0TK = 63 - \left(\frac{Ttrack}{50ns} - 1\right)$<br><br>or<br><br>$Ttrack = (64 - AD0TK)50ns$ |

**Notes:**
1. If AD0TM is set to 1, an additional 3 SAR clock cycles of Track time will be inserted prior to starting the conversion.
2. The Burst Mode Track delay is not inserted prior to the first conversion. The required tracking time for the first conversion should be met by the Burst Mode Power-Up Time.

## SFR Definition 5.6. ADC0H: ADC0 Data Word High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0[15:8] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xBE

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7:0 | ADC0[15:8] | **ADC0 Data Word High Byte.** | Most Significant Byte of the 16-bit ADC0 Accumulator formatted according to the settings in AD0SJST[2:0]. | Set the most significant byte of the 16-bit ADC0 Accumulator to the value written. |
| **Note:** If Accumulator shifting is enabled, the most significant bits of the value read will be zeros. This register should not be written when the SYNC bit is set to 1. | | | | |

## SFR Definition 5.7. ADC0L: ADC0 Data Word Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xBD;

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7:0 | ADC0[7:0] | **ADC0 Data Word Low Byte.** | Least Significant Byte of the 16-bit ADC0 Accumulator formatted according to the settings in AD0SJST[2:0]. | Set the least significant byte of the 16-bit ADC0 Accumulator to the value written. |
| **Note:** If Accumulator shifting is enabled, the most significant bits of the value read will be the least significant bits of the accumulator high byte. This register should not be written when the SYNC bit is set to 1. | | | | |

## 5.4. Programmable Window Detector

The ADC Programmable Window Detector continuously compares the ADC0 output registers to user-programmed limits, and notifies the system when a desired condition is detected. This is especially effective in an interrupt-driven system, saving code space and CPU bandwidth while delivering faster system response times. The window detector interrupt flag (AD0WINT in register ADC0CN) can also be used in polled mode. The ADC0 Greater-Than (ADC0GTH, ADC0GTL) and Less-Than (ADC0LTH, ADC0LTL) registers hold the comparison values. The window detector flag can be programmed to indicate when measured data is inside or outside of the user-programmed limits, depending on the contents of the ADC0 Less-Than and ADC0 Greater-Than registers.

### SFR Definition 5.8. ADC0GTH: ADC0 Greater-Than High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0GT[15:8] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xC4

| Bit | Name | Function |
|---|---|---|
| 7:0 | AD0GT[15:8] | **ADC0 Greater-Than High Byte.**<br>Most Significant Byte of the 16-bit Greater-Than window compare register. |

### SFR Definition 5.9. ADC0GTL: ADC0 Greater-Than Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0GT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xC3

| Bit | Name | Function |
|---|---|---|
| 7:0 | AD0GT[7:0] | **ADC0 Greater-Than Low Byte.**<br>Least Significant Byte of the 16-bit Greater-Than window compare register. |
| **Note:** In 8-bit mode, this register should be set to 0x00. | | |

## SFR Definition 5.10. ADC0LTH: ADC0 Less-Than High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0LT[15:8] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC6

| Bit | Name | Function |
|---|---|---|
| 7:0 | AD0LT[15:8] | **ADC0 Less-Than High Byte.**<br>Most Significant Byte of the 16-bit Less-Than window compare register. |

## SFR Definition 5.11. ADC0LTL: ADC0 Less-Than Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0LT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC5

| Bit | Name | Function |
|---|---|---|
| 7:0 | AD0LT[7:0] | **ADC0 Less-Than Low Byte.**<br>Least Significant Byte of the 16-bit Less-Than window compare register. |
| **Note:** In 8-bit mode, this register should be set to 0x00. | | |

### 5.4.1. Window Detector In Single-Ended Mode

Figure 5.5 shows two example window comparisons for right-justified data, with ADC0LTH:ADC0LTL = 0x0080 (128d) and ADC0GTH:ADC0GTL = 0x0040 (64d). The input voltage can range from 0 to VREF x (1023/1024) with respect to GND, and is represented by a 10-bit unsigned integer value. In the left example, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:ADC0L) is within the range defined by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL (if 0x0040 < ADC0H:ADC0L < 0x0080). In the right example, and AD0WINT interrupt will be generated if the ADC0 conversion word is outside of the range defined by the ADC0GT and ADC0LT registers (if ADC0H:ADC0L < 0x0040 or ADC0H:ADC0L > 0x0080). Figure 5.6 shows an example using left-justified data with the same comparison values.

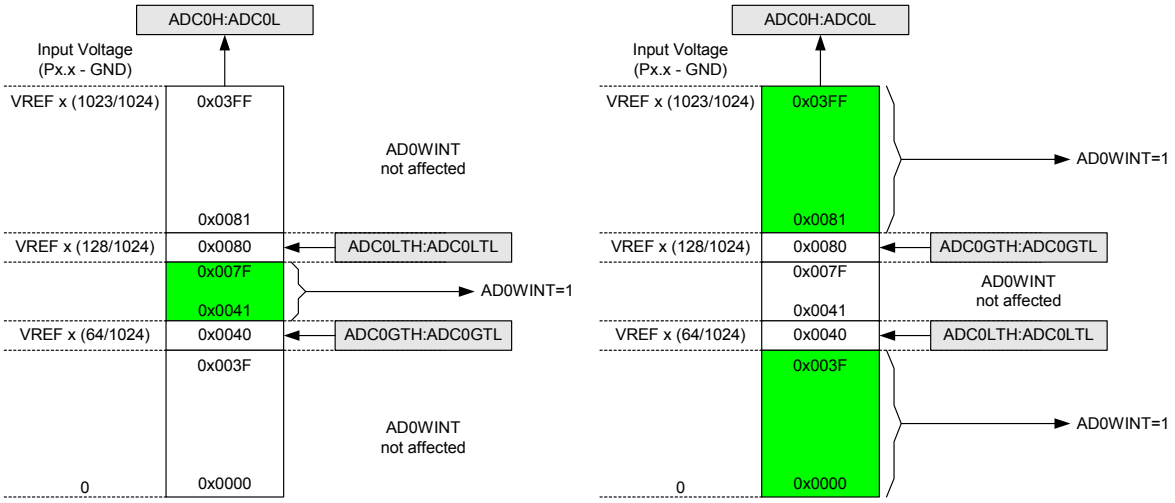



**Figure 5.5. ADC Window Compare Example: Right-Justified Single-Ended Data**

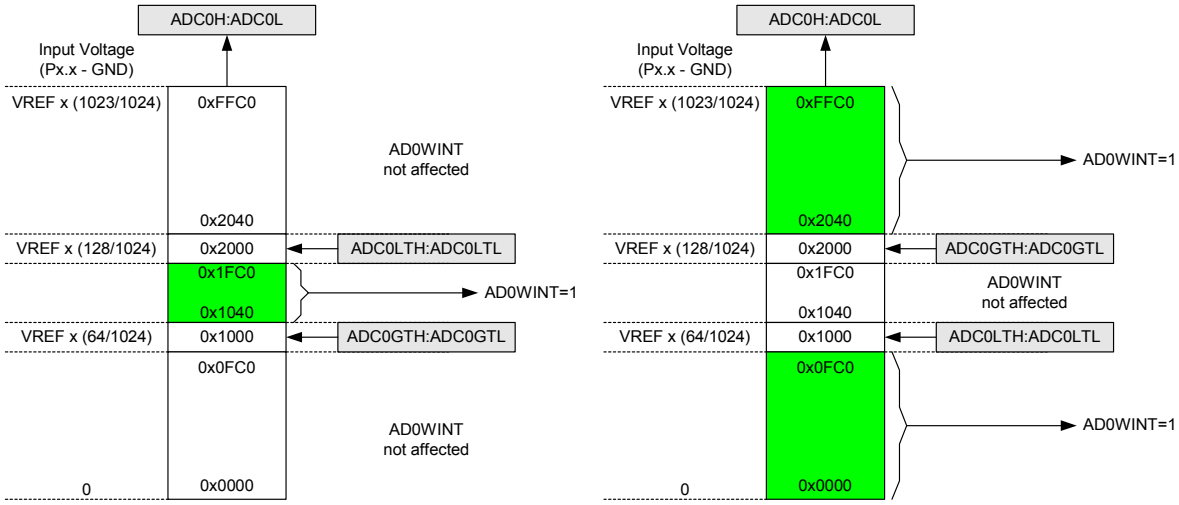



**Figure 5.6. ADC Window Compare Example: Left-Justified Single-Ended Data**

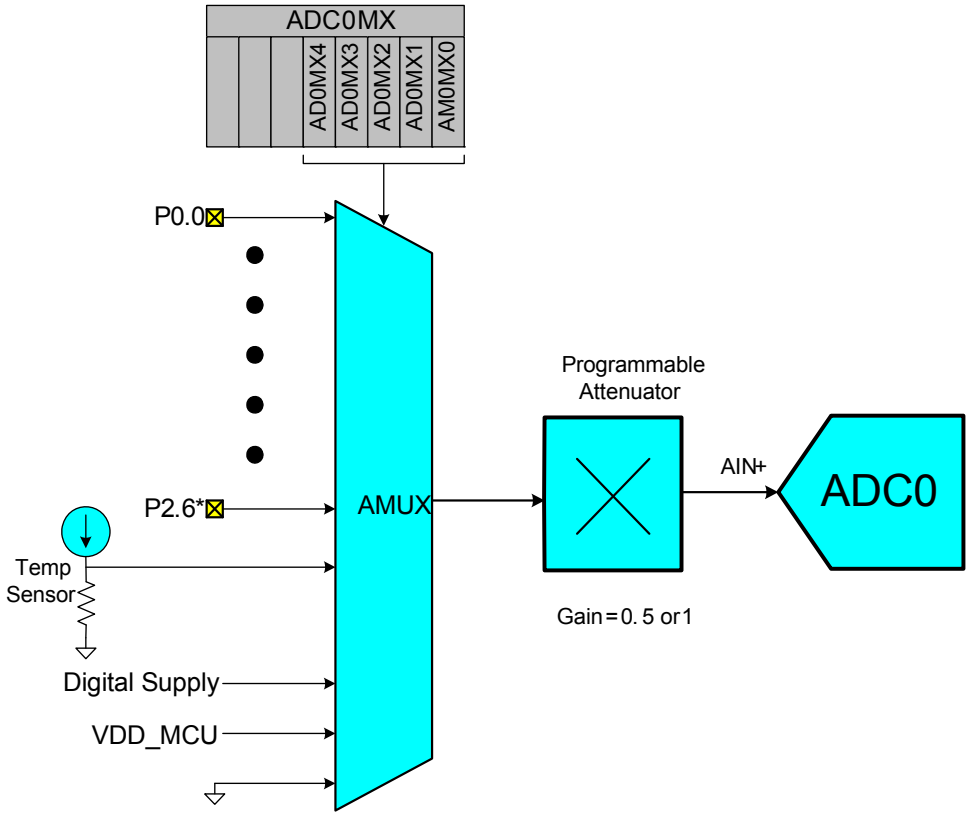
### 5.4.2. ADC0 Specifications

See “4. Electrical Characteristics” on page 40 for a detailed listing of ADC0 specifications.

## 5.5. ADC0 Analog Multiplexer

ADC0 on Si1000/1/2/3/4/5 has an analog multiplexer, referred to as AMUX0.

AMUX0 selects the positive inputs to the single-ended ADC0. Any of the following may be selected as the positive input: Port I/O pins, the on-chip temperature sensor, Regulated Digital Supply Voltage (Output of VREG0), VDD_MCU Supply, or the positive input may be connected to GND. The ADC0 input channels are selected in the ADC0MX register described in SFR Definition 5.12.




*P1.0 – P1.4 are not available as device pins.
P2.4 – P2.6 are only available on Si1000/1/2/3 devices

**Figure 5.7. ADC0 Multiplexer Block Diagram**

**Important Note About ADC0 Input Configuration:** Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to 0 the corresponding bit in register PnMDIN and disable the digital driver (PnMDOUT = 0 and Port Latch = 1). To force the Crossbar to skip a Port pin, set to 1 the corresponding bit in register PnSKIP. See Section “21. Port Input/Output” on page 210 for more Port I/O configuration details.

## SFR Definition 5.12. ADC0MX: ADC0 Input Channel Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | AD0MX | | | | |
| Type | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xBB

| Bit | Name | Function |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't Care. |
| 4:0 | AD0MX | **AMUX0 Positive Input Selection.**<br>Selects the positive input channel for ADC0. |

| | | | |
|---|---|---|---|
| 00000: | P0.0 | 10000: | P2.0 |
| 00001: | P0.1 | 10001: | P2.1 |
| 00010: | P0.2 | 10010: | P2.2 |
| 00011: | P0.3 | 10011: | P2.3 |
| 00100: | P0.4 | 10100: | P2.4 |
| 00101: | P0.5 | 10101: | P2.5 |
| 00110: | P0.6 | 10110: | P2.6 |
| 00111: | P0.7 | 10111: | Reserved. |
| 01000: | Reserved. | 11000: | Reserved. |
| 01001: | Reserved. | 11001: | Reserved. |
| 01010: | Reserved. | 11010: | Reserved. |
| 01011: | Reserved. | 11011: | Temperature Sensor* |
| 01100: | Reserved. | 11100: | VDD_MCU Supply Voltage (1.8–3.6 V) |
| 01101: | P1.5 | | |
| 01110: | P1.6 | 11101: | Digital Supply Voltage (VREG0 Output, 1.7 V Typical) |
| 01111: | P1.7 | | |
| | | 11110: | VDD_MCU Supply Voltage (1.8–3.6 V) |
| | | 11111: | Ground |

***Note:** Before switching the ADC multiplexer from another channel to the temperature sensor, the ADC mux should select the "Ground" channel as an intermediate step. The intermediate "Ground" channel selection step will discharge any voltage on the ADC sampling capacitor from the previous channel selection. This will prevent the possibility of a high voltage (> 2 V) being presented to the temperature sensor circuit, which can otherwise impact its long-term reliability.

## 5.6. Temperature Sensor

An on-chip temperature sensor is included on the Si1000/1/2/3/4/5 which can be directly accessed via the ADC multiplexer in single-ended configuration. To use the ADC to measure the temperature sensor, the ADC mux channel should select the temperature sensor. The temperature sensor transfer function is shown in Figure 5.8. The output voltage ($V_{TEMP}$) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register REF0CN enables/disables the temperature sensor, as described in SFR Definition 5.15. While disabled, the temperature sensor defaults to a high impedance state and any ADC measurements performed on the sensor will result in meaningless data. Refer to Table 4.9 for the slope and offset parameters of the temperature sensor.

**Note:** Before switching the ADC multiplexer from another channel to the temperature sensor, the ADC mux should select the "Ground" channel as an intermediate step. The intermediate "Ground" channel selection step will discharge any voltage on the ADC sampling capacitor from the previous channel selection. This will prevent the possibility of a high voltage (> 2 V) being presented to the temperature sensor circuit, which can otherwise impact its long-term reliability.

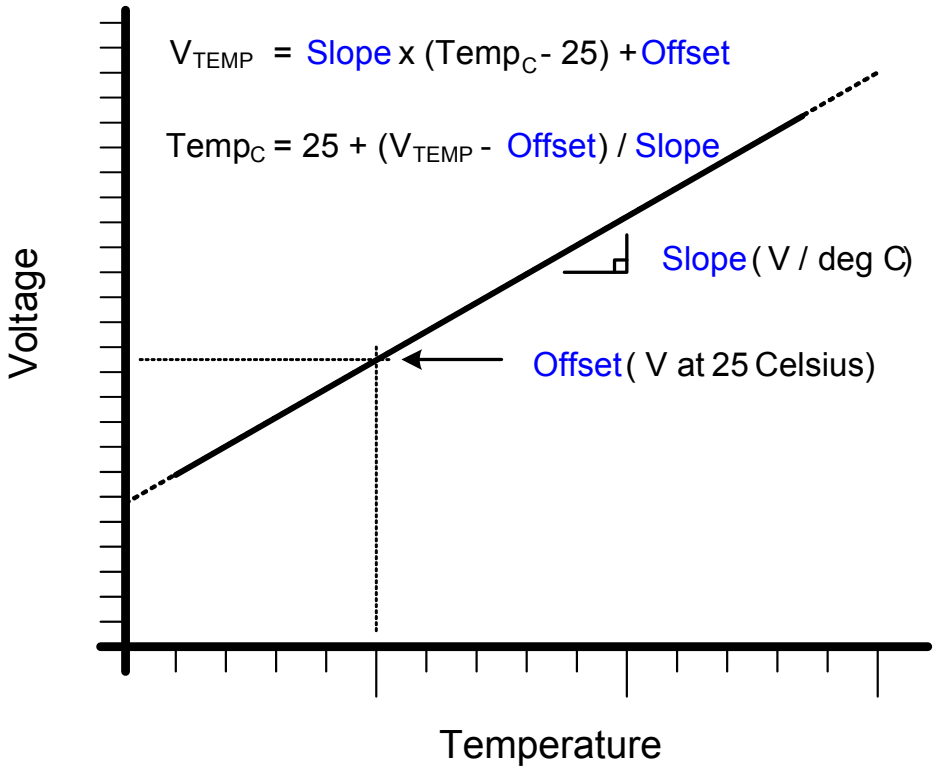



**Figure 5.8. Temperature Sensor Transfer Function**

### 5.6.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 4.10 for linearity specifications). For absolute temperature measurements, offset and/or gain calibration is recommended. Typically a 1-point (offset) calibration includes the following steps:

1. Control/measure the ambient temperature (this temperature must be known).
2. Power the device, and delay for a few seconds to allow for self-heating.
3. Perform an ADC conversion with the temperature sensor selected as the positive input and GND

selected as the negative input.

4. Calculate the offset characteristics, and store this value in non-volatile memory for use with subsequent temperature sensor measurements.

Figure 5.9 shows the typical temperature sensor error assuming a 1-point calibration at 25 °C. **Parameters that affect ADC measurement, in particular the voltage reference value, will also affect temperature measurement.**

A single-point offset measurement of the temperature sensor is performed on each device during production test. The measurement is performed at 25 °C ±5 °C, using the ADC with the internal high speed reference buffer selected as the Voltage Reference. The direct ADC result of the measurement is stored in the SFR registers TOFFH and TOFFL, shown in SFR Definition 5.13 and SFR Definition 5.14.

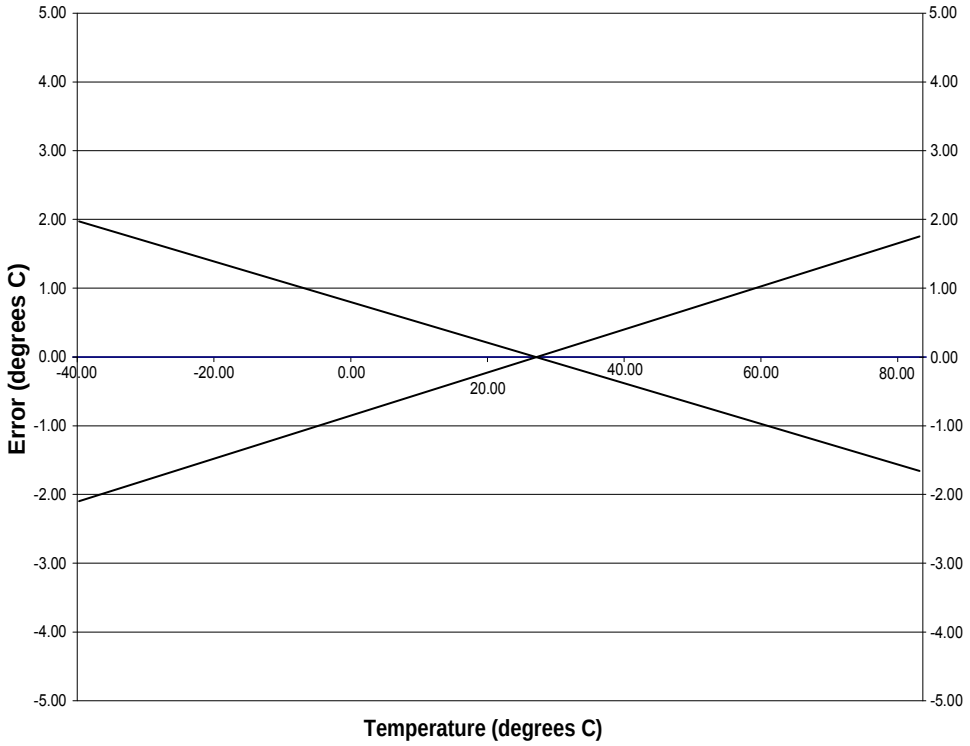



**Figure 5.9. Temperature Sensor Error with 1-Point Calibration ($V_{REF}$ = 1.68 V)**

## SFR Definition 5.13. TOFFH: ADC0 Data Word High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TOFF[9:2] | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Reset | Varies | Varies | Varies | Varies | Varies | Varies | Varies | Varies |

SFR Page = 0xF; SFR Address = 0x86

| Bit | Name | Function |
|---|---|---|
| 7:0 | TOFF[9:2] | **Temperature Sensor Offset High Bits.**<br>Most Significant Bits of the 10-bit temperature sensor offset measurement. |

## SFR Definition 5.14. TOFFL: ADC0 Data Word Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TOFF[1:0] | | | | | | | |
| Type | R | R | | | | | | |
| Reset | Varies | Varies | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x85

| Bit | Name | Function |
|---|---|---|
| 7:6 | TOFF[1:0] | **Temperature Sensor Offset Low Bits.**<br>Least Significant Bits of the 10-bit temperature sensor offset measurement. |
| 5:0 | Unused | Read = 0; Write = Don't Care. |

## 5.7. Voltage and Ground Reference Options

The voltage reference MUX is configurable to use an externally connected voltage reference, one of two internal voltage references, or one of two power supply voltages (see Figure 5.10). The ground reference MUX allows the ground reference for ADC0 to be selected between the ground pin (GND) or a port pin dedicated to analog ground (P0.1/AGND).

The voltage and ground reference options are configured using the REF0CN SFR described on page 98. Electrical specifications are can be found in the Electrical Specifications Chapter.

**Important Note About the $V_{REF}$ and AGND Inputs:** Port pins are used as the external $V_{REF}$ and AGND inputs. When using an external voltage reference or the internal precision reference, P0.0/VREF should be configured as an analog input and skipped by the Digital Crossbar. When using AGND as the ground reference to ADC0, P0.1/AGND should be configured as an analog input and skipped by the Digital Crossbar. Refer to Section “21. Port Input/Output” on page 210 for complete Port I/O configuration details. The external reference voltage must be within the range $0 \le V_{REF} \le$ VDD_MCU and the external ground reference must be at the same DC voltage potential as GND.

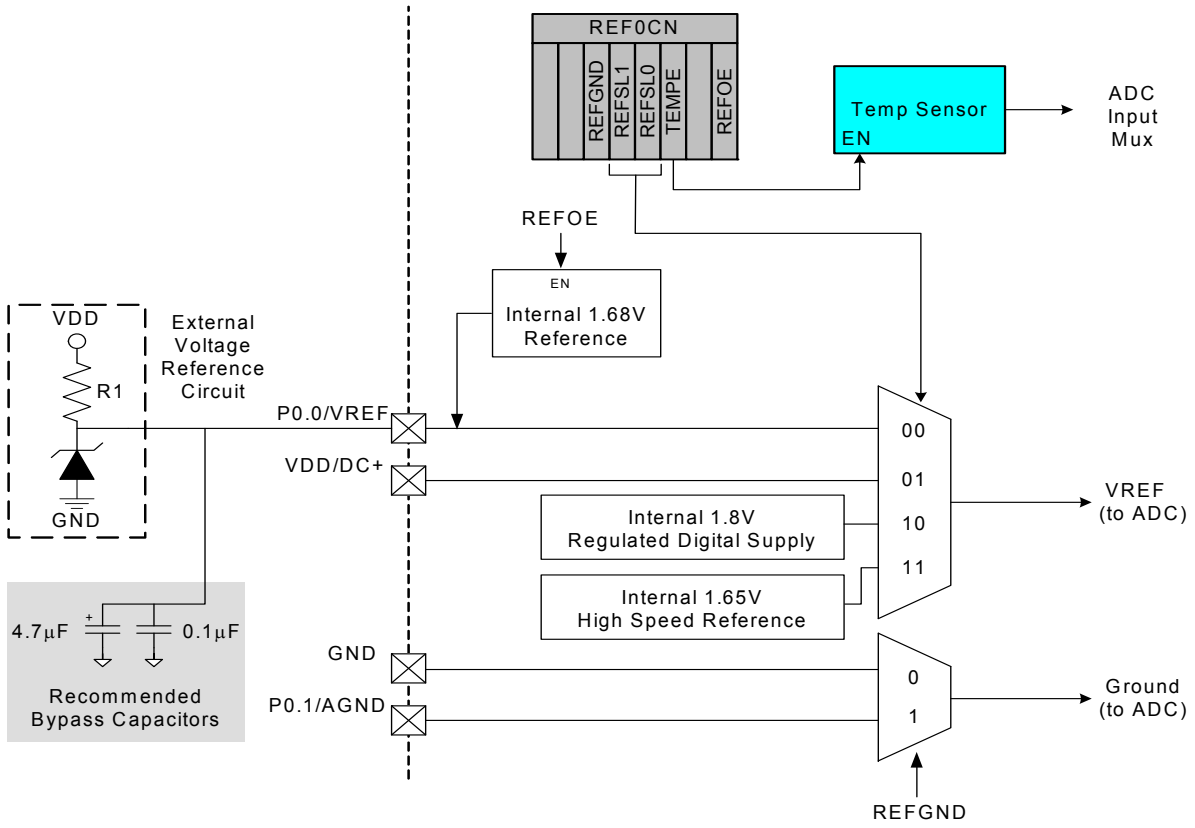



**Figure 5.10. Voltage Reference Functional Block Diagram**

## 5.8. External Voltage References

To use an external voltage reference, REFSL[1:0] should be set to 00 and the internal 1.68 V precision reference should be disabled by setting REFOE to 0. Bypass capacitors should be added as recommended by the manufacturer of the external voltage reference.

## 5.9. Internal Voltage References

For applications requiring the maximum number of port I/O pins, or very short VREF turn-on time, the 1.65 V high-speed reference will be the best internal reference option to choose. The high speed internal reference is selected by setting REFSL[1:0] to 11. When selected, the high speed internal reference will be automatically enabled/disabled on an as-needed basis by ADC0.

For applications requiring the highest absolute accuracy, the 1.68 V precision voltage reference will be the best internal reference option to choose. The 1.68 V precision reference may be enabled and selected by setting REFOE to 1 and REFSL[1:0] to 00. An external capacitor of at least 0.1 µF is recommended when using the precision voltage reference.

In applications that leave the precision internal oscillator always running, there is no additional power required to use the precision voltage reference. In all other applications, using the high speed reference will result in lower overall power consumption due to its minimal startup time and the fact that it remains in a low power state when an ADC conversion is not taking place.

Note: When using the precision internal oscillator as the system clock source, the precision voltage reference should not be enabled from a disabled state. To use the precision oscillator and the precision voltage reference simultaneously, the precision voltage reference should be enabled first and allowed to settle to its final value (charging the external capacitor) before the precision oscillator is started and selected as the system clock.

For applications with a non-varying power supply voltage, using the power supply as the voltage reference can provide ADC0 with added dynamic range at the cost of reduced power supply noise rejection. To use the 1.8 to 3.6 V power supply voltage (VDD_MCU) or the 1.8 V regulated digital supply voltage as the reference source, REFSL[1:0] should be set to 01 or 10, respectively.

## 5.10. Analog Ground Reference

To prevent ground noise generated by switching digital logic from affecting sensitive analog measurements, a separate analog ground reference option is available. When enabled, the ground reference for ADC0 during both the tracking/sampling and the conversion periods is taken from the P0.1/AGND pin. Any external sensors sampled by ADC0 should be referenced to the P0.1/AGND pin. This pin should be connected to the ground terminal of any external sensors sampled by ADC0. If an external voltage reference is used, the P0.1/AGND pin should be connected to the ground of the external reference and its associated decoupling capacitor. If the 1.68 V precision internal reference is used, then P0.1/AGND should be connected to the ground terminal of its external decoupling capacitor. The separate analog ground reference option is enabled by setting REFGND to 1. Note that when sampling the internal temperature sensor, the internal chip ground is always used for the sampling operation, regardless of the setting of the REFGND bit. Similarly, whenever the internal 1.65 V high-speed reference is selected, the internal chip ground is always used during the conversion period, regardless of the setting of the REFGND bit.

## 5.11. Temperature Sensor Enable

The TEMPE bit in register REF0CN enables/disables the temperature sensor. While disabled, the temperature sensor defaults to a high impedance state and any ADC0 measurements performed on the sensor result in meaningless data. See Section “5.6. Temperature Sensor” on page 93 for details on temperature sensor characteristics when it is enabled.

## SFR Definition 5.15. REF0CN: Voltage Reference Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | REFGND | REFSL | | TEMPE | | REFOE |
| Type | R | R | R/W | R/W | R/W | R/W | R | R/W |
| Reset | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD1

| Bit | Name | Function |
|---|---|---|
| 7:6 | Unused | Read = 00b; Write = Don't Care. |
| 5 | REFGND | **Analog Ground Reference.**<br>Selects the ADC0 ground reference.<br>0: The ADC0 ground reference is the GND pin.<br>1: The ADC0 ground reference is the P0.1/AGND pin. |
| 4:3 | REFSL | **Voltage Reference Select.**<br>Selects the ADC0 voltage reference.<br>00: The ADC0 voltage reference is the P0.0/VREF pin.<br>01: The ADC0 voltage reference is the VDD_MCU pin.<br>10: The ADC0 voltage reference is the internal 1.8 V digital supply voltage.<br>11: The ADC0 voltage reference is the internal 1.65 V high speed voltage reference. |
| 2 | TEMPE | **Temperature Sensor Enable.**<br>Enables/Disables the internal temperature sensor.<br>0: Temperature Sensor Disabled.<br>1: Temperature Sensor Enabled. |
| 1 | Unused | Read = 0b; Write = Don't Care. |
| 0 | REFOE | **Internal Voltage Reference Output Enable.**<br>Connects/Disconnects the internal voltage reference to the P0.0/VREF pin.<br>0: Internal 1.68 V Precision Voltage Reference disabled and not connected to P0.0/VREF.<br>1: Internal 1.68 V Precision Voltage Reference enabled and connected to P0.0/VREF. |

## 5.12. Voltage Reference Electrical Specifications

See Table 4.11 on page 60 for detailed Voltage Reference Electrical Specifications.

# 6. Programmable Current Reference (IREF0)

Si1000/1/2/3/4/5 devices include an on-chip programmable current reference (source or sink) with two output current settings: Low Power Mode and High Current Mode. The maximum current output in Low Power Mode is 63 µA (1 µA steps) and the maximum current output in High Current Mode is 504 µA (8 µA steps).

The current source/sink is controlled though the IREF0CN special function register. It is enabled by setting the desired output current to a non-zero value. It is disabled by writing 0x00 to IREF0CN. The port I/O pin associated with ISRC0 should be configured as an analog input and skipped in the Crossbar. See Section "21. Port Input/Output" on page 210 for more details.

## SFR Definition 6.1. IREF0CN: Current Reference Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SINK | MODE | IREF0DAT | | | | | |
| Type | R/W | R/W | R/W | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB9

| Bit | Name | Function |
|---|---|---|
| 7 | SINK | **IREF0 Current Sink Enable.**<br>Selects if IREF0 is a current source or a current sink.<br>0: IREF0 is a current source.<br>1: IREF0 is a current sink. |
| 6 | MDSEL | **IREF0 Output Mode Select.**<br>Selects Low Power or High Current Mode.<br>0: Low Power Mode is selected (step size = 1 µA).<br>1: High Current Mode is selected (step size = 8 µA). |
| 5:0 | IREF0DAT[5:0] | **IREF0 Data Word.**<br>Specifies the number of steps required to achieve the desired output current.<br>Output current = direction x step size x IREF0DAT.<br>IREF0 is in a low power state when IREF0DAT is set to 0x00. |

## 6.1. IREF0 Specifications

See Table 4.12 on page 61 for a detailed listing of IREF0 specifications.

# 7. Comparators

Si1000/1/2/3/4/5 devices include two on-chip programmable voltage comparators: Comparator 0 (CPT0) is shown in Figure 7.1; Comparator 1 (CPT1) is shown in Figure 7.2. The two comparators operate identically, but may differ in their ability to be used as reset or wake-up sources. See the Reset Sources chapter and the Power Management chapter for details on reset sources and low power mode wake-up sources, respectively.

The Comparator offers programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0, CP1), or an asynchronous “raw” output (CP0A, CP1A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the Comparator to operate and generate an output when the device is in some low power modes.

## 7.1. Comparator Inputs

Each Comparator performs an analog comparison of the voltage levels at its positive (CP0+ or CP1+) and negative (CP0- or CP1-) input. Both comparators support multiple port pin inputs multiplexed to their positive and negative comparator inputs using analog input multiplexers. The analog input multiplexers are completely under software control and configured using SFR registers. See Section “7.6. Comparator0 and Comparator1 Analog Multiplexers” on page 107 for details on how to select and configure Comparator inputs.

**Important Note About Comparator Inputs:** The Port pins selected as Comparator inputs should be configured as analog inputs and skipped by the Crossbar. See the Port I/O chapter for more details on how to configure Port I/O pins as Analog Inputs. The Comparator may also be used to compare the logic level of digital signals, however, Port I/O pins configured as digital inputs must be driven to a valid logic state (HIGH or LOW) to avoid increased power consumption.

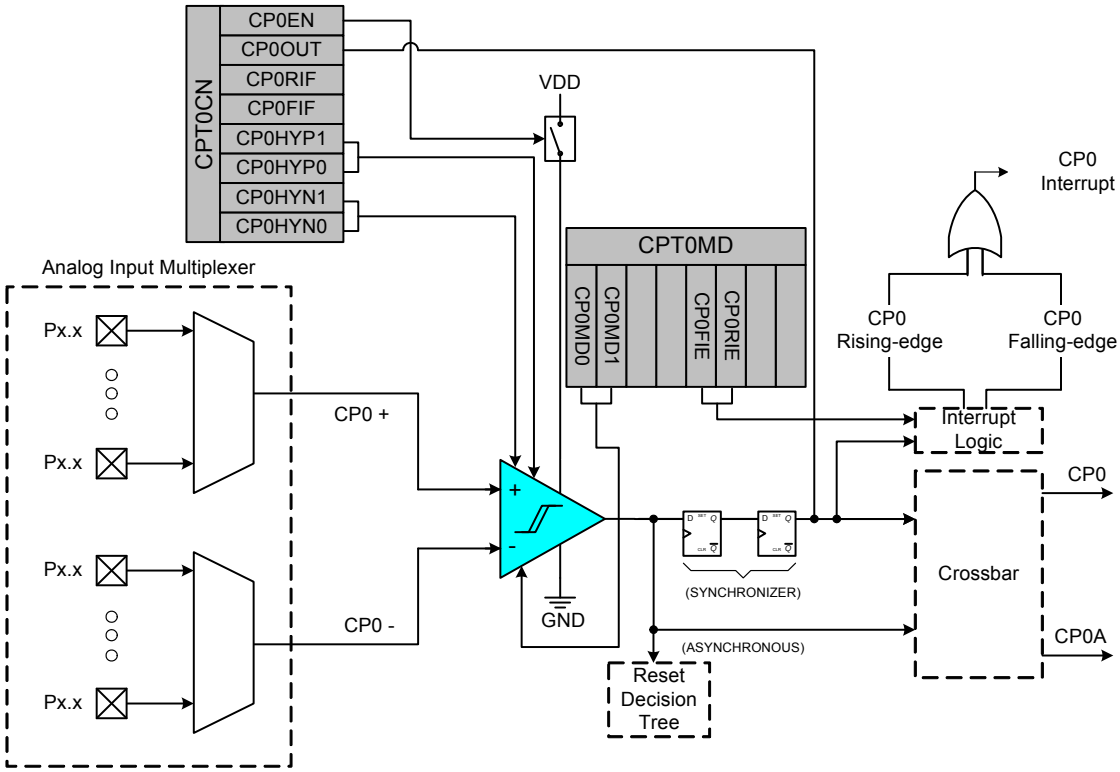



**Figure 7.1. Comparator 0 Functional Block Diagram**

## 7.2. Comparator Outputs

When a comparator is enabled, its output is a logic 1 if the voltage at the positive input is higher than the voltage at the negative input. When disabled, the comparator output is a logic 0. The comparator output is synchronized with the system clock as shown in Figure 7.2. The synchronous “latched” output (CP0, CP1) can be polled in software (CPnOUT bit), used as an interrupt source, or routed to a Port pin through the Crossbar.

The asynchronous “raw” comparator output (CP0A, CP1A) is used by the low power mode wakeup logic and reset decision logic. See the Power Options chapter and the Reset Sources chapter for more details on how the asynchronous comparator outputs are used to make wake-up and reset decisions. The asynchronous comparator output can also be routed directly to a Port pin through the Crossbar, and is available for use outside the device even if the system clock is stopped.

When using a Comparator as an interrupt source, Comparator interrupts can be generated on rising-edge and/or falling-edge comparator output transitions. Two independent interrupt flags (CPnRIF and CPnFIF) allow software to determine which edge caused the Comparator interrupt. The comparator rising-edge and falling-edge interrupt flags are set by hardware when a corresponding edge is detected regardless of the interrupt enable state. Once set, these bits remain set until cleared by software.

The rising-edge and falling-edge interrupts can be individually enabled using the CPnRIE and CPnFIE interrupt enable bits in the CPTnMD register. In order for the CPnRIF and/or CPnFIF interrupt flags to generate an interrupt request to the CPU, the Comparator must be enabled as an interrupt source and global interrupts must be enabled. See the Interrupt Handler chapter for additional information.

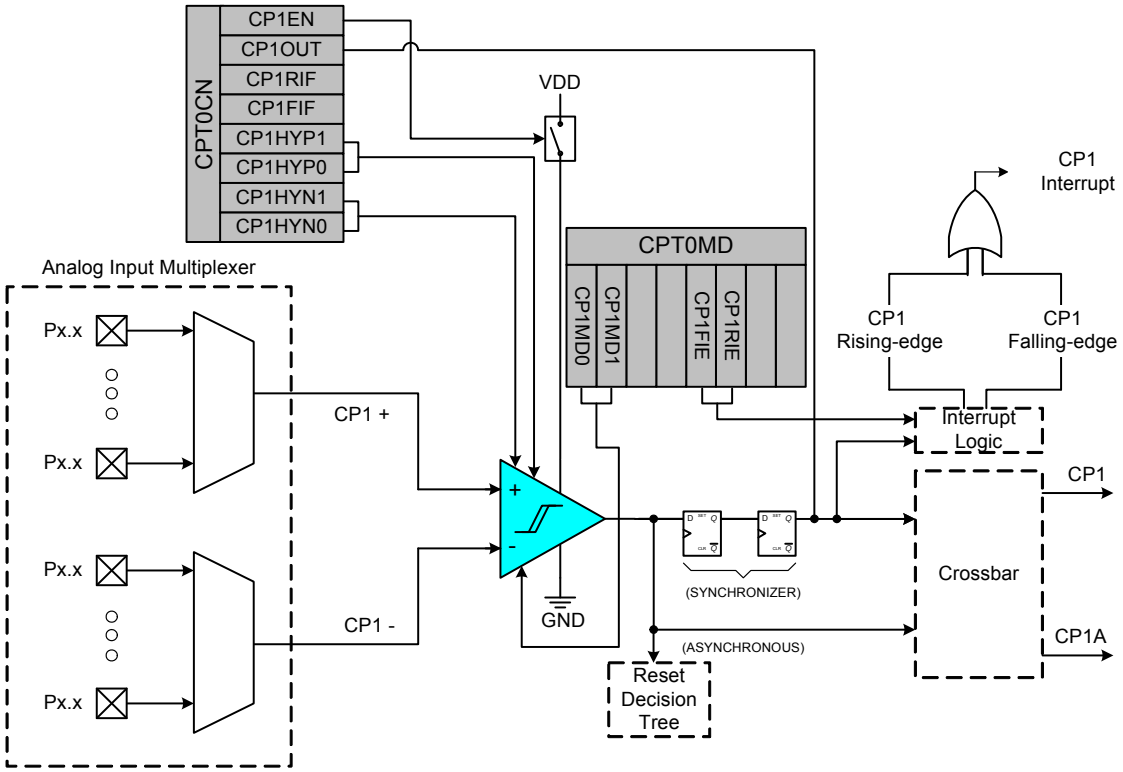



**Figure 7.2. Comparator 1 Functional Block Diagram**

## 7.3. Comparator Response Time

Comparator response time may be configured in software via the CPTnMD registers described on "CPT0MD: Comparator 0 Mode Selection" on page 104 and "CPT1MD: Comparator 1 Mode Selection" on page 106. Four response time settings are available: Mode 0 (Fastest Response Time), Mode 1, Mode 2, and Mode 3 (Lowest Power). Selecting a longer response time reduces the Comparator active supply current. The Comparators also have low power shutdown state, which is entered any time the comparator is disabled. Comparator rising edge and falling edge response times are typically not equal. See Table 4.13 on page 62 for complete comparator timing and supply current specifications.

## 7.4. Comparator Hysteresis

The Comparators feature software-programmable hysteresis that can be used to stabilize the comparator output while a transition is occurring on the input. Using the CPTnCN registers, the user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage (i.e., the comparator negative input).

Figure 7.3 shows that when positive hysteresis is enabled, the comparator output does not transition from logic 0 to logic 1 until the comparator positive input voltage has exceeded the threshold voltage by an amount equal to the programmed hysteresis. It also shows that when negative hysteresis is enabled, the comparator output does not transition from logic 1 to logic 0 until the comparator positive input voltage has fallen below the threshold voltage by an amount equal to the programmed hysteresis.

The amount of positive hysteresis is determined by the settings of the CPnHYP bits in the CPTnCN register and the amount of negative hysteresis voltage is determined by the settings of the CPnHYN bits in the same register. Settings of 20, 10, 5, or 0 mV can be programmed for both positive and negative hysteresis. See Section "Table 4.13. Comparator Electrical Characteristics" on page 62 for complete comparator hysteresis specifications.

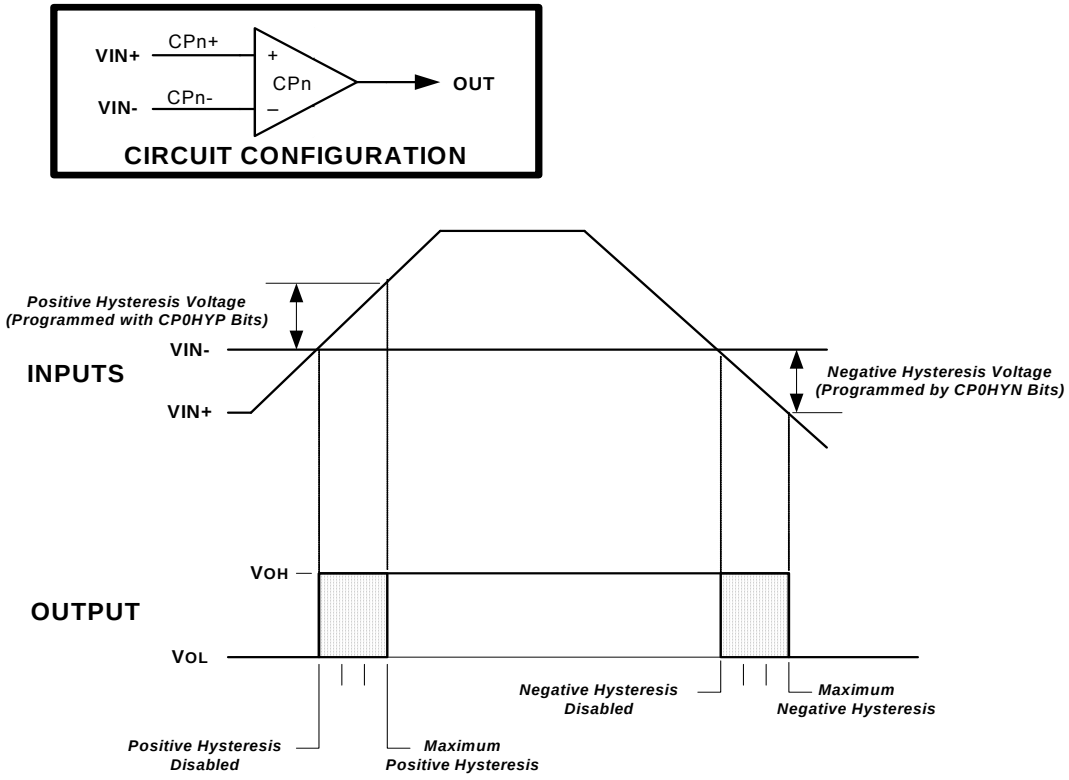



**Figure 7.3. Comparator Hysteresis Plot**

## 7.5. Comparator Register Descriptions

The SFRs used to enable and configure the comparators are described in the following register descriptions. A Comparator must be enabled by setting the CPnEN bit to logic 1 before it can be used. From an enabled state, a comparator can be disabled and placed in a low power state by clearing the CPnEN bit to logic 0.

**Important Note About Comparator Settings:** False rising and falling edges can be detected by the Comparator while powering on or if changes are made to the hysteresis or response time control bits. Therefore, it is recommended that the rising-edge and falling-edge flags be explicitly cleared to logic 0 a short time after the comparator is enabled or its mode bits have been changed. The Comparator Power Up Time is specified in Section “Table 4.13. Comparator Electrical Characteristics” on page 62.

### SFR Definition 7.1. CPT0CN: Comparator 0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CP0EN | CP0OUT | CP0RIF | CP0FIF | CP0HYP[1:0] | | CP0HYN[1:0] | |
| Type | R/W | R | R/W | R/W | R/W | | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0x9B

| Bit | Name | Function |
|---|---|---|
| 7 | CP0EN | **Comparator0 Enable Bit.**<br>0: Comparator0 Disabled.<br>1: Comparator0 Enabled. |
| 6 | CP0OUT | **Comparator0 Output State Flag.**<br>0: Voltage on CP0+ < CP0–.<br>1: Voltage on CP0+ > CP0–. |
| 5 | CP0RIF | **Comparator0 Rising-Edge Flag. Must be cleared by software.**<br>0: No Comparator0 Rising Edge has occurred since this flag was last cleared.<br>1: Comparator0 Rising Edge has occurred. |
| 4 | CP0FIF | **Comparator0 Falling-Edge Flag. Must be cleared by software.**<br>0: No Comparator0 Falling-Edge has occurred since this flag was last cleared.<br>1: Comparator0 Falling-Edge has occurred. |
| 3:2 | CP0HYP[1:0] | **Comparator0 Positive Hysteresis Control Bits.**<br>00: Positive Hysteresis Disabled.<br>01: Positive Hysteresis = 5 mV.<br>10: Positive Hysteresis = 10 mV.<br>11: Positive Hysteresis = 20 mV. |
| 1:0 | CP0HYN[1:0] | **Comparator0 Negative Hysteresis Control Bits.**<br>00: Negative Hysteresis Disabled.<br>01: Negative Hysteresis = 5 mV.<br>10: Negative Hysteresis = 10 mV.<br>11: Negative Hysteresis = 20 mV. |

## SFR Definition 7.2. CPT0MD: Comparator 0 Mode Selection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | CP0RIE | CP0FIE | | | CP0MD[1:0] | |
| Type | R/W | R | R/W | R/W | R | R | R/W | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

SFR Page = All Pages; SFR Address = 0x9D

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Read = 1b, Must Write 1b. |
| 6 | Unused | Read = 0b, Write = don't care. |
| 5 | CP0RIE | **Comparator0 Rising-Edge Interrupt Enable.**<br>0: Comparator0 Rising-edge interrupt disabled.<br>1: Comparator0 Rising-edge interrupt enabled. |
| 4 | CP0FIE | **Comparator0 Falling-Edge Interrupt Enable.**<br>0: Comparator0 Falling-edge interrupt disabled.<br>1: Comparator0 Falling-edge interrupt enabled. |
| 3:2 | Unused | Read = 00b, Write = don't care. |
| 1:0 | CP0MD[1:0] | **Comparator0 Mode Select.**<br>These bits affect the response time and power consumption for Comparator0.<br>00: Mode 0 (Fastest Response Time, Highest Power Consumption)<br>01: Mode 1<br>10: Mode 2<br>11: Mode 3 (Slowest Response Time, Lowest Power Consumption) |

## SFR Definition 7.3. CPT1CN: Comparator 1 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CP1EN | CP1OUT | CP1RIF | CP1FIF | CP1HYP[1:0] | | CP1HYN[1:0] | |
| Type | R/W | R | R/W | R/W | R/W | | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0x9A

| Bit | Name | Function |
|---|---|---|
| 7 | CP1EN | **Comparator1 Enable Bit.**<br>0: Comparator1 Disabled.<br>1: Comparator1 Enabled. |
| 6 | CP1OUT | **Comparator1 Output State Flag.**<br>0: Voltage on CP1+ < CP1–.<br>1: Voltage on CP1+ > CP1–. |
| 5 | CP1RIF | **Comparator1 Rising-Edge Flag. Must be cleared by software.**<br>0: No Comparator1 Rising Edge has occurred since this flag was last cleared.<br>1: Comparator1 Rising Edge has occurred. |
| 4 | CP1FIF | **Comparator1 Falling-Edge Flag. Must be cleared by software.**<br>0: No Comparator1 Falling-Edge has occurred since this flag was last cleared.<br>1: Comparator1 Falling-Edge has occurred. |
| 3:2 | CP1HYP[1:0] | **Comparator1 Positive Hysteresis Control Bits.**<br>00: Positive Hysteresis Disabled.<br>01: Positive Hysteresis = 5 mV.<br>10: Positive Hysteresis = 10 mV.<br>11: Positive Hysteresis = 20 mV. |
| 1:0 | CP1HYN[1:0] | **Comparator1 Negative Hysteresis Control Bits.**<br>00: Negative Hysteresis Disabled.<br>01: Negative Hysteresis = 5 mV.<br>10: Negative Hysteresis = 10 mV.<br>11: Negative Hysteresis = 20 mV. |

## SFR Definition 7.4. CPT1MD: Comparator 1 Mode Selection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | CP1RIE | CP1FIE | | | CP1MD[1:0] | |
| Type | R/W | R | R/W | R/W | R | R | R/W | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

SFR Page = 0x0; SFR Address = 0x9C

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Read = 1b, Must Write 1b. |
| 6 | Unused | Read = 00b, Write = don't care. |
| 5 | CP1RIE | **Comparator1 Rising-Edge Interrupt Enable.**<br>0: Comparator1 Rising-edge interrupt disabled.<br>1: Comparator1 Rising-edge interrupt enabled. |
| 4 | CP1FIE | **Comparator1 Falling-Edge Interrupt Enable.**<br>0: Comparator1 Falling-edge interrupt disabled.<br>1: Comparator1 Falling-edge interrupt enabled. |
| 3:2 | Unused | Read = 00b, Write = don't care. |
| 1:0 | CP1MD[1:0] | **Comparator1 Mode Select**<br>These bits affect the response time and power consumption for Comparator1.<br>00: Mode 0 (Fastest Response Time, Highest Power Consumption)<br>01: Mode 1<br>10: Mode 2<br>11: Mode 3 (Slowest Response Time, Lowest Power Consumption) |

## 7.6. Comparator0 and Comparator1 Analog Multiplexers

Comparator0 and Comparator1 on Si1000/1/2/3/4/5 devices have analog input multiplexers to connect Port I/O pins and internal signals the comparator inputs; CP0+/CP0- are the positive and negative input multiplexers for Comparator0 and CP1+/CP1- are the positive and negative input multiplexers for Comparator1.

The comparator input multiplexers directly support capacitive touch switches. When the Capacitive Touch Sense Compare input is selected on the positive or negative multiplexer, any Port I/O pin connected to the other multiplexer can be directly connected to a capacitive touch switch with no additional external components. The Capacitive Touch Sense Compare provides the appropriate reference level for detecting when the capacitive touch switches have charged or discharged through the on-chip Rsense resistor. The Comparator outputs can be routed to Timer2 or Timer3 for capturing sense capacitor's charge and discharge time. See Section "27. Timers" on page 335 for details. See Application Note AN338 for details on Capacitive Touch Switch sensing.

Any of the following may be selected as comparator inputs: Port I/O pins, Capacitive Touch Sense Compare, VDD_MCU Supply Voltage, Regulated Digital Supply Voltage (Output of VREG0) or ground. The Comparator's supply voltage divided by 2 is also available as an input; the resistors used to divide the voltage only draw current when this setting is selected. The Comparator input multiplexers are configured using the CPT0MX and CPT1MX registers described in SFR Definition 7.5 and SFR Definition 7.6.

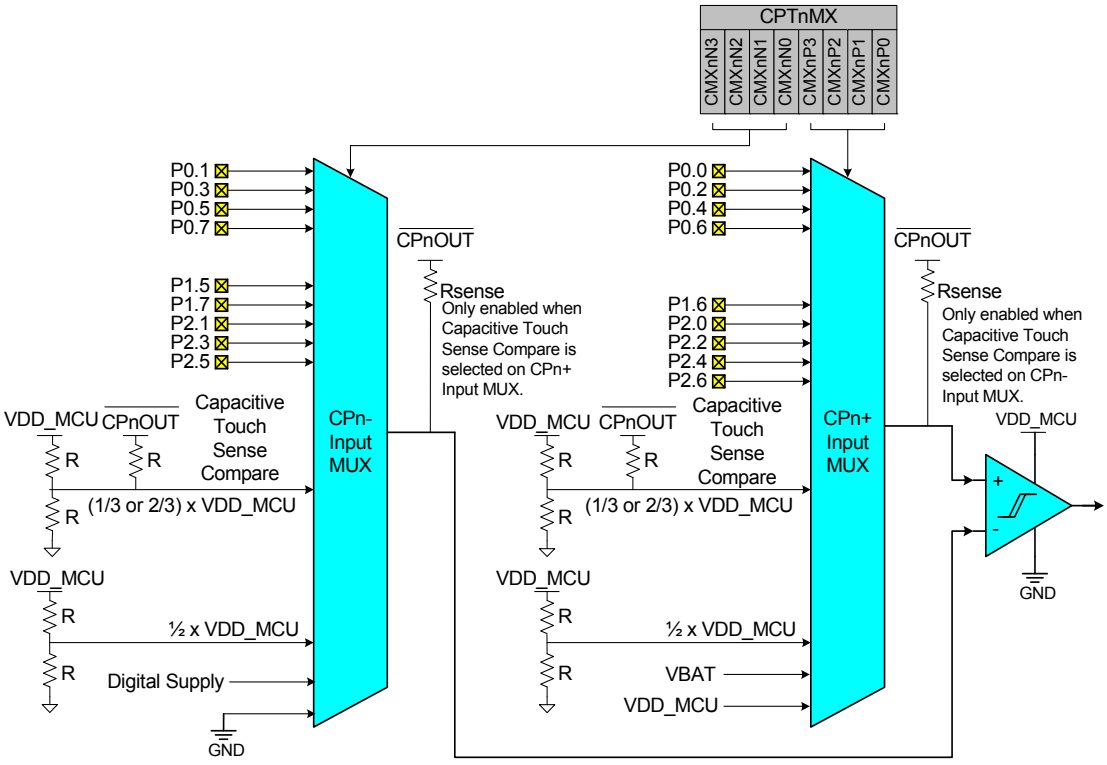



**Figure 7.4. CPn Multiplexer Block Diagram**

**Important Note About Comparator Input Configuration:** Port pins selected as comparator inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to 0 the corresponding bit in register PnMDIN and disable the digital driver (PnMDOUT = 0 and Port Latch = 1). To force the Crossbar to skip a Port pin, set to 1 the corresponding bit in register PnSKIP. See Section "21. Port Input/Output" on page 210 for more Port I/O configuration details.

## SFR Definition 7.5. CPT0MX: Comparator0 Input Channel Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CMX0N[3:0] | | | | CMX0P[3:0] | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0x9F

| Bit | Name | Function | | | |
|---|---|---|---|---|---|
| 7:4 | CMX0N | **Comparator0 Negative Input Selection.**<br>Selects the negative input channel for Comparator0. | | | |
| | | 0000: | P0.1 | 1000: | P2.1 |
| | | 0001: | P0.3 | 1001: | P2.3 |
| | | 0010: | P0.5 | 1010: | P2.5 |
| | | 0011: | P0.7 | 1011: | Reserved |
| | | 0100: | Reserved | 1100: | Capacitive Touch Sense Compare |
| | | 0101: | Reserved | 1101: | VDD_MCU divided by 2 |
| | | 0110: | P1.5 | 1110: | Digital Supply Voltage |
| | | 0111: | P1.7 | 1111: | Ground |
| 3:0 | CMX0P | **Comparator0 Positive Input Selection.**<br>Selects the positive input channel for Comparator0. | | | |
| | | 0000: | P0.0 | 1000: | P2.0 |
| | | 0001: | P0.2 | 1001: | P2.2 |
| | | 0010: | P0.4 | 1010: | P2.4 |
| | | 0011: | P0.6 | 1011: | P2.6 |
| | | 0100: | Reserved | 1100: | Capacitive Touch Sense Compare |
| | | 0101: | Reserved | 1101: | VDD_MCU divided by 2 |
| | | 0110: | Reserved | 1110: | VBAT Supply Voltage |
| | | 0111: | P1.6 | 1111: | VDD_MCU Supply Voltage |

## SFR Definition 7.6. CPT1MX: Comparator1 Input Channel Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CMX1N[3:0] | | | | CMX1P[3:0] | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0x9E

| Bit | Name | Function | | | |
|---|---|---|---|---|---|
| 7:4 | CMX1N | **Comparator1 Negative Input Selection.**<br>Selects the negative input channel for Comparator1. | | | |
| | | 0000: | P0.1 | 1000: | P2.1 |
| | | 0001: | P0.3 | 1001: | P2.3 |
| | | 0010: | P0.5 | 1010: | P2.5 |
| | | 0011: | P0.7 | 1011: | Reserved |
| | | 0100: | Reserved | 1100: | Capacitive Touch Sense Compare |
| | | 0101: | Reserved | 1101: | VDD_MCU divided by 2 |
| | | 0110: | P1.5 | 1110: | Digital Supply Voltage |
| | | 0111: | P1.7 | 1111: | Ground |
| 3:0 | CMX1P | **Comparator1 Positive Input Selection.**<br>Selects the positive input channel for Comparator1. | | | |
| | | 0000: | P0.0 | 1000: | P2.0 |
| | | 0001: | P0.2 | 1001: | P2.2 |
| | | 0010: | P0.4 | 1010: | P2.4 |
| | | 0011: | P0.6 | 1011: | P2.6 |
| | | 0100: | Reserved | 1100: | Capacitive Touch Sense Compare |
| | | 0101: | Reserved | 1101: | VDD_MCU divided by 2 |
| | | 0110: | Reserved | 1110: | VBAT Supply Voltage |
| | | 0111: | P1.6 | 1111: | VDD_MCU Supply Voltage |

# 8. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 29), and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 8.1 for a block diagram). The CIP-51 includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 25 MIPS Peak Throughput with 25 MHz Clock
- 0 to 25 MHz Clock Frequency
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- On-chip Debug Logic
- Program and Data Memory Security

## 8.1. Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. In a standard 8051, all instructions except for MUL and DIV take 12 or 24 system clock cycles to execute, and usually have a maximum system clock of 12 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with no instructions taking more than eight system clock cycles.

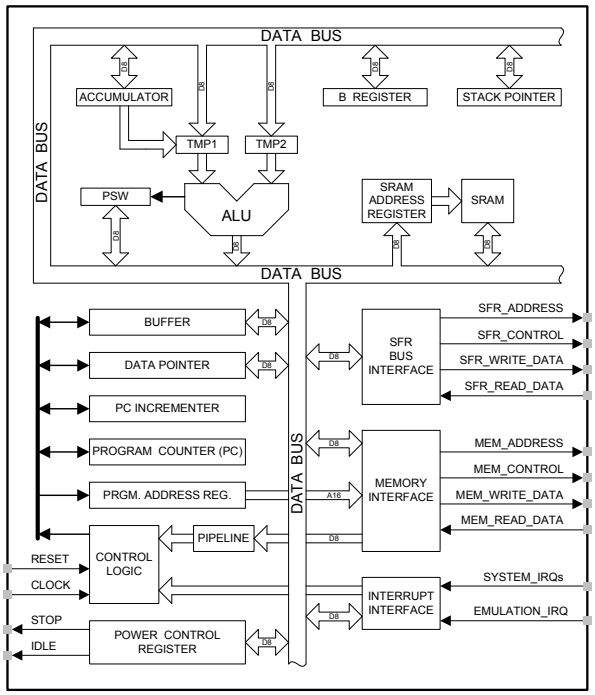



**Figure 8.1. CIP-51 Block Diagram**

With the CIP-51's maximum system clock at 25 MHz, it has a peak throughput of 25 MIPS. The CIP-51 has a total of 109 instructions. The table below shows the total number of instructions that require each execution time.

| Clocks to Execute | 1 | 2 | 2/3 | 3 | 3/4 | 4 | 4/5 | 5 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| Number of Instructions | 26 | 50 | 5 | 14 | 7 | 3 | 1 | 2 | 1 |

## 8.2. Programming and Debugging Support

In-system programming of the flash program memory and communication with on-chip debug support logic is accomplished via the Silicon Labs 2-Wire Development Interface (C2).

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debugging is completely non-intrusive, requiring no RAM, Stack, timers, or other on-chip resources. C2 details can be found in Section “29. Device Specific Behavior” on page 376.

The CIP-51 is supported by development tools from Silicon Labs and third party vendors. Silicon Labs provides an integrated development environment (IDE) including editor, debugger and programmer. The IDE's debugger and programmer interface to the CIP-51 via the C2 interface to provide fast and efficient in-system device programming and debugging. Third party macro assemblers and C compilers are also available.

## 8.3. Instruction Set

The instruction set of the CIP-51 System Controller is fully compatible with the standard MCS-51™ instruction set. Standard 8051 development tools can be used to develop software for the CIP-51. All CIP-51 instructions are the binary and functional equivalent of their MCS-51™ counterparts, including opcodes, addressing modes and effect on PSW flags. However, instruction timing is different than that of the standard 8051.

### 8.3.1. Instruction and CPU Timing

In many 8051 implementations, a distinction is made between machine cycles and clock cycles, with machine cycles varying from 2 to 12 clock cycles in length. However, the CIP-51 implementation is based solely on clock cycle timing. All instruction timings are specified in terms of clock cycles.

Due to the pipelined architecture of the CIP-51, most instructions execute in the same number of clock cycles as there are program bytes in the instruction. Conditional branch instructions take one less clock cycle to complete when the branch is not taken as opposed to when the branch is taken. Table 8.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

**Table 8.1. CIP-51 Instruction Set Summary**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| **Arithmetic Operations** | | | |
| ADD A, Rn | Add register to A | 1 | 1 |
| ADD A, direct | Add direct byte to A | 2 | 2 |
| ADD A, @Ri | Add indirect RAM to A | 1 | 2 |
| ADD A, #data | Add immediate to A | 2 | 2 |
| ADDC A, Rn | Add register to A with carry | 1 | 1 |
| ADDC A, direct | Add direct byte to A with carry | 2 | 2 |
| ADDC A, @Ri | Add indirect RAM to A with carry | 1 | 2 |
| ADDC A, #data | Add immediate to A with carry | 2 | 2 |
| SUBB A, Rn | Subtract register from A with borrow | 1 | 1 |
| SUBB A, direct | Subtract direct byte from A with borrow | 2 | 2 |
| SUBB A, @Ri | Subtract indirect RAM from A with borrow | 1 | 2 |
| SUBB A, #data | Subtract immediate from A with borrow | 2 | 2 |
| INC A | Increment A | 1 | 1 |
| INC Rn | Increment register | 1 | 1 |
| INC direct | Increment direct byte | 2 | 2 |
| INC @Ri | Increment indirect RAM | 1 | 2 |
| DEC A | Decrement A | 1 | 1 |
| DEC Rn | Decrement register | 1 | 1 |
| DEC direct | Decrement direct byte | 2 | 2 |
| DEC @Ri | Decrement indirect RAM | 1 | 2 |
| INC DPTR | Increment Data Pointer | 1 | 1 |
| MUL AB | Multiply A and B | 1 | 4 |
| DIV AB | Divide A by B | 1 | 8 |
| DA A | Decimal adjust A | 1 | 1 |
| **Logical Operations** | | | |
| ANL A, Rn | AND Register to A | 1 | 1 |
| ANL A, direct | AND direct byte to A | 2 | 2 |
| ANL A, @Ri | AND indirect RAM to A | 1 | 2 |
| ANL A, #data | AND immediate to A | 2 | 2 |
| ANL direct, A | AND A to direct byte | 2 | 2 |
| ANL direct, #data | AND immediate to direct byte | 3 | 3 |
| ORL A, Rn | OR Register to A | 1 | 1 |
| ORL A, direct | OR direct byte to A | 2 | 2 |
| ORL A, @Ri | OR indirect RAM to A | 1 | 2 |
| ORL A, #data | OR immediate to A | 2 | 2 |
| ORL direct, A | OR A to direct byte | 2 | 2 |
| ORL direct, #data | OR immediate to direct byte | 3 | 3 |
| XRL A, Rn | Exclusive-OR Register to A | 1 | 1 |
| XRL A, direct | Exclusive-OR direct byte to A | 2 | 2 |
| XRL A, @Ri | Exclusive-OR indirect RAM to A | 1 | 2 |
| XRL A, #data | Exclusive-OR immediate to A | 2 | 2 |
| XRL direct, A | Exclusive-OR A to direct byte | 2 | 2 |

**Table 8.1. CIP-51 Instruction Set Summary (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| XRL direct, #data | Exclusive-OR immediate to direct byte | 3 | 3 |
| CLR A | Clear A | 1 | 1 |
| CPL A | Complement A | 1 | 1 |
| RL A | Rotate A left | 1 | 1 |
| RLC A | Rotate A left through Carry | 1 | 1 |
| RR A | Rotate A right | 1 | 1 |
| RRC A | Rotate A right through Carry | 1 | 1 |
| SWAP A | Swap nibbles of A | 1 | 1 |
| **Data Transfer** | | | |
| MOV A, Rn | Move Register to A | 1 | 1 |
| MOV A, direct | Move direct byte to A | 2 | 2 |
| MOV A, @Ri | Move indirect RAM to A | 1 | 2 |
| MOV A, #data | Move immediate to A | 2 | 2 |
| MOV Rn, A | Move A to Register | 1 | 1 |
| MOV Rn, direct | Move direct byte to Register | 2 | 2 |
| MOV Rn, #data | Move immediate to Register | 2 | 2 |
| MOV direct, A | Move A to direct byte | 2 | 2 |
| MOV direct, Rn | Move Register to direct byte | 2 | 2 |
| MOV direct, direct | Move direct byte to direct byte | 3 | 3 |
| MOV direct, @Ri | Move indirect RAM to direct byte | 2 | 2 |
| MOV direct, #data | Move immediate to direct byte | 3 | 3 |
| MOV @Ri, A | Move A to indirect RAM | 1 | 2 |
| MOV @Ri, direct | Move direct byte to indirect RAM | 2 | 2 |
| MOV @Ri, #data | Move immediate to indirect RAM | 2 | 2 |
| MOV DPTR, #data16 | Load DPTR with 16-bit constant | 3 | 3 |
| MOVC A, @A+DPTR | Move code byte relative DPTR to A | 1 | 3 |
| MOVC A, @A+PC | Move code byte relative PC to A | 1 | 3 |
| MOVX A, @Ri | Move external data (8-bit address) to A | 1 | 3 |
| MOVX @Ri, A | Move A to external data (8-bit address) | 1 | 3 |
| MOVX A, @DPTR | Move external data (16-bit address) to A | 1 | 3 |
| MOVX @DPTR, A | Move A to external data (16-bit address) | 1 | 3 |
| PUSH direct | Push direct byte onto stack | 2 | 2 |
| POP direct | Pop direct byte from stack | 2 | 2 |
| XCH A, Rn | Exchange Register with A | 1 | 1 |
| XCH A, direct | Exchange direct byte with A | 2 | 2 |
| XCH A, @Ri | Exchange indirect RAM with A | 1 | 2 |
| XCHD A, @Ri | Exchange low nibble of indirect RAM with A | 1 | 2 |
| **Boolean Manipulation** | | | |
| CLR C | Clear Carry | 1 | 1 |
| CLR bit | Clear direct bit | 2 | 2 |
| SETB C | Set Carry | 1 | 1 |
| SETB bit | Set direct bit | 2 | 2 |
| CPL C | Complement Carry | 1 | 1 |
| CPL bit | Complement direct bit | 2 | 2 |

**Table 8.1. CIP-51 Instruction Set Summary (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| ANL C, bit | AND direct bit to Carry | 2 | 2 |
| ANL C, /bit | AND complement of direct bit to Carry | 2 | 2 |
| ORL C, bit | OR direct bit to carry | 2 | 2 |
| ORL C, /bit | OR complement of direct bit to Carry | 2 | 2 |
| MOV C, bit | Move direct bit to Carry | 2 | 2 |
| MOV bit, C | Move Carry to direct bit | 2 | 2 |
| JC rel | Jump if Carry is set | 2 | 2/3 |
| JNC rel | Jump if Carry is not set | 2 | 2/3 |
| JB bit, rel | Jump if direct bit is set | 3 | 3/4 |
| JNB bit, rel | Jump if direct bit is not set | 3 | 3/4 |
| JBC bit, rel | Jump if direct bit is set and clear bit | 3 | 3/4 |
| **Program Branching** | | | |
| ACALL addr11 | Absolute subroutine call | 2 | 3 |
| LCALL addr16 | Long subroutine call | 3 | 4 |
| RET | Return from subroutine | 1 | 5 |
| RETI | Return from interrupt | 1 | 5 |
| AJMP addr11 | Absolute jump | 2 | 3 |
| LJMP addr16 | Long jump | 3 | 4 |
| SJMP rel | Short jump (relative address) | 2 | 3 |
| JMP @A+DPTR | Jump indirect relative to DPTR | 1 | 3 |
| JZ rel | Jump if A equals zero | 2 | 2/3 |
| JNZ rel | Jump if A does not equal zero | 2 | 2/3 |
| CJNE A, direct, rel | Compare direct byte to A and jump if not equal | 3 | 4/5 |
| CJNE A, #data, rel | Compare immediate to A and jump if not equal | 3 | 3/4 |
| CJNE Rn, #data, rel | Compare immediate to Register and jump if not equal | 3 | 3/4 |
| CJNE @Ri, #data, rel | Compare immediate to indirect and jump if not equal | 3 | 4/5 |
| DJNZ Rn, rel | Decrement Register and jump if not zero | 2 | 2/3 |
| DJNZ direct, rel | Decrement direct byte and jump if not zero | 3 | 3/4 |
| NOP | No operation | 1 | 1 |

| Notes on Registers, Operands and Addressing Modes: |
|---|
| **Rn** - Register R0–R7 of the currently selected register bank.<br><br>@**Ri** - Data RAM location addressed indirectly through R0 or R1.<br><br>**rel** - 8-bit, signed (twos complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps.<br><br>**direct** - 8-bit internal data location's address. This could be a direct-access Data RAM location (0x00–0x7F) or an SFR (0x80–0xFF).<br><br>#**data** - 8-bit constant<br><br>#**data16** - 16-bit constant<br><br>**bit** - Direct-accessed bit in Data RAM or SFR<br><br>**addr11** - 11-bit destination address used by ACALL and AJMP. The destination must be within the same 2 kB page of program memory as the first byte of the following instruction.<br><br>**addr16** - 16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8 kB program memory space.<br><br>There is one unused opcode (0xA5) that performs the same function as NOP.<br>All mnemonics copyrighted © Intel Corporation 1980. |

## 8.4. CIP-51 Register Descriptions

Following are descriptions of SFRs related to the operation of the CIP-51 System Controller. Reserved bits should not be set to logic l. Future product versions may use these bits to implement new features in which case the reset value of the bit will be logic 0, selecting the feature's default state. Detailed descriptions of the remaining SFRs are included in the sections of the data sheet associated with their corresponding system function.

### SFR Definition 8.1. DPL: Data Pointer Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DPL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0x82

| Bit | Name | Function |
|---|---|---|
| 7:0 | DPL[7:0] | **Data Pointer Low.**<br>The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed flash memory or XRAM. |

### SFR Definition 8.2. DPH: Data Pointer High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DPH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0x83

| Bit | Name | Function |
|---|---|---|
| 7:0 | DPH[7:0] | **Data Pointer High.**<br>The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed flash memory or XRAM. |

## SFR Definition 8.3. SP: Stack Pointer

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

SFR Page = All Pages; SFR Address = 0x81

| Bit | Name | Function |
|---|---|---|
| 7:0 | SP[7:0] | **Stack Pointer.**<br>The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset. |

## SFR Definition 8.4. ACC: Accumulator

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ACC[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xE0; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7:0 | ACC[7:0] | **Accumulator.**<br>This register is the accumulator for arithmetic operations. |

## SFR Definition 8.5. B: B Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | B[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xF0; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7:0 | B[7:0] | **B Register.**<br>This register serves as a second accumulator for certain arithmetic operations. |

## SFR Definition 8.6. PSW: Program Status Word

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CY | AC | F0 | RS[1:0] | | OV | F1 | PARITY |
| Type | R/W | R/W | R/W | R/W | | R/W | R/W | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xD0; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | CY | **Carry Flag.**<br>This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to logic 0 by all other arithmetic operations. |
| 6 | AC | **Auxiliary Carry Flag.**<br>This bit is set when the last arithmetic operation resulted in a carry into (addition) or a borrow from (subtraction) the high order nibble. It is cleared to logic 0 by all other arithmetic operations. |
| 5 | F0 | **User Flag 0.**<br>This is a bit-addressable, general purpose flag for use under software control. |
| 4:3 | RS[1:0] | **Register Bank Select.**<br>These bits select which register bank is used during register accesses.<br>00: Bank 0, Addresses 0x00-0x07<br>01: Bank 1, Addresses 0x08-0x0F<br>10: Bank 2, Addresses 0x10-0x17<br>11: Bank 3, Addresses 0x18-0x1F |
| 2 | OV | **Overflow Flag.**<br>This bit is set to 1 under the following circumstances:<br>■ An ADD, ADDC, or SUBB instruction causes a sign-change overflow.<br>■ A MUL instruction results in an overflow (result is greater than 255).<br>■ A DIV instruction causes a divide-by-zero condition.<br>The OV bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases. |
| 1 | F1 | **User Flag 1.**<br>This is a bit-addressable, general purpose flag for use under software control. |
| 0 | PARITY | **Parity Flag.**<br>This bit is set to logic 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even. |

# 9. Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. The memory organization of the Si1000/1/2/3/4/5 device family is shown in Figure 9.1

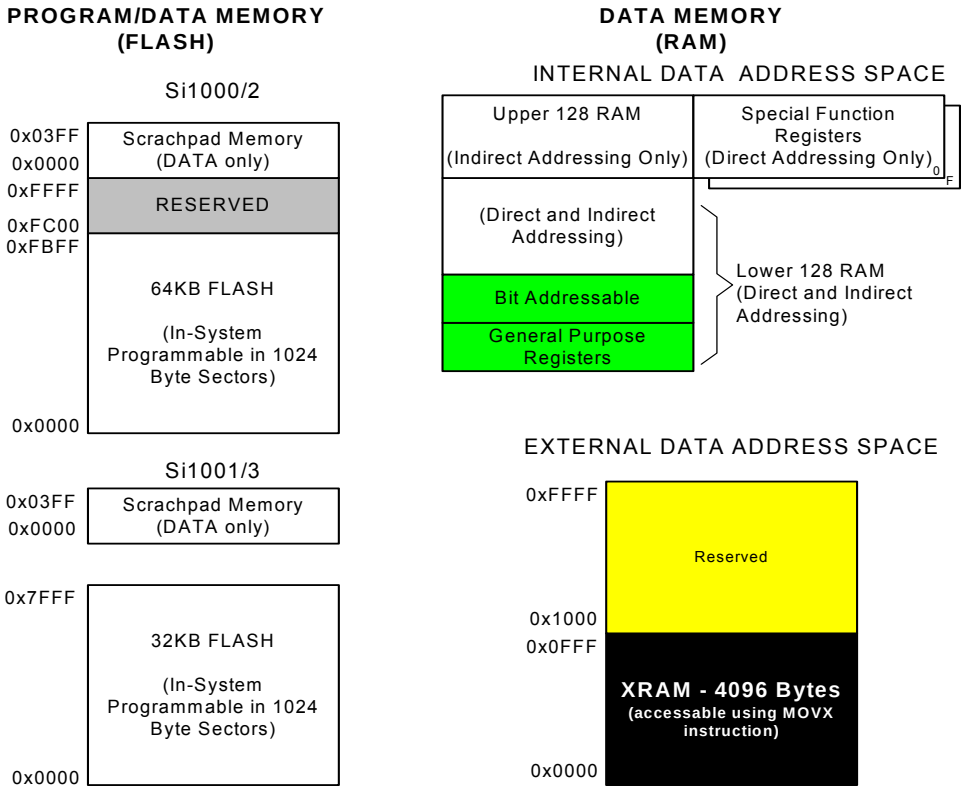



**Figure 9.1. Si1000/1/2/3/4/5 Memory Map**

## 9.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The Si1000/1/2/3/4/5 implements 64 kB (Si1000/2) or 32 kB (Si1001/3) of this program memory space as in-system, re-programmable flash memory, organized in a contiguous block from addresses 0x0000 to 0xFBFF (Si1000/2) or 0x7FFF (Si1001/3). The address 0xFBFF (Si1000/2) or 0x7FFF (Si1001/3) serves as the security lock byte for the device. Any addresses above the lock byte are reserved.

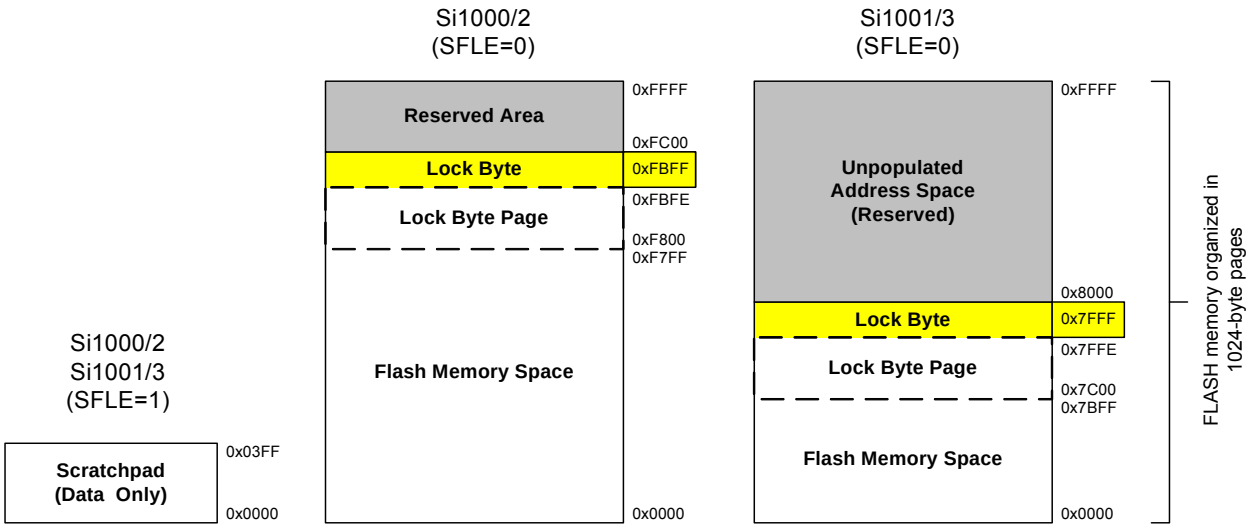



**Figure 9.2. Flash Program Memory Map**

### 9.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the Si1000/1/2/3/4/5 devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip flash memory space. MOVC instructions are always used to read flash memory, while MOVX write instructions are used to erase and write flash. This flash access feature provides a mechanism for the Si1000/1/2/3/4/5 to update program code and use the program memory space for non-volatile data storage. Refer to Section “13. Flash Memory” on page 142 for further details.

## 9.2. Data Memory

The Si1000/1/2/3/4/5 device family includes 4352 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. 4096 bytes of this memory is on-chip “external” memory. The data memory map is shown in Figure 9.1 for reference.

### 9.2.1. Internal RAM

There are 256 bytes of internal RAM mapped into the data memory space from 0x00 through 0xFF. The lower 128 bytes of data memory are used for general purpose registers and scratch pad memory. Either direct or indirect addressing may be used to access the lower 128 bytes of data memory. Locations 0x00 through 0x1F are addressable as four banks of general purpose registers, each bank consisting of eight byte-wide registers. The next 16 bytes, locations 0x20 through 0x2F, may either be addressed as bytes or as 128 bit locations accessible with the direct addressing mode.

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the

upper 128 bytes of data memory. Figure 9.1 illustrates the data memory organization of the Si1000/1/2/3/4/5.

#### 9.2.1.1. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 8.6). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

#### 9.2.1.2. Bit Addressable Locations

In addition to direct access to data memory organized as bytes, the sixteen data memory locations at 0x20 through 0x2F are also accessible as 128 individually addressable bits. Each bit has a bit address from 0x00 to 0x7F. Bit 0 of the byte at 0x20 has bit address 0x00 while bit7 of the byte at 0x20 has bit address 0x07. Bit 7 of the byte at 0x2F has bit address 0x7F. A bit access is distinguished from a full byte access by the type of instruction used (bit source or destination operands as opposed to a byte source or destination).

The MCS-51™ assembly language allows an alternate notation for bit addressing of the form XX.B where XX is the byte address and B is the bit position within the byte. For example, the instruction:

```
MOV    C, 22.3h
```

moves the Boolean value at 0x13 (bit 3 of the byte at location 0x22) into the Carry flag.

#### 9.2.1.3. Stack

A programmer's stack can be located anywhere in the 256-byte data memory. The stack area is designated using the Stack Pointer (SP) SFR. The SP will point to the last location used. The next value pushed on the stack is placed at SP+1 and then SP is incremented. A reset initializes the stack pointer to location 0x07. Therefore, the first value pushed on the stack is placed at location 0x08, which is also the first register (R0) of register bank 1. Thus, if more than one register bank is to be used, the SP should be initialized to a location in the data memory not being used for data storage. The stack depth can extend up to 256 bytes.

### 9.2.2. External RAM

There are 4096 bytes of on-chip RAM mapped into the external data memory space. All of these address locations may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using MOVX indirect addressing mode (such as @R1) in combination with the EMI0CN register.

# 10. On-Chip XRAM

The Si1000/1/2/3/4/5 MCUs include on-chip RAM mapped into the external data memory space (XRAM). The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either the data pointer (DPTR), or with the target address low byte in R0 or R1 and the target address high byte in the External Memory Interface Control Register (EMI0CN, shown in SFR Definition 10.1).

When using the MOVX instruction to access on-chip RAM, no additional initialization is required and the MOVX instruction execution time is as specified in the CIP-51 chapter.

**Important Note**: MOVX write operations can be configured to target flash memory, instead of XRAM. See Section “13. Flash Memory” on page 142 for more details. The MOVX instruction accesses XRAM by default.

## 10.1. Accessing XRAM

The XRAM memory space is accessed using the MOVX instruction. The MOVX instruction has two forms, both of which use an indirect addressing method. The first method uses the Data Pointer, DPTR, a 16-bit register which contains the effective address of the XRAM location to be read from or written to. The second method uses R0 or R1 in combination with the EMI0CN register to generate the effective XRAM address. Examples of both of these methods are given below.

### 10.1.1. 16-Bit MOVX Example

The 16-bit form of the MOVX instruction accesses the memory location pointed to by the contents of the DPTR register. The following series of instructions reads the value of the byte at address 0x1234 into the accumulator A:

```
MOV    DPTR, #1234h        ; load DPTR with 16-bit address to read (0x1234)
MOVX   A, @DPTR            ; load contents of 0x1234 into accumulator A
```

The above example uses the 16-bit immediate MOV instruction to set the contents of DPTR. Alternately, the DPTR can be accessed through the SFR registers DPH, which contains the upper 8-bits of DPTR, and DPL, which contains the lower 8-bits of DPTR.

### 10.1.2. 8-Bit MOVX Example

The 8-bit form of the MOVX instruction uses the contents of the EMI0CN SFR to determine the upper 8-bits of the effective address to be accessed and the contents of R0 or R1 to determine the lower 8-bits of the effective address to be accessed. The following series of instructions read the contents of the byte at address 0x1234 into the accumulator A.

```
MOV    EMI0CN, #12h        ; load high byte of address into EMI0CN
MOV    R0, #34h            ; load low byte of address into R0 (or R1)
MOVX   a, @R0              ; load contents of 0x1234 into accumulator A
```

## 10.2. Special Function Registers

The special function register used for configuring XRAM access is EMI0CN.

### SFR Definition 10.1. EMI0CN: External Memory Interface Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | PGSEL[3:0] | | | |
| Type | R | R | R | R | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xAA

| Bit | Name | Function |
|---|---|---|
| 7:4 | Unused | Read = 0000b; Write = Don't Care. |
| 3:0 | PGSEL | **XRAM Page Select.**<br>The EMI0CN register provides the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. Since the upper (unused) bits of the register are always zero, EMI0CN determines which page of XRAM is accessed.<br><br>For Example:<br>If EMI0CN = 0x01, addresses 0x0100 through 0x01FF will be accessed.<br>If EMI0CN = 0x0F, addresses 0x0F00 through 0x0FFF will be accessed. |

## 11. Special Function Registers

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the Si1000/1/2/3/4/5's resources and peripherals. The CIP-51 controller core duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the Si1000/1/2/3/4/5. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 11.1 and Table 11.2 list the SFRs implemented in the Si1000/1/2/3/4/5 device family.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g. P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the data sheet, as indicated in Table 11.3, for a detailed description of each register.

**Table 11.1. Special Function Register (SFR) Memory Map (Page 0x0)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | SPI0CN | PCA0L | PCA0H | PCA0CPL0 | PCA0CPH0 | PCA0CPL4 | PCA0CPH4 | VDM0CN |
| F0 | B | P0MDIN | P1MDIN | P2MDIN | SMB0ADR | SMB0ADM | EIP1 | EIP2 |
| E8 | ADC0CN | PCA0CPL1 | PCA0CPH1 | PCA0CPL2 | PCA0CPH2 | PCA0CPL3 | PCA0CPH3 | RSTSRC |
| E0 | ACC | XBR0 | XBR1 | XBR2 | IT01CF | | EIE1 | EIE2 |
| D8 | PCA0CN | PCA0MD | PCA0CPM0 | PCA0CPM1 | PCA0CPM2 | PCA0CPM3 | PCA0CPM4 | PCA0PWM |
| D0 | PSW | REF0CN | PCA0CPL5 | PCA0CPH5 | P0SKIP | P1SKIP | P2SKIP | P0MAT |
| C8 | TMR2CN | REG0CN | TMR2RLL | TMR2RLH | TMR2L | TMR2H | PCA0CPM5 | P1MAT |
| C0 | SMB0CN | SMB0CF | SMB0DAT | ADC0GTL | ADC0GTH | ADC0LTL | ADC0LTH | P0MASK |
| B8 | IP | IREF0CN | ADC0AC | ADC0MX | ADC0CF | ADC0L | ADC0H | P1MASK |
| B0 | SPI1CN | OSCXCN | OSCICN | OSCICL | | PMU0CF | FLSCL | FLKEY |
| A8 | IE | CLKSEL | EMI0CN | Reserved | RTC0ADR | RTC0DAT | RTC0KEY | Reserved |
| A0 | P2 | SPI0CFG | SPI0CKR | SPI0DAT | P0MDOUT | P1MDOUT | P2MDOUT | SFRPAGE |
| 98 | SCON0 | SBUF0 | CPT1CN | CPT0CN | CPT1MD | CPT0MD | CPT1MX | CPT0MX |
| 90 | P1 | TMR3CN | TMR3RLL | TMR3RLH | TMR3L | TMR3H | DC0CF | DC0CN |
| 88 | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | CKCON | PSCTL |
| 80 | P0 | SP | DPL | DPH | SPI1CFG | SPI1CKR | SPI1DAT | PCON |
| | 0(8) | 1(9) | 2(A) | 3(B) | 4(C) | 5(D) | 6(E) | 7(F) |

(bit addressable)

## 11.1. SFR Paging

To accommodate more than 128 SFRs in the 0x80 to 0xFF address space, SFR paging has been implemented. By default, all SFR accesses target SFR Page 0x0 to allow access to the registers listed in Table 11.1. During device initialization, some SFRs located on SFR Page 0xF may need to be accessed. Table 11.2 lists the SFRs accessible from SFR Page 0x0F. Some SFRs are accessible from both pages, including the SFRPAGE register. SFRs accessible only from Page 0xF are in **bold**.

The following procedure should be used when accessing SFRs from Page 0xF:

1. Save the current interrupt state (EA_save = EA).
2. Disable Interrupts (EA = 0).
3. Set SFRPAGE = 0xF.
4. Access the SFRs located on SFR Page 0xF.
5. Set SFRPAGE = 0x0.
6. Restore interrupt state (EA = EA_save).

**Table 11.2. Special Function Register (SFR) Memory Map (Page 0xF)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | | | | | | | | |
| F0 | B | | | | | | EIP1 | EIP2 |
| E8 | | | | | | | | |
| E0 | ACC | | | | | | EIE1 | EIE2 |
| D8 | | | | | | | | |
| D0 | PSW | | | | | | | |
| C8 | | | | | | | | |
| C0 | | | | | | | | |
| B8 | | | **ADC0PWR** | | | **ADC0TK** | | |
| B0 | | | | | | | | |
| A8 | IE | CLKSEL | | | | | | |
| A0 | P2 | | | | **P0DRV** | **P1DRV** | **P2DRV** | SFRPAGE |
| 98 | | | | | | | | |
| 90 | P1 | **CRC0DAT** | **CRC0CN** | **CRC0IN** | | **CRC0FLIP** | **CRC0AUTO** | **CRC0CNT** |
| 88 | | | | | | | | |
| 80 | P0 | SP | DPL | DPH | | **TOFFL** | **TOFFH** | PCON |
| | 0(8) | 1(9) | 2(A) | 3(B) | 4(C) | 5(D) | 6(E) | 7(F) |

(bit addressable)

## SFR Definition 11.1. SFRPage: SFR Page

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SFRPAGE[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xA7

| Bit | Name | Function |
|---|---|---|
| 7:0 | SFRPAGE[7:0] | **SFR Page.**<br>Specifies the SFR Page used when reading, writing, or modifying special function registers. |

### Table 11.3. Special Function Registers

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| ACC | 0xE0 | All | Accumulator | 117 |
| ADC0AC | 0xBA | 0x0 | ADC0 Accumulator Configuration | 84 |
| ADC0CF | 0xBC | 0x0 | ADC0 Configuration | 83 |
| ADC0CN | 0xE8 | 0x0 | ADC0 Control | 82 |
| ADC0GTH | 0xC4 | 0x0 | ADC0 Greater-Than Compare High | 88 |
| ADC0GTL | 0xC3 | 0x0 | ADC0 Greater-Than Compare Low | 88 |
| ADC0H | 0xBE | 0x0 | ADC0 High | 87 |
| ADC0L | 0xBD | 0x0 | ADC0 Low | 87 |
| ADC0LTH | 0xC6 | 0x0 | ADC0 Less-Than Compare Word High | 89 |
| ADC0LTL | 0xC5 | 0x0 | ADC0 Less-Than Compare Word Low | 89 |
| ADC0MX | 0xBB | 0x0 | AMUX0 Channel Select | 92 |
| ADC0PWR | 0xBA | 0xF | ADC0 Burst Mode Power-Up Time | 85 |
| ADC0TK | 0xBD | 0xF | ADC0 Tracking Control | 86 |
| B | 0xF0 | All | B Register | 117 |
| CKCON | 0x8E | 0x0 | Clock Control | 336 |
| CLKSEL | 0xA9 | All | Clock Select | 190 |
| CPT0CN | 0x9B | 0x0 | Comparator0 Control | 104 |
| CPT0MD | 0x9D | 0x0 | Comparator0 Mode Selection | 104 |
| CPT0MX | 0x9F | 0x0 | Comparator0 Mux Selection | 108 |
| CPT1CN | 0x9A | 0x0 | Comparator1 Control | 105 |
| CPT1MD | 0x9C | 0x0 | Comparator1 Mode Selection | 106 |
| CPT1MX | 0x9E | 0x0 | Comparator1 Mux Selection | 109 |
| CRC0AUTO | 0x96 | 0xF | CRC0 Automatic Control | 166 |
| CRC0CN | 0x92 | 0xF | CRC0 Control | 164 |
| CRC0CNT | 0x97 | 0xF | CRC0 Automatic Flash Sector Count | 166 |
| CRC0DAT | 0x91 | 0xF | CRC0 Data | 165 |
| CRC0FLIP | 0x95 | 0xF | CRC0 Flip | 167 |

**Table 11.3. Special Function Registers (Continued)**

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| CRC0IN | 0x93 | 0xF | CRC0 Input | 165 |
| DC0CF | 0x96 | 0x0 | DC0 (DC-DC Converter) Configuration | 175 |
| DC0CN | 0x97 | 0x0 | DC0 (DC-DC Converter) Control | 174 |
| DPH | 0x83 | All | Data Pointer High | 116 |
| DPL | 0x82 | All | Data Pointer Low | 116 |
| EIE1 | 0xE6 | All | Extended Interrupt Enable 1 | 136 |
| EIE2 | 0xE7 | All | Extended Interrupt Enable 2 | 138 |
| EIP1 | 0xF6 | 0x0 | Extended Interrupt Priority 1 | 137 |
| EIP2 | 0xF7 | 0x0 | Extended Interrupt Priority 2 | 139 |
| EMI0CN | 0xAA | 0x0 | EMIF Control | 123 |
| FLKEY | 0xB7 | 0x0 | Flash Lock And Key | 150 |
| FLSCL | 0xB6 | 0x0 | Flash Scale | 150 |
| IE | 0xA8 | All | Interrupt Enable | 134 |
| IP | 0xB8 | 0x0 | Interrupt Priority | 135 |
| IREF0CN | 0xB9 | 0x0 | Current Reference IREF Control | 99 |
| IT01CF | 0xE4 | 0x0 | INT0/INT1 Configuration | 141 |
| OSCICL | 0xB3 | 0x0 | Internal Oscillator Calibration | 191 |
| OSCICN | 0xB2 | 0x0 | Internal Oscillator Control | 191 |
| OSCXCN | 0xB1 | 0x0 | External Oscillator Control | 192 |
| P0 | 0x80 | All | Port 0 Latch | 223 |
| P0DRV | 0xA4 | 0xF | Port 0 Drive Strength | 225 |
| P0MASK | 0xC7 | 0x0 | Port 0 Mask | 220 |
| P0MAT | 0xD7 | 0x0 | Port 0 Match | 220 |
| P0MDIN | 0xF1 | 0x0 | Port 0 Input Mode Configuration | 224 |
| P0MDOUT | 0xA4 | 0x0 | Port 0 Output Mode Configuration | 224 |
| P0SKIP | 0xD4 | 0x0 | Port 0 Skip | 223 |
| P1 | 0x90 | All | Port 1 Latch | 226 |
| P1DRV | 0xA5 | 0xF | Port 1 Drive Strength | 228 |
| P1MASK | 0xBF | 0x0 | Port 1 Mask | 221 |
| P1MAT | 0xCF | 0x0 | Port 1 Match | 221 |
| P1MDIN | 0xF2 | 0x0 | Port 1 Input Mode Configuration | 227 |
| P1MDOUT | 0xA5 | 0x0 | Port 1 Output Mode Configuration | 227 |
| P1SKIP | 0xD5 | 0x0 | Port 1 Skip | 226 |
| P2 | 0xA0 | All | Port 2 Latch | 228 |
| P2DRV | 0xA6 | 0xF | Port 2 Drive Strength | 230 |
| P2MDIN | 0xF3 | 0x0 | Port 2 Input Mode Configuration | 229 |
| P2MDOUT | 0xA6 | 0x0 | Port 2 Output Mode Configuration | 230 |
| P2SKIP | 0xD6 | 0x0 | Port 2 Skip | 229 |
| PCA0CN | 0xD8 | 0x0 | PCA0 Control | 370 |
| PCA0CPH0 | 0xFC | 0x0 | PCA0 Capture 0 High | 375 |
| PCA0CPH1 | 0xEA | 0x0 | PCA0 Capture 1 High | 375 |
| PCA0CPH2 | 0xEC | 0x0 | PCA0 Capture 2 High | 375 |
| PCA0CPH3 | 0xEE | 0x0 | PCA0 Capture 3 High | 375 |
| PCA0CPH4 | 0xFE | 0x0 | PCA0 Capture 4 High | 375 |

**Table 11.3. Special Function Registers (Continued)**

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| PCA0CPH5 | 0xD3 | 0x0 | PCA0 Capture 5 High | 375 |
| PCA0CPL0 | 0xFB | 0x0 | PCA0 Capture 0 Low | 375 |
| PCA0CPL1 | 0xE9 | 0x0 | PCA0 Capture 1 Low | 375 |
| PCA0CPL2 | 0xEB | 0x0 | PCA0 Capture 2 Low | 375 |
| PCA0CPL3 | 0xED | 0x0 | PCA0 Capture 3 Low | 375 |
| PCA0CPL4 | 0xFD | 0x0 | PCA0 Capture 4 Low | 375 |
| PCA0CPL5 | 0xD2 | 0x0 | PCA0 Capture 5 Low | 375 |
| PCA0CPM0 | 0xDA | 0x0 | PCA0 Module 0 Mode Register | 373 |
| PCA0CPM1 | 0xDB | 0x0 | PCA0 Module 1 Mode Register | 373 |
| PCA0CPM2 | 0xDC | 0x0 | PCA0 Module 2 Mode Register | 373 |
| PCA0CPM3 | 0xDD | 0x0 | PCA0 Module 3 Mode Register | 373 |
| PCA0CPM4 | 0xDE | 0x0 | PCA0 Module 4 Mode Register | 373 |
| PCA0CPM5 | 0xCE | 0x0 | PCA0 Module 5 Mode Register | 373 |
| PCA0H | 0xFA | 0x0 | PCA0 Counter High | 374 |
| PCA0L | 0xF9 | 0x0 | PCA0 Counter Low | 374 |
| PCA0MD | 0xD9 | 0x0 | PCA0 Mode | 371 |
| PCA0PWM | 0xDF | 0x0 | PCA0 PWM Configuration | 372 |
| PCON | 0x87 | 0x0 | Power Control | 158 |
| PMU0CF | 0xB5 | 0x0 | PMU0 Configuration | 157 |
| PSCTL | 0x8F | 0x0 | Program Store R/W Control | 149 |
| PSW | 0xD0 | All | Program Status Word | 118 |
| REF0CN | 0xD1 | 0x0 | Voltage Reference Control | 98 |
| REG0CN | 0xC9 | 0x0 | Voltage Regulator (VREG0) Control | 177 |
| RSTSRC | 0xEF | 0x0 | Reset Source Configuration/Status | 184 |
| RTC0ADR | 0xAC | 0x0 | RTC0 Address | 198 |
| RTC0DAT | 0xAD | 0x0 | RTC0 Data | 199 |
| RTC0KEY | 0xAE | 0x0 | RTC0 Key | 197 |
| SBUF0 | 0x99 | 0x0 | UART0 Data Buffer | 320 |
| SCON0 | 0x98 | 0x0 | UART0 Control | 319 |
| SFRPAGE | 0xA7 | All | SFR Page | 126 |
| SMB0ADM | 0xF5 | 0x0 | SMBus Slave Address Mask | 302 |
| SMB0ADR | 0xF4 | 0x0 | SMBus Slave Address | 302 |
| SMB0CF | 0xC1 | 0x0 | SMBus0 Configuration | 297 |
| SMB0CN | 0xC0 | 0x0 | SMBus0 Control | 299 |
| SMB0DAT | 0xC2 | 0x0 | SMBus0 Data | 305 |
| SP | 0x81 | All | Stack Pointer | 117 |
| SPI0CFG | 0xA1 | 0x0 | SPI0 Configuration | 329 |
| SPI0CKR | 0xA2 | 0x0 | SPI0 Clock Rate Control | 331 |
| SPI0CN | 0xF8 | 0x0 | SPI0 Control | 330 |
| SPI0DAT | 0xA3 | 0x0 | SPI0 Data | 331 |
| SPI1CFG | 0x84 | 0x0 | SPI1 Configuration | 329 |
| SPI1CKR | 0x85 | 0x0 | SPI1 Clock Rate Control | 331 |
| SPI1CN | 0xB0 | 0x0 | SPI1 Control | 330 |
| SPI1DAT | 0x86 | 0x0 | SPI1 Data | 331 |

**Table 11.3. Special Function Registers (Continued)**

SFRs are listed in alphabetical order. All undefined SFR locations are reserved

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| TCON | 0x88 | 0x0 | Timer/Counter Control | 341 |
| TH0 | 0x8C | 0x0 | Timer/Counter 0 High | 344 |
| TH1 | 0x8D | 0x0 | Timer/Counter 1 High | 344 |
| TL0 | 0x8A | 0x0 | Timer/Counter 0 Low | 343 |
| TL1 | 0x8B | 0x0 | Timer/Counter 1 Low | 343 |
| TMOD | 0x89 | 0x0 | Timer/Counter Mode | 342 |
| TMR2CN | 0xC8 | 0x0 | Timer/Counter 2 Control | 348 |
| TMR2H | 0xCD | 0x0 | Timer/Counter 2 High | 350 |
| TMR2L | 0xCC | 0x0 | Timer/Counter 2 Low | 350 |
| TMR2RLH | 0xCB | 0x0 | Timer/Counter 2 Reload High | 349 |
| TMR2RLL | 0xCA | 0x0 | Timer/Counter 2 Reload Low | 349 |
| TMR3CN | 0x91 | 0x0 | Timer/Counter 3 Control | 354 |
| TMR3H | 0x95 | 0x0 | Timer/Counter 3 High | 356 |
| TMR3L | 0x94 | 0x0 | Timer/Counter 3 Low | 356 |
| TMR3RLH | 0x93 | 0x0 | Timer/Counter 3 Reload High | 355 |
| TMR3RLL | 0x92 | 0x0 | Timer/Counter 3 Reload Low | 355 |
| TOFFH | 0x86 | 0xF | Temperature Offset High | 95 |
| TOFFL | 0x85 | 0xF | Temperature Offset Low | 95 |
| VDM0CN | 0xFF | 0x0 | VDD Monitor Control | 182 |
| XBR0 | 0xE1 | 0x0 | Port I/O Crossbar Control 0 | 217 |
| XBR1 | 0xE2 | 0x0 | Port I/O Crossbar Control 1 | 218 |
| XBR2 | 0xE3 | 0x0 | Port I/O Crossbar Control 2 | 219 |

# 12. Interrupt Handler

The Si1000/1/2/3/4/5 microcontroller family includes an extended interrupt system supporting multiple interrupt sources and two priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Refer to Table 12.1, “Interrupt Summary,” on page 132 for a detailed listing of all interrupt sources supported by the device. Refer to the data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR or an indirect register. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1. If both global interrupts and the specific interrupt source is enabled, a CPU interrupt request is generated when the interrupt-pending flag is set.

As soon as execution of the current instruction is complete, the CPU generates an LCALL to a predetermined address to begin execution of an interrupt service routine (ISR). Each ISR must end with an RETI instruction, which returns program execution to the next instruction that would have been executed if the interrupt request had not occurred. If interrupts are not enabled, the interrupt-pending flag is ignored by the hardware and program execution continues as normal. (The interrupt-pending flag is set to logic 1 regardless of the interrupt's enable/disable state.)

Some interrupt-pending flags are automatically cleared by hardware when the CPU vectors to the ISR. However, most are not cleared by the hardware and must be cleared by software before returning from the ISR. If an interrupt-pending flag remains set after the CPU completes the return-from-interrupt (RETI) instruction, a new interrupt request will be generated immediately and the CPU will re-enter the ISR after the completion of the next instruction.

## 12.1. Enabling Interrupt Sources

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in the Interrupt Enable and Extended Interrupt Enable SFRs. However, interrupts must first be globally enabled by setting the EA bit (IE.7) to logic 1 before the individual interrupt enables are recognized. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings. Note that interrupts which occur when the EA bit is set to logic 0 will be held in a pending state, and will not be serviced until the EA bit is set back to logic 1.

## 12.2. MCU Interrupt Sources and Vectors

The CPU services interrupts by generating an LCALL to a predetermined address (the interrupt vector address) to begin execution of an interrupt service routine (ISR). The interrupt vector addresses associated with each interrupt source are listed in Table 12.1 on page 132. Software should ensure that the interrupt vector for each enabled interrupt source contains a valid interrupt service routine.

Software can simulate an interrupt by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag.

## 12.3. Interrupt Priorities

Each interrupt source can be individually programmed to one of two priority levels: low or high. A low priority interrupt service routine can be preempted by a high priority interrupt. A high priority interrupt cannot be preempted. If a high priority interrupt preempts a low priority interrupt, the low priority interrupt will finish execution after the high priority interrupt completes. Each interrupt has an associated interrupt priority bit in in the Interrupt Priority and Extended Interrupt Priority registers used to configure its priority level. Low priority is the default.

If two interrupts are recognized simultaneously, the interrupt with the higher priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate. See Table 12.1 on page 132 to determine the fixed priority order used to arbitrate between simultaneously recognized interrupts.

## 12.4. Interrupt Latency

Interrupt response time depends on the state of the CPU when the interrupt occurs. Pending interrupts are sampled and priority decoded each system clock cycle. Therefore, the fastest possible response time is 7 system clock cycles: 1 clock cycle to detect the interrupt, 1 clock cycle to execute a single instruction, and 5 clock cycles to complete the LCALL to the ISR. If an interrupt is pending when a RETI is executed, a single instruction is executed before an LCALL is made to service the pending interrupt. Therefore, the maximum response time for an interrupt (when no other interrupt is currently being serviced or the new interrupt is of greater priority) occurs when the CPU is performing an RETI instruction followed by a DIV as the next instruction. In this case, the response time is 19 system clock cycles: 1 clock cycle to detect the interrupt, 5 clock cycles to execute the RETI, 8 clock cycles to complete the DIV instruction and 5 clock cycles to execute the LCALL to the ISR. If the CPU is executing an ISR for an interrupt with equal or higher priority, the new interrupt will not be serviced until the current ISR completes, including the RETI and following instruction.

**Table 12.1. Interrupt Summary**

| Interrupt Source | Interrupt Vector | Priority Order | Pending Flag | Bit addressable? | Cleared by HW? | Enable Flag | Priority Control |
|---|---|---|---|---|---|---|---|
| Reset | 0x0000 | Top | None | N/A | N/A | Always Enabled | Always Highest |
| External Interrupt 0 ($\overline{INT0}$) | 0x0003 | 0 | IE0 (TCON.1) | Y | Y | EX0 (IE.0) | PX0 (IP.0) |
| Timer 0 Overflow | 0x000B | 1 | TF0 (TCON.5) | Y | Y | ET0 (IE.1) | PT0 (IP.1) |
| External Interrupt 1 ($\overline{INT1}$) | 0x0013 | 2 | IE1 (TCON.3) | Y | Y | EX1 (IE.2) | PX1 (IP.2) |
| Timer 1 Overflow | 0x001B | 3 | TF1 (TCON.7) | Y | Y | ET1 (IE.3) | PT1 (IP.3) |
| UART0 | 0x0023 | 4 | RI0 (SCON0.0)<br>TI0 (SCON0.1) | Y | N | ES0 (IE.4) | PS0 (IP.4) |
| Timer 2 Overflow | 0x002B | 5 | TF2H (TMR2CN.7)<br>TF2L (TMR2CN.6) | Y | N | ET2 (IE.5) | PT2 (IP.5) |
| SPI0 | 0x0033 | 6 | SPIF (SPI0CN.7)<br>WCOL (SPI0CN.6)<br>MODF (SPI0CN.5)<br>RXOVRN (SPI0CN.4) | Y | N | ESPI0 (IE.6) | PSPI0 (IP.6) |
| SMB0 | 0x003B | 7 | SI (SMB0CN.0) | Y | N | ESMB0 (EIE1.0) | PSMB0 (EIP1.0) |
| SmaRTClock Alarm | 0x0043 | 8 | ALRM (RTC0CN.2)* | N | N | EARTC0 (EIE1.1) | PARTC0 (EIP1.1) |
| ADC0 Window Comparator | 0x004B | 9 | AD0WINT (ADC0CN.3) | Y | N | EWADC0 (EIE1.2) | PWADC0 (EIP1.2) |
| ADC0 End of Conversion | 0x0053 | 10 | AD0INT (ADC0STA.5) | Y | N | EADC0 (EIE1.3) | PADC0 (EIP1.3) |
| Programmable Counter Array | 0x005B | 11 | CF (PCA0CN.7)<br>CCFn (PCA0CN.n) | Y | N | EPCA0 (EIE1.4) | PPCA0 (EIP1.4) |
| Comparator0 | 0x0063 | 12 | CP0FIF (CPT0CN.4)<br>CP0RIF (CPT0CN.5) | N | N | ECP0 (EIE1.5) | PCP0 (EIP1.5) |
| Comparator1 | 0x006B | 13 | CP1FIF (CPT1CN.4)<br>CP1RIF (CPT1CN.5) | N | N | ECP1 (EIE1.6) | PCP1 (EIP1.6) |
| Timer 3 Overflow | 0x0073 | 14 | TF3H (TMR3CN.7)<br>TF3L (TMR3CN.6) | N | N | ET3 (EIE1.7) | PT3 (EIP1.7) |
| VDD_MCU Supply Monitor Early Warning | 0x007B | 15 | VDDOK (VDM0CN.5)[1] | | | EWARN (EIE2.0) | PWARN (EIP2.0) |
| Port Match | 0x0083 | 16 | None | | | EMAT (EIE2.1) | PMAT (EIP2.1) |

**Table 12.1. Interrupt Summary (Continued)**

| Interrupt Source | Interrupt Vector | Priority Order | Pending Flag | Bit addressable? | Cleared by HW? | Enable Flag | Priority Control |
|---|---|---|---|---|---|---|---|
| SmaRTClock Oscillator Fail | 0x008B | 17 | OSCFAIL (RTC0CN.5)[2] | N | N | ERTC0F (EIE2.2) | PFRTC0F (EIP2.2) |
| EZRadioPRO Serial Interface (SPI1) | 0x0093 | 18 | SPIF (SPI1CN.7)<br>WCOL (SPI1CN.6)<br>MODF (SPI1CN.5)<br>RXOVRN (SPI1CN.4) | N | N | ESPI1 (EIE2.3) | PSPI1 (EIP2.3) |

**Notes:**
1. Indicates a read-only interrupt pending flag. The interrupt enable may be used to prevent software from vectoring to the associated interrupt service routine.
2. Indicates a register located in an indirect memory space.

## 12.5. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described in the following register descriptions. Refer to the data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

## SFR Definition 12.1. IE: Interrupt Enable

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | EA | ESPI0 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xA8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | EA | **Enable All Interrupts.**<br>Globally enables/disables all interrupts. It overrides individual interrupt mask settings.<br>0: Disable all interrupt sources.<br>1: Enable each interrupt according to its individual mask setting. |
| 6 | ESPI0 | **Enable Serial Peripheral Interface (SPI0) Interrupt.**<br>This bit sets the masking of the SPI0 interrupts.<br>0: Disable all SPI0 interrupts.<br>1: Enable interrupt requests generated by SPI0. |
| 5 | ET2 | **Enable Timer 2 Interrupt.**<br>This bit sets the masking of the Timer 2 interrupt.<br>0: Disable Timer 2 interrupt.<br>1: Enable interrupt requests generated by the TF2L or TF2H flags. |
| 4 | ES0 | **Enable UART0 Interrupt.**<br>This bit sets the masking of the UART0 interrupt.<br>0: Disable UART0 interrupt.<br>1: Enable UART0 interrupt. |
| 3 | ET1 | **Enable Timer 1 Interrupt.**<br>This bit sets the masking of the Timer 1 interrupt.<br>0: Disable all Timer 1 interrupt.<br>1: Enable interrupt requests generated by the TF1 flag. |
| 2 | EX1 | **Enable External Interrupt 1.**<br>This bit sets the masking of External Interrupt 1.<br>0: Disable external interrupt 1.<br>1: Enable interrupt requests generated by the $\overline{\text{INT1}}$ input. |
| 1 | ET0 | **Enable Timer 0 Interrupt.**<br>This bit sets the masking of the Timer 0 interrupt.<br>0: Disable all Timer 0 interrupt.<br>1: Enable interrupt requests generated by the TF0 flag. |
| 0 | EX0 | **Enable External Interrupt 0.**<br>This bit sets the masking of External Interrupt 0.<br>0: Disable external interrupt 0.<br>1: Enable interrupt requests generated by the $\overline{\text{INT0}}$ input. |

## SFR Definition 12.2. IP: Interrupt Priority

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | PSPI0 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| Type | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 1b, Write = don't care. |
| 6 | PSPI0 | **Serial Peripheral Interface (SPI0) Interrupt Priority Control.**<br>This bit sets the priority of the SPI0 interrupt.<br>0: SPI0 interrupt set to low priority level.<br>1: SPI0 interrupt set to high priority level. |
| 5 | PT2 | **Timer 2 Interrupt Priority Control.**<br>This bit sets the priority of the Timer 2 interrupt.<br>0: Timer 2 interrupt set to low priority level.<br>1: Timer 2 interrupt set to high priority level. |
| 4 | PS0 | **UART0 Interrupt Priority Control.**<br>This bit sets the priority of the UART0 interrupt.<br>0: UART0 interrupt set to low priority level.<br>1: UART0 interrupt set to high priority level. |
| 3 | PT1 | **Timer 1 Interrupt Priority Control.**<br>This bit sets the priority of the Timer 1 interrupt.<br>0: Timer 1 interrupt set to low priority level.<br>1: Timer 1 interrupt set to high priority level. |
| 2 | PX1 | **External Interrupt 1 Priority Control.**<br>This bit sets the priority of the External Interrupt 1 interrupt.<br>0: External Interrupt 1 set to low priority level.<br>1: External Interrupt 1 set to high priority level. |
| 1 | PT0 | **Timer 0 Interrupt Priority Control.**<br>This bit sets the priority of the Timer 0 interrupt.<br>0: Timer 0 interrupt set to low priority level.<br>1: Timer 0 interrupt set to high priority level. |
| 0 | PX0 | **External Interrupt 0 Priority Control.**<br>This bit sets the priority of the External Interrupt 0 interrupt.<br>0: External Interrupt 0 set to low priority level.<br>1: External Interrupt 0 set to high priority level. |

## SFR Definition 12.3. EIE1: Extended Interrupt Enable 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ET3 | ECP1 | ECP0 | EPCA0 | EADC0 | EWADC0 | ERTC0A | ESMB0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xE6

| Bit | Name | Function |
|---|---|---|
| 7 | ET3 | **Enable Timer 3 Interrupt.**<br>This bit sets the masking of the Timer 3 interrupt.<br>0: Disable Timer 3 interrupts.<br>1: Enable interrupt requests generated by the TF3L or TF3H flags. |
| 6 | ECP1 | **Enable Comparator1 (CP1) Interrupt.**<br>This bit sets the masking of the CP1 interrupt.<br>0: Disable CP1 interrupts.<br>1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags. |
| 5 | ECP0 | **Enable Comparator0 (CP0) Interrupt.**<br>This bit sets the masking of the CP0 interrupt.<br>0: Disable CP0 interrupts.<br>1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags. |
| 4 | EPCA0 | **Enable Programmable Counter Array (PCA0) Interrupt.**<br>This bit sets the masking of the PCA0 interrupts.<br>0: Disable all PCA0 interrupts.<br>1: Enable interrupt requests generated by PCA0. |
| 3 | EADC0 | **Enable ADC0 Conversion Complete Interrupt.**<br>This bit sets the masking of the ADC0 Conversion Complete interrupt.<br>0: Disable ADC0 Conversion Complete interrupt.<br>1: Enable interrupt requests generated by the AD0INT flag. |
| 2 | EWADC0 | **Enable Window Comparison ADC0 Interrupt.**<br>This bit sets the masking of ADC0 Window Comparison interrupt.<br>0: Disable ADC0 Window Comparison interrupt.<br>1: Enable interrupt requests generated by ADC0 Window Compare flag (AD0WINT). |
| 1 | ERTC0A | **Enable SmaRTClock Alarm Interrupts.**<br>This bit sets the masking of the SmaRTClock Alarm interrupt.<br>0: Disable SmaRTClock Alarm interrupts.<br>1: Enable interrupt requests generated by a SmaRTClock Alarm. |
| 0 | ESMB0 | **Enable SMBus (SMB0) Interrupt.**<br>This bit sets the masking of the SMB0 interrupt.<br>0: Disable all SMB0 interrupts.<br>1: Enable interrupt requests generated by SMB0. |

## SFR Definition 12.4. EIP1: Extended Interrupt Priority 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PT3 | PCP1 | PCP0 | PPCA0 | PADC0 | PWADC0 | PRTC0A | PSMB0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xF6

| Bit | Name | Function |
|---|---|---|
| 7 | PT3 | **Timer 3 Interrupt Priority Control.**<br>This bit sets the priority of the Timer 3 interrupt.<br>0: Timer 3 interrupts set to low priority level.<br>1: Timer 3 interrupts set to high priority level. |
| 6 | PCP1 | **Comparator1 (CP1) Interrupt Priority Control.**<br>This bit sets the priority of the CP1 interrupt.<br>0: CP1 interrupt set to low priority level.<br>1: CP1 interrupt set to high priority level. |
| 5 | PCP0 | **Comparator0 (CP0) Interrupt Priority Control.**<br>This bit sets the priority of the CP0 interrupt.<br>0: CP0 interrupt set to low priority level.<br>1: CP0 interrupt set to high priority level. |
| 4 | PPCA0 | **Programmable Counter Array (PCA0) Interrupt Priority Control.**<br>This bit sets the priority of the PCA0 interrupt.<br>0: PCA0 interrupt set to low priority level.<br>1: PCA0 interrupt set to high priority level. |
| 3 | PADC0 | **ADC0 Conversion Complete Interrupt Priority Control.**<br>This bit sets the priority of the ADC0 Conversion Complete interrupt.<br>0: ADC0 Conversion Complete interrupt set to low priority level.<br>1: ADC0 Conversion Complete interrupt set to high priority level. |
| 2 | PWADC0 | **ADC0 Window Comparator Interrupt Priority Control.**<br>This bit sets the priority of the ADC0 Window interrupt.<br>0: ADC0 Window interrupt set to low priority level.<br>1: ADC0 Window interrupt set to high priority level. |
| 1 | PRTC0A | **SmaRTClock Alarm Interrupt Priority Control.**<br>This bit sets the priority of the SmaRTClock Alarm interrupt.<br>0: SmaRTClock Alarm interrupt set to low priority level.<br>1: SmaRTClock Alarm interrupt set to high priority level. |
| 0 | PSMB0 | **SMBus (SMB0) Interrupt Priority Control.**<br>This bit sets the priority of the SMB0 interrupt.<br>0: SMB0 interrupt set to low priority level.<br>1: SMB0 interrupt set to high priority level. |

## SFR Definition 12.5. EIE2: Extended Interrupt Enable 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | ESPI1 | ERTC0F | EMAT | EWARN |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages;SFR Address = 0xE7

| Bit | Name | Function |
|---|---|---|
| 7:4 | Unused | Read = 0000b. Write = Don't care. |
| 3 | ESPI1 | **Enable Serial Peripheral Interface (SPI1) Interrupt.**<br>This bit sets the masking of the SPI1 interrupts.<br>0: Disable all SPI1 interrupts.<br>1: Enable interrupt requests generated by SPI1. |
| 2 | ERTC0F | **Enable SmaRTClock Oscillator Fail Interrupt.**<br>This bit sets the masking of the SmaRTClock Alarm interrupt.<br>0: Disable SmaRTClock Alarm interrupts.<br>1: Enable interrupt requests generated by SmaRTClock Alarm. |
| 1 | EMAT | **Enable Port Match Interrupts.**<br>This bit sets the masking of the Port Match Event interrupt.<br>0: Disable all Port Match interrupts.<br>1: Enable interrupt requests generated by a Port Match. |
| 0 | EWARN | **Enable VDD_MCU Supply Monitor Early Warning Interrupt.**<br>This bit sets the masking of the VDD_MCU Supply Monitor Early Warning interrupt.<br>0: Disable the VDD_MCU Supply Monitor Early Warning interrupt.<br>1: Enable interrupt requests generated by VDD_MCU Supply Monitor. |

## SFR Definition 12.6. EIP2: Extended Interrupt Priority 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | PSPI1 | PRTC0F | PMAT | PWARN |
| Type | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xF7

| Bit | Name | Function |
|---|---|---|
| 7:4 | Unused | Read = 0000b. Write = Don't care. |
| 3 | PSPI1 | **Serial Peripheral Interface (SPI1) Interrupt Priority Control.**<br>This bit sets the priority of the SPI1 interrupt.<br>0: SP1 interrupt set to low priority level.<br>1: SPI1 interrupt set to high priority level. |
| 2 | PRTC0F | **SmaRTClock Oscillator Fail Interrupt Priority Control.**<br>This bit sets the priority of the SmaRTClock Alarm interrupt.<br>0: SmaRTClock Alarm interrupt set to low priority level.<br>1: SmaRTClock Alarm interrupt set to high priority level. |
| 1 | PMAT | **Port Match Interrupt Priority Control.**<br>This bit sets the priority of the Port Match Event interrupt.<br>0: Port Match interrupt set to low priority level.<br>1: Port Match interrupt set to high priority level. |
| 0 | PWARN | **VDD_MCU Supply Monitor Early Warning Interrupt Priority Control.**<br>This bit sets the priority of the VDD_MCU Supply Monitor Early Warning interrupt.<br>0: VDD_MCU Supply Monitor Early Warning interrupt set to low priority level.<br>1: VDD_MCU Supply Monitor Early Warning interrupt set to high priority level. |

## 12.6. External Interrupts $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL ($\overline{\text{INT0}}$ Polarity) and IN1PL ($\overline{\text{INT1}}$ Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (Section “27.1. Timer 0 and Timer 1” on page 337) select level or edge sensitive. The table below lists the possible configurations.

| IT0 | IN0PL | $\overline{\text{INT0}}$ Interrupt |
|---|---|---|
| 1 | 0 | Active low, edge sensitive |
| 1 | 1 | Active high, edge sensitive |
| 0 | 0 | Active low, level sensitive |
| 0 | 1 | Active high, level sensitive |

| IT1 | IN1PL | $\overline{\text{INT1}}$ Interrupt |
|---|---|---|
| 1 | 0 | Active low, edge sensitive |
| 1 | 1 | Active high, edge sensitive |
| 0 | 0 | Active low, level sensitive |
| 0 | 1 | Active high, level sensitive |

$\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ are assigned to Port pins as defined in the IT01CF register (see SFR Definition 12.7). Note that $\overline{\text{INT0}}$ and $\overline{\text{INT0}}$ Port pin assignments are independent of any Crossbar assignments. $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ will monitor their assigned Port pins without disturbing the peripheral that was assigned the Port pin via the Crossbar. To assign a Port pin only to $\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see Section “21.3. Priority Crossbar Decoder” on page 214 for complete details on configuring the Crossbar).

IE0 (TCON.1) and IE1 (TCON.3) serve as the interrupt-pending flags for the $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupts, respectively. If an $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ external interrupt is configured as edge-sensitive, the corresponding interrupt-pending flag is automatically cleared by the hardware when the CPU vectors to the ISR. When configured as level sensitive, the interrupt-pending flag remains logic 1 while the input is active as defined by the corresponding polarity bit (IN0PL or IN1PL); the flag remains logic 0 while the input is inactive. The external interrupt source must hold the input active until the interrupt request is recognized. It must then deactivate the interrupt request before execution of the ISR completes or another interrupt request will be generated.

## SFR Definition 12.7. IT01CF: $\overline{\text{INT0}}$/$\overline{\text{INT1}}$ Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | IN1PL | IN1SL[2:0] | | | IN0PL | IN0SL[2:0] | | |
| Type | R/W | R/W | | | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

SFR Page = 0x0; SFR Address = 0xE4

| Bit | Name | Function |
|---|---|---|
| 7 | IN1PL | **$\overline{\text{INT1}}$ Polarity.**<br>0: $\overline{\text{INT1}}$ input is active low.<br>1: $\overline{\text{INT1}}$ input is active high. |
| 6:4 | IN1SL[2:0] | **$\overline{\text{INT1}}$ Port Pin Selection Bits.**<br>These bits select which Port pin is assigned to $\overline{\text{INT1}}$. Note that this pin assignment is independent of the Crossbar; $\overline{\text{INT1}}$ will monitor the assigned Port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin.<br>000: Select P0.0<br>001: Select P0.1<br>010: Select P0.2<br>011: Select P0.3<br>100: Select P0.4<br>101: Select P0.5<br>110: Select P0.6<br>111: Select P0.7 |
| 3 | IN0PL | **$\overline{\text{INT0}}$ Polarity.**<br>0: $\overline{\text{INT0}}$ input is active low.<br>1: $\overline{\text{INT0}}$ input is active high. |
| 2:0 | IN0SL[2:0] | **$\overline{\text{INT0}}$ Port Pin Selection Bits.**<br>These bits select which Port pin is assigned to $\overline{\text{INT0}}$. Note that this pin assignment is independent of the Crossbar; $\overline{\text{INT0}}$ will monitor the assigned Port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin.<br>000: Select P0.0<br>001: Select P0.1<br>010: Select P0.2<br>011: Select P0.3<br>100: Select P0.4<br>101: Select P0.5<br>110: Select P0.6<br>111: Select P0.7 |

# 13. Flash Memory

On-chip, re-programmable flash memory is included for program code and non-volatile data storage. The flash memory can be programmed in-system through the C2 interface or by software using the MOVX write instruction. Once cleared to logic 0, a flash bit must be erased to set it back to logic 1. Flash bytes would typically be erased (set to 0xFF) before being reprogrammed. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operations is not required. Code execution is stalled during flash write/erase operations. Refer to Table 4.6 for complete flash memory electrical characteristics.

## 13.1. Programming the Flash Memory

The simplest means of programming the flash memory is through the C2 interface using programming tools provided by Silicon Laboratories or a third party vendor. This is the only means for programming a non-initialized device. For details on the C2 commands to program flash memory, see Section “29. Device Specific Behavior” on page 376.

The flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before programming flash memory using MOVX, flash programming operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target flash memory); and (2) Writing the flash key codes in sequence to the flash lock register (FLKEY). The PSWE bit remains set until cleared by software. For detailed guidelines on programming flash from firmware, please see Section “13.5. Flash Write and Erase Guidelines” on page 146.

To ensure the integrity of the flash contents, the on-chip VDD Monitor must be enabled and enabled as a reset source in any system that includes code that writes and/or erases flash memory from software. Furthermore, there should be no delay between enabling the $V_{DD}$ Monitor and enabling the $V_{DD}$ Monitor as a reset source. Any attempt to write or erase flash memory while the $V_{DD}$ Monitor is disabled, or not enabled as a reset source, will cause a flash error device reset.

### 13.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The flash lock and key register (FLKEY) must be written with the correct key codes, in sequence, before flash operations may be performed. The key codes are: 0xA5, 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order, or the wrong codes are written, flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a flash write or erase is attempted before the key codes have been written properly. The flash lock resets after each write or erase; the key codes must be written again before a following flash operation can be performed. The FLKEY register is detailed in SFR Definition 13.2.

### 13.1.2. Flash Erase Procedure

The flash memory is organized in 1024-byte pages. The erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire 1024-byte page, perform the following steps:

1. Save current interrupt state and disable interrupts.
2. Set the PSEE bit (register PSCTL).
3. Set the PSWE bit (register PSCTL).
4. Write the first key code to FLKEY: 0xA5.
5. Write the second key code to FLKEY: 0xF1.
6. Using the MOVX instruction, write a data byte to any location within the 1024-byte page to be erased.
7. Clear the PSWE and PSEE bits.
8. Restore previous interrupt state.

Steps 4–6 must be repeated for each 1024-byte page to be erased.

**Notes:**

1. Future 16 and 8 kB derivatives in this product family will use a 512-byte page size. To maintain code compatibility across the entire family, the erase procedure should be performed on each 512-byte section of memory.
2. Flash security settings may prevent erasure of some flash pages, such as the reserved area and the page containing the lock bytes. For a summary of flash security settings and restrictions affecting flash erase operations, please see Section “13.3. Security Options” on page 144.
3. 8-bit MOVX instructions cannot be used to erase or write to flash memory at addresses higher than 0x00FF.

### 13.1.3. Flash Write Procedure

A write to flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in flash. **A byte location to be programmed should be erased before a new value is written.**

The recommended procedure for writing a single byte in flash is as follows:

1. Save current interrupt state and disable interrupts.
2. Ensure that the flash byte has been erased (has a value of 0xFF).
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. Write the first key code to FLKEY: 0xA5.
6. Write the second key code to FLKEY: 0xF1.
7. Using the MOVX instruction, write a single data byte to the desired location within the 1024-byte sector.
8. Clear the PSWE bit.
9. Restore previous interrupt state.

Steps 5–7 must be repeated for each byte to be written.

**Notes:**

1. Future 16 and 8 kB derivatives in this product family will use a 512-byte page size. To maintain code compatibility across the entire family, the erase procedure should be performed on each 512-byte section of memory.
2. Flash security settings may prevent writes to some areas of flash, such as the reserved area. For a summary of flash security settings and restrictions affecting flash write operations, please see Section “13.3. Security Options” on page 144.

## 13.2. Non-volatile Data Storage

The flash memory can be used for non-volatile data storage as well as program code. This allows data such as calibration coefficients to be calculated and stored at run time. Data is written using the MOVX write instruction and read using the MOVC instruction. Note: MOVX read instructions always target XRAM.

An additional 1024-byte scratchpad is available for non-volatile data storage. It is accessible at addresses 0x0000 to 0x03FF when SFLE is set to 1. The scratchpad area cannot be used for code execution.

## 13.3. Security Options

The CIP-51 provides security options to protect the flash memory from inadvertent modification by software as well as to prevent the viewing of proprietary program code and constants. The Program Store Write Enable (bit PSWE in register PSCTL) and the Program Store Erase Enable (bit PSEE in register PSCTL) bits protect the flash memory from accidental modification by software. PSWE must be explicitly set to 1 before software can modify the flash memory; both PSWE and PSEE must be set to 1 before software can erase flash memory. Additional security features prevent proprietary program code and data constants from being read or altered across the C2 interface.

A Security Lock Byte located at the last byte of flash user space offers protection of the flash program memory from access (reads, writes, or erases) by unprotected code or the C2 interface. The flash security mechanism allows the user to lock *n* 1024-byte flash pages, starting at page 0 (addresses 0x0000 to 0x03FF), where *n* is the 1s complement number represented by the Security Lock Byte. **Note that the page containing the flash Security Lock Byte is unlocked when no other flash pages are locked (all bits of the Lock Byte are 1) and locked when any other flash pages are locked (any bit of the Lock Byte is 0).** See the Si1000 example below.

| | |
|---|---|
| Security Lock Byte: | 11111101b |
| ones Complement: | 00000010b |
| Flash pages locked: | 3 (First two flash pages + Lock Byte Page) |
| Addresses locked: | 0x0000 to 0x07FF (first two flash pages) and 0xF800 to 0xFBFF (Lock Byte Page) |

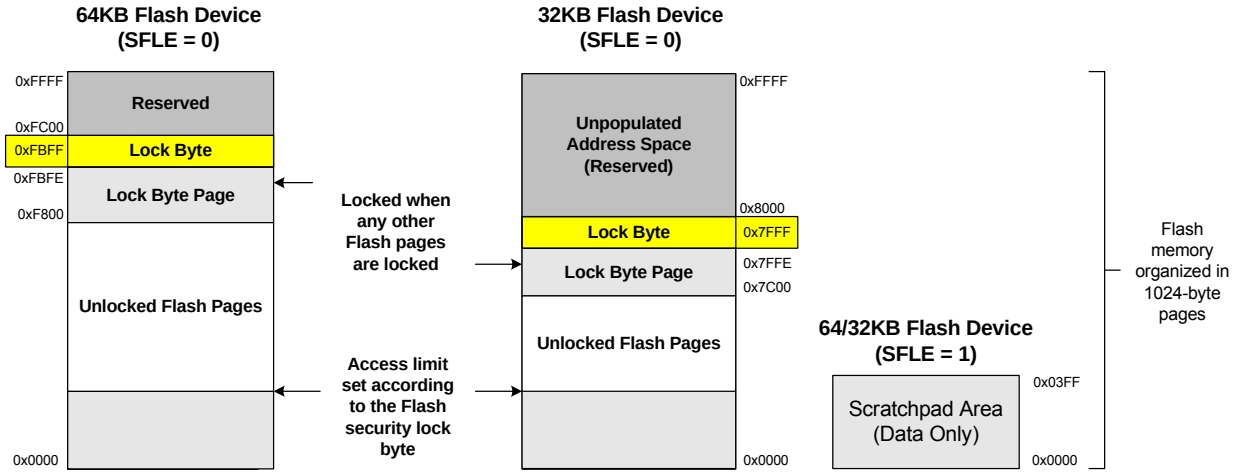



**Figure 13.1. Flash Program Memory Map**

The level of flash security depends on the flash access method. The three flash access methods that can be restricted are reads, writes, and erases from the C2 debug interface, user firmware executing on unlocked pages, and user firmware executing on locked pages. Table 13.1 summarizes the flash security features of the Si1000/1/2/3/4/5 devices.

**Table 13.1. Flash Security Summary**

| Action | C2 Debug Interface | User Firmware executing from: | |
|---|---|---|---|
| | | an unlocked page | a locked page |
| Read, Write or Erase unlocked pages (except page with Lock Byte) | Permitted | Permitted | Permitted |
| Read, Write or Erase locked pages (except page with Lock Byte) | Not Permitted | FEDR | Permitted |
| Read or Write page containing Lock Byte (if no pages are locked) | Permitted | Permitted | Permitted |
| Read or Write page containing Lock Byte (if any page is locked) | Not Permitted | FEDR | Permitted |
| Read contents of Lock Byte (if no pages are locked) | Permitted | Permitted | Permitted |
| Read contents of Lock Byte (if any page is locked) | Not Permitted | FEDR | Permitted |
| Erase page containing Lock Byte (if no pages are locked) | Permitted | FEDR | FEDR |
| Erase page containing Lock Byte - Unlock all pages (if any page is locked) | Only by C2DE | FEDR | FEDR |
| Lock additional pages (change 1s to 0s in the Lock Byte) | Not Permitted | FEDR | FEDR |
| Unlock individual pages (change 0s to 1s in the Lock Byte) | Not Permitted | FEDR | FEDR |
| Read, Write or Erase Reserved Area | Not Permitted | FEDR | FEDR |

C2DE—C2 Device Erase (Erases all flash pages including the page containing the Lock Byte)
FEDR—Not permitted; Causes Flash Error Device Reset (FERROR bit in RSTSRC is 1 after reset)

- All prohibited operations that are performed via the C2 interface are ignored (do not cause device reset).
- Locking any flash page also locks the page containing the Lock Byte.
- Once written to, the Lock Byte cannot be modified except by performing a C2 Device Erase.
- If user code writes to the Lock Byte, the Lock does not take effect until the next device reset.
- The scratchpad is locked when all other flash pages are locked.
- The scratchpad is erased when a Flash Device Erase command is performed.

## 13.4. Determining the Device Part Number at Run Time

In many applications, user software may need to determine the MCU part number at run time in order to determine the hardware capabilities. The part number can be determined by reading the value of the flash byte at address 0xFFFE.

The value of the flash byte at address 0xFFFE can be decoded as follows:

0xD0—Si1000
0xD1—Si1001
0xD2—Si1002
0xD3—Si1003
0xD4 - Si1004
0xD5 - Si1005

## 13.5. Flash Write and Erase Guidelines

Any system which contains routines which write or erase flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of VDD, system clock frequency, or temperature. This accidental execution of flash modifying code can result in alteration of flash memory contents causing a system failure that is only recoverable by re-flashing the code in the device.

To help prevent the accidental modification of flash by firmware, the VDD Monitor must be enabled and enabled as a reset source on Si100x devices for the flash to be successfully modified. **If either the VDD Monitor or the VDD Monitor reset source is not enabled, a Flash Error Device Reset will be generated when the firmware attempts to modify the flash.**

The following guidelines are recommended for any system that contains routines which write or erase flash from code.

### 13.5.1. VDD Maintenance and the VDD Monitor

1. If the system power supply is subject to voltage or current "spikes," add sufficient transient protection devices to the power supply to ensure that the supply voltages listed in the Absolute Maximum Ratings table are not exceeded.
2. Make certain that the minimum $V_{DD}$ rise time specification of 1 ms is met. If the system cannot meet this rise time specification, then add an external VDD brownout circuit to the $\overline{\text{RST}}$ pin of the device that holds the device in reset until $V_{DD}$ reaches the minimum device operating voltage and re-asserts $\overline{\text{RST}}$ if $V_{DD}$ drops below the minimum device operating voltage.
3. Keep the on-chip VDD Monitor enabled and enable the $V_{DD}$ Monitor as a reset source as early in code as possible. This should be the first set of instructions executed after the Reset Vector. For C-based systems, this will involve modifying the startup code added by the C compiler. See your compiler documentation for more details. Make certain that there are no delays in software between enabling the $V_{DD}$ Monitor and enabling the $V_{DD}$ Monitor as a reset source. Code examples showing this can be found in “AN201: Writing to Flash from Firmware," available from the Silicon Laboratories website.

**Notes:** On Si1000/1/2/3/4/5 devices, both the $V_{DD}$ Monitor and the $V_{DD}$ Monitor reset source must be enabled to write or erase flash without generating a Flash Error Device Reset.

On Si1000/1/2/3/4/5 devices, both the $V_{DD}$ Monitor and the $V_{DD}$ Monitor reset source are enabled by hardware after a power-on reset.

4. As an added precaution, explicitly enable the $V_{DD}$ Monitor and enable the $V_{DD}$ Monitor as a reset source inside the functions that write and erase flash memory. The $V_{DD}$ Monitor enable instructions should be placed just after the instruction to set PSWE to a 1, but before the flash write or erase operation instruction.

5. Make certain that all writes to the RSTSRC (Reset Sources) register use direct assignment operators and explicitly DO NOT use the bit-wise operators (such as AND or OR). For example, "RSTSRC = 0x02" is correct, but "RSTSRC |= 0x02" is incorrect.
6. Make certain that all writes to the RSTSRC register explicitly set the PORSF bit to a 1. Areas to check are initialization code which enables other reset sources, such as the Missing Clock Detector or Comparator, for example, and instructions which force a Software Reset. A global search on "RSTSRC" can quickly verify this.

#### 13.5.2. PSWE Maintenance

7. Reduce the number of places in code where the PSWE bit (b0 in PSCTL) is set to a 1. There should be exactly one routine in code that sets PSWE to a 1 to write flash bytes and one routine in code that sets both PSWE and PSEE both to a 1 to erase flash pages.
8. Minimize the number of variable accesses while PSWE is set to a 1. Handle pointer address updates and loop maintenance outside the "PSWE = 1;... PSWE = 0;" area. Code examples showing this can be found in “AN201: Writing to Flash from Firmware,” available from the Silicon Laboratories website.
9. Disable interrupts prior to setting PSWE to a 1 and leave them disabled until after PSWE has been reset to 0. Any interrupts posted during the flash write or erase operation will be serviced in priority order after the flash operation has been completed and interrupts have been re-enabled by software.
10. Make certain that the flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory areas.
11. Add address bounds checking to the routines that write or erase flash memory to ensure that a routine called with an illegal address does not result in modification of the flash.

#### 13.5.3. System Clock

12. If operating from an external crystal, be advised that crystal performance is susceptible to electrical interference and is sensitive to layout and to changes in temperature. If the system is operating in an electrically noisy environment, use the internal oscillator or use an external CMOS clock.
13. If operating from the external oscillator, switch to the internal oscillator during flash write or erase operations. The external oscillator can continue to run, and the CPU can switch back to the external oscillator after the flash operation has completed.

Additional flash recommendations and example code can be found in “AN201: Writing to Flash from Firmware,” available from the Silicon Laboratories website.

## 13.6. Minimizing Flash Read Current

The flash memory in the Si1000/1/2/3/4/5 devices is responsible for a substantial portion of the total digital supply current when the device is executing code. Below are suggestions to minimize flash read current.

1. Use Idle, Suspend, or Sleep Modes while waiting for an interrupt, rather than polling the interrupt flag. Idle Mode is particularly well-suited for use in implementing short pauses, since the wake-up time is no more than three system clock cycles. See the Power Management chapter for details on the various low-power operating modes.
2. Si1000/1/2/3/4/5 devices have a one-shot timer that saves power when operating at system clock frequencies of 10 MHz or less. The one-shot timer generates a minimum-duration enable signal for the flash sense amps on each clock cycle in which the flash memory is accessed. This allows the flash to remain in a low power state for the remainder of the long clock cycle.
   At clock frequencies above 10 MHz, the system clock cycle becomes short enough that the one-shot timer no longer provides a power benefit. Disabling the one-shot timer at higher frequencies reduces power consumption. The one-shot is enabled by default, and it can be disabled (bypassed) by setting the BYPASS bit (FLSCL.6) to logic 1. To re-enable the one-shot, clear the BYPASS bit to logic 0. After changing the BYPASS bit from 1 to 0, the third opcode byte fetched from program memory is indeterminate. Therefore, the operation which clears the BYPASS bit should be immediately followed by a benign 3-byte instruction whose third byte is a don't care. An example of such an instruction is a 3-byte MOV that targets the FLWR register. When programming in C, the dummy value written to FLWR should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.
3. Flash read current depends on the number of address lines that toggle between sequential flash read operations. In most cases, the difference in power is relatively small (on the order of 5%).
4. The flash memory is organized in rows. Each row in the Si1000/1/2/3/4/5 flash contains 128 bytes. A substantial current increase can be detected when the read address jumps from one row in the flash memory to another. Consider a 3-cycle loop (e.g., SJMP $, or while(1);) which straddles a 128-byte flash row boundary. The flash address jumps from one row to another on two of every three clock cycles. This can result in a current increase of up 30% when compared to the same 3-cycle loop contained entirely within a single row.
5. To minimize the power consumption of small loops, it is best to locate them within a single row, if possible. To check if a loop is contained within a flash row, divide the starting address of the first instruction in the loop by 128. If the remainder (result of modulo operation) plus the length of the loop is less than 127, then the loop fits inside a single flash row. Otherwise, the loop will be straddling two adjacent flash rows. If a loop executes in 20 or more clock cycles, then the transitions from one row to another will occur on relatively few clock cycles, and any resulting increase in operating current will be negligible.

**Note:** Future 16 and 8 kB derivatives in this product family will use a flash memory that is organized in rows of 64 bytes each. To maintain code compatibility across the entire family, it is best to locate small loops within a single 64-byte segment.

## SFR Definition 13.1. PSCTL: Program Store R/W Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | | SFLE | PSEE | PSWE |
| Type | R | R | R | R | R | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page =0x0; SFR Address = 0x8F

| Bit | Name | Function |
|---|---|---|
| 7:3 | Unused | Read = 00000b, Write = don't care. |
| 2 | SFLE | **Scratchpad Flash Memory Access Enable.**<br>When this bit is set, flash MOVC reads and MOVX writes from user software are directed to the Scratchpad flash sector. Flash accesses outside the address range 0x0000-0x03FF should not be attempted and may yield undefined results when SFLE is set to 1.<br>0: Flash access from user software directed to the Program/Data Flash sector.<br>1: Flash access from user software directed to the Scratchpad Sector. |
| 1 | PSEE | **Program Store Erase Enable.**<br>Setting this bit (in combination with PSWE) allows an entire page of flash program memory to be erased. If this bit is logic 1 and flash writes are enabled (PSWE is logic 1), a write to flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter.<br>0: Flash program memory erasure disabled.<br>1: Flash program memory erasure enabled. |
| 0 | PSWE | **Program Store Write Enable.**<br>Setting this bit allows writing a byte of data to the flash program memory using the MOVX write instruction. The flash location should be erased before writing data.<br>0: Writes to flash program memory disabled.<br>1: Writes to flash program memory enabled; the MOVX write instruction targets flash memory. |

## SFR Definition 13.2. FLKEY: Flash Lock and Key

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FLKEY[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB6

| Bit | Name | Function |
|---|---|---|
| 7:0 | FLKEY[7:0] | **Flash Lock and Key Register.**<br>Write:<br>This register provides a lock and key function for flash erasures and writes. Flash writes and erases are enabled by writing 0xA5 followed by 0xF1 to the FLKEY register. Flash writes and erases are automatically disabled after the next write or erase is complete. If any writes to FLKEY are performed incorrectly, or if a flash write or erase operation is attempted while these operations are disabled, the flash will be permanently<br>locked from writes or erasures until the next device reset. If an application never writes to flash, it can intentionally lock the flash by writing a non-0xA5 value to FLKEY from software.<br>Read:<br>When read, bits 1–0 indicate the current flash lock state.<br>00: Flash is write/erase locked.<br>01: The first key code has been written (0xA5).<br>10: Flash is unlocked (writes/erases allowed).<br>11: Flash writes/erases disabled until the next reset. |

## SFR Definition 13.3. FLSCL: Flash Scale

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | BYPASS | | | | | | |
| Type | R | R/W | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB6

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Always Write to 0. |
| 6 | BYPASS | **Flash Read Timing One-Shot Bypass.**<br>0: The one-shot determines the flash read time. This setting should be used for operating frequencies less than 10 MHz.<br>1: The system clock determines the flash read time. This setting should be used for frequencies greater than 10 MHz. |
| 5:0 | Reserved | Always Write to 000000. |

**Note:** When changing the BYPASS bit from 1 to 0, the third opcode byte fetched from program memory is indeterminate. Therefore, the operation which clears the BYPASS bit should be immediately followed by a benign 3-byte instruction whose third byte is a don't care. An example of such an instruction is a 3-byte MOV that targets the FLWR register. When programming in C, the dummy value written to FLWR should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

## SFR Definition 13.4. FLWR: Flash Write Only

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FLWR[7:0] | | | | | | | |
| Type | W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE5

| Bit | Name | Function |
|---|---|---|
| 7:0 | FLWR[7:0] | **Flash Write Only.**<br>All writes to this register have no effect on system operation. |

## 14. Power Management

Si1000/1/2/3/4/5 devices support 5 power modes: Normal, Idle, Stop, Suspend, and Sleep. The power management unit (PMU0) allows the device to enter and wake-up from the available power modes. A brief description of each power mode is provided in Table 14.1. Detailed descriptions of each mode can be found in the following sections.

**Table 14.1. Power Modes**

| Power Mode | Description | Wake-Up Sources | Power Savings |
|---|---|---|---|
| Normal | Device fully functional | N/A | Excellent MIPS/mW |
| Idle | All peripherals fully functional. Very easy to wake up. | Any Interrupt | Good<br>No Code Execution |
| Stop | Legacy 8051 low power mode. A reset is required to wake up. | Any Reset | Good<br>No Code Execution<br>Precision Oscillator Disabled |
| Suspend | Similar to Stop Mode, but very fast wake-up time and code resumes execution at the next instruction. | SmaRTClock, Port Match, Comparator0, $\overline{\text{RST}}$ pin | Very Good<br>No Code Execution<br>All Internal Oscillators Disabled<br>System Clock Gated |
| Sleep | Ultra Low Power and flexible wake-up sources. Code resumes execution at the next instruction. Comparator0 only functional in two-cell mode. | SmaRTClock, Port Match, Comparator0, $\overline{\text{RST}}$ pin | Excellent<br>Power Supply Gated<br>All Oscillators except SmaRT-Clock Disabled |

In battery powered systems, the system should spend as much time as possible in Sleep mode in order to preserve battery life. When a task with a fixed number of clock cycles needs to be performed, the device should switch to Normal mode, finish the task as quickly as possible, and return to Sleep mode. Idle Mode and Suspend modes provide a very fast wake-up time; however, the power savings in these modes will not be as much as in Sleep Mode. Stop Mode is included for legacy reasons; the system will be more power efficient and easier to wake up when Idle, Suspend, or Sleep Mode are used.

Although switching power modes is an integral part of power management, enabling/disabling individual peripherals as needed will help lower power consumption in all power modes. Each analog peripheral can be disabled when not in use or placed in a low power mode. Digital peripherals such as timers or serial busses draw little power whenever they are not in use. Digital peripherals draw no power in Sleep Mode.

## 14.1. Normal Mode

The MCU is fully functional in Normal Mode. Figure 14.1 shows the on-chip power distribution to various peripherals. There are three supply voltages powering various sections of the chip: VBAT, VDD/DC+, and the 1.8 V internal core supply. VREG0, PMU0 and the SmaRTClock are always powered directly from the VBAT pin. All analog peripherals are directly powered from the VDD/DC+ pin, which is an output in one-cell mode and an input in two-cell mode. All digital peripherals and the CIP-51 core are powered from the 1.8 V internal core supply. The RAM is also powered from the core supply in Normal mode.

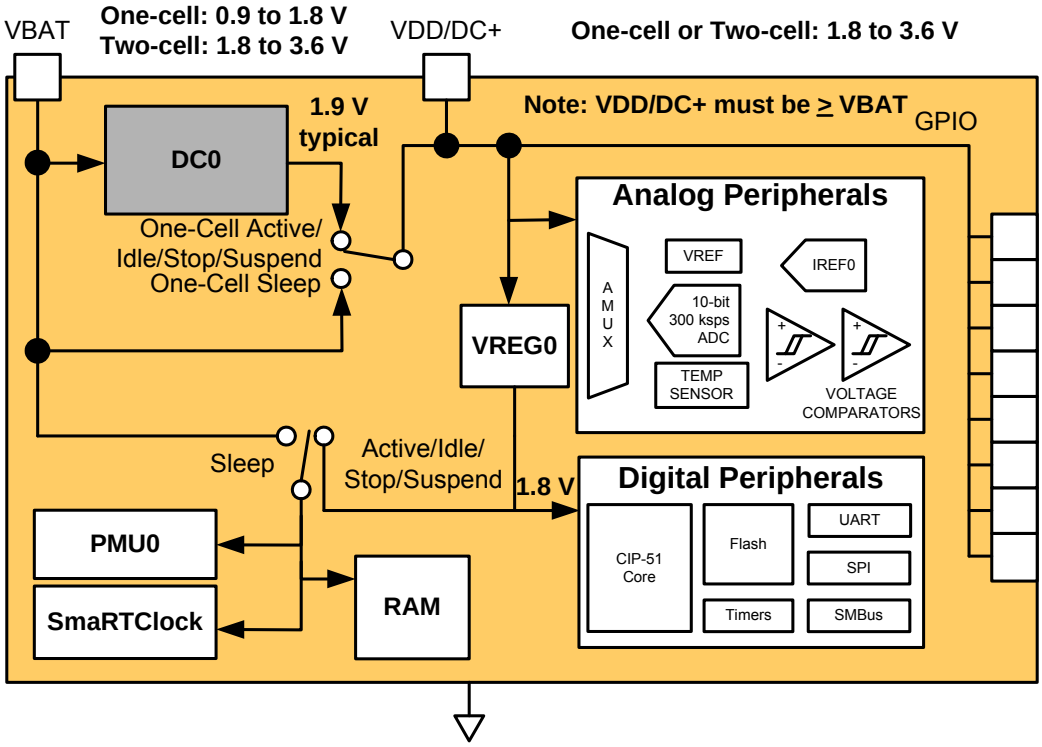



**Figure 14.1. Si1000/1/2/3/4/5 Power Distribution**

## 14.2. Idle Mode

Setting the Idle Mode Select bit (PCON.0) causes the CIP-51 to halt the CPU and enter Idle mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. All analog and digital peripherals can remain active during Idle mode.

**Note:** To ensure the MCU enters a low power state upon entry into Idle Mode, the one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6) to logic 0. See the note in SFR Definition 13.3. FLSCL: Flash Scale for more information on how to properly clear the BYPASS bit.

Idle mode is terminated when an enabled interrupt is asserted or a reset occurs. The assertion of an enabled interrupt will cause the Idle Mode Selection bit (PCON.0) to be cleared and the CPU to resume operation. The pending interrupt will be serviced and the next instruction to be executed after the return from interrupt (RETI) will be the instruction immediately following the one that set the Idle Mode Select bit. If Idle mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

If enabled, the Watchdog Timer (WDT) will eventually cause an internal watchdog reset and thereby terminate the Idle mode. This feature protects the system from an unintended permanent shutdown in the event of an inadvertent write to the PCON register. If this behavior is not desired, the WDT may be disabled by software prior to entering the Idle mode if the WDT was initially configured to allow this operation. This provides the opportunity for additional power savings, allowing the system to remain in the Idle mode indefinitely, waiting for an external stimulus to wake up the system. Refer to Section “18.6. PCA Watchdog Timer Reset” on page 183 for more information on the use and configuration of the WDT.

## 14.3. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the CIP-51 to enter Stop mode as soon as the instruction that sets the bit completes execution. In Stop mode the precision internal oscillator and CPU are stopped; the state of the low power oscillator and the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may be shut down individually prior to entering Stop Mode. Stop mode can only be terminated by an internal or external reset. On reset, the CIP-51 performs the normal reset sequence and begins program execution at address 0x0000.

If enabled, the Missing Clock Detector will cause an internal reset and thereby terminate the Stop mode. The Missing Clock Detector should be disabled if the CPU is to be put to in STOP mode for longer than the MCD timeout of 100 µs.

Stop Mode is a legacy 8051 power mode; it will not result in optimal power savings. Sleep or Suspend mode will provide more power savings if the MCU needs to be inactive for a long period of time.

On Si1000/1/2/3/4/5 devices, the Precision Oscillator Bias is not automatically disabled and should be disabled by software to achieve the lowest possible Stop mode current.

**Note:** To ensure the MCU enters a low power state upon entry into Stop Mode, the one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6) to logic 0. See the note in SFR Definition 13.3. FLSCL: Flash Scale for more information on how to properly clear the BYPASS bit.

## 14.4. Suspend Mode

Setting the Suspend Mode Select bit (PMU0CF.6) causes the system clock to be gated off and all internal oscillators disabled. All digital logic (timers, communication peripherals, interrupts, CPU, etc.) stops functioning until one of the enabled wake-up sources occurs.

**Important Notes:**

- **When entering Suspend Mode, the global clock divider must be set to "divide by 1" by setting CLKDIV[2:0] = 000b in the CLKSEL register.**
- **The one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6) to logic 0. See the note in SFR Definition 13.3. FLSCL: Flash Scale for more information on how to properly clear the BYPASS bit.**
- **Upon wake-up from suspend mode, PMU0 requires two system clocks in order to update the PMU0CF wake-up flags. All flags will read back a value of 0 during the first two system clocks following a wake-up from suspend mode.**
- **The system clock source must be set to the low power internal oscillator or the precision oscillator prior to entering suspend mode.**

The following wake-up sources can be configured to wake the device from suspend mode:

- SmaRTClock Oscillator Fail
- SmaRTClock Alarm
- Port Match Event
- Comparator0 Rising Edge

In addition, a noise glitch on $\overline{\text{RST}}$ that is not long enough to reset the device will cause the device to exit suspend. In order for the MCU to respond to the pin reset event, software must not place the device back into suspend mode for a period of 15 µs. The PMU0CF register may be checked to determine if the wake-up was due to a falling edge on the /RST pin. If the wake-up source is not due to a falling edge on $\overline{\text{RST}}$, there is no time restriction on how soon software may place the device back into suspend mode. A 4.7 kΩ pullup resistor to VDD_MCU/DC+ is recommend for RST to prevent noise glitches from waking the device.

## 14.5. Sleep Mode

Setting the Sleep Mode Select bit (PMU0CF.6) turns off the internal 1.8 V regulator (VREG0) and switches the power supply of all on-chip RAM to the VDD_MCU pin (see Figure 14.1). Power to most digital logic on the chip is disconnected; only PMU0 and the SmaRTClock remain powered. Analog peripherals remain powered. The Comparators remain functional when the device enters sleep mode. All other analog peripherals (ADC0, IREF0, External Oscillator, etc.) should be disabled prior to entering sleep mode. The system clock source must be set to the low power internal oscillator or the precision oscillator prior to entering sleep mode.

**Important Notes:**

- **When entering Sleep Mode, the global clock divider must be set to "divide by 1" by setting CLKDIV[2:0] = 000b in the CLKSEL register.**
- Any write to PMU0CF which places the device in sleep mode should be immediately followed by two NOP instructions. Software that does not place two NOP instructions immediately following the write to PMU0CF should continue to behave the same way as during software development.

GPIO pins configured as digital outputs will retain their output state during sleep mode. In two-cell mode, they will maintain the same current drive capability in sleep mode as they have in normal mode. In one-cell mode, the VDD_MCU/DC+ supply will drop to the level of VBAT, which will reduce the output high-voltage level and the source and sink current drive capability.

GPIO pins configured as digital inputs can be used during sleep mode as wakeup sources using the port match feature. In two-cell mode, they will maintain the same input level specifications in sleep mode as they have in normal mode. In one-cell mode, the VDD supply will drop to the level of VBAT, which will lower the switching threshold and increase the propagation delay.

**Note:** By default, the VDD/DC+ supply is connected to VBAT upon entry into Sleep Mode (one-cell mode). If the VDDSLP bit (DC0CF.1) is set to logic 1, the VDD/DC+ supply will float in Sleep Mode. This allows the decoupling capacitance on the VDD/DC+ supply to maintain the supply rail until the capacitors are discharged. For relatively short sleep intervals, this can result in substantial power savings because the decoupling capacitance is not continuously charged and discharged.

RAM and SFR register contents are preserved in sleep mode as long as the voltage on VBAT (or VDD_MCU on Si1000/1/2/3 devices) does not fall below $V_{POR}$. The PC counter and all other volatile state information is preserved allowing the device to resume code execution upon waking up from sleep mode. The following wake-up sources can be configured to wake the device from sleep mode:

- SmaRTClock Oscillator Fail
- SmaRTClock Alarm
- Port Match Event
- Comparator0 Rising Edge

The Comparator0 Rising Edge wakeup is only valid in two-cell mode. The comparator requires a supply voltage of at least 1.8 V to operate properly.

In addition, any falling edge on $\overline{\text{RST}}$ (due to a pin reset or a noise glitch) will cause the device to exit sleep mode. In order for the MCU to respond to the pin reset event, software must not place the device back into sleep mode for a period of 15 µs. The PMU0CF register may be checked to determine if the wake-up was due to a falling edge on the $\overline{\text{RST}}$ pin. If the wake-up source is not due to a falling edge on $\overline{\text{RST}}$, there is no time restriction on how soon software may place the device back into sleep mode. A 4.7 kΩ pullup resistor to VDD_MCU/DC+ is recommend for $\overline{\text{RST}}$ to prevent noise glitches from waking the device.

## 14.6. Configuring Wakeup Sources

Before placing the device in a low power mode, one or more wakeup sources should be enabled so that the device does not remain in the low power mode indefinitely. For Idle Mode, this includes enabling any interrupt. For stop mode, this includes enabling any reset source or relying on the $\overline{\text{RST}}$ pin to reset the device.

Wake-up sources for suspend and sleep modes are configured through the PMU0CF register. Wake-up sources are enabled by writing 1 to the corresponding wake-up source enable bit. Wake-up sources must be re-enabled each time the device is placed in suspend or sleep mode, in the same write that places the device in the low power mode.

The reset pin is always enabled as a wake-up source. On the falling edge of $\overline{\text{RST}}$, the device will be awaken from sleep mode. The device must remain awake for more than 15 µs in order for the reset to take place.

## 14.7. Determining the Event that Caused the Last Wakeup

When waking from Idle Mode, the CPU will vector to the interrupt which caused it to wake up. When waking from Stop mode, the RSTSRC register may be read to determine the cause of the last reset.

Upon exit from Suspend or Sleep mode, the wake-up flags in the PMU0CF register can be read to determine the event which caused the device to wake up. After waking up, the wake-up flags will continue to be updated if any of the wake-up events occur. Wake-up flags are always updated, even if they are not enabled as wake-up sources.

All wake-up flags enabled as wake-up sources in PMU0CF must be cleared before the device can enter suspend or sleep mode. After clearing the wake-up flags, each of the enabled wake-up events should be checked in the individual peripherals to ensure that a wake-up event did not occur while the wake-up flags were being cleared.

## SFR Definition 14.1. PMU0CF: Power Management Unit Configuration[1,2]

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SLEEP | SUSPEND | CLEAR | RSTWK | RTCFWK | RTCAWK | PMATWK | CPT0WK |
| Type | W | W | W | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | Varies | Varies | Varies | Varies | Varies |

SFR Page = 0x0; SFR Address = 0xB5

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7 | SLEEP | **Sleep Mode Select** | Writing 1 places the device in Sleep Mode. | N/A |
| 6 | SUSPEND | **Suspend Mode Select** | Writing 1 places the device in Suspend Mode. | N/A |
| 5 | CLEAR | **Wake-up Flag Clear** | Writing 1 clears all wake-up flags. | N/A |
| 4 | RSTWK | **Reset Pin Wake-up Flag** | N/A | Set to 1 if a falling edge has been detected on $\overline{RST}$. |
| 3 | RTCFWK | **SmaRTClock Oscillator Fail Wake-up Source Enable and Flag** | 0: Disable wake-up on SmaRTClock Osc. Fail.<br>1: Enable wake-up on SmaRTClock Osc. Fail. | Set to 1 if the SmaRTClock Oscillator has failed. |
| 2 | RTCAWK | **SmaRTClock Alarm Wake-up Source Enable and Flag** | 0: Disable wake-up on SmaRTClock Alarm.<br>1: Enable wake-up on SmaRTClock Alarm. | Set to 1 if a SmaRTClock Alarm has occurred. |
| 1 | PMATWK | **Port Match Wake-up Source Enable and Flag** | 0: Disable wake-up on Port Match Event.<br>1: Enable wake-up on Port Match Event. | Set to 1 if a Port Match Event has occurred. |
| 0 | CPT0WK | **Comparator0 Wake-up Source Enable and Flag** | 0: Disable wake-up on Comparator0 rising edge.<br>1: Enable wake-up on Comparator0 rising edge. | Set to 1 if Comparator0 rising edge caused the last wake-up. |

**Notes:**

1. Read-modify-write operations (ORL, ANL, etc.) should not be used on this register. Wake-up sources must be re-enabled each time the SLEEP or SUSPEND bits are written to 1.
2. The Low Power Internal Oscillator cannot be disabled and the MCU cannot be placed in Suspend or Sleep Mode if any wake-up flags are set to 1. Software should clear all wake-up sources after each reset and after each wake-up from Suspend or Sleep Modes.

## SFR Definition 14.2. PCON: Power Management Control Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GF[5:0] | | | | | | STOP | IDLE |
| Type | R/W | | | | | | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0x87

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:2 | GF[5:0] | **General Purpose Flags** | Sets the logic value. | Returns the logic value. |
| 1 | STOP | **Stop Mode Select** | Writing 1 places the device in Stop Mode. | N/A |
| 0 | IDLE | **Idle Mode Select** | Writing 1 places the device in Idle Mode. | N/A |

## 14.8. Power Management Specifications

See Table 4.5 on page 57 for detailed Power Management Specifications.

# 15. Cyclic Redundancy Check Unit (CRC0)

Si1000/1/2/3/4/5 devices include a cyclic redundancy check unit (CRC0) that can perform a CRC using a 16-bit or 32-bit polynomial. CRC0 accepts a stream of 8-bit data written to the CRC0IN register. CRC0 posts the 16-bit or 32-bit result to an internal register. The internal result register may be accessed indirectly using the CRC0PNT bits and CRC0DAT register, as shown in Figure 15.1. CRC0 also has a bit reverse register for quick data manipulation.

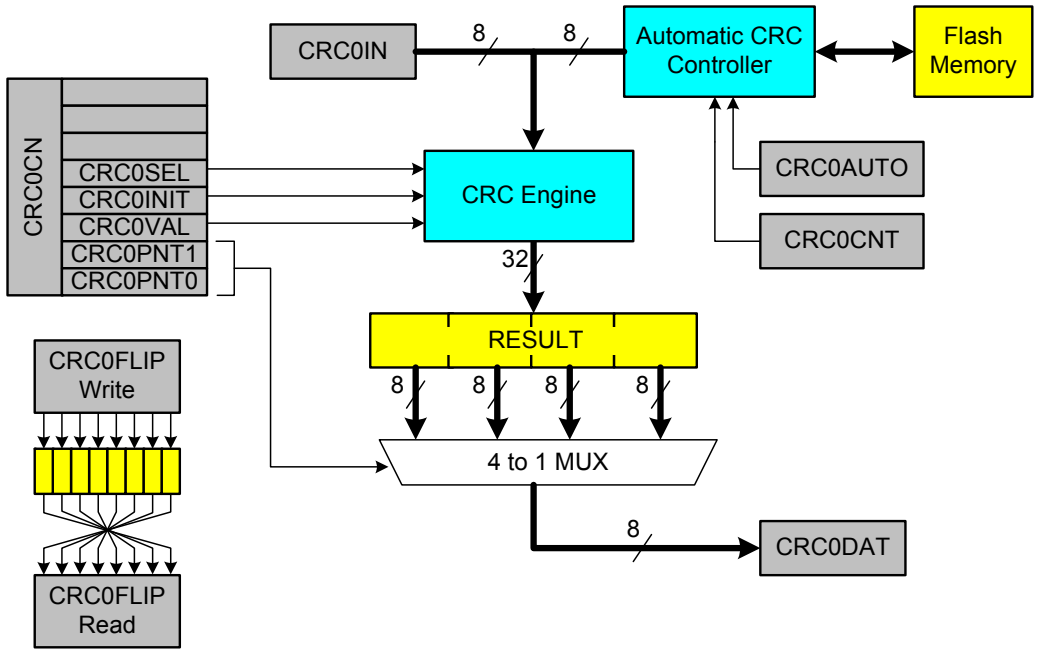



**Figure 15.1. CRC0 Block Diagram**

## 15.1. 16-bit CRC Algorithm

The Si1000/1/2/3/4/5 CRC unit calculates the 16-bit CRC MSB-first, using a poly of 0x1021. The following describes the 16-bit CRC algorithm performed by the hardware:

1. XOR the input with the most-significant bits of the current CRC result. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x0000 or 0xFFFF).

2a. If the MSB of the CRC result is set, left-shift the CRC result and XOR the result with the selected polynomial (0x1021).

2b. If the MSB of the CRC result is not set, left-shift the CRC result.

Repeat Steps 2a/2b for the number of input bits (8). The algorithm is also described in the following example.

The 16-bit Si1000/1/2/3/4/5 CRC algorithm can be described by the following code:

```
unsigned short UpdateCRC (unsigned short CRC_acc, unsigned char CRC_input)
{
   unsigned char i;                    // loop counter

   #define POLY 0x1021
```

```
   // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
   // with no carries)
   CRC_acc = CRC_acc ^ (CRC_input << 8);

   // "Divide" the poly into the dividend using CRC XOR subtraction
   // CRC_acc holds the "remainder" of each divide
   //
   // Only complete this division for 8 bits since input is 1 byte
   for (i = 0; i < 8; i++)
   {
      // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
      // into the "dividend")
      if ((CRC_acc & 0x8000) == 0x8000)
      {
         // if so, shift the CRC value, and XOR "subtract" the poly
         CRC_acc = CRC_acc << 1;
         CRC_acc ^= POLY;
      }
      else
      {
         // if not, just shift the CRC value
         CRC_acc = CRC_acc << 1;
      }
   }

   // Return the final remainder (CRC value)
   return CRC_acc;
}
```

The following table lists several input values and the associated outputs using the 16-bit Si1000/1/2/3/4/5 CRC algorithm:

**Table 15.1. Example 16-bit CRC Outputs**

| Input | Output |
|---|---|
| 0x63 | 0xBD35 |
| 0x8C | 0xB1F4 |
| 0x7D | 0x4ECA |
| 0xAA, 0xBB, 0xCC | 0x6CF6 |
| 0x00, 0x00, 0xAA, 0xBB, 0xCC | 0xB166 |

## 15.2. 32-bit CRC Algorithm

The Si1000 CRC unit calculates the 32-bit CRC using a poly of 0x04C11DB7. The CRC-32 algorithm is "reflected", meaning that all of the input bytes and the final 32-bit output are bit-reversed in the processing engine. The following is a description of a simplified CRC algorithm that produces results identical to the hardware:

Step 1. XOR the least-significant byte of the current CRC result with the input byte. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x00000000 or 0xFFFFFFFF).
Step 2. Right-shift the CRC result.
Step 3. If the LSB of the CRC result is set, XOR the CRC result with the reflected polynomial (0xEDB88320).
Step 4. Repeat at Step 2 for the number of input bits (8).

For example, the 32-bit Si1000 CRC algorithm can be described by the following code:

```
unsigned long UpdateCRC (unsigned long CRC_acc, unsigned char CRC_input)
{
   unsigned char i; // loop counter
   #define POLY 0xEDB88320 // bit-reversed version of the poly 0x04C11DB7
   // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
   // with no carries)

   CRC_acc = CRC_acc ^ CRC_input;

   // "Divide" the poly into the dividend using CRC XOR subtraction
   // CRC_acc holds the "remainder" of each divide
   //
   // Only complete this division for 8 bits since input is 1 byte
   for (i = 0; i < 8; i++)
   {
      // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
      // into the "dividend")
      if ((CRC_acc & 0x00000001) == 0x00000001)
      {
         // if so, shift the CRC value, and XOR "subtract" the poly
         CRC_acc = CRC_acc >> 1;
         CRC_acc ^= POLY;
      }
      else
      {
         // if not, just shift the CRC value
         CRC_acc = CRC_acc >> 1;
      }
   }
   // Return the final remainder (CRC value)
   return CRC_acc;
}
```

The following table lists several input values and the associated outputs using the 32-bit Si1000 CRC algorithm (an initial value of 0xFFFFFFFF is used):

**Table 15.2. Example 32-bit CRC Outputs**

| Input | Output |
|---|---|
| 0x63 | 0xF9462090 |
| 0xAA, 0xBB, 0xCC | 0x41B207B3 |
| 0x00, 0x00, 0xAA, 0xBB, 0xCC | 0x78D129BC |

## 15.3. Preparing for a CRC Calculation

To prepare CRC0 for a CRC calculation, software should select the desired polynomial and set the initial value of the result. Two polynomials are available: 0x1021 (16-bit) and 0x04C11DB7 (32-bit). The CRC0 result may be initialized to one of two values: 0x00000000 or 0xFFFFFFFF. The following steps can be used to initialize CRC0.

1. Select a polynomial (Set CRC0SEL to 0 for 32-bit or 1 for 16-bit).
2. Select the initial result value (Set CRC0VAL to 0 for 0x00000000 or 1 for 0xFFFFFFFF).
3. Set the result to its initial value (Write 1 to CRC0INIT).

## 15.4. Performing a CRC Calculation

Once CRC0 is initialized, the input data stream is sequentially written to CRC0IN, one byte at a time. The CRC0 result is automatically updated after each byte is written. The CRC engine may also be configured to automatically perform a CRC on one or more Flash sectors. The following steps can be used to automatically perform a CRC on Flash memory.

1. Prepare CRC0 for a CRC calculation as shown above.
2. Write the index of the starting page to CRC0AUTO.
3. Set the AUTOEN bit in CRC0AUTO.
4. Write the number of Flash sectors to perform in the CRC calculation to CRC0CNT.
   Note: Each Flash sector is 1024 bytes.
5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes.
6. After initiating an automatic CRC calculation, the third opcode byte fetched from program memory is indeterminate. Therefore, writes to CRC0CN that initiate a CRC operation must be immediately followed by a benign 3-byte instruction whose third byte is a don't care. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in C, the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.
7. Clear the AUTOEN bit in CRC0AUTO.
8. Read the CRC result using the procedure below.

## 15.5. Accessing the CRC0 Result

The internal CRC0 result is 32-bits (CRC0SEL = 0b) or 16-bits (CRC0SEL = 1b). The CRC0PNT bits select the byte that is targeted by read and write operations on CRC0DAT and increment after each read or write. The calculation result will remain in the internal CR0 result register until it is set, overwritten, or additional data is written to CRC0IN.

## SFR Definition 15.1. CRC0CN: CRC0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | CRC0SEL | CRC0INIT | CRC0VAL | CRC0PNT[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x92

| Bit | Name | Function |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't Care. |
| 4 | CRC0SEL | **CRC0 Polynomial Select Bit.**<br>This bit selects the CRC0 polynomial and result length (32-bit or 16-bit).<br>0: CRC0 uses the 32-bit polynomial 0x04C11DB7 for calculating the CRC result.<br>1: CRC0 uses the 16-bit polynomial 0x1021 for calculating the CRC result. |
| 3 | CRC0INIT | **CRC0 Result Initialization Bit.**<br>Writing a 1 to this bit initializes the entire CRC result based on CRC0VAL. |
| 2 | CRC0VAL | **CRC0 Set Value Initialization Bit.**<br>This bit selects the set value of the CRC result.<br>0: CRC result is set to 0x00000000 on write of 1 to CRC0INIT.<br>1: CRC result is set to 0xFFFFFFFF on write of 1 to CRC0INIT. |
| 1:0 | CRC0PNT[1:0] | **CRC0 Result Pointer.**<br>Specifies the byte of the CRC result to be read/written on the next access to CRC0DAT. The value of these bits will auto-increment upon each read or write.<br>For CRC0SEL = 0:<br>00: CRC0DAT accesses bits 7–0 of the 32-bit CRC result.<br>01: CRC0DAT accesses bits 15–8 of the 32-bit CRC result.<br>10: CRC0DAT accesses bits 23–16 of the 32-bit CRC result.<br>11: CRC0DAT accesses bits 31–24 of the 32-bit CRC result.<br>For CRC0SEL = 1:<br>00: CRC0DAT accesses bits 7–0 of the 16-bit CRC result.<br>01: CRC0DAT accesses bits 15–8 of the 16-bit CRC result.<br>10: CRC0DAT accesses bits 7–0 of the 16-bit CRC result.<br>11: CRC0DAT accesses bits 15–8 of the 16-bit CRC result. |

**Note:** Upon initiation of an automatic CRC calculation, the third opcode byte fetched from program memory is indeterminate. Therefore, writes to CRC0CN that initiate a CRC operation must be immediately followed by a benign 3-byte instruction whose third byte is a don't care. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in 'C', the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

## SFR Definition 15.2. CRC0IN: CRC0 Data Input

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0IN[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x93

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0IN[7:0] | **CRC0 Data Input.**<br>Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm described in Section 15.1 |

## SFR Definition 15.3. CRC0DAT: CRC0 Data Output

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x91

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0DAT[7:0] | **CRC0 Data Output.**<br>Each read or write performed on CRC0DAT targets the CRC result bits pointed to by the CRC0 Result Pointer (CRC0PNT bits in CRC0CN). |

## SFR Definition 15.4. CRC0AUTO: CRC0 Automatic Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AUTOEN | CRCDONE | CRC0ST[5:0] | | | | | |
| Type | R/W | | | | | | | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x96

| Bit | Name | Function |
|---|---|---|
| 7 | AUTOEN | **Automatic CRC Calculation Enable.**<br>When AUTOEN is set to 1, any write to CRC0CN will initiate an automatic CRC starting at Flash sector CRC0ST and continuing for CRC0CNT sectors. |
| 6 | CRCDONE | **CRCDONE Automatic CRC Calculation Complete.**<br>Set to '0' when a CRC calculation is in progress. Note that code execution is stopped during a CRC calculation, therefore reads from firmware will always return '1'. |
| 5:0 | CRC0ST[5:0] | **Automatic CRC Calculation Starting Flash Sector.**<br>These bits specify the Flash sector to start the automatic CRC calculation. The starting address of the first Flash sector included in the automatic CRC calculation is CRC0ST x 1024. |

## SFR Definition 15.5. CRC0CNT: CRC0 Automatic Flash Sector Count

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | CRC0CNT[5:0] | | | | | |
| Type | R/W | | | | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x97

| Bit | Name | Function |
|---|---|---|
| 7:6 | Unused | Read = 00b; Write = Don’t Care. |
| 5:0 | CRC0CNT[5:0] | **Automatic CRC Calculation Flash Sector Count.**<br>These bits specify the number of Flash sectors to include in an automatic CRC calculation. The starting address of the last Flash sector included in the automatic CRC calculation is (CRC0ST+CRC0CNT) x 1024. |

## 15.6. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 15.2. Each byte of data written to CRC0FLIP is read back bit reversed. For example, if 0xC0 is written to CRC0FLIP, the data read back is 0x03. Bit reversal is a useful mathematical function used in algorithms such as the FFT.

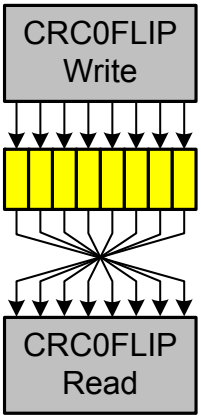



**Figure 15.2. Bit Reverse Register**

### SFR Definition 15.6. CRC0FLIP: CRC0 Bit Flip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0FLIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x95

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0FLIP[7:0] | **CRC0 Bit Flip.**<br>Any byte written to CRC0FLIP is read back in a bit-reversed order, i.e. the written LSB becomes the MSB. For example:<br>If 0xC0 is written to CRC0FLIP, the data read back will be 0x03.<br>If 0x05 is written to CRC0FLIP, the data read back will be 0xA0. |

## 16. On-Chip DC-DC Converter (DC0)

Si1004/5 devices include an on-chip dc-dc converter to allow operation from a single cell battery with a supply voltage as low as 0.9 V. The dc-dc converter is a switching boost converter with an input voltage range of 0.9 to 1.8 V and a programmable output voltage range of 1.8 to 3.3 V. The default output voltage is 1.9 V. The dc-dc converter can supply the system with up to 65 mW of regulated power (or up to 100 mW in some applications) and can be used for powering other devices in the system. This allows the most flexibility when interfacing to sensors and other analog signals which typically require a higher supply voltage than a single-cell battery can provide.

Figure 16.1 shows a block diagram of the dc-dc converter. During normal operation in the first half of the switching cycle, the Duty Cycle Control switch is closed and the Diode Bypass switch is open. Since the output voltage is higher than the voltage at the DCEN pin, no current flows through the diode and the load is powered from the output capacitor. During this stage, the DCEN pin is connected to ground through the Duty Cycle Control switch, generating a positive voltage across the inductor and forcing its current to ramp up.

In the second half of the switching cycle, the Duty Cycle control switch is opened and the Diode Bypass switch is closed. This connects DCEN directly to VDD_MCU/DC+ and forces the inductor current to charge the output capacitor. Once the inductor transfers its stored energy to the output capacitor, the Duty Cycle Control switch is closed, the Diode Bypass switch is opened, and the cycle repeats.

The dc-dc converter has a built in voltage reference and oscillator, and will automatically limit or turn off the switching activity in case the peak inductor current rises beyond a safe limit or the output voltage rises above the programmed target value. This allows the dc-dc converter output to be safely overdriven by a secondary power source (when available) in order to preserve battery life. The dc-dc converter's settings can be modified using SFR registers which provide the ability to change the target output voltage, oscillator frequency or source, Diode Bypass switch resistance, peak inductor current, and minimum duty cycle.

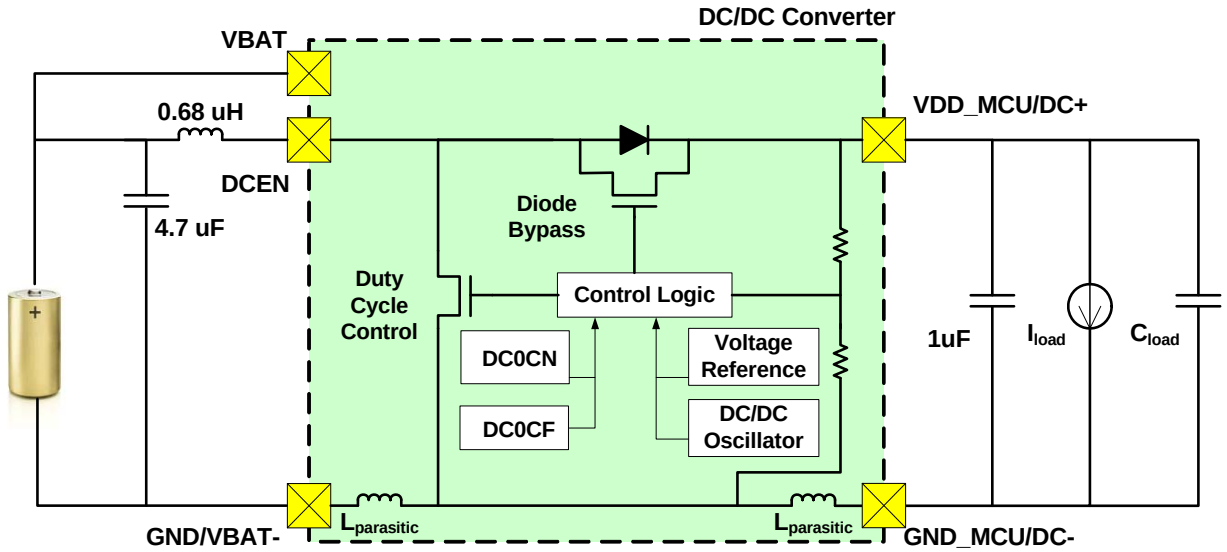



**Figure 16.1. DC-DC Converter Block Diagram**

## 16.1. Startup Behavior

On initial power-on, the dc-dc converter outputs a constant 50% duty cycle until there is sufficient voltage on the output capacitor to maintain regulation. The size of the output capacitor and the amount of load current present during startup will determine the length of time it takes to charge the output capacitor.

During initial power-on reset, the maximum peak inductor current threshold, which triggers the overcurrent protection circuit, is set to approximately 125 mA. This generates a “soft-start” to limit the output voltage slew rate and prevent excessive in-rush current at the output capacitor. In order to ensure reliable startup of the dc-dc converter, the following restrictions have been imposed:

- The maximum dc load current allowed during startup is given in Table 4.15 on page 65. If the dc-dc converter is powering external sensors or devices through the VDD_MCU/DC+ pin or through GPIO pins, then the current supplied to these sensors or devices is counted towards this limit. The in-rush current into capacitors does not count towards this limit.

- The maximum total output capacitance is given in Table 4.15 on page 65. This value includes the required 1 µF ceramic output capacitor and any additional capacitance connected to the VDD_MCU/DC+ pin.

Once initial power-on is complete, the peak inductor current limit can be increased by software as shown in Table 16.1. Limiting the peak inductor current can allow the device to start up near the battery’s end of life.

**Table 16.1. IPeak Inductor Current Limit Settings**

| SWSEL | ILIMIT | Peak Current (mA) |
|---|---|---|
| 1 | 0 | 100 |
| 0 | 0 | 125 |
| 1 | 1 | 250 |
| 0 | 1 | 500 |

The peak inductor current is dependent on several factors including the dc load current and can be estimated using following equation:

$$I_{PK} = \sqrt{\frac{2\ I_{LOAD}(VDD/\mathrm{DC+} - VBAT)}{efficiency \times \mathrm{inductance} \times frequency}}$$

efficiency = 0.80

inductance = 0.68 µH

frequency = 2.4 MHz

## 16.2. High Power Applications

The dc-dc converter is designed to provide the system with 65 mW of output power, however, it can safely provide up to 100 mW of output power without any risk of damage to the device. For high power applications, the system should be carefully designed to prevent unwanted VBAT and VDD_MCU/DC+ Supply Monitor resets, which are more likely to occur when the dc-dc converter output power exceeds 65mW. In addition, output power above 65 mW causes the dc-dc converter to have relaxed output regulation, high output ripple and more analog noise. At high output power, an inductor with low DC resistance should be chosen in order to minimize power loss and maximize efficiency.

The combination of high output power and low input voltage will result in very high peak and average inductor currents. If the power supply has a high internal resistance, the transient voltage on the VBAT terminal could drop below 0.9 V and trigger a VBAT Supply Monitor Reset, even if the open-circuit voltage is well above the 0.9 V threshold. While this problem is most often associated with operation from very small batteries or batteries that are near the end of their useful life, it can also occur when using bench power supplies that have a slow transient response; the supply's display may indicate a voltage above 0.9 V, but the minimum voltage on the VBAT pin may be lower. A similar problem can occur at the output of the dc-dc converter: using the default low current limit setting (125 mA) can trigger $V_{DD}$ Supply Monitor resets if there is a high transient load current, particularly if the programmed output voltage is at or near 1.8 V.

## 16.3. Pulse Skipping Mode

The dc-dc converter allows the user to set the minimum pulse width such that if the duty cycle needs to decrease below a certain width in order to maintain regulation, an entire "clock pulse" will be skipped.

Pulse skipping can provide substantial power savings, particularly at low values of load current. The converter will continue to maintain a minimum output voltage at its programmed value when pulse skipping is employed, though the output voltage ripple can be higher. Another consideration is that the dc-dc will operate with pulse-frequency modulation rather than pulse-width modulation, which makes the switching frequency spectrum less predictable; this could be an issue if the dc-dc converter is used to power a radio. Figure 4.5 and Figure 4.6 on page 48 and page 49 show the effect of pulse skipping on power consumption.

## 16.4. Enabling the DC-DC Converter

On power-on reset, the state of the DCEN pin is sampled to determine if the device will power up in one-cell or two-cell mode. In two-cell mode, the dc-dc converter always remains disabled. In one-cell mode, the dc-dc converter remains disabled in Sleep Mode, and enabled in all other power modes. See Section "14. Power Management" on page 152 for complete details on available power modes.

The dc-dc converter is enabled (one-cell mode) in hardware by placing a 0.68 µH inductor between DCEN and VBAT. The dc-dc converter is disabled (two-cell mode) by shorting DCEN directly to GND. The DCEN pin should never be left floating. Note that the device can only switch between one-cell and two-cell mode during a power-on reset. See Section "18. Reset Sources" on page 178 for more information regarding reset behavior.

Figure 16.2 shows the two dc-dc converter configuration options.

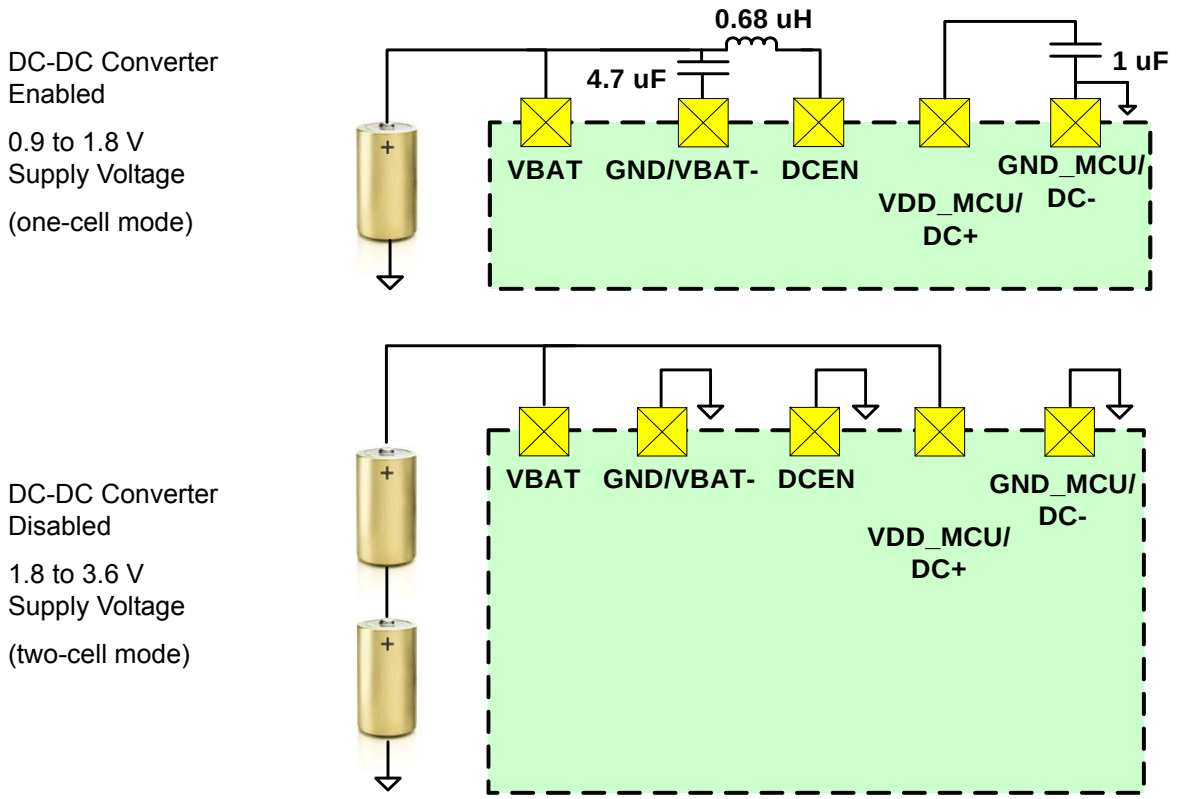



**Figure 16.2. DC-DC Converter Configuration Options**

When the dc-dc converter “Enabled” configuration (one-cell mode) is chosen, the following guidelines apply:

- In most cases, the GND/VBAT– pin should not be externally connected to GND.
- The 0.68 µH inductor should be placed as close as possible to the DCEN pin for maximum efficiency.
- The 4.7 µF capacitor should be placed as close as possible to the inductor.
- The current loop including GND/VBAT–, the 4.7 µF capacitor, the 0.68 µH inductor and the DCEN pin should be made as short as possible to minimize capacitance.
- The PCB traces connecting VDD_MCU/DC+ to the output capacitor and the output capacitor to GND_MCU/DC– should be as short and as thick as possible in order to minimize parasitic inductance.

## 16.5. Minimizing Power Supply Noise

To minimize noise on the power supply lines, the GND/VBAT– and GND_MCU/DC- pins should be kept separate, as shown in Figure 16.2; GND_MCU/DC- should be connected to the pc board ground plane.

The large decoupling capacitors in the input and output circuits ensure that each supply is relatively quiet with respect to its own ground. However, connecting a circuit element "diagonally" (e.g., connecting an external chip between VDD_MCU/DC+ and GND/VBAT-, or between VBAT and GND_MCU/DC-) can result in high supply noise across that circuit element.

To accommodate situations in which ADC0 is sampling a signal that is referenced to one of the external grounds, we recommend using the Analog Ground Reference (P0.1/AGND) option described in Section 5.12. This option prevents any voltage differences between the internal chip ground and the external grounds from modulating the ADC input signal. If this option is enabled, the P0.1 pin should be tied to the ground reference of the external analog input signal. When using the ADC with the dc-dc converter, we also recommend enabling the SYNC bit in the DC0CN register to minimize interference.

These general guidelines provide the best performance in most applications, though some situations may benefit from experimentation to eliminate any residual noise issues. Examples might include tying the grounds together, using additional low-inductance decoupling caps in parallel with the recommended ones, investigating the effects of different dc-dc converter settings, etc.

## 16.6. Selecting the Optimum Switch Size

The dc-dc converter has two built-in switches (the diode bypass switch and duty cycle control switch). To maximize efficiency, one of two switch sizes may be selected. The large switches are ideal for carrying high currents and the small switches are ideal for low current applications. The ideal switchover point to switch from the small switches to the large switches varies with the programmed output voltage. At an output voltage of 2 V, the ideal switchover point is at approximately 4 mA total output current. At an output voltage of 3 V, the ideal switchover point is at approximately 8 mA total output current.

## 16.7. DC-DC Converter Clocking Options

The dc-dc converter may be clocked from its internal oscillator, or from any system clock source, selectable by the CLKSEL bit (DC0CF.0). The dc-dc converter internal oscillator frequency is approximately 2.4 MHz. For a more accurate clock source, the system clock, or a divided version of the system clock may be used as the dc-dc clock source. The dc-dc converter has a built in clock divider (configured using DC0CF[6:5]) which allows any system clock frequency over 1.6 MHz to generate a valid clock in the range of 1.6 to 3.2 MHz.

When the precision internal oscillator is selected as the system clock source, the OSCICL register may be used to fine tune the oscillator frequency and the dc-dc converter clock. The oscillator frequency should only be decreased since it is factory calibrated at its maximum frequency. The minimum frequency which can be reached by the oscillator after taking into account process variations is approximately 16 MHz. The system clock routed to the dc-dc converter clock divider also may be inverted by setting the CLKINV bit (DC0CF.3) to logic 1. These options can be used to minimize interference in noise sensitive applications.

## 16.8. DC-DC Converter Behavior in Sleep Mode

When the Si1000/1/2/3/4/5 devices are placed in Sleep mode, the dc-dc converter is disabled, and the VDD_MCU/DC+ output is internally connected to VBAT by default. This behavior ensures that the GPIO pins are powered from a low-impedance source during sleep mode. If the GPIO pins are not used as inputs or outputs during sleep mode, then the VDD_MCU/DC+ output can be made to float during Sleep mode by setting the VDDSLP bit in the DC0CF register to 1.

Setting this bit can provide power savings in two ways. First, if the sleep interval is relatively short and the VDD_MCU/DC+ load current (include leakage currents) is negligible, then the capacitor on VDD_MCU/DC+ will maintain the output voltage near the programmed value, which means that the VDD_MCU/DC+ capacitor will not need to be recharged upon every wake up event. The second power advantage is that internal or external low-power circuits that require more than 1.8 V can continue to function during Sleep mode without operating the dc-dc converter, powered by the energy stored in the 1 µF output decoupling capacitor. For example, the Si1004/5 comparators require about 0.4 µA when operating in their lowest power mode. If the dc-dc converter output were increased to 3.3 V just before putting the device into Sleep mode, then the comparator could be powered for more than 3 seconds before the output voltage dropped to 1.8 V. In this example, the overall energy consumption would be much lower than if the dc-dc converter were kept running to power the comparator.

If the load current on VDD_MCU/DC+ is high enough to discharge the VDD_MCU/DC+ capacitance to a voltage lower than VBAT during the sleep interval, an internal diode will prevent VDD_MCU/DC+ from dropping more than a few hundred millivolts below VBAT. There may be some additional leakage current from VBAT to ground when the VDD_MCU/DC+ level falls below VBAT, but this leakage current should be small compared to the current from VDD_MCU/DC+.

The amount of time that it takes for a device configured in one-cell mode to wake up from Sleep mode depends on a number of factors, including the dc-dc converter clock speed, the settings of the SWSEL and ILIMIT bits, the battery internal resistance, the load current, and the difference between the VBAT voltage level and the programmed output voltage. The wake up time can be as short as 2 µs, though it is more commonly in the range of 5 to 10 µs, and it can exceed 50 µs under extreme conditions.

See Section “14. Power Management” on page 152 for more information about sleep mode.

## 16.9. DC-DC Converter Register Descriptions

The SFRs used to configure the dc-dc converter are described in the following register descriptions. The reset values for these registers can be used as-is in most systems; therefore, no software intervention or initialization is required.

### SFR Definition 16.1. DC0CN: DC-DC Converter Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | MINPW | | SWSEL | Reserved | SYNC | VSEL | | |
| Type | R/W | | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

SFR Page = 0x0; SFR Address = 0x97

| Bit | Name | Function |
|---|---|---|
| 7:6 | MINPW[1:0] | **DC-DC Converter Minimum Pulse Width.**<br>Specifies the minimum pulse width.<br>00: No minimum duty cycle.<br>01: Minimum pulse width is 20 ns.<br>10: Minimum pulse width is 40 ns.<br>11: Minimum pulse width is 80 ns. |
| 5 | SWSEL | **DC-DC Converter Switch Select.**<br>Selects one of two possible converter switch sizes to maximize efficiency.<br>0: The large switches are selected (best efficiency for high output currents).<br>1: The small switches are selected (best efficiency for low output currents). |
| 4 | Reserved | Always Write to 0. |
| 3 | SYNC | **ADC0 Synchronization Enable.**<br>When synchronization is enabled, the ADC0SC[4:0] bits in the ADC0CF register must be set to 00000b. Behavior as described is valid in REVC and later devices.<br>0: The ADC is not synchronized to the dc-dc converter.<br>1: The ADC is synchronized to the dc-dc converter. ADC0 tracking is performed during the longest quiet time of the dc-dc converter switching cycle and ADC0 SAR clock is also synchronized to the dc-dc converter switching cycle. |
| 2:0 | VSEL[2:0] | **DC-DC Converter Output Voltage Select.**<br>Specifies the target output voltage.<br>000: Target output voltage is 1.8 V.<br>001: Target output voltage is 1.9 V.<br>010: Target output voltage is 2.0 V.<br>011: Target output voltage is 2.1 V.<br>100: Target output voltage is 2.4 V.<br>101: Target output voltage is 2.7 V.<br>110: Target output voltage is 3.0 V.<br>111: Target output voltage is 3.3 V. |

## SFR Definition 16.2. DC0CF: DC-DC Converter Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | CLKDIV[1:0] | | AD0CKINV | CLKINV | ILIMIT | VDDSLP | CLKSEL |
| Type | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x96

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Read = 0b; Must write 0b. |
| 6:5 | CLKDIV[1:0] | **DC-DC Clock Divider.**<br>Divides the dc-dc converter clock when the system clock is selected as the clock source for dc-dc converter. These bits are ignored when the dc-dc converter is clocked from its local oscillator.<br>00: The dc-dc converter clock is system clock divided by 1.<br>01: The dc-dc converter clock is system clock divided by 2.<br>10: The dc-dc converter clock is system clock divided by 4.<br>11: The dc-dc converter clock is system clock divided by 8. |
| 4 | AD0CKINV | **ADC0 Clock Inversion (Clock Invert During Sync).**<br>Inverts the ADC0 SAR clock derived from the dc-dc converter clock when the SYNC bit (DC0CN.3) is enabled. This bit is ignored when the SYNC bit is set to zero.<br>0: ADC0 SAR clock is inverted.<br>1: ADC0 SAR clock is not inverted. |
| 3 | CLKINV | **DC-DC Converter Clock Invert.**<br>Inverts the system clock used as the input to the dc-dc clock divider.<br>0: The dc-dc converter clock is not inverted.<br>1: The dc-dc converter clock is inverted. |
| 2 | ILIMIT | **Peak Current Limit Threshold.**<br>Sets the threshold for the maximum allowed peak inductor current. See Table 16.1 for peak inductor current levels.<br>0: Peak inductor current is set at a lower level.<br>1: Peak inductor current is set at a higher level. |
| 1 | VDDSLP | **VDD_MCU/DC+ Sleep Mode Connection.**<br>Specifies the power source for VDD_MCU/DC+ in Sleep Mode when the dc-dc converter is enabled.<br>0: VDD_MCU/DC+ connected to VBAT in Sleep Mode.<br>1: VDD_MCU/DC+ is floating in Sleep Mode. |
| 0 | CLKSEL | **DC-DC Converter Clock Source Select.**<br>Specifies the dc-dc converter clock source.<br>0: The dc-dc converter is clocked from its local oscillator.<br>1: The dc-dc converter is clocked from the system clock. |

## 16.10. DC-DC Converter Specifications

See Table 4.14 on page 64 for a detailed listing of dc-dc converter specifications.

# 17. Voltage Regulator (VREG0)

Si1000/1/2/3/4/5 devices include an internal voltage regulator (VREG0) to regulate the internal core supply to 1.8 V from a VDD_MCU supply of 1.8 to 3.6 V. Electrical characteristics for the on-chip regulator are specified in the Electrical Specifications chapter.

The REG0CN register allows the Precision Oscillator Bias to be disabled, saving approximately 80 µA in all non-Sleep power modes. This bias should only be disabled when the precision oscillator is not being used.

The internal regulator (VREG0) is disabled when the device enters Sleep Mode and remains enabled when the device enters Suspend Mode. See Section “14. Power Management” on page 152 for complete details about low power modes.

## SFR Definition 17.1. REG0CN: Voltage Regulator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | Reserved | Reserved | OSCBIAS | | | | Reserved |
| Type | R | R/W | R/W | R/W | R | R | R | R/W |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC9

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 0b. Write = Don’t care. |
| 6:5 | Reserved | Read = 0b. Must Write 0b. |
| 4 | OSCBIAS | **Precision Oscillator Bias.**<br>When set to 1, the bias used by the precision oscillator is forced on. If the precision oscillator is not being used, this bit may be cleared to 0 to save approximately 80 µA of supply current in all non-Sleep power modes. If disabled then re-enabled, the precision oscillator bias requires 4 µs of settling time. |
| 3:1 | Unused | Read = 000b. Write = Don’t care. |
| 0 | Reserved | Read = 0b. Must Write 0b. |

## 17.1. Voltage Regulator Electrical Specifications

See Table 4.15 on page 65 for detailed Voltage Regulator Electrical Specifications.

## 18. Reset Sources

Reset circuitry allows the controller to be easily placed in a predefined default condition. On entry to this reset state, the following occur:

- CIP-51 halts program execution
- Special Function Registers (SFRs) are initialized to their defined reset values
- External Port pins are forced to a known state
- Interrupts and timers are disabled

All SFRs are reset to the predefined values noted in the SFR descriptions. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost. Since the stack pointer SFR is reset, the stack is effectively lost, even though the data on the stack is not altered.

The Port I/O latches are reset to 0xFF (all logic ones) in open-drain mode. Weak pullups are enabled during and after the reset. For $V_{DD}$ Monitor and power-on resets, the $\overline{RST}$ pin is driven low until the device exits the reset state.

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to an internal oscillator. Refer to Section “19. Clocking Sources” on page 185 for information on selecting and configuring the system clock source. The Watchdog Timer is enabled with the system clock divided by 12 as its clock source (Section “28.4. Watchdog Timer Mode” on page 368 details the use of the Watchdog Timer). Program execution begins at location 0x0000.

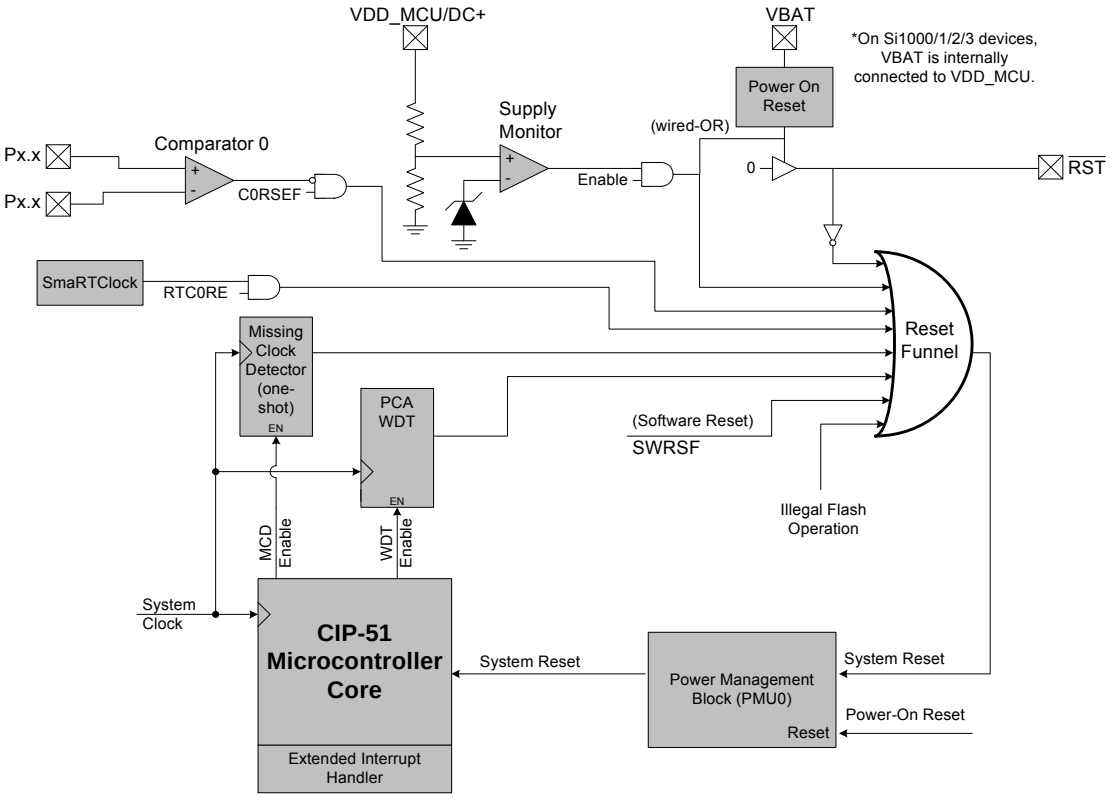



**Figure 18.1. Reset Sources**

## 18.1. Power-On (VBAT Supply Monitor) Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until $V_{BAT}$ settles above $V_{POR}$. An additional delay occurs before the device is released from reset; the delay decreases as the $V_{BAT}$ ramp time increases ($V_{BAT}$ ramp time is defined as how fast $V_{BAT}$ ramps from 0 V to $V_{POR}$). Figure 18.3 plots the power-on and $V_{DD}$ monitor reset timing. For valid ramp times (less than 3 ms), the power-on reset delay ($T_{PORDelay}$) is typically 3 ms ($V_{BAT}$ = 0.9 V), 7 ms ($V_{BAT}$ = 1.8 V), or 15 ms ($V_{BAT}$ = 3.6 V).

**Note:** The maximum $V_{DD}$ ramp time is 3 ms; slower ramp times may cause the device to be released from reset before $V_{BAT}$ reaches the $V_{POR}$ level.

On exit from a power-on reset, the PORSF flag (RSTSRC.1) is set by hardware to logic 1. When PORSF is set, all of the other reset flags in the RSTSRC Register are indeterminate (PORSF is cleared by all other resets). Since all resets cause program execution to begin at the same location (0x0000), software can read the PORSF flag to determine if a power-up was the cause of reset. The contents of internal data memory should be assumed to be undefined after a power-on reset.

**Note:** Si1000/1/2/3 have the VBAT signal internally connected to VDD_MCU.

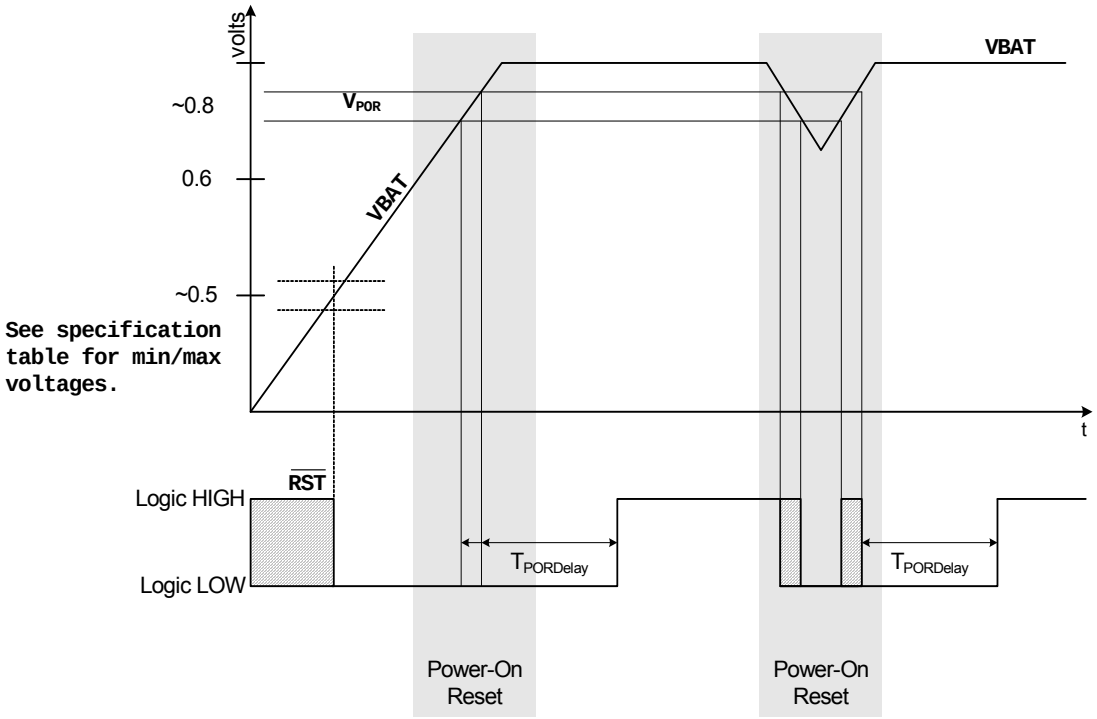



**Figure 18.2. Power-Fail Reset Timing Diagram**

## 18.2. Power-Fail (VDD_MCU Supply Monitor) Reset

Si1000/1/2/3/4/5 devices have a VDD_MCU Supply Monitor that is enabled and selected as a reset source after each power-on or power-fail reset. When enabled and selected as a reset source, any power down transition or power irregularity that causes VDD_MCU to drop below $V_{RST}$ will cause the $\overline{RST}$ pin to be driven low and the CIP-51 will be held in a reset state (see Figure 18.3). When VDD_MCU returns to a level above $V_{RST}$, the CIP-51 will be released from the reset state.

After a power-fail reset, the PORSF flag reads 1, the contents of RAM invalid, and the VDD_MCU supply monitor is enabled and selected as a reset source. The enable state of the VDD_MCU supply monitor and its selection as a reset source is only altered by power-on and power-fail resets. For example, if the VDD_MCU supply monitor is de-selected as a reset source and disabled by software, then a software reset is performed, the VDD_MCU supply monitor will remain disabled and de-selected after the reset.

In battery-operated systems, the contents of RAM can be preserved near the end of the battery's usable life if the device is placed in sleep mode prior to a power-fail reset occurring. When the device is in sleep mode, the power-fail reset is automatically disabled and the contents of RAM are preserved as long as the VBAT supply does not fall below $V_{POR}$. A large capacitor can be used to hold the power supply voltage above $V_{POR}$ while the user is replacing the battery. Upon waking from sleep mode, the enable and reset source select state of the VDD_MCU supply monitor are restored to the value last set by the user.

To allow software early notification that a power failure is about to occur, the VDDOK bit is cleared when the VDD_MCU supply falls below the $V_{WARN}$ threshold. The VDDOK bit can be configured to generate an interrupt. See Section "12. Interrupt Handler" on page 130 for more details.

**Important Note:** To protect the integrity of Flash contents, **the VDD_MCU supply monitor must be enabled and selected as a reset source if software contains routines which erase or write Flash memory.** If the VDD_MCU supply monitor is not enabled, any erase or write performed on Flash memory will cause a Flash Error device reset.

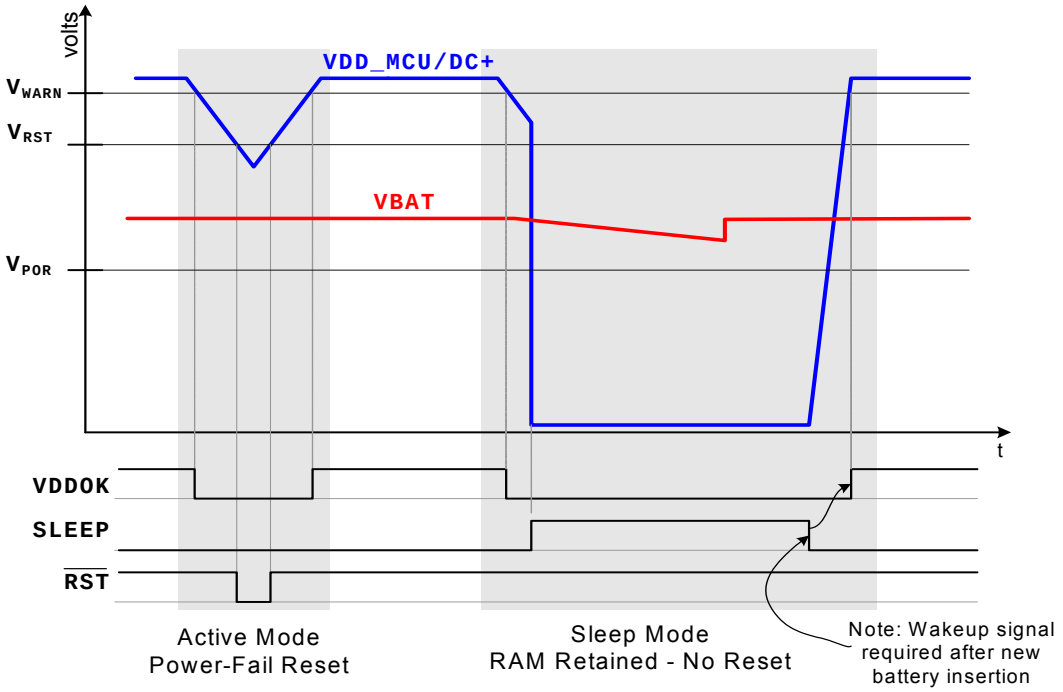



**Figure 18.3. Power-Fail Reset Timing Diagram**

**Important Notes:**

- The Power-on Reset (POR) delay is not incurred after a VDD_MCU supply monitor reset. See Section “4. Electrical Characteristics” on page 40 for complete electrical characteristics of the VDD_MCU monitor.
- Software should take care not to inadvertently disable the $V_{DD}$ Monitor as a reset source when writing to RSTSRC to enable other reset sources or to trigger a software reset. All writes to RSTSRC should explicitly set PORSF to '1' to keep the $V_{DD}$ Monitor enabled as a reset source.
- The VDD_MCU supply monitor must be enabled before selecting it as a reset source**.** Selecting the VDD_MCU supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the VDD_MCU supply monitor and selecting it as a reset source. See Section “4. Electrical Characteristics” on page 40 for minimum VDD_MCU Supply Monitor turn-on time. **No delay should be introduced in systems where software contains routines that erase or write Flash memory.** The procedure for enabling the VDD_MCU supply monitor and selecting it as a reset source is shown below:

1. Enable the VDD_MCU Supply Monitor (VDMEN bit in VDM0CN = 1).
2. Wait for the VDD_MCU Supply Monitor to stabilize (optional).
3. Select the VDD_MCU Supply Monitor as a reset source (PORSF bit in RSTSRC = 1).

### SFR Definition 18.1. VDM0CN: VDD_MCU Supply Monitor Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | VDMEN | VDDSTAT | VDDOK | Reserved | Reserved | Reserved | | |
| Type | R/W | R | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | Varies | Varies | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFF

| Bit | Name | Function |
|---|---|---|
| 7 | VDMEN | **VDD_MCU Supply Monitor Enable.**<br>This bit turns the VDD_MCU supply monitor circuit on/off. The VDD_MCU Supply Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 18.2).<br>0: VDD_MCU Supply Monitor Disabled.<br>1: VDD_MCU Supply Monitor Enabled. |
| 6 | VDDSTAT | **VDD_MCU Supply Status.**<br>This bit indicates the current power supply status.<br>0: VDD_MCU is at or below the $V_{RST}$ threshold.<br>1: VDD_MCU is above the $V_{RST}$ threshold. |
| 5 | VDDOK | **VDD_MCU Supply Status (Early Warning).**<br>This bit indicates the current power supply status.<br>0: VDD_MCU is at or below the $V_{WARN}$ threshold.<br>1: VDD_MCU is above the $V_{WARN}$ monitor threshold. |
| 4:2 | Reserved | Read = 000b. Must Write 000b. |
| 1:0 | Unused | Read = 00b. Write = Don't Care. |

## 18.3. External Reset

The external $\overline{\text{RST}}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{\text{RST}}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{\text{RST}}$ pin may be necessary to avoid erroneous noise-induced resets. See Table 4.4 for complete $\overline{\text{RST}}$ pin specifications. The external reset remains functional even when the device is in the low power Suspend and Sleep Modes. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

## 18.4. Missing Clock Detector Reset

The Missing Clock Detector (MCD) is a one-shot circuit that is triggered by the system clock. If the system clock remains high or low for more than 100 µs, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag (RSTSRC.2) will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDRSF bit enables the Missing Clock Detector; writing a 0 disables it. The missing clock detector reset is automatically disabled when the device is in the low power Suspend or Sleep mode. Upon exit from either low power state, the enabled/disabled state of this reset source is restored to its previous value. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.5. Comparator0 Reset

Comparator0 can be configured as a reset source by writing a 1 to the C0RSEF flag (RSTSRC.5). Comparator0 should be enabled and allowed to settle prior to writing to C0RSEF to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator0 reset is active-low: if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0-), the device is put into the reset state. After a Comparator0 reset, the C0RSEF flag (RSTSRC.5) will read 1 signifying Comparator0 as the reset source; otherwise, this bit reads 0. The Comparator0 reset source remains functional even when the device is in the low power Suspend and Sleep states as long as Comparator0 is also enabled as a wake-up source. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.6. PCA Watchdog Timer Reset

The programmable Watchdog Timer (WDT) function of the Programmable Counter Array (PCA) can be used to prevent software from running out of control during a system malfunction. The PCA WDT function can be enabled or disabled by software as described in Section “28.4. Watchdog Timer Mode” on page 368; the WDT is enabled and clocked by SYSCLK / 12 following any reset. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit (RSTSRC.5) is set to 1. The PCA Watchdog Timer reset source is automatically disabled when the device is in the low power Suspend or Sleep mode. Upon exit from either low power state, the enabled/disabled state of this reset source is restored to its previous value.The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.7. Flash Error Reset

If a Flash read/write/erase or program read targets an illegal address, a system reset is generated. This may occur due to any of the following:

- A Flash write or erase is attempted above user code space. This occurs when PSWE is set to 1 and a MOVX write operation targets an address above the Lock Byte address.
- A Flash read is attempted above user code space. This occurs when a MOVC operation targets an address above the Lock Byte address.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above the Lock Byte address.
- A Flash read, write or erase attempt is restricted due to a Flash security setting (see Section “13.3. Security Options” on page 144).
- A Flash write or erase is attempted while the $V_{DD}$ Monitor is disabled.

The FERROR bit (RSTSRC.6) is set following a Flash error reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.8. SmaRTClock (Real Time Clock) Reset

The SmaRTClock can generate a system reset on two events: SmaRTClock Oscillator Fail or SmaRTClock Alarm. The SmaRTClock Oscillator Fail event occurs when the SmaRTClock Missing Clock Detector is enabled and the SmaRTClock clock is below approximately 20 kHz. A SmaRTClock alarm event occurs when the SmaRTClock Alarm is enabled and the SmaRTClock timer value matches the ALARMn registers. The SmaRTClock can be configured as a reset source by writing a 1 to the RTC0RE flag (RSTSRC.7). The SmaRTClock reset remains functional even when the device is in the low power Suspend or Sleep mode. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.9. Software Reset

Software may force a reset by writing a 1 to the SWRSF bit (RSTSRC.4). The SWRSF bit will read 1 following a software forced reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## SFR Definition 18.2. RSTSRC: Reset Source

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0RE | FERROR | C0RSEF | SWRSF | WDTRSF | MCDRSF | PORSF | PINRSF |
| Type | R/W | R | R/W | R/W | R | R/W | R/W | R |
| Reset | Varies | Varies | Varies | Varies | Varies | Varies | Varies | Varies |

SFR Page = 0x0; SFR Address = 0xEF.

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7 | RTC0RE | **SmaRTClock Reset Enable and Flag** | 0: Disable SmaRTClock as a reset source.<br>1: Enable SmaRTClock as a reset source. | Set to 1 if SmaRTClock alarm or oscillator fail caused the last reset. |
| 6 | FERROR | **Flash Error Reset Flag.** | N/A | Set to 1 if Flash read/write/erase error caused the last reset. |
| 5 | C0RSEF | **Comparator0 Reset Enable and Flag.** | 0: Disable Comparator0 as a reset source.<br>1: Enable Comparator0 as a reset source. | Set to 1 if Comparator0 caused the last reset. |
| 4 | SWRSF | **Software Reset Force and Flag.** | Writing a 1 forces a system reset. | Set to 1 if last reset was caused by a write to SWRSF. |
| 3 | WDTRSF | **Watchdog Timer Reset Flag.** | N/A | Set to 1 if Watchdog Timer overflow caused the last reset. |
| 2 | MCDRSF | **Missing Clock Detector (MCD) Enable and Flag.** | 0: Disable the MCD.<br>1: Enable the MCD.<br>The MCD triggers a reset if a missing clock condition is detected. | Set to 1 if Missing Clock Detector timeout caused the last reset. |
| 1 | PORSF | **Power-On / Power-Fail Reset Flag, and Power-Fail Reset Enable.** | 0: Disable the VDD_MCU Supply Monitor as a reset source.<br>1: Enable the VDD_MCU Supply Monitor as a reset source.[3] | Set to 1 anytime a power-on or $V_{DD}$ monitor reset occurs.[2] |
| 0 | PINRSF | **HW Pin Reset Flag.** | N/A | Set to 1 if $\overline{\text{RST}}$ pin caused the last reset. |

**Notes:**
1. It is safe to use read-modify-write operations (ORL, ANL, etc.) to enable or disable specific interrupt sources.
2. If PORSF read back 1, the value read from all other bits in this register are indeterminate.
3. Writing a 1 to PORSF before the VDD_MCU Supply Monitor is stabilized may generate a system reset.

# 19. Clocking Sources

Si1000/1/2/3/4/5 devices include a programmable precision internal oscillator, an external oscillator drive circuit, a low power internal oscillator, and a SmaRTClock real time clock oscillator. The precision internal oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 19.1. The external oscillator can be configured using the OSCXCN register. The low power internal oscillator is automatically enabled and disabled when selected and deselected as a clock source. SmaRT-Clock operation is described in the SmaRTClock oscillator chapter.

The system clock (SYSCLK) can be derived from the precision internal oscillator, external oscillator, low power internal oscillator, or SmaRTClock oscillator. The global clock divider can generate a system clock that is 1, 2, 4, 8, 16, 32, 64, or 128 times slower that the selected input clock source. Oscillator electrical specifications can be found in the Electrical Specifications Chapter.

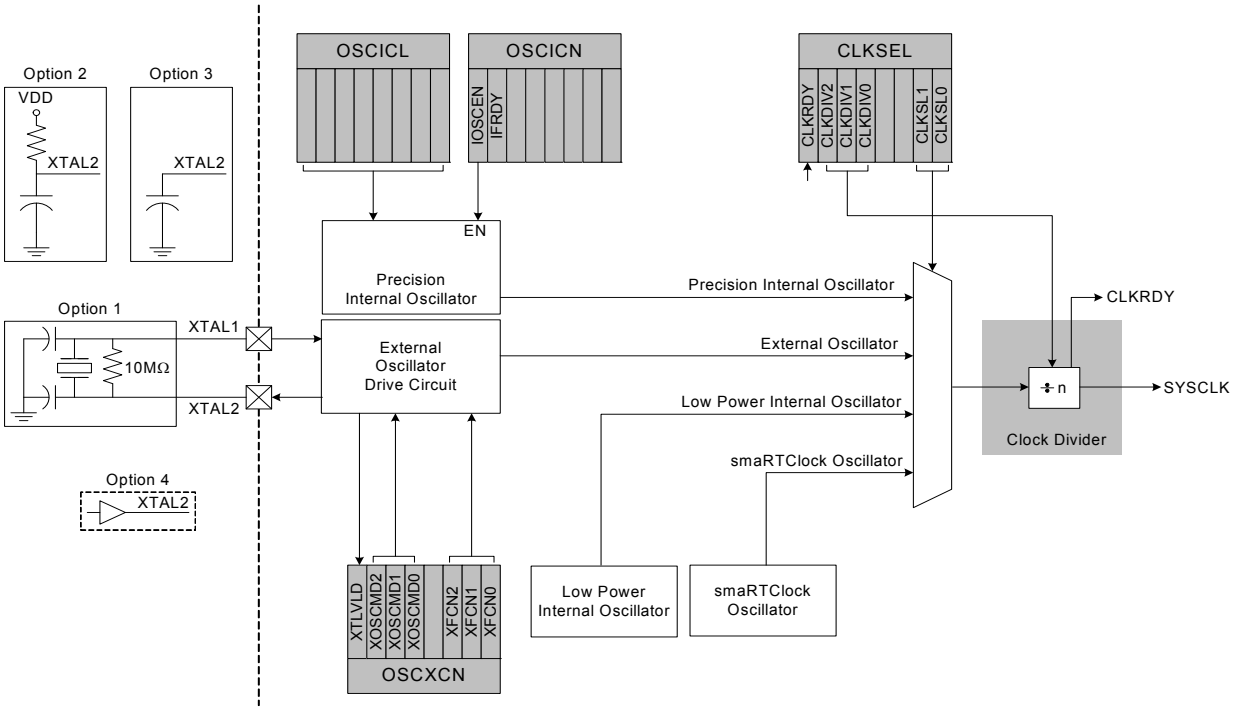



**Figure 19.1. Clocking Sources Block Diagram**

The proper way of changing the system clock when both the clock source and the clock divide value are being changed is as follows:

If switching from a fast “undivided” clock to a slower “undivided” clock:

1. Change the clock divide value.
2. Poll for CLKRDY > 1.
3. Change the clock source.

If switching from a slow “undivided” clock to a faster “undivided” clock:

1. Change the clock source.
2. Change the clock divide value.
3. Poll for CLKRDY > 1.

## 19.1. Programmable Precision Internal Oscillator

All Si1000/1/2/3/4/5 devices include a programmable precision internal oscillator that may be selected as the system clock. OSCICL is factory calibrated to obtain a 24.5 MHz frequency. See Table 4.7, “Internal Precision Oscillator Electrical Characteristics,” on page 58 for complete oscillator specifications.

The precision oscillator supports a spread spectrum mode which modulates the output frequency in order to reduce the EMI generated by the system. When enabled (SSE = 1), the oscillator output frequency is modulated by a stepped triangle wave whose frequency is equal to the oscillator frequency divided by 384 (63.8 kHz using the factory calibration). The deviation from the nominal oscillator frequency is +0%, –1.6%, and the step size is typically 0.26% of the nominal frequency. When using this mode, the typical average oscillator frequency is lowered from 24.5 MHz to 24.3 MHz.

## 19.2. Low Power Internal Oscillator

All Si1000/1/2/3/4/5 devices include a low power internal oscillator that defaults as the system clock after a system reset. The low power internal oscillator frequency is 20 MHz ± 10% and is automatically enabled when selected as the system clock and disabled when not in use. See Table 4.8, “Internal Low-Power Oscillator Electrical Characteristics,” on page 58 for complete oscillator specifications.

## 19.3. External Oscillator Drive Circuit

All Si1000/1/2/3/4/5 devices include an external oscillator circuit that may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. Figure 19.1 shows a block diagram of the four external oscillator options. The external oscillator is enabled and configured using the OSCXCN register.

The external oscillator output may be selected as the system clock or used to clock some of the digital peripherals (e.g., Timers, PCA, etc.). See the data sheet chapters for each digital peripheral for details. See Section “4. Electrical Characteristics” on page 40 for complete oscillator specifications.

### 19.3.1. External Crystal Mode

If a crystal or ceramic resonator is used as the external oscillator, the crystal/resonator and a 10 MΩ resistor must be wired across the XTAL1 and XTAL2 pins as shown in Figure 19.1, Option 1. Appropriate loading capacitors should be added to XTAL1 and XTAL2, and both pins should be configured for analog I/O with the digital output drivers disabled.

Figure 19.2 shows the external oscillator circuit for a 20 MHz quartz crystal with a manufacturer recommended load capacitance of 12.5 pF. Loading capacitors are "in series" as seen by the crystal and "in parallel" with the stray capacitance of the XTAL1 and XTAL2 pins. The total value of the each loading capacitor and the stray capacitance of each XTAL pin should equal 12.5pF x 2 = 25 pF. With a stray capacitance of 10 pF per pin, the 15 pF capacitors yield an equivalent series capacitance of 12.5 pF across the crystal.

**Note:** The recommended load capacitance depends upon the crystal and the manufacturer. Please refer to the crystal data sheet when completing these calculations.

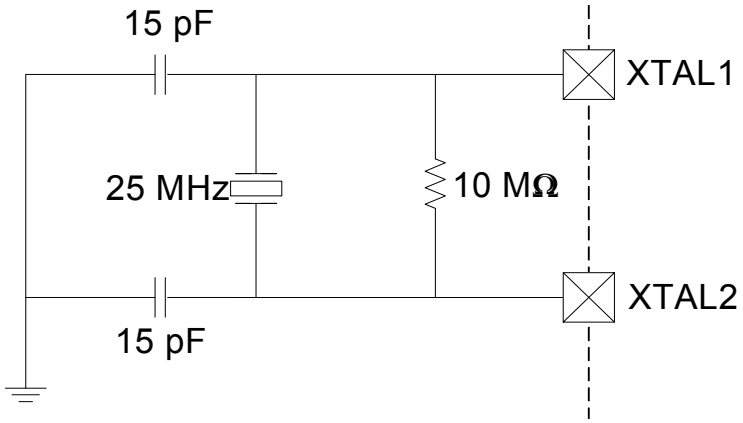



**Figure 19.2. 25 MHz External Crystal Example**

**Important Note on External Crystals:** Crystal oscillator circuits are quite sensitive to PCB layout. The crystal should be placed as close as possible to the XTAL pins on the device. The traces should be as short as possible and shielded with ground plane from any other traces which could introduce noise or interference.

When using an external crystal, the external oscillator drive circuit must be configured by software for *Crystal Oscillator Mode* or *Crystal Oscillator Mode with divide by 2 stage*. The divide by 2 stage ensures that the clock derived from the external oscillator has a duty cycle of 50%. The External Oscillator Frequency Control value (XFCN) must also be specified based on the crystal frequency. The selection should be based on Table 19.1. For example, a 25 MHz crystal requires an XFCN setting of 111b.

**Table 19.1. Recommended XFCN Settings for Crystal Mode**

| XFCN | Crystal Frequency | Bias Current | Typical Supply Current (VDD = 2.4 V) |
|---|---|---|---|
| 000 | f ≤ 20 kHz | 0.5 µA | 3.0 µA, f = 32.768 kHz |
| 001 | 20 kHz < f ≤ 58 kHz | 1.5 µA | 4.8 µA, f = 32.768 kHz |
| 010 | 58 kHz < f ≤ 155 kHz | 4.8 µA | 9.6 µA, f = 32.768 kHz |
| 011 | 155 kHz < f ≤ 415 kHz | 14 µA | 28 µA, f = 400 kHz |
| 100 | 415 kHz < f ≤ 1.1 MHz | 40 µA | 71 µA, f = 400 kHz |
| 101 | 1.1 MHz < f ≤ 3.1 MHz | 120 µA | 193 µA, f = 400 kHz |
| 110 | 3.1 MHz < f ≤ 8.2 MHz | 550 µA | 940 µA, f = 8 MHz |
| 111 | 8.2 MHz < f ≤ 25 MHz | 2.6 mA | 3.9 mA, f = 25 MHz |

When the crystal oscillator is first enabled, the external oscillator valid detector allows software to determine when the external system clock has stabilized. Switching to the external oscillator before the crystal oscillator has stabilized can result in unpredictable behavior. The recommended procedure for starting the crystal is:

1. Configure XTAL1 and XTAL2 for analog I/O and disable the digital output drivers.
2. Configure and enable the external oscillator.
3. Poll for XTLVLD => 1.
4. Switch the system clock to the external oscillator.

### 19.3.2. External RC Mode

If an RC network is used as the external oscillator, the circuit should be configured as shown in Figure 19.1, Option 2. The RC network should be added to XTAL2, and XTAL2 should be configured for analog I/O with the digital output drivers disabled. XTAL1 is not affected in RC mode.

The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. The resistor should be no smaller than 10kΩ. The oscillation frequency can be determined by the following equation:

$$f = \frac{1.23 \times 10^3}{R \times C}$$

where
f = frequency of clock in MHzR = pull-up resistor value in kΩ
$V_{DD}$ = power supply voltage in VoltsC = capacitor value on the XTAL2 pin in pF

To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, first select the RC network value to produce the desired frequency of oscillation. For example, if the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

$$f = \frac{1.23 \times 10^3}{R \times C} = \frac{1.23 \times 10^3}{246 \times 50} = 100 \text{ kHz}$$

where
f = frequency of clock in MHz; R = pull-up resistor value in kΩ
$V_{DD}$ = power supply voltage in Volts; C = capacitor value on the XTAL2 pin in pF

Referencing Table 19.2, the recommended XFCN setting is 010.

**Table 19.2. Recommended XFCN Settings for RC and C modes**

| XFCN | Approximate Frequency Range (RC and C Mode) | K Factor (C Mode) | Typical Supply Current/ Actual Measured Frequency (C Mode, VDD = 2.4 V) |
|---|---|---|---|
| 000 | f ≤ 25 kHz | K Factor = 0.87 | 3.0 µA, f = 11 kHz, C = 33 pF |
| 001 | 25 kHz < f ≤ 50 kHz | K Factor = 2.6 | 5.5 µA, f = 33 kHz, C = 33 pF |
| 010 | 50 kHz < f ≤ 100 kHz | K Factor = 7.7 | 13 µA, f = 98 kHz, C = 33 pF |
| 011 | 100 kHz < f ≤ 200 kHz | K Factor = 22 | 32 µA, f = 270 kHz, C = 33 pF |
| 100 | 200 kHz < f ≤ 400 kHz | K Factor = 65 | 82 µA, f = 310 kHz, C = 46 pF |
| 101 | 400 kHz < f ≤ 800 kHz | K Factor = 180 | 242 µA, f = 890 kHz, C = 46 pF |
| 110 | 800 kHz < f ≤ 1.6 MHz | K Factor = 664 | 1.0 mA, f = 2.0 MHz, C = 46 pF |
| 111 | 1.6 MHz < f ≤ 3.2 MHz | K Factor = 1590 | 4.6 mA, f = 6.8 MHz, C = 46 pF |

When the RC oscillator is first enabled, the external oscillator valid detector allows software to determine when oscillation has stabilized. The recommended procedure for starting the RC oscillator is:

1. Configure XTAL2 for analog I/O and disable the digital output drivers.
2. Configure and enable the external oscillator.

3. Poll for XTLVLD ≥ 1.
4. Switch the system clock to the external oscillator.

### 19.3.3. External Capacitor Mode

If a capacitor is used as the external oscillator, the circuit should be configured as shown in Figure 19.1, Option 3. The capacitor should be added to XTAL2, and XTAL2 should be configured for analog I/O with the digital output drivers disabled. XTAL1 is not affected in RC mode.

The capacitor should be no greater than 100 pF; however, for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. The oscillation frequency and the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register can be determined by the following equation:

$$f = \frac{KF}{C \times V_{DD}}$$

where
f = frequency of clock in MHzR = pull-up resistor value in kΩ
$V_{DD}$ = power supply voltage in VoltsC = capacitor value on the XTAL2 pin in pF

Below is an example of selecting the capacitor and finding the frequency of oscillation Assume $V_{DD}$ = 3.0 V and f = 150 kHz:

$$f = \frac{KF}{C \times V_{DD}}$$

$$0.150\ \text{MHz} = \frac{KF}{C \times 3.0}$$

Since a frequency of roughly 150 kHz is desired, select the K Factor from Table 19.2 as KF = 22:

$$0.150\ \text{MHz} = \frac{22}{C \times 3.0\ \text{V}}$$

$$C = \frac{22}{0.150\ \text{MHz} \times 3.0\ \text{V}}$$

$$C = 48.8\ \text{pF}$$

Therefore, the XFCN value to use in this example is 011 and C is approximately 50 pF.

The recommended startup procedure for C mode is the same as RC mode.

### 19.3.4. External CMOS Clock Mode

If an external CMOS clock is used as the external oscillator, the clock should be directly routed into XTAL2. The XTAL2 pin should be configured as a digital input. XTAL1 is not used in external CMOS clock mode.

The external oscillator valid detector will always return zero when the external oscillator is configured to External CMOS Clock mode.

## 19.4. Special Function Registers for Selecting and Configuring the System Clock

The clocking sources on Si1000/1/2/3/4/5 devices are enabled and configured using the OSCICN, OSCICL, OSCXCN and the SmaRTClock internal registers. See Section “20. SmaRTClock (Real Time Clock)” on page 193 for SmaRTClock register descriptions. The system clock source for the MCU can be selected using the CLKSEL register. To minimize active mode current, the oneshot timer which sets Flash read time should by bypassed when the system clock is greater than 10 MHz. See the FLSCL register description for details.

The clock selected as the system clock can be divided by 1, 2, 4, 8, 16, 32, 64, or 128. When switching between two clock divide values, the transition may take up to 128 cycles of the undivided clock source. The CLKRDY flag can be polled to determine when the new clock divide value has been applied. The clock divider must be set to "divide by 1" when entering Suspend or Sleep Mode.

The system clock source may also be switched on-the-fly. The switchover takes effect after one clock period of the slower oscillator.

### SFR Definition 19.1. CLKSEL: Clock Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CLKRDY | CLKDIV[2:0] | | | | CLKSEL[2:0] | | |
| Type | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |

SFR Page = All Pages; SFR Address = 0xA9

| Bit | Name | Function |
|---|---|---|
| 7 | CLKRDY | **System Clock Divider Clock Ready Flag.**<br>0: The selected clock divide setting has not been applied to the system clock.<br>1: The selected clock divide setting has been applied to the system clock. |
| 6:4 | CLKDIV[2:0] | **System Clock Divider Bits.**<br>Selects the clock division to be applied to the undivided system clock source.<br>000: System clock is divided by 1.<br>001: System clock is divided by 2.<br>010: System clock is divided by 4.<br>011: System clock is divided by 8.<br>100: System clock is divided by 16.<br>101: System clock is divided by 32.<br>110: System clock is divided by 64.<br>111: System clock is divided by 128. |
| 3 | Unused | Read = 0b. Must Write 0b. |
| 2:0 | CLKSEL[2:0] | **System Clock Select.**<br>Selects the oscillator to be used as the undivided system clock source.<br>000: Precision Internal Oscillator.<br>001: External Oscillator.<br>010: Reserved.<br>011: SmaRTClock Oscillator.<br>1xx: Low Power Oscillator. |

## SFR Definition 19.2. OSCICN: Internal Oscillator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | IOSCEN | IFRDY | Reserved[5:0] | | | | | |
| Type | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xB2

| Bit | Name | Function |
|---|---|---|
| 7 | IOSCEN | **Internal Oscillator Enable.**<br>0: Internal oscillator disabled.<br>1: Internal oscillator enabled. |
| 6 | IFRDY | **Internal Oscillator Frequency Ready Flag**.<br>0: Internal oscillator is not running at its programmed frequency.<br>1: Internal oscillator is running at its programmed frequency. |
| 5:0 | Reserved | Read = 001111b. Must Write 001111b. |

**Note:** It is recommended to use read-modify-write operations such as ORL and ANL to set or clear the enable bit of this register.

## SFR Definition 19.3. OSCICL: Internal Oscillator Calibration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SSE | OSCICL[6:0] | | | | | | |
| Type | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | Varies | Varies | Varies | Varies | Varies | Varies | Varies |

SFR Page = 0x0; SFR Address = 0xB3

| Bit | Name | Function |
|---|---|---|
| 7 | SSE | **Spread Spectrum Enable**.<br>0: Spread Spectrum clock dithering disabled.<br>1: Spread Spectrum clock dithering enabled. |
| 6:0 | OSCICL | **Internal Oscillator Calibration**.<br>Factory calibrated to obtain a frequency of 24.5 MHz. Incrementing this register decreases the oscillator frequency and decrementing this register increases the oscillator frequency. The step size is approximately 1% of the calibrated frequency. The recommended calibration frequency range is between 16 and 24.5 MHz. |

**Note:** If the Precision Internal Oscillator is selected as the system clock, the following procedure should be used when changing the value of the internal oscillator calibration bits.
1. Switch to a different clock source.
2. Disable the oscillator by writing OSCICN.7 to 0.
3. Change OSCICL to the desired setting.
4. Enable the oscillator by writing OSCICN.7 to 1.

## SFR Definition 19.4. OSCXCN: External Oscillator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | XCLKVLD | XOSCMD[2:0] | | | Reserved | XFCN[2:0] | | |
| Type | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB1

| Bit | Name | Function |
|---|---|---|
| 7 | XCLKVLD | **External Oscillator Valid Flag.**<br>Provides External Oscillator status and is valid at all times for all modes of operation except External CMOS Clock Mode and External CMOS Clock Mode with divide by 2. In these modes, XCLKVLD always returns 0.<br>0: External Oscillator is unused or not yet stable.<br>1: External Oscillator is running and stable. |
| 6:4 | XOSCMD | **External Oscillator Mode Bits.**<br>Configures the external oscillator circuit to the selected mode.<br>00x: External Oscillator circuit disabled.<br>010: External CMOS Clock Mode.<br>011: External CMOS Clock Mode with divide by 2 stage.<br>100: RC Oscillator Mode.<br>101: Capacitor Oscillator Mode.<br>110: Crystal Oscillator Mode.<br>111: Crystal Oscillator Mode with divide by 2 stage. |
| 3 | Reserved | Read = 0b. Must Write 0b. |
| 2:0 | XFCN | **External Oscillator Frequency Control Bits.**<br>Controls the external oscillator bias current.<br>000-111: See Table 19.1 on page 187 (Crystal Mode) or Table 19.2 on page 188 (RC or C Mode) for recommended settings. |

# 20. SmaRTClock (Real Time Clock)

Si1000/1/2/3/4/5 devices include an ultra low power 32-bit SmaRTClock Peripheral (Real Time Clock) with alarm. The SmaRTClock has a dedicated 32 kHz oscillator that can be configured for use with or without a crystal. No external resistor or loading capacitors are required. The on-chip loading capacitors are programmable to 16 discrete levels allowing compatibility with a wide range of crystals. The SmaRTClock can operate directly from a 0.9–3.6 V battery voltage and remains operational even when the device goes into its lowest power down mode.

The SmaRTClock allows a maximum of 36 hour 32-bit independent time-keeping when used with a 32.768 kHz Watch Crystal. The SmaRTClock provides an Alarm and Missing SmaRTClock events, which could be used as reset or wakeup sources. See Section “18. Reset Sources” on page 178 and Section “14. Power Management” on page 152 for details on reset sources and low power mode wake-up sources, respectively.

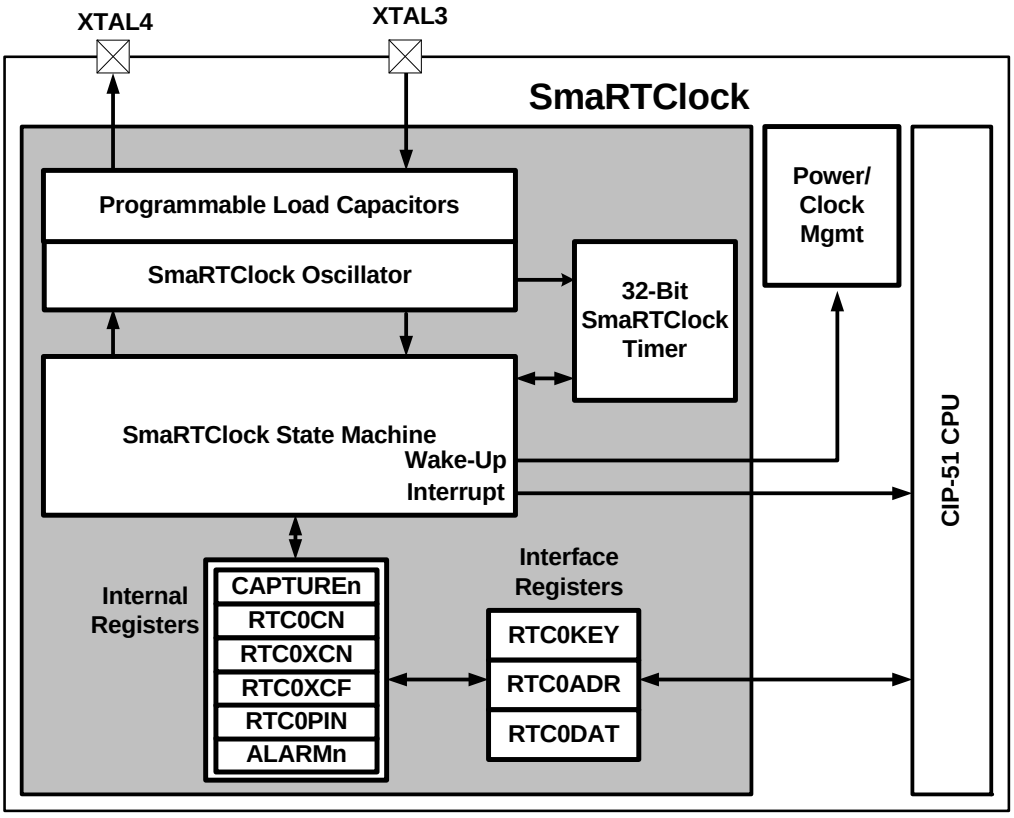



**Figure 20.1. SmaRTClock Block Diagram**

## 20.1. SmaRTClock Interface

The SmaRTClock Interface consists of three registers: RTC0KEY, RTC0ADR, and RTC0DAT. These interface registers are located on the CIP-51’s SFR map and provide access to the SmaRTClock internal registers listed in Table 20.1. The SmaRTClock internal registers can only be accessed indirectly through the SmaRTClock Interface.

**Table 20.1. SmaRTClock Internal Registers**

| SmaRTClock Address | SmaRTClock Register | Register Name | Description |
|---|---|---|---|
| 0x00–0x03 | CAPTUREn | SmaRTClock Capture Registers | Four Registers used for setting the 32-bit SmaRTClock timer or reading its current value. |
| 0x04 | RTC0CN | SmaRTClock Control Register | Controls the operation of the SmaRTClock State Machine. |
| 0x05 | RTC0XCN | SmaRTClock Oscillator Control Register | Controls the operation of the SmaRTClock Oscillator. |
| 0x06 | RTC0XCF | SmaRTClock Oscillator Configuration Register | Controls the value of the progammable oscillator load capacitance and enables/disables AutoStep. |
| 0x07 | RTC0PIN | SmaRTClock Pin Configuration Register | **Note:** Forces XTAL3 and XTAL4 to be internally shorted.<br>This register also contains other reserved bits which should not be modified. |
| 0x08–0x0B | ALARMn | SmaRTClock Alarm Registers | Four registers used for setting or reading the 32-bit SmaRTClock alarm value. |

### 20.1.1. SmaRTClock Lock and Key Functions

The SmaRTClock Interface is protected with a lock and key function. The SmaRTClock Lock and Key Register (RTC0KEY) must be written with the correct key codes, in sequence, before writes and reads to RTC0ADR and RTC0DAT may be performed. The key codes are: 0xA5, 0xF1. There are no timing restrictions, but the key codes must be written in order. If the key codes are written out of order, the wrong codes are written, or an indirect register read or write is attempted while the interface is locked, the SmaRTClock interface will be disabled, and the RTC0ADR and RTC0DAT registers will become inaccessible until the next system reset. Once the SmaRTClock interface is unlocked, software may perform any number of accesses to the SmaRTClock registers until the interface is re-locked or the device is reset. Any write to RTC0KEY while the SmaRTClock interface is unlocked will re-lock the interface.

Reading the RTC0KEY register at any time will provide the SmaRTClock Interface status and will not interfere with the sequence that is being written. The RTC0KEY register description in SFR Definition 20.1 lists the definition of each status code.

### 20.1.2. Using RTC0ADR and RTC0DAT to Access SmaRTClock Internal Registers

The SmaRTClock internal registers can be read and written using RTC0ADR and RTC0DAT. The RTC0ADR register selects the SmaRTClock internal register that will be targeted by subsequent reads or writes. Recommended instruction timing is provided in this section. If the recommended instruction timing is not followed, then BUSY (RTC0ADR.7) should be checked prior to each read or write operation to make sure the SmaRTClock Interface is not busy performing the previous read or write operation. A SmaRTClock Write operation is initiated by writing to the RTC0DAT register. Below is an example of writing to a SmaRTClock internal register.

1. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommended instruction timing.
2. Write 0x05 to RTC0ADR. This selects the internal RTC0CN register at SmaRTClock Address 0x05.
3. Write 0x00 to RTC0DAT. This operation writes 0x00 to the internal RTC0CN register.

A SmaRTClock Read operation is initiated by setting the SmaRTClock Interface Busy bit. This transfers the contents of the internal register selected by RTC0ADR to RTC0DAT. The transferred data will remain in RTC0DAT until the next read or write operation. Below is an example of reading a SmaRTClock internal register.

1. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommended instruction timing.
2. Write 0x05 to RTC0ADR. This selects the internal RTC0CN register at SmaRTClock Address 0x05.
3. Write 1 to BUSY. This initiates the transfer of data from RTC0CN to RTC0DAT.
4. Poll BUSY (RTC0ADR.7) until it returns 0 or follow recommend instruction timing.
5. Read data from RTC0DAT. This data is a copy of the RTC0CN register.

**Note:** The RTC0ADR and RTC0DAT registers will retain their state upon a device reset.

### 20.1.3. RTC0ADR Short Strobe Feature

Reads and writes to indirect SmaRTClock registers normally take 7 system clock cycles. To minimize the indirect register access time, the Short Strobe feature decreases the read and write access time to 6 system clocks. The Short Strobe feature is automatically enabled on reset and can be manually enabled/disabled using the SHORT (RTC0ADR.4) control bit.

Recommended Instruction Timing for a single register read with short strobe enabled:

```
mov RTC0ADR, #095h
nop
nop
nop
mov A, RTC0DAT
```

Recommended Instruction Timing for a single register write with short strobe enabled:

```
mov RTC0ADR, #095h
mov RTC0DAT, #000h
nop
```

### 20.1.4. SmaRTClock Interface Autoread Feature

When Autoread is enabled, each read from RTC0DAT initiates the next indirect read operation on the SmaRTClock internal register selected by RTC0ADR. Software should set the BUSY bit once at the beginning of each series of consecutive reads. Software should follow recommended instruction timing or check if the SmaRTClock Interface is busy prior to reading RTC0DAT. Autoread is enabled by setting AUTORD (RTC0ADR.6) to logic 1.

### 20.1.5. RTC0ADR Autoincrement Feature

For ease of reading and writing the 32-bit CAPTURE and ALARM values, RTC0ADR automatically increments after each read or write to a CAPTUREn or ALARMn register. This speeds up the process of setting an alarm or reading the current SmaRTClock timer value. Autoincrement is always enabled.

Recommended Instruction Timing for a multi-byte register read with short strobe and autoread enabled:

```
mov RTC0ADR, #0d0h
nop
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
nop
nop
mov A, RTC0DAT
```

Recommended Instruction Timing for a multi-byte register write with short strobe enabled:

```
mov RTC0ADR, #010h
mov RTC0DAT, #05h
nop
mov RTC0DAT, #06h
nop
mov RTC0DAT, #07h
nop
mov RTC0DAT, #08h
nop
```

## SFR Definition 20.1. RTC0KEY: SmaRTClock Lock and Key

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0ST[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xAE

| Bit | Name | Function |
|---|---|---|
| 7:0 | RTC0ST | **SmaRTClock Interface Lock/Key and Status.**<br>Locks/unlocks the SmaRTClock interface when written. Provides lock status when read.<br><br>**Read:**<br>**0x00:** SmaRTClock Interface is locked.<br>**0x01:** SmaRTClock Interface is locked.<br>First key code (0xA5) has been written, waiting for second key code.<br>**0x02:** SmaRTClock Interface is unlocked.<br>First and second key codes (0xA5, 0xF1) have been written.<br>**0x03:** SmaRTClock Interface is disabled until the next system reset.<br><br>**Write:**<br>When RTC0ST = 0x00 (locked), writing 0xA5 followed by 0xF1 unlocks the SmaRTClock Interface.<br><br>When RTC0ST = 0x01 (waiting for second key code), writing any value other than the second key code (0xF1) will change RTC0STATE to 0x03 and disable the SmaRTClock Interface until the next system reset.<br><br>When RTC0ST = 0x02 (unlocked), any write to RTC0KEY will lock the SmaRT-Clock Interface.<br><br>When RTC0ST = 0x03 (disabled), writes to RTC0KEY have no effect. |

## SFR Definition 20.2. RTC0ADR: SmaRTClock Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | BUSY | AUTORD | | SHORT | ADDR[3:0] | | | |
| Type | R/W | R/W | R | R/W | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xAC

| Bit | Name | Function |
|---|---|---|
| 7 | BUSY | **SmaRTClock Interface Busy Indicator.**<br>Indicates SmaRTClock interface status. Writing 1 to this bit initiates an indirect read. |
| 6 | AUTORD | **SmaRTClock Interface Autoread Enable.**<br>Enables/disables Autoread.<br>0: Autoread Disabled.<br>1: Autoread Enabled. |
| 5 | Unused | Read = 0b; Write = Don't Care. |
| 4 | SHORT | **Short Strobe Enable.**<br>Enables/disables the Short Strobe Feature.<br>0: Short Strobe disabled.<br>1: Short Strobe enabled. |
| 3:0 | ADDR[3:0] | **SmaRTClock Indirect Register Address.**<br>Sets the currently selected SmaRTClock register.<br>See Table 20.1 for a listing of all SmaRTClock indirect registers. |
| **Note:** The ADDR bits increment after each indirect read/write operation that targets a CAPTUREn or ALARMn internal SmaRTClock register. | | |

## SFR Definition 20.3. RTC0DAT: SmaRTClock Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xAD

| Bit | Name | Function |
|---|---|---|
| 7:0 | RTC0DAT | **SmaRTClock Data Bits.**<br>Holds data transferred to/from the internal SmaRTClock register selected by RTC0ADR. |
| **Note:** Read-modify-write instructions (orl, anl, etc.) should not be used on this register. | | |

## 20.2. SmaRTClock Clocking Sources

The SmaRTClock peripheral is clocked from its own timebase, independent of the system clock. The SmaRTClock timebase is derived from the SmaRTClock oscillator circuit, which has two modes of operation: Crystal Mode, and Self-Oscillate Mode. The oscillation frequency is 32.768 kHz in Crystal Mode and can be programmed in the range of 10 kHz to 40 kHz in Self-Oscillate Mode. The frequency of the SmaRTClock oscillator can be measured with respect to another oscillator using an on-chip timer. See Section "27. Timers" on page 335 for more information on how this can be accomplished.

**Note:** The SmaRTClock timebase can be selected as the system clock and routed to a port pin. See Section "19. Clocking Sources" on page 185 for information on selecting the system clock source and Section "21. Port Input/Output" on page 210 for information on how to route the system clock to a port pin.

### 20.2.1. Using the SmaRTClock Oscillator with a Crystal or External CMOS Clock

When using Crystal Mode, a 32.768 kHz crystal should be connected between XTAL3 and XTAL4. No other external components are required. The following steps show how to start the SmaRTClock crystal oscillator in software:

1. Set SmaRTClock to Crystal Mode (XMODE = 1).
2. Disable Automatic Gain Control (AGCEN) and enable Bias Doubling (BIASX2) for fast crystal startup.
3. Set the desired loading capacitance (RTC0XCF).
4. Enable power to the SmaRTClock oscillator circuit (RTC0EN = 1).
5. Wait 20 ms.
6. Poll the SmaRTClock Clock Valid Bit (CLKVLD) until the crystal oscillator stabilizes.
7. Poll the SmaRTClock Load Capacitance Ready Bit (LOADRDY) until the load capacitance reaches its programmed value.
8. Enable Automatic Gain Control (AGCEN) and disable Bias Doubling (BIASX2) for maximum power savings.
9. Enable the SmaRTClock missing clock detector.
10. Wait 2 ms.
11. Clear the PMU0CF wake-up source flags.

In Crystal Mode, the SmaRTClock oscillator may be driven by an external CMOS clock. The CMOS clock should be applied to XTAL3. XTAL4 should be left floating. The input low voltage (VIL) and input high voltage (VIH) for XTAL3 when used with an external CMOS clock are 0.1 and 0.8 V, respectively. The SmaRTClock oscillator should be configured to its lowest bias setting with AGC disabled. The CLKVLD bit is indeterminate when using a CMOS clock, however, the OSCFAIL bit may be checked 2 ms after SmaRTClock oscillator is powered on to ensure that there is a valid clock on XTAL3.

### 20.2.2. Using the SmaRTClock Oscillator in Self-Oscillate Mode

When using Self-Oscillate Mode, the XTAL3 and XTAL4 pins should be shorted together. The RTC0PIN register can be used to internally short XTAL3 and XTAL4. The following steps show how to configure SmaRTClock for use in Self-Oscillate Mode:

1. Set SmaRTClock to Self-Oscillate Mode (XMODE = 0).
2. Set the desired oscillation frequency:
   For oscillation at about 20 kHz, set BIASX2 = 0.
   For oscillation at about 40 kHz, set BIASX2 = 1.
3. The oscillator starts oscillating instantaneously.
4. Fine tune the oscillation frequency by adjusting the load capacitance (RTC0XCF).

### 20.2.3. Programmable Load Capacitance

The programmable load capacitance has 16 values to support crystal oscillators with a wide range of recommended load capacitance. If Automatic Load Capacitance Stepping is enabled, the crystal load capacitors start at the smallest setting to allow a fast startup time, then slowly increase the capacitance until the final programmed value is reached. The final programmed loading capacitor value is specified using the LOADCAP bits in the RTC0XCF register. The LOADCAP setting specifies the amount of on-chip load capacitance and does not include any stray PCB capacitance. Once the final programmed loading capacitor value is reached, the LOADRDY flag will be set by hardware to logic 1.

When using the SmaRTClock oscillator in Self-Oscillate mode, the programmable load capacitance can be used to fine tune the oscillation frequency. In most cases, increasing the load capacitor value will result in a decrease in oscillation frequency.Table 20.2 shows the crystal load capacitance for various settings of LOADCAP.

.

**Table 20.2. SmaRTClock Load Capacitance Settings**

| LOADCAP | Crystal Load Capacitance | Equivalent Capacitance seen on XTAL3 and XTAL4 |
|---|---|---|
| 0000 | 4.0 pF | 8.0 pF |
| 0001 | 4.5 pF | 9.0 pF |
| 0010 | 5.0 pF | 10.0 pF |
| 0011 | 5.5 pF | 11.0 pF |
| 0100 | 6.0 pF | 12.0 pF |
| 0101 | 6.5 pF | 13.0 pF |
| 0110 | 7.0 pF | 14.0 pF |
| 0111 | 7.5 pF | 15.0 pF |
| 1000 | 8.0 pF | 16.0 pF |
| 1001 | 8.5 pF | 17.0 pF |
| 1010 | 9.0 pF | 18.0 pF |
| 1011 | 9.5 pF | 19.0 pF |
| 1100 | 10.5 pF | 21.0 pF |
| 1101 | 11.5 pF | 23.0 pF |
| 1110 | 12.5 pF | 25.0 pF |
| 1111 | 13.5 pF | 27.0 pF |

### 20.2.4. Automatic Gain Control (Crystal Mode Only) and SmaRTClock Bias Doubling

Automatic Gain Control allows the SmaRTClock oscillator to trim the oscillation amplitude of a crystal in order to achieve the lowest possible power consumption. Automatic Gain Control automatically detects when the oscillation amplitude has reached a point where it safe to reduce the drive current, therefore, it may be enabled during crystal startup. It is recommended to enable Automatic Gain Control in most systems which use the SmaRTClock oscillator in Crystal Mode. The following are recommended crystal specifications and operating conditions when Automatic Gain Control is enabled:

- ESR < 50 kΩ
- Load Capacitance < 10 pF
- Supply Voltage < 3.0 V
- Temperature > –20 °C

When using Automatic Gain Control, it is recommended to perform an oscillation robustness test to ensure that the chosen crystal will oscillate under the worst case condition to which the system will be exposed. The worst case condition that should result in the least robust oscillation is at the following system conditions: lowest temperature, highest supply voltage, highest ESR, highest load capacitance, and lowest bias current (AGC enabled, Bias Double Disabled).

To perform the oscillation robustness test, the SmaRTClock oscillator should be enabled and selected as the system clock source. Next, the SYSCLK signal should be routed to a port pin configured as a push-pull digital output. The positive duty cycle of the output clock can be used as an indicator of oscillation robustness. As shown in Figure 20.2, duty cycles less than 55% indicate a robust oscillation. As the duty cycle approaches 60%, oscillation becomes less reliable and the risk of clock failure increases. Increasing the bias current (by disabling AGC) will always improve oscillation robustness and will reduce the output clock's duty cycle. This test should be performed at the worst case system conditions, as results at very low temperatures or high supply voltage will vary from results taken at room temperature or low supply voltage.

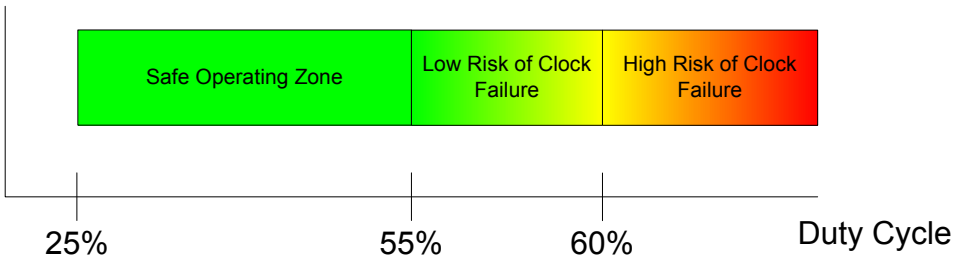



**Figure 20.2. Interpreting Oscillation Robustness (Duty Cycle) Test Results**

As an alternative to performing the oscillation robustness test, Automatic Gain Control may be disabled at the cost of increased power consumption (approximately 200 nA). Disabling Automatic Gain Control will provide the crystal oscillator with higher immunity against external factors which may lead to clock failure. Automatic Gain Control must be disabled if using the SmaRTClock oscillator in self-oscillate mode.

Table 20.3 shows a summary of the oscillator bias settings. The SmaRTClock Bias Doubling feature allows the self-oscillation frequency to be increased (almost doubled) and allows a higher crystal drive strength in crystal mode. High crystal drive strength is recommended when the crystal is exposed to poor environmental conditions such as excessive moisture. SmaRTClock Bias Doubling is enabled by setting BIASX2 (RTC0XCN.5) to 1.

**Table 20.3. SmaRTClock Bias Settings**

| Mode | Setting | Power Consumption |
|---|---|---|
| Crystal | Bias Double Off, AGC On | Lowest<br>600 nA |
| | Bias Double Off, AGC Off | Low<br>800 nA |
| | Bias Double On, AGC On | High |
| | Bias Double On, AGC Off | Highest |
| Self-Oscillate | Bias Double Off | Low |
| | Bias Double On | High |

### 20.2.5. Missing SmaRTClock Detector

The missing SmaRTClock detector is a one-shot circuit enabled by setting MCLKEN (RTC0CN.6) to 1. When the SmaRTClock Missing Clock Detector is enabled, OSCFAIL (RTC0CN.5) is set by hardware if SmaRTClock oscillator remains high or low for more than 100 µs.

A SmaRTClock Missing Clock detector timeout can trigger an interrupt, wake the device from a low power mode, or reset the device. See Section “12. Interrupt Handler” on page 130, Section “14. Power Management” on page 152, and Section “18. Reset Sources” on page 178 for more information.

**Note:** The SmaRTClock Missing Clock Detector should be disabled when making changes to the oscillator settings in RTC0XCN.

### 20.2.6. SmaRTClock Oscillator Crystal Valid Detector

The SmaRTClock oscillator crystal valid detector is an oscillation amplitude detector circuit used during crystal startup to determine when oscillation has started and is nearly stable. The output of this detector can be read from the CLKVLD bit (RTX0XCN.4).

**Notes:**

- The CLKVLD bit has a blanking interval of 2 ms. During the first 2 ms after turning on the crystal oscillator, the output of CLKVLD is not valid.
- This SmaRTClock crystal valid detector (CLKVLD) is not intended for detecting an oscillator failure. The missing SmaRTClock detector (CLKFAIL) should be used for this purpose.

## 20.3. SmaRTClock Timer and Alarm Function

The SmaRTClock timer is a 32-bit counter that, when running (RTC0TR = 1), is incremented every SmaRTClock oscillator cycle. The timer has an alarm function that can be set to generate an interrupt, wake the device from a low power mode, or reset the device at a specific time. See Section “12. Interrupt Handler” on page 130, Section “14. Power Management” on page 152, and Section “18. Reset Sources” on page 178 for more information.

The SmaRTClock timer includes an Auto Reset feature, which automatically resets the timer to zero one SmaRTClock cycle after the alarm signal is deasserted. When using Auto Reset, the Alarm match value should always be set to 2 counts less than the desired match value. Auto Reset can be enabled by writing a 1 to ALRM (RTC0CN.2).

### 20.3.1. Setting and Reading the SmaRTClock Timer Value

The 32-bit SmaRTClock timer can be set or read using the six CAPTUREn internal registers. Note that the timer does not need to be stopped before reading or setting its value. The following steps can be used to set the timer value:

1. Write the desired 32-bit set value to the CAPTUREn registers.
2. Write 1 to RTC0SET. This will transfer the contents of the CAPTUREn registers to the SmaRTClock timer.
3. Operation is complete when RTC0SET is cleared to 0 by hardware.

The following steps can be used to read the current timer value:

1. Write 1 to RTC0CAP. This will transfer the contents of the timer to the CAPTUREn registers.
2. Poll RTC0CAP until it is cleared to 0 by hardware.
3. A snapshot of the timer value can be read from the CAPTUREn registers

### 20.3.2. Setting a SmaRTClock Alarm

The SmaRTClock alarm function compares the 32-bit value of SmaRTClock Timer to the value of the ALARMn registers. An alarm event is triggered if the SmaRTClock timer is **equal to** the ALARMn registers. If Auto Reset is enabled, the 32-bit timer will be cleared to zero one SmaRTClock cycle after the alarm event.

The SmaRTClock alarm event can be configured to reset the MCU, wake it up from a low power mode, or generate an interrupt. See Section “12. Interrupt Handler” on page 130, Section “14. Power Management” on page 152, and Section “18. Reset Sources” on page 178 for more information.

The following steps can be used to set up a SmaRTClock Alarm:

1. Disable SmaRTClock Alarm Events (RTC0AEN = 0).
2. Set the ALARMn registers to the desired value.
3. Enable SmaRTClock Alarm Events (RTC0AEN = 1).

**Notes:**

- The ALRM bit, which is used as the SmaRTClock Alarm Event flag, is cleared by disabling SmaRTClock Alarm Events (RTC0AEN = 0).
- If AutoReset is disabled, disabling (RTC0AEN = 0) then Re-enabling Alarm Events (RTC0AEN = 1) after a SmaRTClock Alarm without modifying ALARMn registers will automatically schedule the next alarm after 2^32 SmaRTClock cycles (approximately 36 hours using a 32.768 kHz crystal).
- The SmaRTClock Alarm Event flag will remain asserted for a maximum of one SmaRTClock cycle. See Section “14. Power Management” on page 152 for information on how to capture a SmaRTClock Alarm event using a flag which is not automatically cleared by hardware.

### 20.3.3. Software Considerations for using the SmaRTClock Timer and Alarm

The SmaRTClock timer and alarm have two operating modes to suit varying applications. The two modes are described below:

**Mode 1**:
The first mode uses the SmaRTClock timer as a perpetual timebase which is never reset to zero. Every 36 hours, the timer is allowed to overflow without being stopped or disrupted. The alarm interval is software managed and is added to the ALRMn registers by software after each alarm. This allows the alarm match value to always stay ahead of the timer by one software managed interval. If software uses 32-bit unsigned addition to increment the alarm match value, then it does not need to handle overflows since both the timer and the alarm match value will overflow in the same manner.

This mode is ideal for applications which have a long alarm interval (e.g. 24 or 36 hours) and/or have a need for a perpetual timebase. An example of an application that needs a perpetual timebase is one whose wake-up interval is constantly changing. For these applications, software can keep track of the number of timer overflows in a 16-bit variable, extending the 32-bit (36 hour) timer to a 48-bit (272 year) perpetual timebase.

**Mode 2**:
The second mode uses the SmaRTClock timer as a general purpose up counter which is auto reset to zero by hardware after each alarm. The alarm interval is managed by hardware and stored in the ALRMn registers. Software only needs to set the alarm interval once during device initialization. After each alarm, software should keep a count of the number of alarms that have occurred in order to keep track of time.

This mode is ideal for applications that require minimal software intervention and/or have a fixed alarm interval. This mode is the most power efficient since it requires less CPU time per alarm.

## Internal Register Definition 20.4. RTC0CN: SmaRTClock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0EN | MCLKEN | OSCFAIL | RTC0TR | RTC0AEN | ALRM | RTC0SET | RTC0CAP |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | Varies | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Address = 0x04

| Bit | Name | Function | | |
|---|---|---|---|---|
| 7 | RTC0EN | **SmaRTClock Enable.**<br>Enables/disables the SmaRTClock oscillator and associated bias currents.<br>0: SmaRTClock oscillator disabled.<br>1: SmaRTClock oscillator enabled. | | |
| 6 | MCLKEN | **Missing SmaRTClock Detector Enable.**<br>Enables/disables the missing SmaRTClock detector.<br>0: Missing SmaRTClock detector disabled.<br>1: Missing SmaRTClock detector enabled. | | |
| 5 | OSCFAIL | **SmaRTClock Oscillator Fail Event Flag.**<br>Set by hardware when a missing SmaRTClock detector timeout occurs. Must be cleared by software. The value of this bit is not defined when the SmaRTClock oscillator is disabled. | | |
| 4 | RTC0TR | **SmaRTClock Timer Run Control.**<br>Controls if the SmaRTClock timer is running or stopped (holds current value).<br>0: SmaRTClock timer is stopped.<br>1: SmaRTClock timer is running. | | |
| 3 | RTC0AEN | **SmaRTClock Alarm Enable.**<br>Enables/disables the SmaRTClock alarm function. Also clears the ALRM flag.<br>0: SmaRTClock alarm disabled.<br>1: SmaRTClock alarm enabled. | | |
| 2 | ALRM | **SmaRTClock Alarm Event Flag and Auto Reset Enable**<br>Reads return the state of the alarm event flag.<br>Writes enable/disable the Auto Reset function. | Read:<br>0: SmaRTClock alarm event flag is de-asserted.<br>1: SmaRTClock alarm event flag is asserted. | Write:<br>0: Disable Auto Reset.<br>1: Enable Auto Reset. |
| 1 | RTC0SET | **SmaRTClock Timer Set.**<br>Writing 1 initiates a SmaRTClock timer set operation. This bit is cleared to 0 by hardware to indicate that the timer set operation is complete. | | |
| 0 | RTC0CAP | **SmaRTClock Timer Capture.**<br>Writing 1 initiates a SmaRTClock timer capture operation. This bit is cleared to 0 by hardware to indicate that the timer capture operation is complete. | | |
| **Note:** The ALRM flag will remain asserted for a maximum of one SmaRTClock cycle. See Section “Power Management” on page 152 for information on how to capture a SmaRTClock Alarm event using a flag which is not automatically cleared by hardware. | | | | |

## Internal Register Definition 20.5. RTC0XCN: SmaRTClock Oscillator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AGCEN | XMODE | BIASX2 | CLKVLD | | | | |
| Type | R/W | R/W | R/W | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Address = 0x05

| Bit | Name | Function |
|---|---|---|
| 7 | AGCEN | **SmaRTClock Oscillator Automatic Gain Control (AGC) Enable.**<br>0: AGC disabled.<br>1: AGC enabled. |
| 6 | XMODE | **SmaRTClock Oscillator Mode.**<br>Selects Crystal or Self Oscillate Mode.<br>0: Self-Oscillate Mode selected.<br>1: Crystal Mode selected. |
| 5 | BIASX2 | **SmaRTClock Oscillator Bias Double Enable.**<br>**Enables/disables the Bias Double feature.**<br>0: Bias Double disabled.<br>1: Bias Double enabled. |
| 4 | CLKVLD | **SmaRTClock Oscillator Crystal Valid Indicator.**<br>Indicates if oscillation amplitude is sufficient for maintaining oscillation.<br>0: Oscillation has not started or oscillation amplitude is too low to maintain oscillation.<br>1: Sufficient oscillation amplitude detected. |
| 3:0 | Unused | Read = 0000b; Write = Don't Care. |

## Internal Register Definition 20.6. RTC0XCF: SmaRTClock Oscillator Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AUTOSTP | LOADRDY | | | LOADCAP | | | |
| Type | R/W | R | R | R | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | Varies | Varies | Varies | Varies |

SmaRTClock Address = 0x06

| Bit | Name | Function |
|---|---|---|
| 7 | AUTOSTP | **Automatic Load Capacitance Stepping Enable.**<br>Enables/disables automatic load capacitance stepping.<br>0: Load capacitance stepping disabled.<br>1: Load capacitance stepping enabled. |
| 6 | LOADRDY | **Load Capacitance Ready Indicator.**<br>Set by hardware when the load capacitance matches the programmed value.<br>0: Load capacitance is currently stepping.<br>1: Load capacitance has reached it programmed value. |
| 5:4 | Unused | Read = 00b; Write = Don't Care. |
| 3:0 | LOADCAP | **Load Capacitance Programmed Value.**<br>Holds the user's desired value of the load capacitance. See Table 20.2 on page 201. |

## Internal Register Definition 20.7. RTC0PIN: SmaRTClock Pin Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0PIN | | | | | | | |
| Type | W | | | | | | | |
| Reset | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |

SmaRTClock Address = 0x07

| Bit | Name | Function |
|---|---|---|
| 7:0 | RTC0PIN | **SmaRTClock Pin Configuration.**<br>Writing 0xE7 to this register forces XTAL3 and XTAL4 to be internally shorted for use with Self Oscillate Mode.<br>Writing 0x67 returns XTAL3 and XTAL4 to their normal configuration. |

## Internal Register Definition 20.8. CAPTUREn: SmaRTClock Timer Capture

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CAPTURE[31:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Addresses: CAPTURE0 = 0x00; CAPTURE1 = 0x01; CAPTURE2 =0x02; CAPTURE3: 0x03.

| Bit | Name | Function |
|---|---|---|
| 7:0 | CAPTURE[31:0] | **SmaRTClock Timer Capture.**<br>These 4 registers (CAPTURE3–CAPTURE0) are used to read or set the 32-bit SmaRTClock timer. Data is transferred to or from the SmaRTClock timer when the RTC0SET or RTC0CAP bits are set. |
| **Note:** The least significant bit of the timer capture value is in CAPTURE0.0. | | |

## Internal Register Definition 20.9. ALARMn: SmaRTClock Alarm Programmed Value

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ALARM[31:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Addresses: ALARM0 = 0x08; ALARM1 = 0x09; ALARM2 = 0x0A; ALARM3 = 0x0B

| Bit | Name | Function |
|---|---|---|
| 7:0 | ALARM[31:0] | **SmaRTClock Alarm Programmed Value.**<br>These 4 registers (ALARM3–ALARM0) are used to set an alarm event for the SmaRTClock timer. The SmaRTClock alarm should be disabled (RTC0AEN=0) when updating these registers. |
| **Note:** The least significant bit of the alarm programmed value is in ALARM0.0. | | |

## 21. Port Input/Output

Digital and analog resources are available through 19 or 16 I/O pins. The EZRadioPRO peripheral provides an additional 3 GPIO pins which are independent of the pins described in this chapter. Port pins are organized as three byte-wide ports. Port pins P0.0–P2.6 can be defined as digital or analog I/O. Digital I/O pins can be assigned to one of the internal digital resources or used as general purpose I/O (GPIO). Analog I/O pins are used by the internal analog resources. P1.0, P1.1, P1.2, and P1.4 are dedicated for communication with the EZRadioPRO peripheral. P1.3 is not available. P2.4, P2.5, and P2.6 are only available on the Si1000/1/2/3. P2.7 can be used as GPIO and is shared with the C2 Interface Data signal (C2D). See Section “29. Device Specific Behavior” on page 376 for more details.

The designer has complete control over which digital and analog functions are assigned to individual Port pins, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. See Section 21.3 for more information on the Crossbar.

All Px.x Port I/Os are 5V tolerant when used as digital inputs or open-drain outputs. For Port I/Os configured as push-pull outputs, current is sourced from the VDD_MCU supply. Port I/Os used for analog functions can operate up to the VDD_MCU supply voltage. See Section 21.1 for more information on Port I/O operating modes and the electrical specifications chapter for detailed electrical specifications.

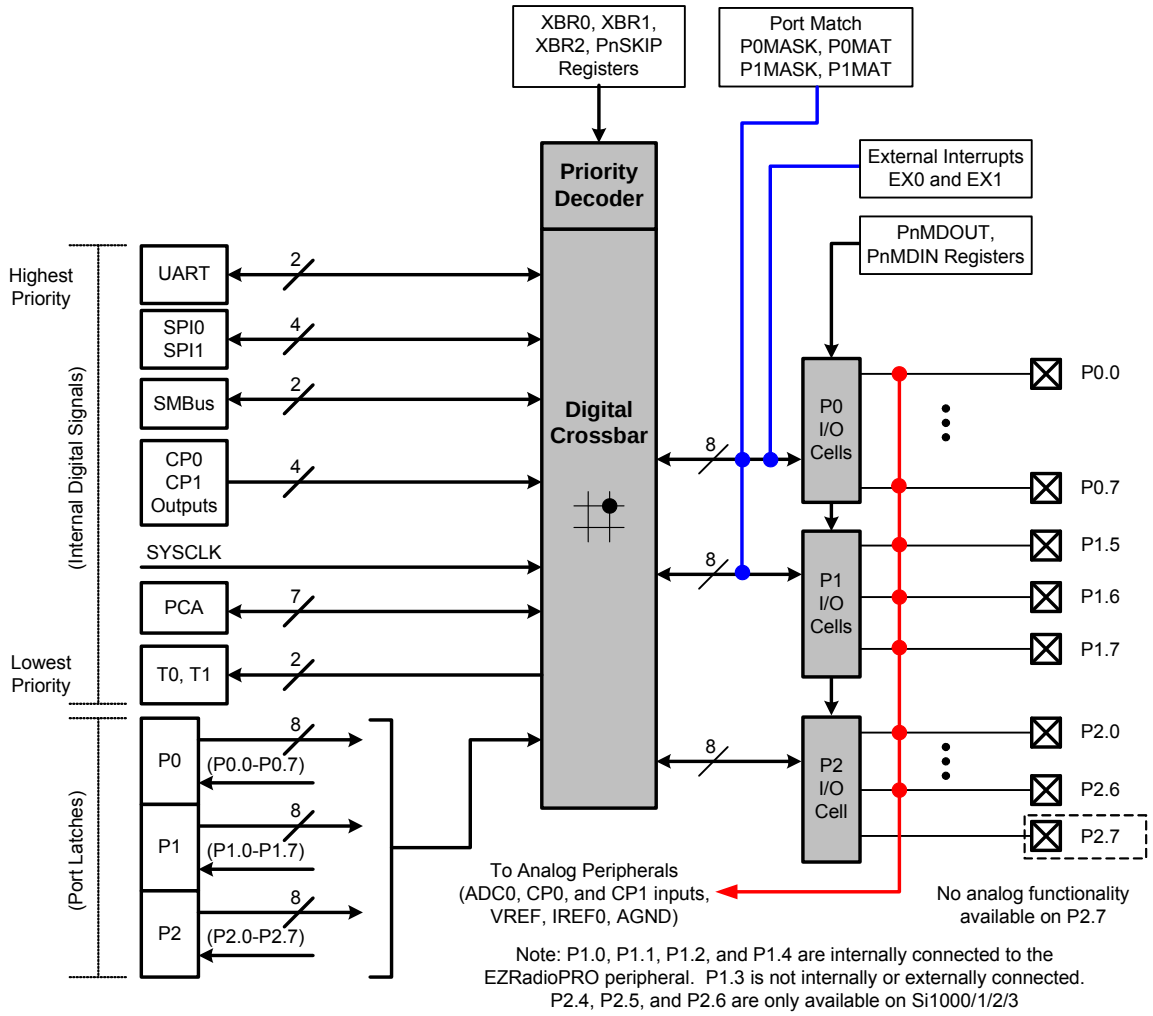



**Figure 21.1. Port I/O Functional Block Diagram**

## 21.1. Port I/O Modes of Operation

Port pins P0.0–P2.6 use the Port I/O cell shown in Figure 21.2. Each Port I/O cell can be configured by software for analog I/O or digital I/O using the PnMDIN registers. On reset, all Port I/O cells default to a digital high impedance state with weak pull-ups enabled.

### 21.1.1. Port Pins Configured for Analog I/O

Any pins to be used as Comparator or ADC input, external oscillator input/output, or AGND, VREF, or Current Reference output should be configured for analog I/O (PnMDIN.n = 0). When a pin is configured for analog I/O, its weak pullup and digital receiver are disabled. In most cases, software should also disable the digital output drivers. Port pins configured for analog I/O will always read back a value of 0 regardless of the actual voltage on the pin.

Configuring pins as analog I/O saves power and isolates the Port pin from digital interference. Port pins configured as digital inputs may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

### 21.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external digital event capture functions, or as GPIO should be configured as digital I/O (PnMDIN.n = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

Push-pull outputs (PnMDOUT.n = 1) drive the Port pad to the VDD_MCU or GND supply rails based on the output logic value of the Port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the Port pad to GND when the output logic value is 0 and become high impedance inputs (both high and low drivers turned off) when the output logic value is 1.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the Port pad to the VDD_MCU supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to GND to minimize power consumption and may be globally disabled by setting WEAKPUD to 1. The user must ensure that digital I/O are always internally or externally pulled or driven to a valid logic state. Port pins configured for digital I/O always read back the logic state of the Port pad, regardless of the output logic value of the Port pin.

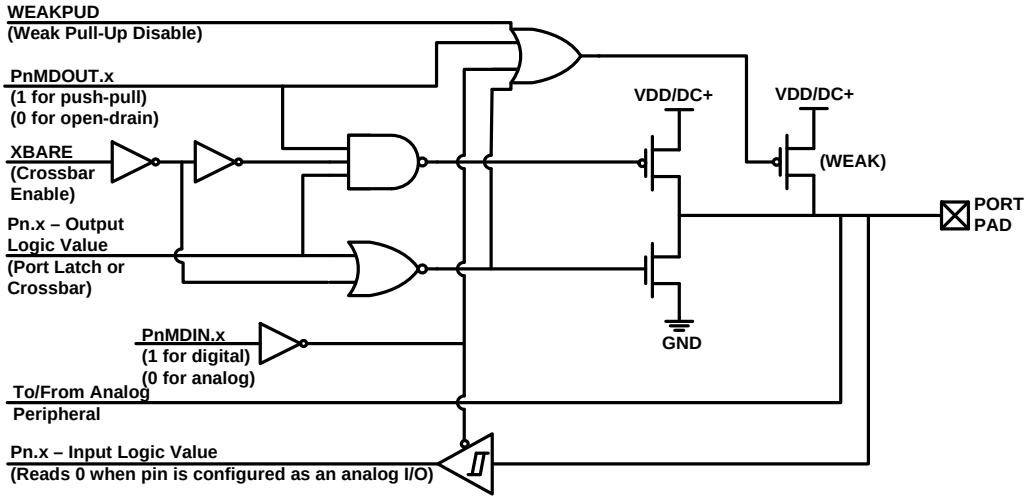



**Figure 21.2. Port I/O Cell Block Diagram**

### 21.1.3. Interfacing Port I/O to 5 V and 3.3 V Logic

All Port I/O configured for digital, open-drain operation are capable of interfacing to digital logic operating at a supply voltage higher than 4.5 V and less than 5.25 V. When the supply voltage is in the range of 1.8 to 2.2 V, the I/O may also interface to digital logic operating between 3.0 to 3.6 V if the input signal frequency is less than 12.5 MHz or less than 25 MHz if the signal rise time (10% to 90%) is less than 1.2 ns. When operating at a supply voltage above 2.2 V, the device should not interface to 3.3 V logic; however, interfacing to 5 V logic is permitted. An external pull-up resistor to the higher supply voltage is typically required for most systems.

**Important Notes:**

- When interfacing to a signal that is between 4.5 and 5.25 V, the maximum clock frequency that may be input on a GPIO pin is 12.5 MHz. The exception to this rule is when routing an external CMOS clock to P0.3, in which case, a signal up to 25 MHz is valid as long as the rise time (10% to 90%) is shorter than 1.8 ns.
- When the supply voltage is less than 2.2 V and interfacing to a signal that is between 3.0 and 3.6 V, the maximum clock frequency that may be input on a GPIO pin is 3.125 MHz. The exception to this rule is when routing an external CMOS clock to P0.3, in which case, a signal up to 25 MHz is valued as long as the rise time (10% to 90%) is shorter than 1.2 ns.
- In a multi-voltage interface, the external pull-up resistor should be sized to allow a current of at least 150 µA to flow into the Port pin when the supply voltage is between (VDD_MCU/DC+ plus 0.4 V) and (VDD_MCU/DC+ plus 1.0 V). Once the Port pad voltage increases beyond this range, the current flowing into the Port pin is minimal.

These guidelines only apply to multi-voltage interfaces. Port I/Os may always interface to digital logic operating at the same supply voltage.

### 21.1.4. Increasing Port I/O Drive Strength

Port I/O output drivers support a high and low drive strength; the default is low drive strength. The drive strength of a Port I/O can be configured using the PnDRV registers. See Section “4. Electrical Characteristics” on page 40 for the difference in output drive strength between the two modes.

## 21.2. Assigning Port I/O Pins to Analog and Digital Functions

Port I/O pins P0.0–P2.6 can be assigned to various analog, digital, and external interrupt functions. The Port pins assuaged to analog functions should be configured for analog I/O and Port pins assuaged to digital or external interrupt functions should be configured for digital I/O.

### 21.2.1. Assigning Port I/O Pins to Analog Functions

Table 21.1 shows all available analog functions that need Port I/O assignments. **Port pins selected for these analog functions should have their digital drivers disabled (PnMDOUT.n = 0 and Port Latch = 1) and their corresponding bit in PnSKIP set to 1.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. Table 21.1 shows the potential mapping of Port I/O to each analog function.

**Table 21.1. Port I/O Assignment for Analog Functions**

| Analog Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| ADC Input | P0.0–P2.6 | ADC0MX, PnSKIP |
| Comparator0 Input | P0.0–P2.6 | CPT0MX, PnSKIP |
| Comparator1 Input | P0.0–P2.6 | CPT1MX, PnSKIP |
| Voltage Reference (VREF0) | P0.0 | REF0CN, PnSKIP |
| Analog Ground Reference (AGND) | P0.1 | REF0CN, PnSKIP |
| Current Reference (IREF0) | P0.7 | IREF0CN, PnSKIP |
| External Oscillator Input (XTAL1) | P0.2 | OSCXCN, PnSKIP |
| External Oscillator Output (XTAL2) | P0.3 | OSCXCN, PnSKIP |

### 21.2.2. Assigning Port I/O Pins to Digital Functions

Any Port pins not assigned to analog functions may be assigned to digital functions or used as GPIO. Most digital functions rely on the Crossbar for pin assignment; however, some digital functions bypass the Crossbar in a manner similar to the analog functions listed above. **Port pins used by these digital functions and any Port pins selected for use as GPIO should have their corresponding bit in PnSKIP set to 1.** Table 21.2 shows all available digital functions and the potential mapping of Port I/O to each digital function.

**Table 21.2. Port I/O Assignment for Digital Functions**

| Digital Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| UART0, SPI1, SPI0, SMBus, CP0 and CP1 Outputs, System Clock Output, PCA0, Timer0 and Timer1 External Inputs. | Any Port pin available for assignment by the Crossbar. This includes P0.0–P2.6 pins which have their PnSKIP bit set to 0.<br>**Note:** The Crossbar will always assign UART0 and SPI1 pins to fixed locations. | XBR0, XBR1, XBR2 |
| Any pin used for GPIO | P0.0–P2.6 | P0SKIP, P1SKIP, P2SKIP |

### 21.2.3. Assigning Port I/O Pins to External Digital Event Capture Functions

External digital event capture functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The digital event capture functions do not require dedicated pins and will function on both GPIO pins (PnSKIP = 1) and pins in use by the Crossbar (PnSKIP = 0). External digital even capture functions cannot be used on pins configured for analog I/O. Table 21.3 shows all available external digital event capture functions.

**Table 21.3. Port I/O Assignment for External Digital Event Capture Functions**

| Digital Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| External Interrupt 0 | P0.0–P0.7 | IT01CF |
| External Interrupt 1 | P0.0–P0.7 | IT01CF |
| Port Match | P0.0–P1.7 | P0MASK, P0MAT<br>P1MASK, P1MAT |

## 21.3. Priority Crossbar Decoder

The Priority Crossbar Decoder assigns a Port I/O pin to each software selected digital function using the fixed peripheral priority order shown in Figure 21.3. The registers XBR0, XBR1, and XBR2 defined in SFR Definition 21.1, SFR Definition 21.2, and SFR Definition 21.3 are used to select digital functions in the Crossbar. The Port pins available for assignment by the Crossbar include all Port pins (P0.0–P2.6) which have their corresponding bit in PnSKIP set to 0.

From Figure 21.3, the highest priority peripheral is UART0. If UART0 is selected in the Crossbar (using the XBRn registers), then P0.4 and P0.5 will be assigned to UART0. The next highest priority peripheral is SPI1. If SPI1 is selected in the Crossbar, then P1.0–P1.2 will be assigned to SPI1. P1.3 will be assigned if SPI1 is configured for 4-wire mode. The user should ensure that the pins to be assigned by the Crossbar have their PnSKIP bits set to 0.

For all remaining digital functions selected in the Crossbar, starting at the top of Figure 21.3 going down, the least-significant unskipped, unassigned Port pin(s) are assigned to that function. If a Port pin is already assigned (e.g., UART0 or SPI1 pins), or if its PnSKIP bit is set to 1, then the Crossbar will skip over the pin and find next available unskipped, unassigned Port pin. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to 1.

Figure 21.3 shows the Crossbar Decoder priority with no Port pins skipped (P0SKIP, P1SKIP, P2SKIP = 0x00); Figure 21.4 shows the Crossbar Decoder priority with the External Oscillator pins (XTAL1 and XTAL2) skipped (P0SKIP = 0x0C).

**Notes:**

- The Crossbar must be enabled (XBARE = 1) before any Port pin is used as a digital output. Port output drivers are disabled while the Crossbar is disabled.
- When SMBus is selected in the Crossbar, the pins associated with SDA and SCL will automatically be forced into open-drain output mode regardless of the PnMDOUT setting.
- SPI0 can be operated in either 3-wire or 4-wire modes, depending on the state of the NSSMD1-NSSMD0 bits in register SPI0CN. The NSS signal is only routed to a Port pin when 4-wire mode is selected. When SPI0 is selected in the Crossbar, the SPI0 mode (3-wire or 4-wire) will affect the pinout of all digital functions lower in priority than SPI0.
- For given XBRn, PnSKIP, and SPInCN register settings, one can determine the I/O pin-out of the device using Figure 21.3 and Figure 21.4.

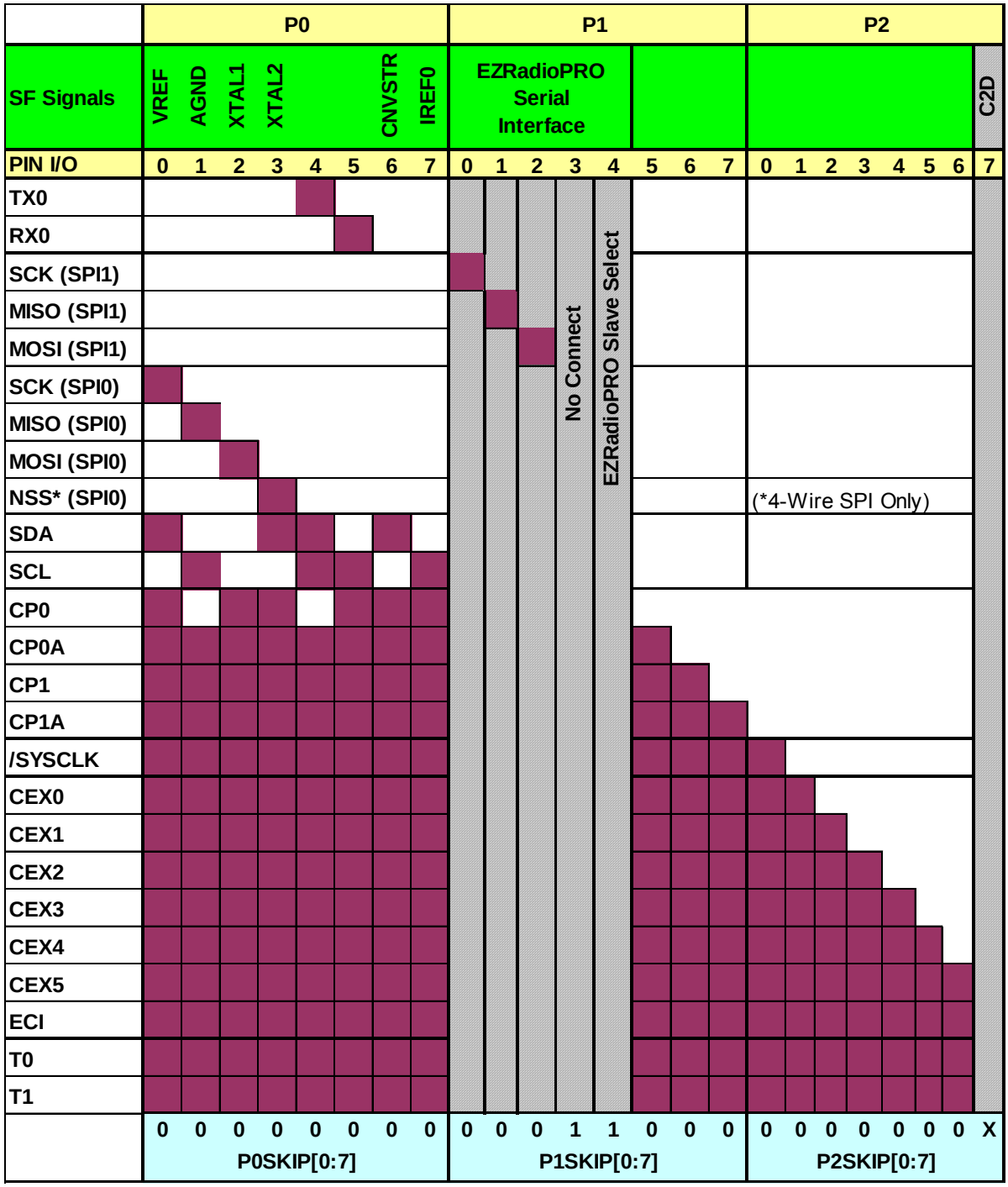



**Figure 21.3. Crossbar Priority Decoder with No Pins Skipped**

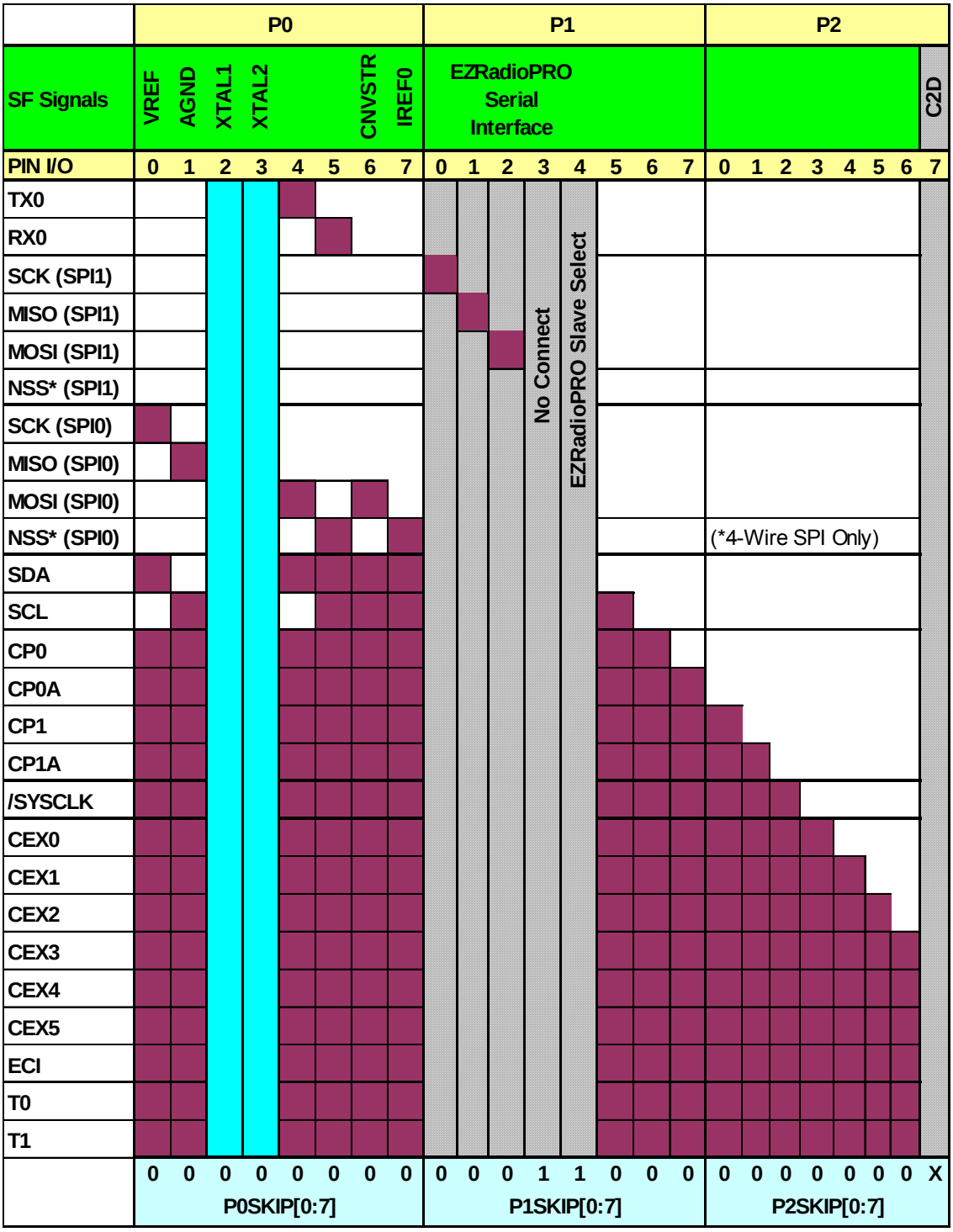



**Figure 21.4. Crossbar Priority Decoder with Crystal Pins Skipped**

## SFR Definition 21.1. XBR0: Port I/O Crossbar Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CP1AE | CP1E | CP0AE | CP0E | SYSCKE | SMB0E | SPI0E | URT0E |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE1

| Bit | Name | Function |
|---|---|---|
| 7 | CP1AE | **Comparator1 Asynchronous Output Enable.**<br>0: Asynchronous CP1 output unavailable at Port pin.<br>1: Asynchronous CP1 output routed to Port pin. |
| 6 | CP1E | **Comparator1 Output Enable.**<br>0: CP1 output unavailable at Port pin.<br>1: CP1 output routed to Port pin. |
| 5 | CP0AE | **Comparator0 Asynchronous Output Enable.**<br>0: Asynchronous CP0 output unavailable at Port pin.<br>1: Asynchronous CP0 output routed to Port pin. |
| 4 | CP0E | **Comparator0 Output Enable.**<br>0: CP1 output unavailable at Port pin.<br>1: CP1 output routed to Port pin. |
| 3 | SYSCKE | **$\overline{\text{SYSCLK}}$ Output Enable.**<br>0: $\overline{\text{SYSCLK}}$ output unavailable at Port pin.<br>1: $\overline{\text{SYSCLK}}$ output routed to Port pin. |
| 2 | SMB0E | **SMBus I/O Enable.**<br>0: SMBus I/O unavailable at Port pin.<br>1: SDA and SCL routed to Port pins. |
| 1 | SPI0E | **SPI0 I/O Enable**<br>0: SPI0 I/O unavailable at Port pin.<br>1: SCK, MISO, and MOSI (for SPI0) routed to Port pins.<br>NSS (for SPI0) routed to Port pin only if SPI0 is configured to 4-wire mode. |
| 0 | URT0E | **UART0 Output Enable.**<br>0: UART I/O unavailable at Port pin.<br>1: TX0 and RX0 routed to Port pins P0.4 and P0.5. |
| **Note:** SPI0 can be assigned either 3 or 4 Port I/O pins. | | |

## SFR Definition 21.2. XBR1: Port I/O Crossbar Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | SPI1E | T1E | T0E | ECIE | PCA0ME[2:0] | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE2

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 0b; Write = Don't Care. |
| 6 | SPI1E | **EZRadioPRO Serial Interface (SPI1) Enable.**<br>0: EZRadioPRO peripheral unavailable.<br>1: SCK (for EZRadioPRO) routed to P1.0.<br>MISO (for EZRadioPRO) routed to P1.1.<br>MOSI (for EZRadioPRO) routed to P1.2.<br>NSS (for EZRadioPRO) routed to P1.3 only if SPI1 is configured to 4-wire mode.<br>Note: When communicating with EZRadioPRO, the SPI1 should be configured to 3-wire mode and P1.4 should be used as a standard Port I/O pin to control NSS. |
| 5 | T1E | **Timer1 Input Enable.**<br>0: T1 input unavailable at Port pin.<br>1: T1 input routed to Port pin. |
| 4 | T0E | **Timer0 Input Enable.**<br>0: T0 input unavailable at Port pin.<br>1: T0 input routed to Port pin. |
| 3 | ECIE | **PCA0 External Counter Input (ECI) Enable.**<br>0: PCA0 external counter input unavailable at Port pin.<br>1: PCA0 external counter input routed to Port pin. |
| 2:0 | PCA0ME | **PCA0 Module I/O Enable.**<br>000: All PCA0 I/O unavailable at Port pin.<br>001: CEX0 routed to Port pin.<br>010: CEX0, CEX1 routed to Port pins.<br>011: CEX0, CEX1, CEX2 routed to Port pins.<br>100: CEX0, CEX1, CEX2 CEX3 routed to Port pins.<br>101: CEX0, CEX1, CEX2, CEX3, CEX4 routed to Port pins.<br>110: CEX0, CEX1, CEX2, CEX3, CEX4, CEX5 routed to Port pins.<br>111: Reserved. |
| **Note:** SPI1 can be assigned either 3 or 4 Port I/O pins. | | |

## SFR Definition 21.3. XBR2: Port I/O Crossbar Register 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | WEAKPUD | XBARE | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE3

| Bit | Name | Function |
|---|---|---|
| 7 | WEAKPUD | **Port I/O Weak Pullup Disable**<br>0: Weak Pullups enabled (except for Port I/O pins configured for analog mode). |
| 6 | XBARE | **Crossbar Enable**<br>0: Crossbar disabled.<br>1: Crossbar enabled. |
| 5:0 | Unused | Read = 000000b; Write = Don't Care. |
| **Note:** The Crossbar must be enabled (XBARE = 1) to use any Port pin as a digital output. | | |

## 21.4. Port Match

Port match functionality allows system events to be triggered by a logic value change on P0 or P1. A software controlled value stored in the PnMAT registers specifies the expected or normal logic values of P0 and P1. A Port mismatch event occurs if the logic levels of the Port's input pins no longer match the software controlled value. This allows Software to be notified if a certain change or pattern occurs on P0 or P1 input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which P0 and P1 pins should be compared against the PnMAT registers. A Port mismatch event is generated if (P0 & P0MASK) does not equal (PnMAT & P0MASK) or if (P1 & P1MASK) does not equal (PnMAT & P1MASK).

A Port mismatch event may be used to generate an interrupt or wake the device from a low power mode. See Section "12. Interrupt Handler" on page 130 and Section "14. Power Management" on page 152 for more details on interrupt and wake-up sources.

## SFR Definition 21.4. P0MASK: Port0 Mask Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MASK[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xC7

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MASK[7:0] | **Port0 Mask Value.**<br>Selects the P0 pins to be compared with the corresponding bits in P0MAT.<br>0: P0.n pin pad logic value is ignored and cannot cause a Port Mismatch event.<br>1: P0.n pin pad logic value is compared to P0MAT.n. |

## SFR Definition 21.5. P0MAT: Port0 Match Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page= 0x0; SFR Address = 0xD7

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MAT[7:0] | **Port 0 Match Value.**<br>Match comparison value used on Port 0 for bits in P0MASK which are set to 1.<br>0: P0.n pin logic value is compared with logic LOW.<br>1: P0.n pin logic value is compared with logic HIGH. |

## SFR Definition 21.6. P1MASK: Port1 Mask Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MASK[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xBF

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MASK[7:0] | **Port 1 Mask Value.**<br>Selects P1 pins to be compared to the corresponding bits in P1MAT.<br>0: P1.n pin logic value is ignored and cannot cause a Port Mismatch event.<br>1: P1.n pin logic value is compared to P1MAT.n. |
| **Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. | | |

## SFR Definition 21.7. P1MAT: Port1 Match Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xCF

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MAT[7:0] | **Port 1 Match Value.**<br>Match comparison value used on Port 1 for bits in P1MASK which are set to 1.<br>0: P1.n pin logic value is compared with logic LOW.<br>1: P1.n pin logic value is compared with logic HIGH. |
| **Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. | | |

## 21.5. Special Function Registers for Accessing and Configuring Port I/O

All Port I/O are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. The read-modify-write instructions when operating on a Port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SETB, when the destination is an individual bit in a Port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

Each Port has a corresponding PnSKIP register which allows its individual Port pins to be assigned to digital functions or skipped by the Crossbar. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to 1.

The Port input mode of the I/O pins is defined using the Port Input Mode registers (PnMDIN). Each Port cell can be configured for analog or digital I/O. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is P2.7, which can only be used for digital I/O.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). Each Port Output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the PnMDOUT settings.

The drive strength of the output drivers are controlled by the Port Drive Strength (PnDRV) registers. The default is low drive strength. See Section “4. Electrical Characteristics” on page 40 for the difference in output drive strength between the two modes.

## SFR Definition 21.8. P0: Port0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = All Pages; SFR Address = 0x80; Bit-Addressable

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:0 | P0[7:0] | **Port 0 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P0.n Port pin is logic LOW.<br>1: P0.n Port pin is logic HIGH. |

## SFR Definition 21.9. P0SKIP: Port0 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0SKIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xD4

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0SKIP[7:0] | **Port 0 Crossbar Skip Enable Bits.**<br>These bits select Port 0 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar.<br>0: Corresponding P0.n pin is not skipped by the Crossbar.<br>1: Corresponding P0.n pin is skipped by the Crossbar. |

## SFR Definition 21.10. P0MDIN: Port0 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDIN[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page= 0x0; SFR Address = 0xF1

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDIN[7:0] | **Analog Configuration Bits for P0.7–P0.0 (respectively).**<br>Port pins configured for analog mode have their weak pullup, and digital receiver disabled. The digital driver is not explicitly disabled.<br>0: Corresponding P0.n pin is configured for analog mode.<br>1: Corresponding P0.n pin is not configured for analog mode. |

## SFR Definition 21.11. P0MDOUT: Port0 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDOUT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA4

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDOUT[7:0] | **Output Configuration Bits for P0.7–P0.0 (respectively).**<br>These bits control the digital driver even when the corresponding bit in register P0MDIN is logic 0.<br>0: Corresponding P0.n Output is open-drain.<br>1: Corresponding P0.n Output is push-pull. |

## SFR Definition 21.12. P0DRV: Port0 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0DRV[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0xA4

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0DRV[7:0] | **Drive Strength Configuration Bits for P0.7–P0.0 (respectively).**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: Corresponding P0.n Output has low output drive strength.<br>1: Corresponding P0.n Output has high output drive strength. |

## SFR Definition 21.13. P1: Port1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = All Pages; SFR Address = 0x90; Bit-Addressable

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:0 | P1[7:0] | **Port 1 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P1.n Port pin is logic LOW.<br>1: P1.n Port pin is logic HIGH. |

**Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected.

## SFR Definition 21.14. P1SKIP: Port1 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1SKIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD5

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1SKIP[7:0] | **Port 1 Crossbar Skip Enable Bits.**<br>These bits select Port 1 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar.<br>0: Corresponding P1.n pin is not skipped by the Crossbar.<br>1: Corresponding P1.n pin is skipped by the Crossbar. |

**Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. P1.3 and P1.4 should always be skipped in the crossbar.

## SFR Definition 21.15. P1MDIN: Port1 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDIN[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xF2

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDIN[7:0] | **Analog Configuration Bits for P1.7–P1.0 (respectively).**<br>Port pins configured for analog mode have their weak pullup and digital receiver disabled. The digital driver is not explicitly disabled.<br>0: Corresponding P1.n pin is configured for analog mode.<br>1: Corresponding P1.n pin is not configured for analog mode. |
| **Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. | | |

## SFR Definition 21.16. P1MDOUT: Port1 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDOUT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA5

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDOUT[7:0] | **Output Configuration Bits for P1.7–P1.0 (respectively).**<br>These bits control the digital driver even when the corresponding bit in register P1MDIN is logic 0.<br>0: Corresponding P1.n Output is open-drain.<br>1: Corresponding P1.n Output is push-pull. |
| **Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. | | |

## SFR Definition 21.17. P1DRV: Port1 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1DRV[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0xA5

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1DRV[7:0] | **Drive Strength Configuration Bits for P1.7–P1.0 (respectively).**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: Corresponding P1.n Output has low output drive strength.<br>1: Corresponding P1.n Output has high output drive strength. |
| **Note:** P1.0, P1.1, P1.2, and P1.4 are internally connected to the EZRadioPRO peripheral. P1.3 is not externally or internally connected. | | |

## SFR Definition 21.18. P2: Port2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = All Pages; SFR Address = 0xA0; Bit-Addressable

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7:0 | P2[7:0] | **Port 2 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P2.n Port pin is logic LOW.<br>1: P2.n Port pin is logic HIGH. |

## SFR Definition 21.19. P2SKIP: Port2 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2SKIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD6

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7:0 | P2SKIP[7:0] | **Port 1 Crossbar Skip Enable Bits.**<br>These bits select Port 2 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar.<br>0: Corresponding P2.n pin is not skipped by the Crossbar.<br>1: Corresponding P2.n pin is skipped by the Crossbar. | | |

## SFR Definition 21.20. P2MDIN: Port2 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | P2MDIN[6:0] | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xF3

| Bit | Name | Function |
|---|---|---|
| 7 | | Reserved. Read = 1b; Must Write 1b. |
| 6:0 | P2MDIN[3:0] | **Analog Configuration Bits for P2.6–P2.0 (respectively).**<br>Port pins configured for analog mode have their weak pullup and digital receiver disabled. The digital driver is not explicitly disabled.<br>0: Corresponding P2.n pin is configured for analog mode.<br>1: Corresponding P2.n pin is not configured for analog mode. |

## SFR Definition 21.21. P2MDOUT: Port2 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2MDOUT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA6

| Bit | Name | Function |
|---|---|---|
| 7:0 | P2MDOUT[7:0] | **Output Configuration Bits for P2.7–P2.0 (respectively).**<br>These bits control the digital driver even when the corresponding bit in register P2MDIN is logic 0.<br>0: Corresponding P2.n Output is open-drain.<br>1: Corresponding P2.n Output is push-pull. |

## SFR Definition 21.22. P2DRV: Port2 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2DRV[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0F; SFR Address = 0xA6

| Bit | Name | Function |
|---|---|---|
| 7:0 | P2DRV[7:0] | **Drive Strength Configuration Bits for P2.7–P2.0 (respectively).**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: Corresponding P2.n Output has low output drive strength.<br>1: Corresponding P2.n Output has high output drive strength. |

## 22. EZRadioPRO® Serial Interface (SPI1)

The EZRadioPRO serial interface (SPI1) provides access to the EZRadioPRO peripheral registers from software executing on the MCU core. The serial interface consists of two SPI peripherals—a dedicated SPI Master accessible from the MCU core and dedicated SPI Slave residing inside the EZRadioPRO peripheral. The SPI1 peripheral on the MCU core side can only be used in master mode to communicate with the EZRadioPRO slave device in three wire mode. NSS for the EZRadioPRO is provided using Port 1.4, which is internally routed to the EZRadioPRO peripheral. The EZRadioPRO Serial Interface provides a system interrupt to regulate SPI traffic between the MCU core and the EZRadioPRO peripheral. This interrupt is internally routed to the MCU core. The EZRadioPRO peripheral also has an nIRQ pin which should be routed external to the package back into an external interrupt pin. The nIRQ interrupt pin is independent of the EZRadioPRO Serial Interface.

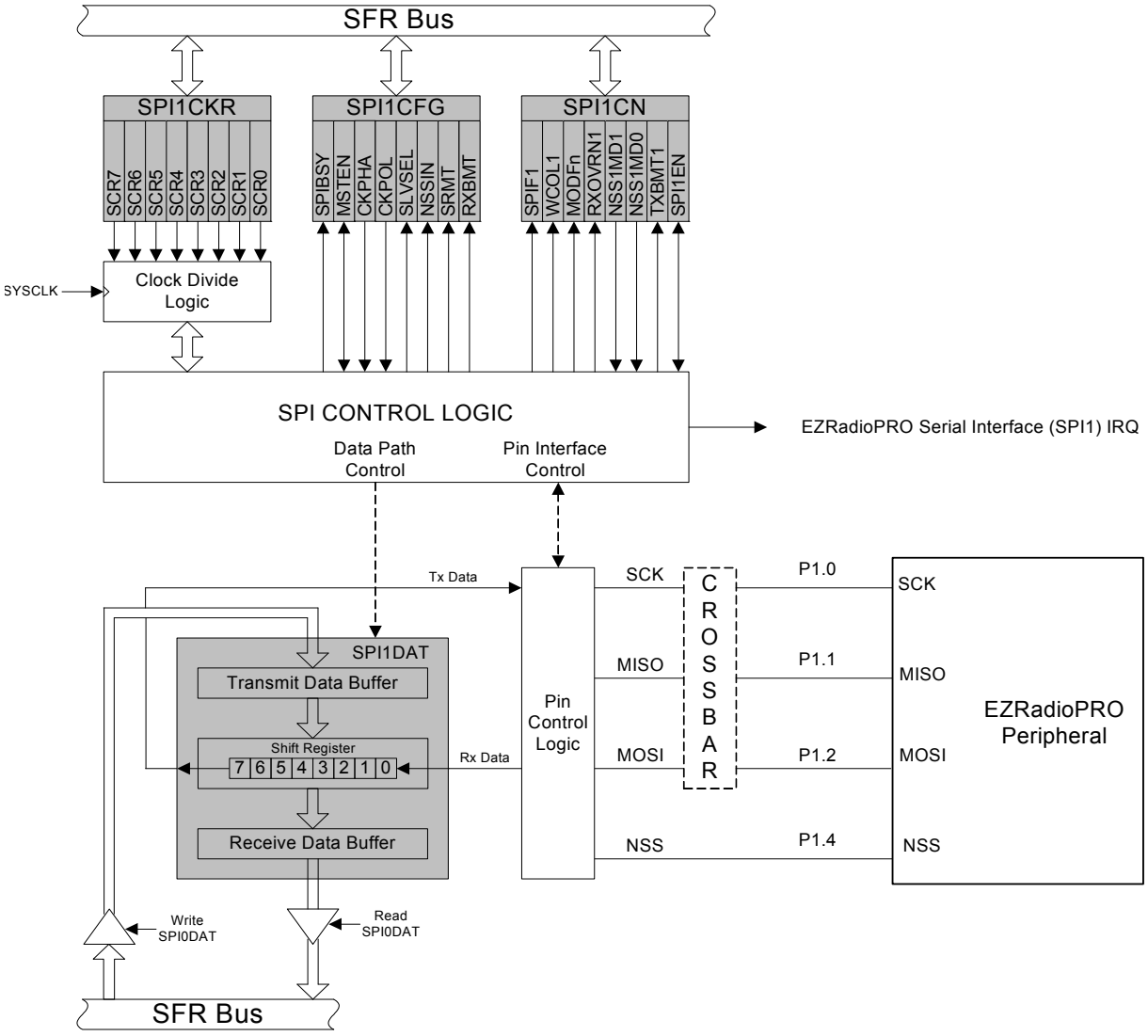



**Figure 22.1. EZRadioPRO Serial Interface Block Diagram**

## 22.1. Signal Descriptions

The four signals used by SPI1 (MOSI, MISO, SCK, NSS) are described below.

### 22.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output from the MCU core and an input to the EZRadioPRO peripheral. Data is transferred most-significant bit first. MOSI is driven by the MSB of the shift register.

### 22.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to master devices. It is used to serially transfer data from the EZRadioPRO to the MCU core. This signal is an input to the MCU core and an output from the EZRadioPRO peripheral. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled.

### 22.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI1 generates this signal.

### 22.1.4. Slave Select (NSS)

Since SPI1 operates in three wire mode, the NSS functionality built into the SPI state machine is not used. Instead, a Port pin must be configured to control the chip select on the EZRadioPRO peripheral.

## 22.2. SPI Master Operation on the MCU Core Side

A SPI master device initiates all data transfers on a SPI bus. SPI1 is placed in master mode by setting the Master Enable flag (MSTENn, SPI1CN.6). Writing a byte of data to the SPI1 data register (SPI1DAT) when in master mode writes to the transmit buffer. If the SPI shift register is empty, the byte in the transmit buffer is moved to the shift register, and a data transfer begins. The SPI1 master immediately shifts out the data serially on the MOSI line while providing the serial clock on SCK. The SPIF1 (SPI1CN.7) flag is set to logic 1 at the end of the transfer. If interrupts are enabled, an interrupt request is generated when the SPIF flag is set. While the SPI1 master transfers data to a slave on the MOSI line, the addressed SPI slave device simultaneously transfers the contents of its shift register to the SPI master on the MISO line in a full-duplex operation. Therefore, the SPIF flag serves as both a transmit-complete and receive-data-ready flag. The data byte received from the slave is transferred MSB-first into the master's shift register. When a byte is fully shifted into the register, it is moved to the receive buffer where it can be read by the processor by reading SPI1DAT.

## 22.3. SPI Slave Operation on the EZRadioPRO Peripheral Side

The EZRadioPRO peripheral presents a standard 4-wire SPI interface: SCK, MISO, MOSI and NSS. The SPI master can read data from the device on the MOSI output pin. A SPI transaction is a 16-bit sequence which consists of a Read-Write ($\overline{R}$/W) select bit, followed by a 7-bit address field (ADDR), and an 8-bit data field (DATA) as demonstrated in Figure 22.2. The 7-bit address field is used to select one of the 128, 8-bit control registers. The $\overline{R}$/W select bit determines whether the SPI transaction is a read or write transaction. If $\overline{R}$/W = 1 it signifies a WRITE transaction, while $\overline{R}$/W = 0 signifies a READ transaction. The contents (ADDR or DATA) are latched into the transceiver every eight clock cycles. The timing parameters for the SPI interface are shown in Table 22.1. The SCK rate is flexible with a maximum rate of 10 MHz.

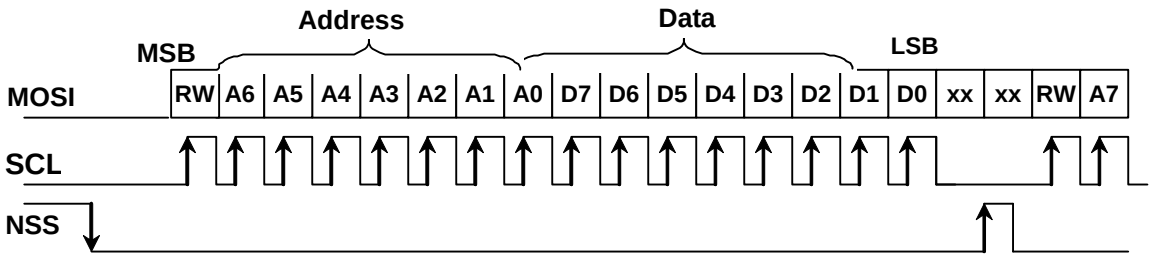
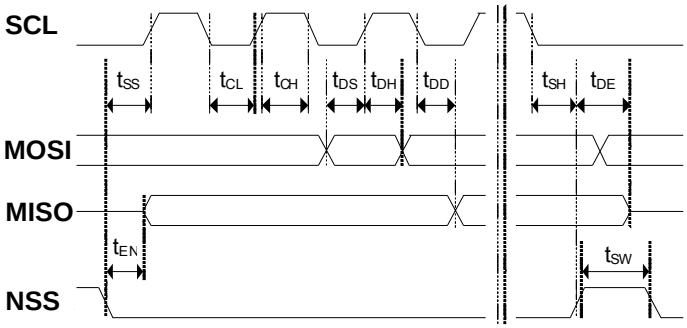



**Figure 22.2. SPI Timing**

**Table 22.1. Serial Interface Timing Parameters**

| Symbol | Parameter | Min (nsec) | Diagram |
|---|---|---|---|
| $t_{CH}$ | Clock high time | 40 |  |
| $t_{CL}$ | Clock low time | 40 | |
| $t_{DS}$ | Data setup time | 20 | |
| $t_{DH}$ | Data hold time | 20 | |
| $t_{DD}$ | Output data delay time | 20 | |
| $t_{EN}$ | Output enable time | 20 | |
| $t_{DE}$ | Output disable time | 50 | |
| $t_{SS}$ | Select setup time | 20 | |
| $t_{SH}$ | Select hold time | 50 | |
| $t_{SW}$ | Select high period | 80 | |



To read back data from the transceiver, the R/W bit must be set to 0 followed by the 7-bit address of the register from which to read. The 8 bit DATA field following the 7-bit ADDR field is ignored on the MOSI pin when R/W = 0. The next eight negative edge transitions of the SCK signal will clock out the contents of the selected register. The data read from the selected register will be available on the MISO output. The READ function is shown in Figure 22.3. After the READ function is completed the MISO signal will remain at either a logic 1 or logic 0 state depending on the last data bit clocked out (D0). When NSS goes high the MISO output pin will be pulled high by internal pullup.

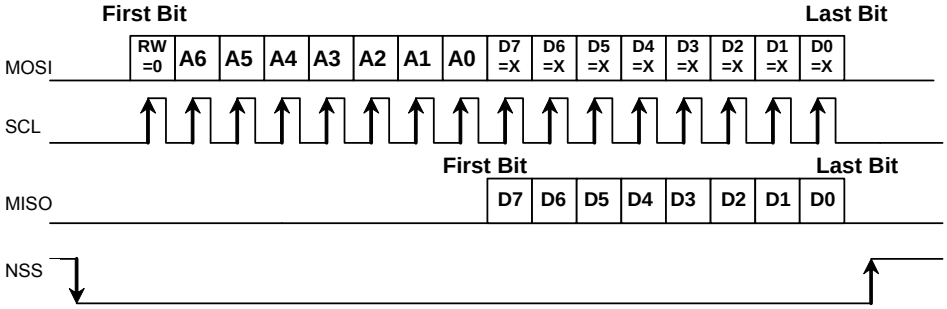



**Figure 22.3. SPI Timing—READ Mode**

The SPI interface contains a burst read/write mode which allows for reading/writing sequential registers without having to re-send the SPI address. When the NSS bit is held low while continuing to send SCK pulses, the SPI interface will automatically increment the ADDR and read from/write to the next address. An example burst write transaction is illustrated in Figure 22.4 and a burst read in Figure 22.5. As long as NSS is held low, input data will be latched into the transceiver every eight SCK cycles.

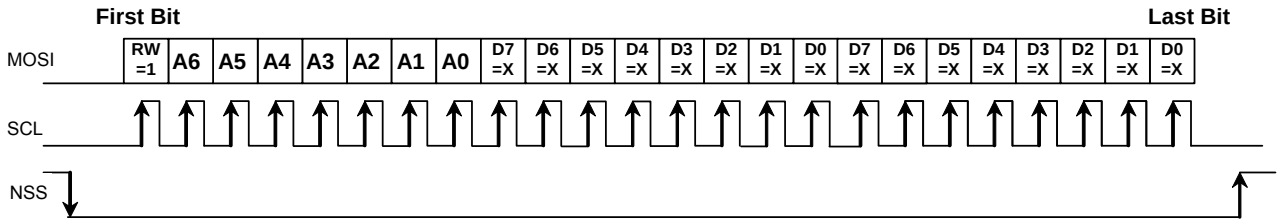



**Figure 22.4. SPI Timing—Burst Write Mode**

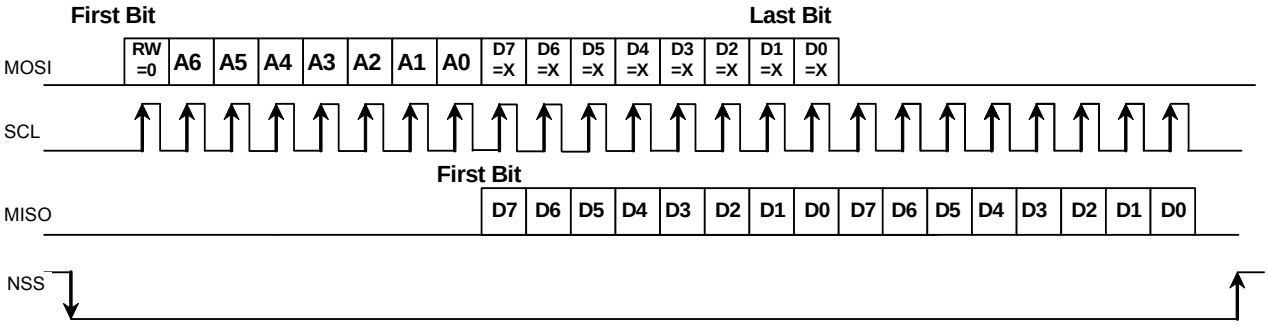



**Figure 22.5. SPI Timing—Burst Read Mode**

## 22.4. EZRadioPRO Serial Interface Interrupt Sources

When SPI1 interrupts are enabled, the following flags will generate an interrupt when they are set to logic 1:

*All of the following bits must be cleared by software.*

1. The SPI Interrupt Flag, SPIFn (SPInCN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPIn modes.
2. The Write Collision Flag, WCOLn (SPInCN.6) is set to logic 1 if a write to SPInDAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPInDAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPIn modes.
3. The Mode Fault Flag MODFn (SPInCN.5) is set to logic 1 when SPIn is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTENn and SPIENn bits in SPI0CN are set to logic 0 to disable SPIn and allow another master device to access the bus.
4. The Receive Overrun Flag RXOVRNn (SPInCN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

## 22.5. Serial Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI Configuration Register (SPI1CFG). The CKPHA bit (SPI1CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI1CFG.4) selects between an active-high or active-low clock. Both CKPOL and CKPHA must be set to zero in order to communicate with the EZRadioPRO peripheral.

The SPI1 Clock Rate Register (SPI1CKR) as shown in SFR Definition 22.3 controls the master mode serial clock frequency. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower.

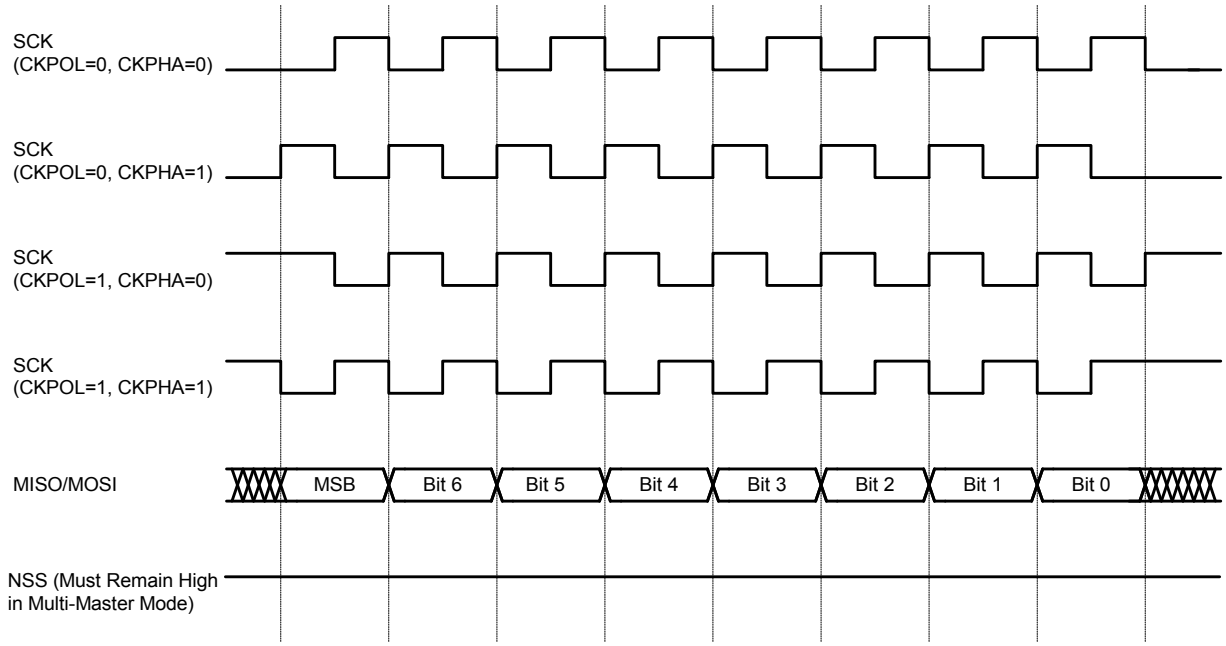



**Figure 22.6. Master Mode Data/Clock Timing**

## 22.6. SPI Special Function Registers

SPI1 is accessed and controlled through four special function registers in the system controller: SPI1CN Control Register, SPI1DAT Data Register, SPI1CFG Configuration Register, and SPI1CKR Clock Rate Register. The special function registers related to the operation of the SPI1 Bus are described in the following figures.

## SFR Definition 22.1. SPI1CFG: SPI Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIBSY | MSTEN | CKPHA | CKPOL | | | | |
| Type | R | R/W | R/W | R/W | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0x84

| Bit | Name | Function |
|---|---|---|
| 7 | SPIBSY | **SPI Busy.**<br>This bit is set to logic 1 when a SPI transfer is in progress. |
| 6 | MSTEN | **Master Mode Enable.**<br>When set to '1', enables master mode. This bit must be set to 1 to communicate with the EZRadioPRO peripheral. |
| 5 | CKPHA | **SPI Clock Phase.**<br>0: Data centered on first edge of SCK period.*<br>1: Data centered on second edge of SCK period.* |
| 4 | CKPOL | **SPI Clock Polarity.**<br>0: SCK line low in idle state.<br>1: SCK line high in idle state. |
| 3:0 | | **Reserved.**<br>Read = 0000, Write = don't care. |

**Note:** In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 22.2 for timing parameters.

## SFR Definition 22.2. SPI1CN: SPI Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIF1 | WCOL1 | MODF1 | | NSS1MD1 | NSS1MD0 | TXBMT1 | SPI1EN |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

SFR Page = 0x0; SFR Address = 0xB0; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | SPIF1 | **SPI1 Interrupt Flag.**<br>This bit is set to logic 1 by hardware at the end of a data transfer. If interrupts are enabled, setting this bit causes the CPU to vector to the SPI1 interrupt service routine. This bit is not automatically cleared by hardware. It must be cleared by software. |
| 6 | WCOL1 | **Write Collision Flag.**<br>This bit is set to logic 1 by hardware (and generates a SPI1 interrupt) to indicate a write to the SPI1 data register was attempted while a data transfer was in progress. It must be cleared by software. |
| 5 | MODF1 | **Mode Fault Flag.**<br>This bit is set to logic 1 by hardware (and generates a SPI1 interrupt) when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). This bit is not automatically cleared by hardware. It must be cleared by software. |
| 4 | | **Reserved.**<br>Read = varies; Write = must write zero. |
| 3:2 | NSS1MD[1:0] | **Slave Select Mode.**<br>Must be set to 00b. SPI1 can only be used in 3-wire master mode. |
| 1 | TXBMT1 | **Transmit Buffer Empty.**<br>This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer. |
| 0 | SPI1EN | **SPI1 Enable.**<br>0: SPI1 disabled.<br>1: SPI1 enabled. |

## SFR Definition 22.3. SPI1CKR: SPI Clock Rate

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SCR1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x85

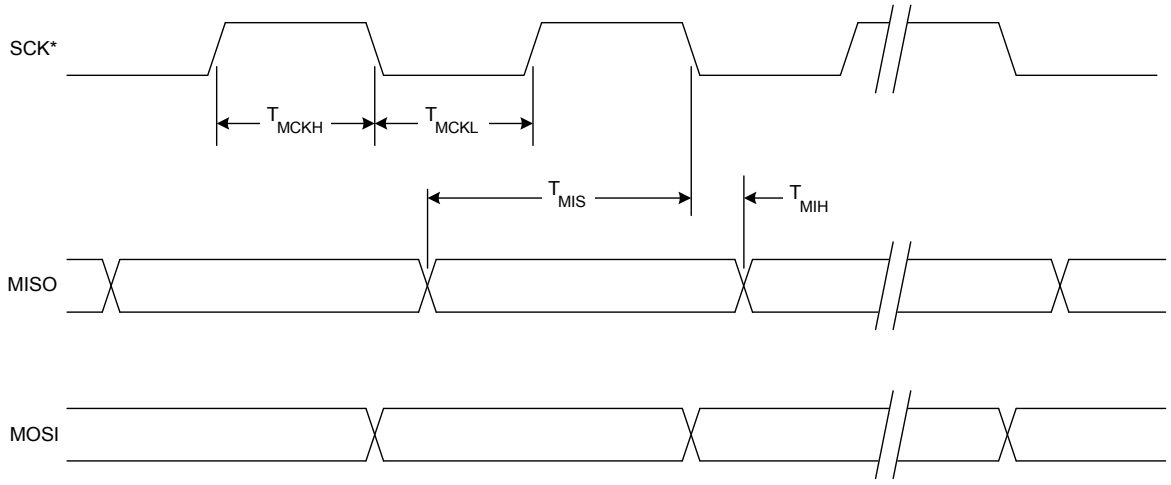
| Bit | Name | Function |
|---|---|---|
| 7:0 | SCR1 | **SPI Clock Rate.**<br>These bits determine the frequency of the SCK output when the SPI module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI1CKR* is the 8-bit value held in the SPI1CKR register.<br>$f_{SCK} = \frac{SYSCLK}{2 \times (SPI1CKR[7:0] + 1)}$<br>for 0 <= SPI0CKR <= 255<br><br>Example: If SYSCLK = 2 MHz and SPI1CKR = 0x04,<br>$f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$<br>$f_{SCK} = 200kHz$ |

## SFR Definition 22.4. SPI1DAT: SPI Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPI1DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x86

| Bit | Name | Function |
|---|---|---|
| 7:0 | SPI1DAT | **SPI1 Transmit and Receive Data.**<br>The SPI1DAT register is used to transmit and receive SPI1 data. Writing data to SPI1DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI1DAT returns the contents of the receive buffer. |




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 22.7. SPI Master Timing**

**Table 22.2. SPI Timing Parameters**

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing** | | | | |
| $T_{MCKH}$ | SCK High Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MCKL}$ | SCK Low Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MIS}$ | MISO Valid to SCK Shift Edge | 1 x $T_{SYSCLK}$ + 20 | — | ns |
| $T_{MIH}$ | SCK Shift Edge to MISO Change | 0 | — | ns |
| **Note:** $T_{SYSCLK}$ is equal to one period of the device system clock (SYSCLK). | | | | |

## 23. EZRadioPRO® 240–960 MHz Transceiver

Si1000/1/2/3/4/5 devices include the EZRadioPRO family of ISM wireless transceivers with continuous frequency tuning over 240–960 MHz. The wide operating voltage range of 1.8–3.6 V and low current consumption makes the EZRadioPRO an ideal solution for battery powered applications.

The EZRadioPRO transceiver operates as a time division duplexing (TDD) transceiver where the device alternately transmits and receives data packets. The device uses a single-conversion mixer to downconvert the 2-level FSK/GFSK/OOK modulated receive signal to a low IF frequency. Following a programmable gain amplifier (PGA) the signal is converted to the digital domain by a high performance ΔΣ ADC allowing filtering, demodulation, slicing, and packet handling to be performed in the built-in DSP increasing the receiver's performance and flexibility versus analog based architectures. The demodulated signal is then output to the system MCU through a programmable GPIO or via the standard SPI bus by reading the 64-byte RX FIFO.

A single high precision local oscillator (LO) is used for both transmit and receive modes since the transmitter and receiver do not operate at the same time. The LO is generated by an integrated VCO and ΔΣ Fractional-N PLL synthesizer. The synthesizer is designed to support configurable data rates, output frequency and frequency deviation at any frequency between 240–960 MHz. The transmit FSK data is modulated directly into the ΔΣ data stream and can be shaped by a Gaussian low-pass filter to reduce unwanted spectral content.

The Si1000's PA output power can be configured between –1 and +20 dBm in 3 dB steps, while the Si1002/3/4/5's PA output power can be configured between –8 and +13 dBm in 3 dB steps. The PA is single-ended to allow for easy antenna matching and low BOM cost. The PA incorporates automatic ramp-up and rampdown control to reduce unwanted spectral spreading. The +20 dBm power amplifier of the Si1000/1 can also be used to compensate for the reduced performance of a lower cost, lower performance antenna or antenna with size constraints due to a small form-factor. Competing solutions require large and expensive external PAs to achieve comparable performance. The EZRadioPRO transceivers support frequency hopping, TX/RX switch control, and antenna diversity switch control to extend the link range and improve performance.

The EZRadioPRO peripheral also controls three GPIO pins: GPIO_0, GPIO_1, and GPIO_2. See Application Note "AN415: EZRadioPRO Programming Guide" for details on initializing and using the EZRadioPRO peripheral.

## 23.1. EZRadioPRO Operating Modes

The EZRadioPRO transceivers provide several operating modes which can be used to optimize the power consumption for a given application. Depending upon the system communication protocol, an optimal trade-off between the radio wake time and power consumption can be achieved.

Table 23.1 summarizes the operating modes of the EZRadioPRO transceivers. In general, any given operating mode may be classified as an active mode or a power saving mode. The table indicates which block(s) are enabled (active) in each corresponding mode. With the exception of the SHUTDOWN mode, all can be dynamically selected by sending the appropriate commands over the SPI. An “X” in any cell means that, in the given mode of operation, that block can be independently programmed to be either ON or OFF, without noticeably impacting the current consumption. The SPI circuit block includes the SPI interface hardware and the device register space. The 32 kHz OSC block includes the 32.768 kHz RC oscillator or 32.768 kHz crystal oscillator and wake-up timer. AUX (Auxiliary Blocks) includes the temperature sensor, general purpose ADC, and low-battery detector.

**Table 23.1. EZRadioPRO Operating Modes**

| Mode Name | Circuit Blocks | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Digital LDO | SPI | 32 kHz OSC | AUX | 30 MHz XTAL | PLL | PA | RX | $I_{VDD}$ |
| SHUTDOWN | OFF (Register contents lost) | OFF | OFF | OFF | OFF | OFF | OFF | OFF | 15 nA |
| STANDBY | ON (Register contents retained) | ON | OFF | OFF | OFF | OFF | OFF | OFF | 450 nA |
| SLEEP | | ON | ON | X | OFF | OFF | OFF | OFF | 1 µA |
| SENSOR | | ON | X | ON | OFF | OFF | OFF | OFF | 1 µA |
| READY | | ON | X | X | ON | OFF | OFF | OFF | 600 µA |
| TUNING | | ON | X | X | ON | ON | OFF | OFF | 8.5 mA |
| TRANSMIT | | ON | X | X | ON | ON | ON | OFF | 30 mA* |
| RECEIVE | | ON | X | X | ON | ON | OFF | ON | 18.5 mA |

**Note:** Using Si1002/3 at +13 dBm using recommended reference design. These power modes are for the EZRadioPRO peripheral only and are independent of the MCU power modes.

### 23.1.1. Operating Mode Control

There are four primary states in the EZRadioPRO transceiver radio state machine: SHUTDOWN, IDLE, TX, and RX (see Figure 23.1). The SHUTDOWN state completely shuts down the radio to minimize current consumption. There are five different configurations/options for the IDLE state which can be selected to optimize the chip to the applications needs. "Register 07h. Operating Mode and Function Control 1" controls which operating mode/state is selected with the exception of SHUTDOWN which is controlled by SDN pin 20. The TX and RX state may be reached automatically from any of the IDLE states by setting the txon/rxon bits in "Register 07h. Operating Mode and Function Control 1". Table 23.2 shows each of the operating modes with the time required to reach either RX or TX mode as well as the current consumption of each mode.

The transceivers include a low-power digital regulated supply (LPLDO) which is internally connected in parallel to the output of the main digital regulator (and is available externally at the VR_DIG pin). This common digital supply voltage is connected to all digital circuit blocks including the digital modem, crystal oscillator, SPI, and register space. The LPLDO has extremely low quiescent current consumption but limited current supply capability; it is used only in the IDLE-STANDBY and IDLE-SLEEP modes. The main digital regulator is automatically enabled in all other modes.

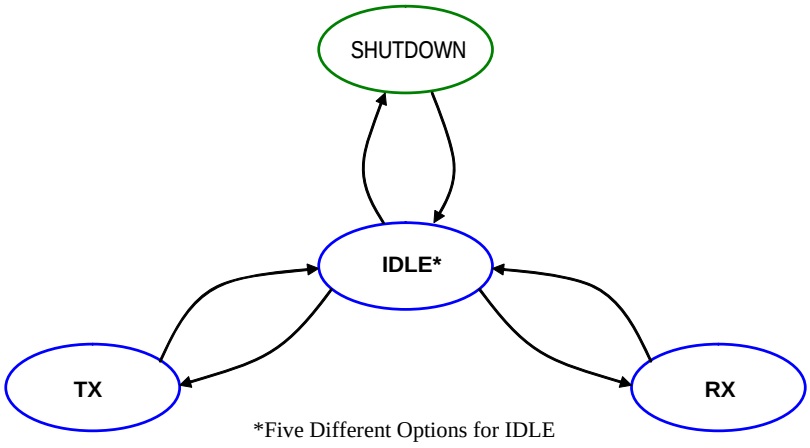



**Figure 23.1. State Machine Diagram**

**Table 23.2. EZRadioPRO Operating Modes Response Time**

| State/Mode | Response Time to | | Current in State /Mode [µA] |
|---|---|---|---|
| | TX | RX | |
| Shut Down State | 16.8 ms | 16.8 ms | 15 nA |
| Idle States: | | | |
| Standby Mode | 800 µs | 800 µs | 450 nA |
| Sleep Mode | 800 µs | 800 µs | 1 µA |
| Sensor Mode | 800 µs | 800 µs | 1 µA |
| Ready Mode | 200 µs | 200 µs | 800 µA |
| Tune Mode | 200 µs | 200 µs | 8.5 mA |
| TX State | NA | 200 µs | 30 mA @ +13 dBm |
| RX State | 200 µs | NA | 18.5 mA |

#### 23.1.1.1. SHUTDOWN State

The SHUTDOWN state is the lowest current consumption state of the device with nominally less than 15 nA of current consumption. The shutdown state may be entered by driving the SDN pin (Pin 20) high. The SDN pin should be held low in all states except the SHUTDOWN state. In the SHUTDOWN state, the contents of the registers are lost and there is no SPI access.

When the chip is connected to the power supply, a POR will be initiated after the falling edge of SDN. After a POR, the device will be in READY mode with the buffers enabled.

#### 23.1.1.1.1. IDLE State

There are five different modes in the IDLE state which may be selected by "Register 07h. Operating Mode and Function Control 1". All modes have a tradeoff between current consumption and response time to TX/RX mode. This tradeoff is shown in Table 23.2. After the POR event, SWRESET, or exiting from the SHUTDOWN state the chip will default to the IDLE-READY mode. After a POR event the interrupt registers must be read to properly enter the SLEEP, SENSOR, or STANDBY mode and to control the 32 kHz clock correctly.

#### 23.1.1.1.2. STANDBY Mode

STANDBY mode has the lowest current consumption of the five IDLE states with only the LPLDO enabled to maintain the register values. In this mode the registers can be accessed in both read and write mode. The STANDBY mode can be entered by writing 0h to "Register 07h. Operating Mode and Function Control 1". If an interrupt has occurred (i.e., the nIRQ pin = 0) the interrupt registers must be read to achieve the minimum current consumption. Additionally, the ADC should not be selected as an input to the GPIO in this mode as it will cause excess current consumption.

#### 23.1.1.1.3. SLEEP Mode

In SLEEP mode the LPLDO is enabled along with the Wake-Up-Timer, which can be used to accurately wake-up the radio at specified intervals. See “Wake-Up Timer and 32 kHz Clock Source” on page 278 for more information on the Wake-Up-Timer. SLEEP mode is entered by setting enwt = 1 (40h) in "Register 07h. Operating Mode and Function Control 1". If an interrupt has occurred (i.e., the nIRQ pin = 0) the interrupt registers must be read to achieve the minimum current consumption. Also, the ADC should not be selected as an input to the GPIO in this mode as it will cause excess current consumption.

#### 23.1.1.1.4. SENSOR Mode

In SENSOR mode either the Low Battery Detector, Temperature Sensor, or both may be enabled in addition to the LPLDO and Wake-Up-Timer. The Low Battery Detector can be enabled by setting enlbd = 1 in "Register 07h. Operating Mode and Function Control 1". See “Temperature Sensor” on page 275 and “Low Battery Detector” on page 277 for more information on these features. If an interrupt has occurred (i.e., the nIRQ pin = 0) the interrupt registers must be read to achieve the minimum current consumption.

#### 23.1.1.1.5. READY Mode

READY Mode is designed to give a fast transition time to TX mode with reasonable current consumption. In this mode the Crystal oscillator remains enabled reducing the time required to switch to TX or RX mode by eliminating the crystal start-up time. READY mode is entered by setting xton = 1 in "Register 07h. Operating Mode and Function Control 1". To achieve the lowest current consumption state the crystal oscillator buffer should be disabled in “Register 62h. Crystal Oscillator Control and Test.” To exit READY mode, bufovr (bit 1) of this register must be set back to 0.

#### 23.1.1.1.6. TUNE Mode

In TUNE mode the PLL remains enabled in addition to the other blocks enabled in the IDLE modes. This will give the fastest response to TX mode as the PLL will remain locked but it results in the highest current consumption. This mode of operation is designed for frequency hopping spread spectrum systems (FHSS). TUNE mode is entered by setting pllon = 1 in "Register 07h. Operating Mode and Function Control 1". It is not necessary to set xton to 1 for this mode, the internal state machine automatically enables the crystal oscillator.

#### 23.1.1.2. TX State

The TX state may be entered from any of the IDLE modes when the txon bit is set to 1 in "Register 07h. Operating Mode and Function Control 1". A built-in sequencer takes care of all the actions required to transition between states from enabling the crystal oscillator to ramping up the PA. The following sequence of events will occur automatically when going from STANDBY mode to TX mode by setting the txon bit.

1. Enable the main digital LDO and the Analog LDOs.
2. Start up crystal oscillator and wait until ready (controlled by an internal timer).
3. Enable PLL.
4. Calibrate VCO (this action is skipped when the skipvco bit is 1, default value is 0).
5. Wait until PLL settles to required transmit frequency (controlled by an internal timer).
6. Activate power amplifier and wait until power ramping is completed (controlled by an internal timer).
7. Transmit packet.

Steps in this sequence may be eliminated depending on which IDLE mode the chip is configured to prior to setting the txon bit. By default, the VCO and PLL are calibrated every time the PLL is enabled.

#### 23.1.1.3. RX State

The RX state may be entered from any of the IDLE modes when the rxon bit is set to 1 in "Register 07h. Operating Mode and Function Control 1". A built-in sequencer takes care of all the actions required to transition from one of the IDLE modes to the RX state. The following sequence of events will occur automatically to get the chip into RX mode when going from STANDBY mode to RX mode by setting the rxon bit:

1. Enable the main digital LDO and the Analog LDOs.
2. Start up crystal oscillator and wait until ready (controlled by an internal timer).
3. Enable PLL.
4. Calibrate VCO (this action is skipped when the skipvco bit is 1, default value is 0).
5. Wait until PLL settles to required receive frequency (controlled by an internal timer).
6. Enable receive circuits: LNA, mixers, and ADC.
7. Enable receive mode in the digital modem.

Depending on the configuration of the radio all or some of the following functions will be performed automatically by the digital modem: AGC, AFC (optional), update status registers, bit synchronization, packet handling (optional) including sync word, header check, and CRC.

#### 23.1.1.4. Device Status

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 02 | R | Device Status | ffovfl | ffunfl | rxffem | headerr | freqerr | | cps[1] | cps[0] | — |

The operational status of the EZRadioPRO peripheral can be read from "Register 02h. Device Status".

## 23.2. Interrupts

The EZRadioPRO peripheral is capable of generating an interrupt signal (nIRQ) when certain events occur. The nIRQ pin is driven low to indicate a pending interrupt request. **The EZRadioPRO interrupt does not have an internal interrupt vector. To use the interrupt, the nIRQ pin must be looped back to an external interrupt input.** This interrupt signal will be generated when any one (or more) of the interrupt events (corresponding to the Interrupt Status bits) shown below occur. The nIRQ pin will remain low until the Interrupt Status Register(s) (Registers 03h–04h) containing the active Interrupt Status bit is read. The nIRQ output signal will then be reset until the next change in status is detected. The interrupts must be

enabled by the corresponding enable bit in the Interrupt Enable Registers (Registers 05h–06h). All enabled interrupt bits will be cleared when the corresponding interrupt status register is read. If the interrupt is not enabled when the event occurs it will not trigger the nIRQ pin, but the status may still be read at anytime in the Interrupt Status registers.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 03 | R | Interrupt Status 1 | ifferr | itxffafull | itxffaem | irxffafull | iext | ipksent | ipkvalid | icrcerror | — |
| 04 | R | Interrupt Status 2 | iswdet | ipreaval | ipreainval | irssi | iwut | ilbd | ichiprdy | ipor | — |
| 05 | R/W | Interrupt Enable 1 | enfferr | entxffafull | entxffaem | enrxffafull | enext | enpksent | enpkvalid | encrcerror | 00h |
| 06 | R/W | Interrupt Enable 2 | enswdet | enpreava | enpreainval | enrssi | enwut | enlbd | enchiprdy | enpor | 01h |

See “AN440: EZRadioPRO Detailed Register Descriptions” for a complete list of interrupts.

## 23.3. System Timing

The system timing for TX and RX modes is shown in Figures 23.2 and 23.3. The figures demonstrate transitioning from STANDBY mode to TX or RX mode through the built-in sequencer of required steps. The user only needs to program the desired mode, and the internal sequencer will properly transition the part from its current mode.

The VCO will automatically calibrate at every frequency change or power up. The PLL T0 time is to allow for bias settling of the VCO. The PLL TS time is for the settling time of the PLL, which has a default setting of 100 µs. The total time for PLL T0, PLL CAL, and PLL TS under all conditions is 200 µs. Under certain applications, the PLL T0 time and the PLL CAL may be skipped for faster turn-around time. Contact applications support if faster turnaround time is desired.

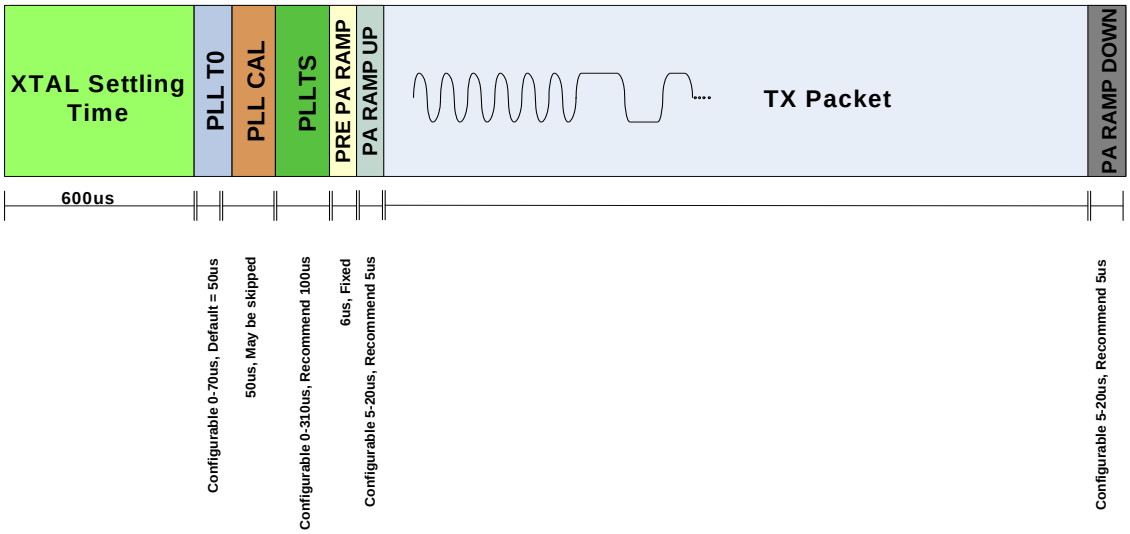



**Figure 23.2. TX Timing**

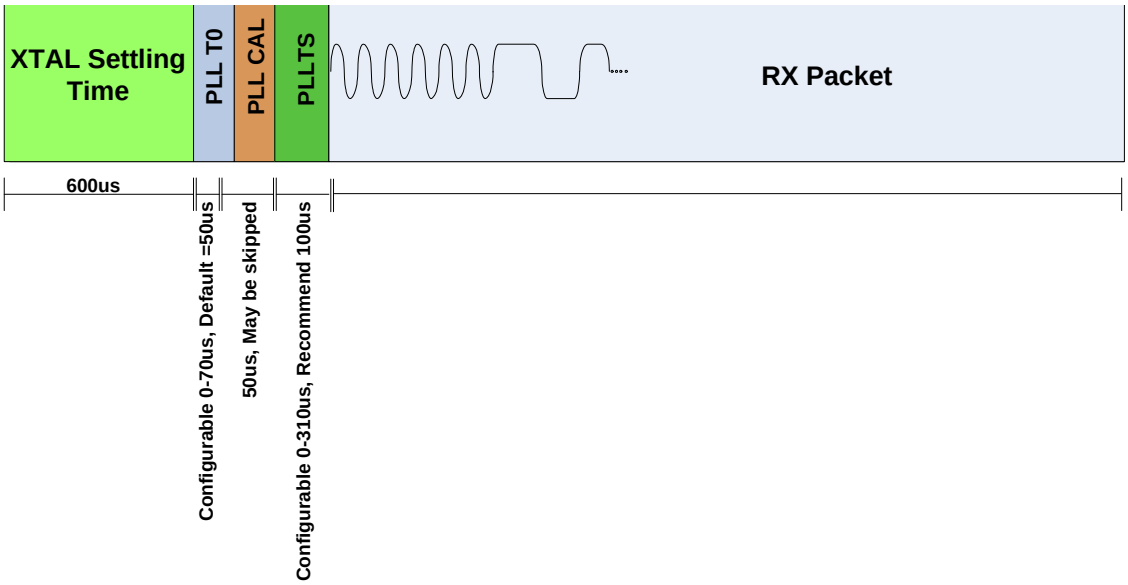



**Figure 23.3. RX Timing**

### 23.3.1. Frequency Control

For calculating the necessary frequency register settings it is recommended that customers use Silicon Labs' Wireless Design Suite (WDS) or the EZRadioPRO Register Calculator worksheet (in Microsoft Excel) available on the product website. These methods offer a simple method to quickly determine the correct settings based on the application requirements. The following information can be used to calculated these values manually.

### 23.3.2. Frequency Programming

In order to receive or transmit an RF signal, the desired channel frequency, $f_{carrier}$, must be programmed into the transceiver. Note that this frequency is the center frequency of the desired channel and not an LO frequency. The carrier frequency is generated by a Fractional-N Synthesizer, using 10 MHz both as the reference frequency and the clock of the ($3^{rd}$ order) ΔΣ modulator. This modulator uses modulo 64000 accumulators. This design was made to obtain the desired frequency resolution of the synthesizer. The overall division ratio of the feedback loop consist of an integer part (N) and a fractional part (F).In a generic sense, the output frequency of the synthesizer is as follows:

$$f_{OUT} = 10MHz \times (N + F)$$

The fractional part (F) is determined by three different values, Carrier Frequency (fc[15:0]), Frequency Offset (fo[8:0]), and Frequency Deviation (fd[7:0]). Due to the fine resolution and high loop bandwidth of the synthesizer, FSK modulation is applied inside the loop and is done by varying F according to the incoming data; this is discussed further in "Frequency Deviation" on page 251. Also, a fixed offset can be added to fine-tune the carrier frequency and counteract crystal tolerance errors. For simplicity assume that only the fc[15:0] register will determine the fractional component. The equation for selection of the carrier frequency is shown below:

$$f_{carrier} = 10MHz \times (hbsel + 1) \times (N + F)$$

$$f_{TX} = 10MHz * (hbsel + 1) * (fb[4:0] + 24 + \frac{fc[15:0]}{64000})$$

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 73 | R/W | Frequency Offset 1 | fo[7] | fo[6] | fo[5] | fo[4] | fo[3] | fo[2] | fo[1] | fo[0] | 00h |
| 74 | R/W | Frequency Offset 2 | | | | | | | fo[9] | fo[8] | 00h |
| 75 | R/W | Frequency Band Select | | sbsel | hbsel | fb[4] | fb[3] | fb[2] | fb[1] | fb[0] | 35h |
| 76 | R/W | Nominal Carrier Frequency 1 | fc[15] | fc[14] | fc[13] | fc[12] | fc[11] | fc[10] | fc[9] | fc[8] | BBh |
| 77 | R/W | Nominal Carrier Frequency 0 | fc[7] | fc[6] | fc[5] | fc[4] | fc[3] | fc[2] | fc[1] | fc[0] | 80h |

The integer part (N) is determined by fb[4:0]. Additionally, the output frequency can be halved by connecting a ÷2 divider to the output. This divider is not inside the loop and is controlled by the hbsel bit in "Register 75h. Frequency Band Select". This effectively partitions the entire 240–960 MHz frequency range into two separate bands: High Band (HB) for hbsel = 1, and Low Band (LB) for hbsel = 0. The valid range of fb[4:0] is from 0 to 23. If a higher value is written into the register, it will default to a value of 23. The integer part has a fixed offset of 24 added to it as shown in the formula above. Table 23.3 demonstrates the selection of fb[4:0] for the corresponding frequency band.

After selection of the fb (N) the fractional component may be solved with the following equation:

$$fc[15:0] = \left( \frac{f_{TX}}{10MHz * (hbsel + 1)} - fb[4:0] - 24 \right) * 64000$$

fb and fc are the actual numbers stored in the corresponding registers.

**Table 23.3. Frequency Band Selection**

| fb[4:0] Value | N | Frequency Band | |
|---|---|---|---|
| | | hbsel=0 | hbsel=1 |
| 0 | 24 | 240–249.9 MHz | 480–499.9 MHz |
| 1 | 25 | 250–259.9 MHz | 500–519.9 MHz |
| 2 | 26 | 260–269.9 MHz | 520–539.9 MHz |
| 3 | 27 | 270–279.9 MHz | 540–559.9 MHz |
| 4 | 28 | 280–289.9 MHz | 560–579.9 MHz |
| 5 | 29 | 290–299.9 MHz | 580–599.9 MHz |
| 6 | 30 | 300–309.9 MHz | 600–619.9 MHz |
| 7 | 31 | 310–319.9 MHz | 620–639.9 MHz |
| 8 | 32 | 320–329.9 MHz | 640–659.9 MHz |

**Table 23.3. Frequency Band Selection (Continued)**

| | | | |
|---|---|---|---|
| 9 | 33 | 330–339.9 MHz | 660–679.9 MHz |
| 10 | 34 | 340–349.9 MHz | 680–699.9 MHz |
| 11 | 35 | 350–359.9 MHz | 700–719.9 MHz |
| 12 | 36 | 360–369.9 MHz | 720–739.9 MHz |
| 13 | 37 | 370–379.9 MHz | 740–759.9 MHz |
| 14 | 38 | 380–389.9 MHz | 760–779.9 MHz |
| 15 | 39 | 390–399.9 MHz | 780–799.9 MHz |
| 16 | 40 | 400–409.9 MHz | 800–819.9 MHz |
| 17 | 41 | 410–419.9 MHz | 820–839.9 MHz |
| 18 | 42 | 420–429.9 MHz | 840–859.9 MHz |
| 19 | 43 | 430–439.9 MHz | 860–879.9 MHz |
| 20 | 44 | 440–449.9 MHz | 880–899.9 MHz |
| 21 | 45 | 450–459.9 MHz | 900–919.9 MHz |
| 22 | 46 | 460–469.9 MHz | 920–939.9 MHz |
| 23 | 47 | 470–479.9 MHz | 940–960 MHz |

The chip will automatically shift the frequency of the Synthesizer down by 937.5 kHz (30 MHz ÷ 32) to achieve the correct Intermediate Frequency (IF) when RX mode is entered. Low-side injection is used in the RX Mixing architecture; therefore, no frequency reprogramming is required when using the same TX frequency and switching between RX/TX modes. For operation in the frequency bands between 240–320 MHz and 480–640 MHz with a temperature above 60 °C, register modifications are required. In these cases, write 03h to register 59h and 02h to register 5Ah.

### 23.3.3. Easy Frequency Programming for FHSS

While Registers 73h–77h may be used to program the carrier frequency of the transceiver, it is often easier to think in terms of “channels” or “channel numbers” rather than an absolute frequency value in Hz. Also, there may be some timing-critical applications (such as for Frequency Hopping Systems) in which it is desirable to change frequency by programming a single register. Once the channel step size is set, the frequency may be changed by a single register corresponding to the channel number. A nominal frequency is first set using Registers 73h–77h, as described above. Registers 79h and 7Ah are then used to set a channel step size and channel number, relative to the nominal setting. The Frequency Hopping Step Size (fhs[7:0]) is set in increments of 10 kHz with a maximum channel step size of 2.56 MHz. The Frequency Hopping Channel Select Register then selects channels based on multiples of the step size.

$$F_{carrier} = Fnom + fhs[7:0] \times (fhch[7:0] \times 10kHz)$$

For example: if the nominal frequency is set to 900 MHz using Registers 73h–77h and the channel step size is set to 1 MHz using "Register 7Ah. Frequency Hopping Step Size". For example, if the "Register 79h. Frequency Hopping Channel Select" is set to 5d, the resulting carrier frequency would be 905 MHz. Once the nominal frequency and channel step size are programmed in the registers, it is only necessary to program the fhch[7:0] register in order to change the frequency.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 79 | R/W | Frequency Hopping Channel Select | fhch[7] | fhch[6] | fhch[5] | fhch[4] | fhch[3] | fhch[2] | fhch[1] | fhch[0] | 00h |
| 7A | R/W | Frequency Hopping Step Size | fhs[7] | fhs[6] | fhs[5] | fhs[4] | fhs[3] | fhs[2] | fhs[1] | fhs[0] | 00h |

### 23.3.4. Automatic State Transition for Frequency Change

If registers 79h or 7Ah are changed in either TX or RX mode, the state machine will automatically transition the chip back to TUNE, change the frequency, and automatically go back to either TX or RX. This feature is useful to reduce the number of SPI commands required in a Frequency Hopping System. This in turn reduces microcontroller activity, reducing current consumption. The exception to this is during TX FIFO mode. If a frequency change is initiated during a TX packet, then the part will complete the current TX packet and will only change the frequency for subsequent packets.

### 23.3.5. Frequency Deviation

The peak frequency deviation is configurable from ±0.625 to ±320 kHz. The Frequency Deviation (Δf) is controlled by the Frequency Deviation Register (fd), address 71 and 72h, and is independent of the carrier frequency setting. When enabled, regardless of the setting of the hbsel bit (high band or low band), the resolution of the frequency deviation will remain in increments of 625 Hz. When using frequency modulation the carrier frequency will deviate from the nominal center channel carrier frequency by ±Δf:

$$\Delta f = fd[8:0] \times 625Hz$$

$$fd[8:0] = \frac{\Delta f}{625Hz} \qquad \Delta\text{f} = \text{peak deviation}$$

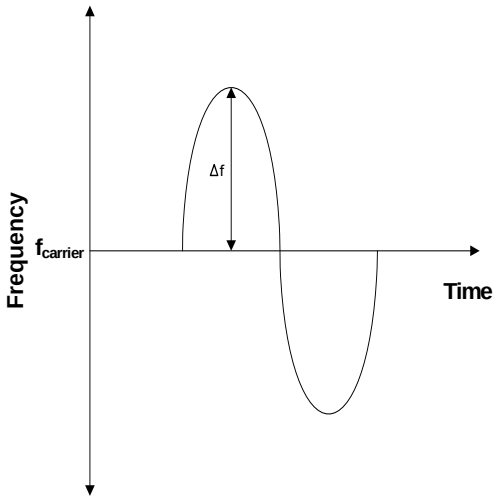



**Figure 23.4. Frequency Deviation**

The previous equation should be used to calculate the desired frequency deviation. If desired, frequency modulation may also be disabled in order to obtain an unmodulated carrier signal at the channel center frequency; see “Modulation Type” on page 254 for further details.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 71 | R/W | Modulation Mode Control 2 | trclk[1] | trclk[0] | dtmod[1] | dtmod[0] | eninv | fd[8] | modtyp[1] | modtyp[0] | 00h |
| 72 | R/W | Frequency Deviation | fd[7] | fd[6] | fd[5] | fd[4] | fd[3] | fd[2] | fd[1] | fd[0] | 20h |

### 23.3.6. Frequency Offset Adjustment

When the AFC is disabled the frequency offset can be adjusted manually by fo[9:0] in registers 73h and 74h. It is not possible to have both AFC and offset as internally they share the same register. The frequency offset adjustment and the AFC both are implemented by shifting the Synthesizer Local Oscillator frequency. This register is a signed register so in order to get a negative offset it is necessary to take the twos complement of the positive offset number. The offset can be calculated by the following:

$$DesiredOffset = 156.25Hz \times (hbsel + 1) \times fo[9:0]$$

$$fo[9:0] = \frac{DesiredOffset}{156.25Hz \times (hbsel + 1)}$$

The adjustment range in high band is ±160 kHz and in low band it is ±80 kHz. For example to compute an offset of +50 kHz in high band mode fo[9:0] should be set to 0A0h. For an offset of –50 kHz in high band mode the fo[9:0] register should be set to 360h.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 73 | R/W | Frequency Offset | fo[7] | fo[6] | fo[5] | fo[4] | fo[3] | fo[2] | fo[1] | fo[0] | 00h |
| 74 | R/W | Frequency Offset | | | | | | | fo[9] | fo[8] | 00h |

### 23.3.7. Automatic Frequency Control (AFC)

All AFC settings can be easily obtained from the settings calculator. This is the recommended method to program all AFC settings. This section is intended to describe the operation of the AFC in more detail to help understand the trade-offs of using AFC.The receiver supports automatic frequency control (AFC) to compensate for frequency differences between the transmitter and receiver reference frequencies. These differences can be caused by the absolute accuracy and temperature dependencies of the reference crystals. Due to frequency offset compensation in the modem, the receiver is tolerant to frequency offsets up to 0.25 times the IF bandwidth when the AFC is disabled. When the AFC is enabled, the received signal will be centered in the pass-band of the IF filter, providing optimal sensitivity and selectivity over a wider range of frequency offsets up to 0.35 times the IF bandwidth. The trade-off of receiver sensitivity (at 1% PER) versus carrier offset and the impact of AFC are illustrated in Figure 23.5.

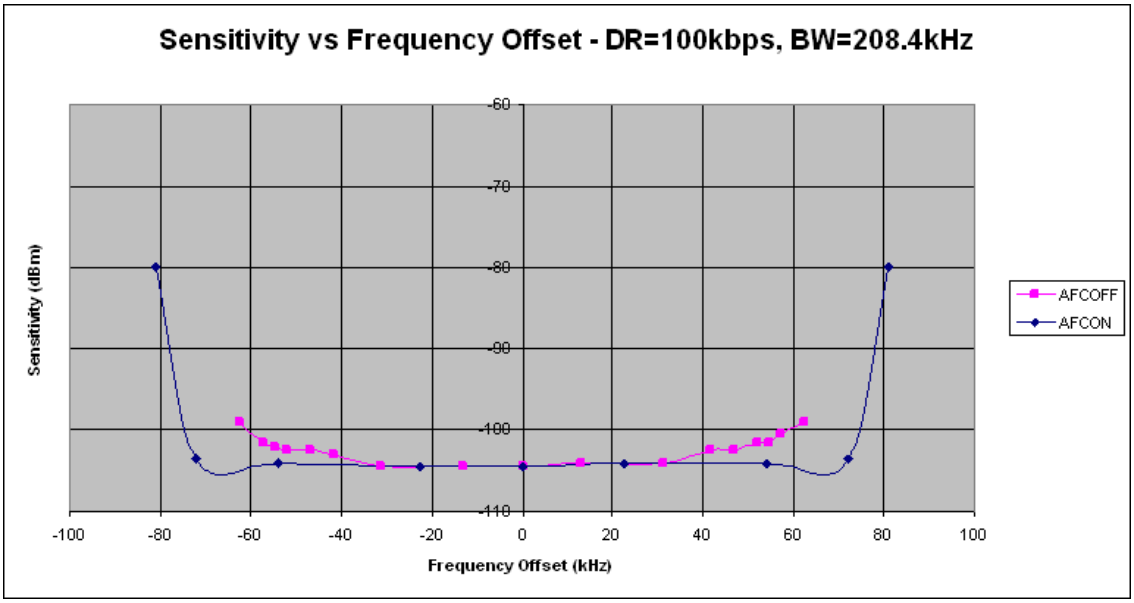



**Figure 23.5. Sensitivity at 1% PER vs. Carrier Frequency Offset**

When AFC is enabled, the preamble length needs to be long enough to settle the AFC. In general, one byte of preamble is sufficient to settle the AFC. Disabling the AFC allows the preamble to be shortened from 40 bits to 32 bits. Note that with the AFC disabled, the preamble length must still be long enough to settle the receiver and to detect the preamble (see “Preamble Length” on page 269). The AFC corrects the detected frequency offset by changing the frequency of the Fractional-N PLL. When the preamble is detected, the AFC will freeze for the remainder of the packet. In multi-packet mode the AFC is reset at the end of every packet and will re-acquire the frequency offset for the next packet. The AFC loop includes a bandwidth limiting mechanism improving the rejection of out of band signals. When the AFC loop is enabled, its pull-in-range is determined by the bandwidth limiter value (AFCLimiter) which is located in register 2Ah.

AFC_pull_in_range = ±AFCLimiter[7:0] x (hbsel+1) x 625 Hz

The AFC Limiter register is an unsigned register and its value can be obtained from the EZRadioPRO Register Calculator spreadsheet.

The amount of error correction feedback to the Fractional-N PLL before the preamble is detected is controlled from afcgearh[2:0]. The default value 000 relates to a feedback of 100% from the measured frequency error and is advised for most applications. Every bit added will half the feedback but will require a longer preamble to settle.

The AFC operates as follows. The frequency error of the incoming signal is measured over a period of two bit times, after which it corrects the local oscillator via the Fractional-N PLL. After this correction, some time is allowed to settle the Fractional-N PLL to the new frequency before the next frequency error is measured. The duration of the AFC cycle before the preamble is detected can be programmed with shwait[2:0]. It is advised to use the default value 001, which sets the AFC cycle to 4 bit times (2 for measurement and 2 for settling). If shwait[2:0] is programmed to 3'b000, there is no AFC correction output. It is advised to use the default value 001, which sets the AFC cycle to 4 bit times (2 for measurement and 2 for settling).

The AFC correction value may be read from register 2Bh. The value read can be converted to kHz with the following formula:

AFC Correction = 156.25Hz x (hbsel +1) x afc_corr[7: 0]

| | Frequency Correction | |
|---|---|---|
| | RX | TX |
| AFC disabled | Freq Offset Register | Freq Offset Register |
| AFC enabled | AFC | Freq Offset Register |

### 23.3.8. TX Data Rate Generator

The data rate is configurable between 0.123–256 kbps. For data rates below 30 kbps the ”txdtrtscale” bit in register 70h should be set to 1. When higher data rates are used this bit should be set to 0**.**

The TX date rate is determined by the following formula in bps:

$$\text{DR_TX (bps)} = \frac{txdr[15:0] \times 1\ \text{MHz}}{2^{16+5\times txdtrtscale}}$$

$$\text{txdr[15:0]} = \frac{\text{DR_TX(bps)} \times 2^{16+5\times txdtrtscale}}{1\ \text{MHz}}$$

For data rates higher than 100 kbps, Register 58h should be changed from its default of 80h to C0h. Non-optimal modulation and increased eye closure will result if this setting is not made for data rates higher than 100 kbps. The txdr register is only applicable to TX mode and does not need to be programmed for RX mode. The RX bandwidth which is partly determined from the data rate is programmed separately.

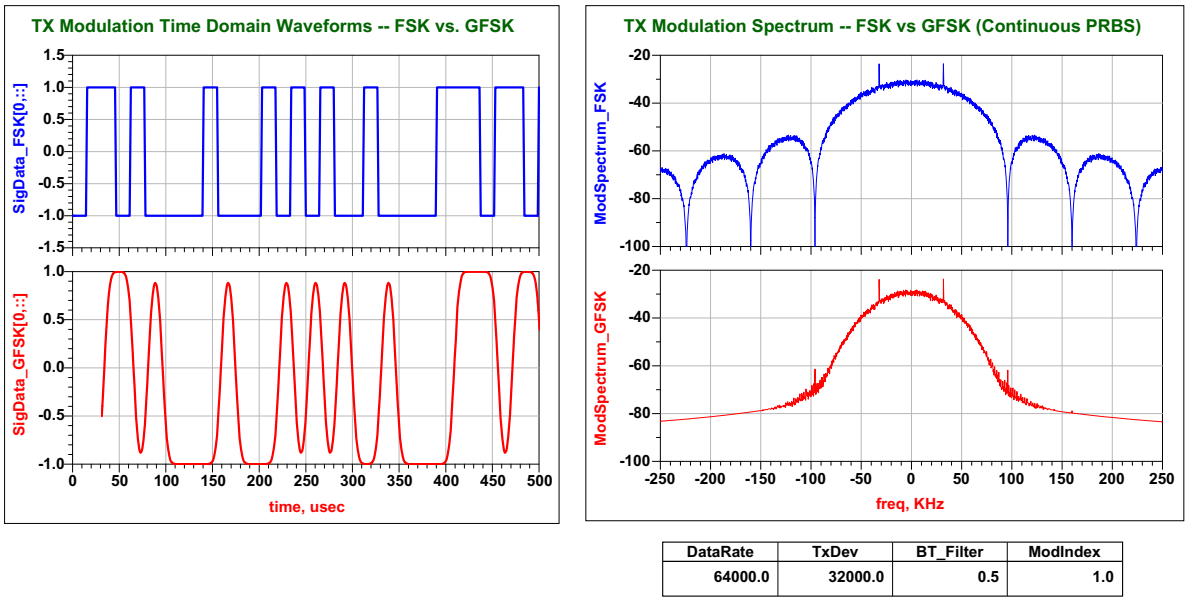
| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 6E | R/W | TX Data Rate 1 | txdr[15] | txdr[14] | txdr[13] | txdr[12] | txdr[11] | txdr[10] | txdr[9] | txdr[8] | 0Ah |
| 6F | R/W | TX Data Rate 0 | txdr[7] | txdr[6] | txdr[5] | txdr[4] | txdr[3] | txdr[2] | txdr[1] | txdr[0] | 3Dh |

## 23.4. Modulation Options

### 23.4.1. Modulation Type

The EZRadioPRO transceivers support three different modulation options: Gaussian Frequency Shift Keying (GFSK), Frequency Shift Keying (FSK), and On-Off Keying (OOK). GFSK is the recommended modulation type as it provides the best performance and cleanest modulation spectrum. Figure 23.6 demonstrates the difference between FSK and GFSK for a Data Rate of 64 kbps. The time domain plots demonstrate the effects of the Gaussian filtering. The frequency domain plots demonstrate the spectral benefit of GFSK over FSK. The type of modulation is selected with the modtyp[1:0] bits in "Register 71h. Modulation Mode Control 2". Note that it is also possible to obtain an unmodulated carrier signal by setting modtyp[1:0] = 00.

| modtyp[1:0] | Modulation Source |
|---|---|
| 00 | Unmodulated Carrier |
| 01 | OOK |
| 10 | FSK |
| 11 | GFSK (enable TX Data CLK when direct mode is used) |




| DataRate | TxDev | BT_Filter | ModIndex |
|---|---|---|---|
| 64000.0 | 32000.0 | 0.5 | 1.0 |

**Figure 23.6. FSK vs. GFSK Spectrums**

### 23.4.2. Modulation Data Source

The transceiver may be configured to obtain its modulation data from one of three different sources: FIFO mode, Direct Mode, and from a PN9 mode. In Direct Mode, the TX modulation data may be obtained from several different input pins. These options are set through the dtmod[1:0] field in "Register 71h. Modulation Mode Control 2".

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 71 | R/W | Modulation Mode Control 2 | trclk[1] | trclk[0] | dtmod[1] | dtmod[0] | eninv | fd[8] | modtyp[1] | modtyp[0] | 00h |

| dtmod[1:0] | Data Source |
|---|---|
| 00 | Direct Mode using TX/RX Data via GPIO pin (GPIO configuration required) |
| 01 | Direct Mode using TX/RX Data via SDI pin (only when nSEL is high) |
| 10 | FIFO Mode |
| 11 | PN9 (internally generated) |

#### 23.4.2.1. FIFO Mode

In FIFO mode, the transmit and receive data is stored in integrated FIFO register memory. The FIFOs are accessed via "Register 7Fh. FIFO Access," and are most efficiently accessed with burst read/write operation.

In TX mode, the data bytes stored in FIFO memory are "packaged" together with other fields and bytes of information to construct the final transmit packet structure. These other potential fields include the Preamble, Sync word, Header, CRC checksum, etc. The configuration of the packet structure in TX mode is

determined by the Automatic Packet Handler (if enabled), in conjunction with a variety of Packet Handler Registers (see Table 23.4 on page 268). If the Automatic Packet Handler is disabled, the entire desired packet structure should be loaded into FIFO memory; no other fields (such as Preamble or Sync word are automatically added to the bytes stored in FIFO memory). For further information on the configuration of the FIFOs for a specific application or packet size, see “Data Handling and Packet Handler” on page 264.

In RX mode, only the bytes of the received packet structure that are considered to be "data bytes" are stored in FIFO memory. Which bytes of the received packet are considered "data bytes" is determined by the Automatic Packet Handler (if enabled), in conjunction with the Packet Handler Registers (see Table 23.4 on page 268). If the Automatic Packet Handler is disabled, all bytes following the Sync word are considered data bytes and are stored in FIFO memory. Thus, even if Automatic Packet Handling operation is not desired, the preamble detection threshold and Sync word still need to be programmed so that the RX Modem knows when to start filling data into the FIFO. When the FIFO is being used in RX mode, all of the received data may still be observed directly (in real-time) by properly programming a GPIO pin as the RXDATA output pin; this can be quite useful during application development.

When in FIFO mode, the chip will automatically exit the TX or RX State when either the ipksent or ipkvalid interrupt occurs. The chip will return to the IDLE mode state programmed in "Register 07h. Operating Mode and Function Control 1". For example, the chip may be placed into TX mode by setting the txon bit, but with the pllon bit additionally set. The chip will transmit all of the contents of the FIFO and the ipksent interrupt will occur. When this interrupt event occurs, the chip will clear the txon bit and return to TUNE mode, as indicated by the set state of the pllon bit. If no other bits are additionally set in register 07h (besides txon initially), then the chip will return to the STANDBY state.

In RX mode, the rxon bit will be cleared if ipkvalid occurs and the rxmpk bit (RX Multi-Packet bit, SPI Register 08h bit [4]) is not set. When the rxmpk bit is set, the part will not exit the RX state after successfully receiving a packet, but will remain in RX mode. The microcontroller will need to decide on the appropriate subsequent action, depending upon information such as an interrupt generated by CRC, packet valid, or preamble detect.

#### 23.4.2.2. Direct Mode

For legacy systems that perform packet handling within an MCU or other baseband chip, it may not be desirable to use the FIFO. For this scenario, a Direct Mode is provided which bypasses the FIFOs entirely.

In TX direct mode, the TX modulation data is applied to an input pin of the chip and processed in "real time" (i.e., not stored in a register for transmission at a later time). A variety of pins may be configured for use as the TX Data input function.

Furthermore, an additional pin may be required for a TX Clock output function if GFSK modulation is desired (only the TX Data input pin is required for FSK). Two options for the source of the TX Data are available in the dtmod[1:0] field, and various configurations for the source of the TX Data Clock may be selected through the trclk[1:0] field.

| trclk[1:0] | TX/RX Data Clock Configuration |
|---|---|
| 00 | No TX Clock (only for FSK) |
| 01 | TX/RX Data Clock is available via GPIO (GPIO needs programming accordingly as well) |
| 10 | TX/RX Data Clock is available via SDO pin (only when nSEL is high) |
| 11 | TX/RX Data Clock is available via the nIRQ pin |

The eninv bit in SPI Register 71h will invert the TX Data; this is most likely useful for diagnostic and testing purposes.

In RX direct mode, the RX Data and RX Clock can be programmed for direct (real-time) output to GPIO pins. The microcontroller may then process the RX data without using the FIFO or packet handler functions

of the RFIC. In RX direct mode, the chip must still acquire bit timing during the Preamble, and thus the preamble detection threshold (SPI Register 35h) must still be programmed. Once the preamble is detected, certain bit timing functions within the RX Modem change their operation for optimized performance over the remainder of the packet. It is not required that a Sync word be present in the packet in RX Direct mode; however, if the Sync word is absent then the skipsyn bit in SPI Register 33h must be set, or else the bit timing and tracking function within the RX Modem will not be configured for optimum performance.

#### 23.4.2.3. Direct Synchronous Mode

In TX direct mode, the chip may be configured for synchronous or asynchronous modes of modulation. In direct synchronous mode, the RFIC is configured to provide a TX Clock signal as an output to the external device that is providing the TX Data stream. This TX Clock signal is a square wave with a frequency equal to the programmed data rate. The external modulation source (e.g., MCU) must accept this TX Clock signal as an input and respond by providing one bit of TX Data back to the RFIC, synchronous with one edge of the TX Clock signal. In this fashion, the rate of the TX Data input stream from the external source is controlled by the programmed data rate of the RFIC; no TX Data bits are made available at the input of the RFIC until requested by another cycle of the TX Clock signal. The TX Data bits supplied by the external source are transmitted directly in real-time (i.e., not stored internally for later transmission).

All modulation types (FSK/GFSK/OOK) are valid in TX direct synchronous mode. As will be discussed in the next section, there are limits on modulation types in TX direct asynchronous mode.

#### 23.4.2.4. Direct Asynchronous Mode

In TX direct asynchronous mode, the RFIC no longer controls the data rate of the TX Data input stream. Instead, the data rate is controlled only by the external TX Data source; the RFIC simply accepts the data applied to its TX Data input pin, at whatever rate it is supplied. This means that there is no longer a need for a TX Clock output signal from the RFIC, as there is no synchronous "handshaking" between the RFIC and the external data source. The TX Data bits supplied by the external source are transmitted directly in real-time (i.e., not stored internally for later transmission).

It is not necessary to program the data rate parameter when operating in TX direct asynchronous mode. The chip still internally samples the incoming TX Data stream to determine when edge transitions occur; however, rather than sampling the data at a pre-programmed data rate, the chip now internally samples the incoming TX Data stream at its maximum possible oversampling rate. This allows the chip to accurately determine the timing of the bit edge transitions without prior knowledge of the data rate. (Of course, it is still necessary to program the desired peak frequency deviation.)

Only FSK and OOK modulation types are valid in TX Direct Asynchronous Mode; GFSK modulation is not available in asynchronous mode. This is because the RFIC does not have knowledge of the supplied data rate, and thus cannot determine the appropriate Gaussian lowpass filter function to apply to the incoming data.

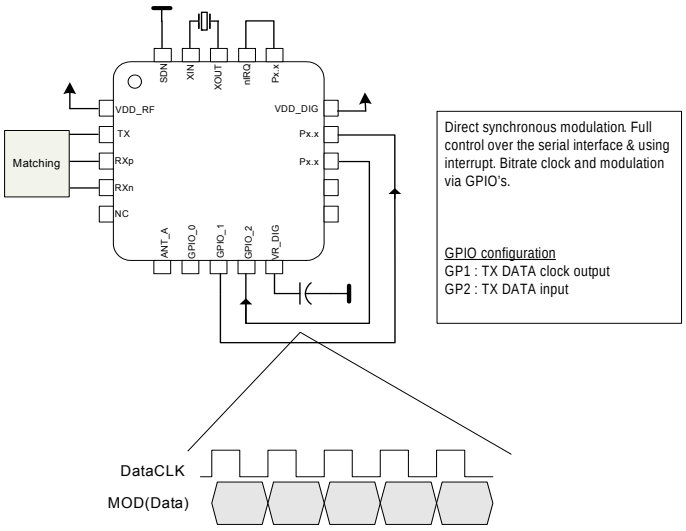



**Figure 23.7. Direct Synchronous Mode Example**

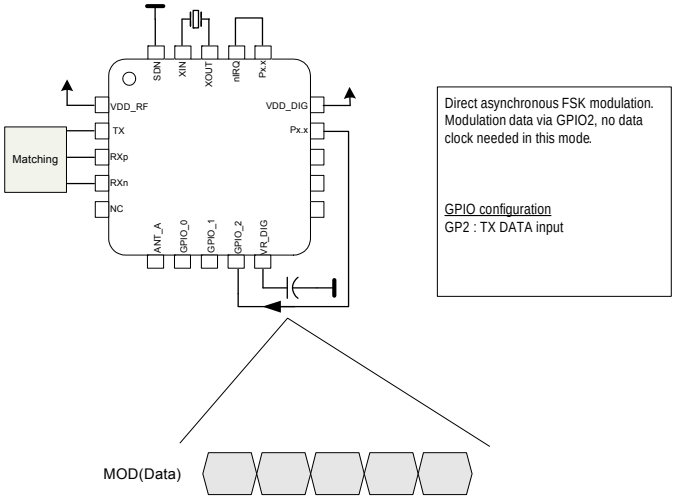



**Figure 23.8. Direct Asynchronous Mode Example**

### 23.4.2.5. Direct Mode using SPI or nIRQ Pins

It is possible to use the EZRadioPRO Serial Interface signals and nIRQ as the modulation clock and data. The MISO signal can be configured to be the data clock by programming trclk = 10. If the NSS signal is LOW then the function of the MISO signal will be SPI data output. If the NSS signal is high and trclk[1:0] is 10 then during RX and TX modes the data clock will be available on the MISO signal. If trclk[1:0] is set to 11 and no interrupts are enabled in registers 05 or 06h, then the nIRQ pin can also be used as the TX/RX data clock.

Note: The MISO and NSS signals are internal connections. The nIRQ signal is accessed through an external package pin.

The MOSI signal can be configured to be the data source in both RX and TX modes if dtmod[1:0] = 01. In a similar fashion, if NSS is LOW the MOSI signal will function as SPI data-in. If NSS is HIGH then in TX

mode it will be the data to be modulated and transmitted. In RX mode it will be the received demodulated data. Figure 23.9 demonstrates using MOSI and MISO as the TX/RX data and clock:

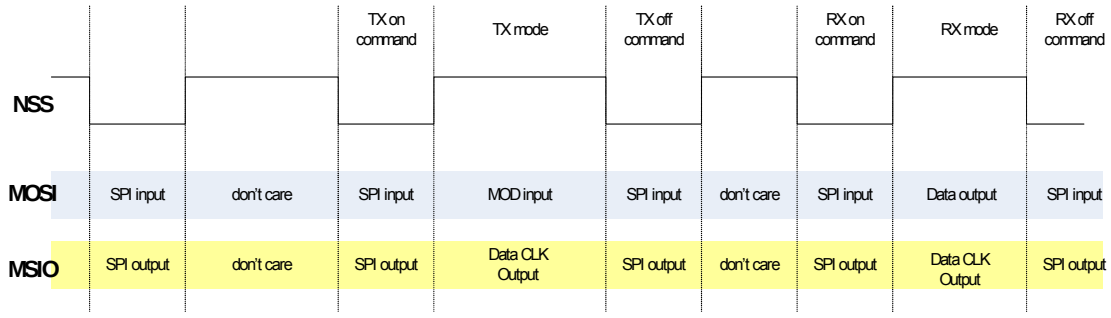



**Figure 23.9. Microcontroller Connections**

If the MISO pin is not used for data clock then it may be programmed to be the interrupt function (nIRQ) by programming Reg 0Eh bit 3.

### 23.4.3. PN9 Mode

In this mode the TX Data is generated internally using a pseudorandom (PN9 sequence) bit generator. The primary purpose of this mode is for use as a test mode to observe the modulated spectrum without having to provide data.

## 23.5. Internal Functional Blocks

This section provides an overview some of the key blocks of the internal radio architecture.

### 23.5.1. RX LNA

Depending on the part, the input frequency range for the LNA is between 240–960 MHz. The LNA provides gain with a noise figure low enough to suppress the noise of the following stages. The LNA has one step of gain control which is controlled by the analog gain control (AGC) algorithm. The AGC algorithm adjusts the gain of the LNA and PGA so the receiver can handle signal levels from sensitivity to +5 dBm with optimal performance.

In the Si1002/3, the TX and RX may be tied directly. See the TX/RX direct-tie reference design available on www.silabs.com. When the direct tie is used the lna_sw bit in Register 6Dh, TX Power must be set.

### 23.5.2. RX I-Q Mixer

The output of the LNA is fed internally to the input of the receive mixer. The receive mixer is implemented as an I-Q mixer that provides both I and Q channel outputs to the programmable gain amplifier. The mixer consists of two double-balanced mixers whose RF inputs are driven in parallel, local oscillator (LO) inputs are driven in quadrature, and separate I and Q Intermediate Frequency (IF) outputs drive the programmable gain amplifier. The receive LO signal is supplied by an integrated VCO and PLL synthesizer operating between 240–960 MHz. The necessary quadrature LO signals are derived from the divider at the VCO output.

### 23.5.3. Programmable Gain Amplifier

The programmable gain amplifier (PGA) provides the necessary gain to boost the signal level into the dynamic range of the ADC. The PGA must also have enough gain switching to allow for large input signals to ensure a linear RSSI range up to –20 dBm. The PGA has steps of 3 dB which are controlled by the AGC algorithm in the digital modem.

### 23.5.4. ADC

The amplified IQ IF signals are digitized using an Analog-to-Digital Converter (ADC), which allows for low current consumption and high dynamic range. The bandpass response of the ADC provides exceptional rejection of out of band blockers.

### 23.5.5. Digital Modem

Using high-performance ADCs allows channel filtering, image rejection, and demodulation to be performed in the digital domain, resulting in reduced area while increasing flexibility. The digital modem performs the following functions:

- Channel selection filter
- TX modulation
- RX demodulation
- AGC
- Preamble detector
- Invalid preamble detector
- Radio signal strength indicator (RSSI)
- Automatic frequency compensation (AFC)
- Packet handling including EZMAC® features
- Cyclic redundancy check (CRC)

The digital channel filter and demodulator are optimized for ultra low power consumption and are highly configurable. Supported modulation types are GFSK, FSK, and OOK. The channel filter can be configured to support bandwidths ranging from 620 kHz down to 2.6 kHz. A large variety of data rates are supported ranging from 0.123 up to 256 kbps. The AGC algorithm is implemented digitally using an advanced control loop optimized for fast response time.

The configurable preamble detector is used to improve the reliability of the sync-word detection. The sync-word detector is only enabled when a valid preamble is detected, significantly reducing the probability of false detection.

The received signal strength indicator (RSSI) provides a measure of the signal strength received on the tuned channel. The resolution of the RSSI is 0.5 dB. This high resolution RSSI enables accurate channel power measurements for clear channel assessment (CCA), carrier sense (CS), and listen before talk (LBT) functionality.

Frequency mistuning caused by crystal inaccuracies can be compensated by enabling the digital automatic frequency control (AFC) in receive mode.

A comprehensive programmable packet handler including key features of Silicon Labs' EZMAC is integrated to create a variety of communication topologies ranging from peer-to-peer networks to mesh networks. The extensive programmability of the packet header allows for advanced packet filtering which in turn enables a mix of broadcast, group, and point-to-point communication.

A wireless communication channel can be corrupted by noise and interference, and it is therefore important to know if the received data is free of errors. A cyclic redundancy check (CRC) is used to detect the presence of erroneous bits in each packet. A CRC is computed and appended at the end of each transmitted packet and verified by the receiver to confirm that no errors have occurred. The packet handler and CRC can significantly reduce the load on the microcontroller reducing the overall current consumption.

The digital modem includes the TX modulator which converts the TX data bits into the corresponding stream of digital modulation values to be summed with the fractional input to the sigma-delta modulator. This modulation approach results in highly accurate resolution of the frequency deviation. A Gaussian filter is implemented to support GFSK, considerably reducing the energy in the adjacent channels. The default bandwidth-time product (BT) is 0.5 for all programmed data rates, but it may be adjusted to other values.

### 23.5.6. Synthesizer

An integrated Sigma Delta (ΣΔ) Fractional-N PLL synthesizer capable of operating from 240–960 MHz is provided on-chip. Using a ΣΔ synthesizer has many advantages; it provides flexibility in choosing data rate, deviation, channel frequency, and channel spacing. The transmit modulation is applied directly to the loop in the digital domain through the fractional divider which results in very precise accuracy and control over the transmit deviation.

Depending on the part, the PLL and Δ-Σ modulator scheme is designed to support any desired frequency and channel spacing in the range from 240–960 MHz with a frequency resolution of 156.25 Hz (Low band) or 312.5 Hz (High band). The transmit data rate can be programmed between 0.123–256 kbps, and the frequency deviation can be programmed between ±1–320 kHz. These parameters may be adjusted via registers as shown in “Frequency Control” on page 248.

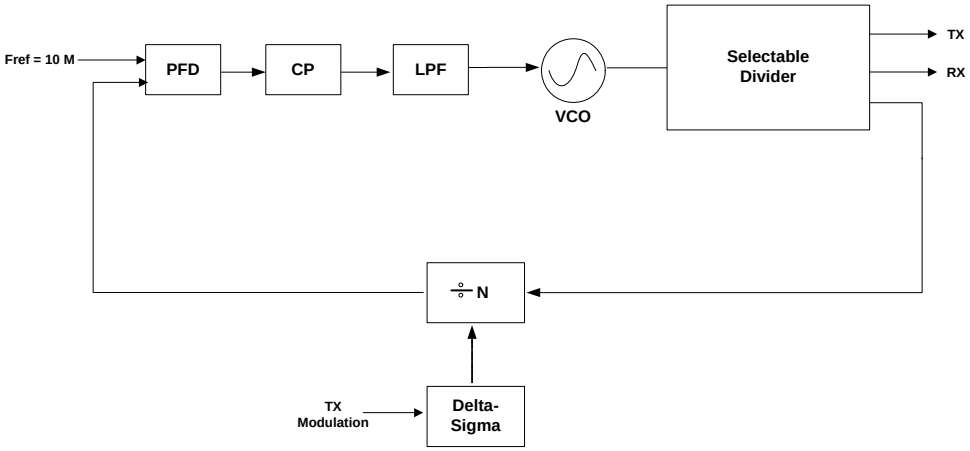



**Figure 23.10. PLL Synthesizer Block Diagram**

The reference frequency to the PLL is 10 MHz. The PLL utilizes a differential L-C VCO, with integrated on-chip inductors. The output of the VCO is followed by a configurable divider which will divide down the signal to the desired output frequency band. The modulus of the variable divide-by-N divider stage is controlled dynamically by the output from the Δ-Σ modulator. The tuning resolution is sufficient to tune to the commanded frequency with a maximum accuracy of 312.5 Hz anywhere in the range between 240–960 MHz.

#### 23.5.6.1. VCO

The output of the VCO is automatically divided down to the correct output frequency depending on the hbsel and fb[4:0] fields in "Register 75h. Frequency Band Select". In receive mode, the LO frequency is automatically shifted downwards by the IF frequency of 937.5 kHz, allowing transmit and receive operation on the same frequency. The VCO integrates the resonator inductor and tuning varactor, so no external VCO components are required.

The VCO uses a capacitance bank to cover the wide frequency range specified. The capacitance bank will automatically be calibrated every time the synthesizer is enabled. In certain fast hopping applications this might not be desirable so the VCO calibration may be skipped by setting the appropriate register.

### 23.5.7. Power Amplifier

The Si1000/1 contains an internal integrated power amplifier (PA) capable of transmitting at output levels between +1 and +20 dBm. The Si1002/3/4/5 contains a PA which is capable of transmitting output levels between –8 to +13 dBm. The PA design is single-ended and is implemented as a two stage class CE amplifier with a high efficiency when transmitting at maximum power. The PA efficiency can only be optimized at one power level. Changing the output power by adjusting txpow[2:0] will scale both the output power and current but the efficiency will not remain constant. The PA output is ramped up and down to prevent unwanted spectral splatter.

In the Si1002/3 the TX and RX may be tied directly. See the TX/RX direct-tie reference design available on the Silicon Labs website for more details. When the direct tie is used the lna_sw bit in Register 6Dh, TX Power must be set to 1.

#### 23.5.7.1. Output Power Selection

The output power is configurable in 3 dB steps with the txpow[2:0] field in "Register 6Dh. TX Power". Extra output power can allow the use of a cheaper smaller antenna, greatly reducing the overall BOM cost. The higher power setting of the chip achieves maximum possible range, but of course comes at the cost of higher TX current consumption. However, depending on the duty cycle of the system, the effect on battery life may be insignificant. Contact Silicon Labs Support for help in evaluating this tradeoff.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 6D | R/W | TX Power | reserved | reserved | reserved | reserved | lna_sw | txpow[2] | txpow[1] | txpow[0] | 18h |

| txpow[2:0] | Si10x0/1 Output Power |
|---|---|
| 000 | +1 dBm |
| 001 | +2 dBm |
| 010 | +5 dBm |
| 011 | +8 dBm |
| 100 | +11 dBm |
| 101 | +14 dBm |
| 110 | +17 dBm |
| 111 | +20 dBm |

| txpow[2:0] | Si10x2/3/4/5 Output Power |
|---|---|
| 000 | –8 dBm |
| 001 | –5 dBm |
| 010 | –2 dBm |
| 011 | +1 dBm |
| 100 | +4 dBm |
| 101 | +7 dBm |
| 110 | +10 dBm |
| 111 | +13 dBm |

### 23.5.8. Crystal Oscillator

The transceiver includes an integrated 30 MHz crystal oscillator with a fast start-up time of less than 600 µs when a suitable parallel resonant crystal is used. The design is differential with the required crystal load capacitance integrated on-chip to minimize the number of external components. By default, all that is required off-chip is the 30 MHz crystal.

The crystal load capacitance can be digitally programmed to accommodate crystals with various load capacitance requirements and to adjust the frequency of the crystal oscillator. The tuning of the crystal load capacitance is programmed through the xlc[6:0] field of "Register 09h. 30 MHz Crystal Oscillator Load Capacitance". The total internal capacitance is 12.5 pF and is adjustable in approximately 127 steps (97fF/step). The xtalshift bit provides a coarse shift in frequency but is not binary with xlc[6:0].

The crystal frequency adjustment can be used to compensate for crystal production tolerances. Utilizing the on-chip temperature sensor and suitable control software, the temperature dependency of the crystal can be canceled.

The typical value of the total on-chip capacitance Cint can be calculated as follows:

Cint = 1.8 pF + 0.085 pF x xlc[6:0] + 3.7 pF x xtalshift

Note that the coarse shift bit xtalshift is not binary with xlc[6:0]. The total load capacitance Cload seen by the crystal can be calculated by adding the sum of all external parasitic PCB capacitances Cext to Cint. If the maximum value of Cint (16.3 pF) is not sufficient, an external capacitor can be added for exact tuning. Additional information on calculating Cext and crystal selection guidelines is provided in “AN417: Si4x3x Family Crystal Oscillator.”

If AFC is disabled then the synthesizer frequency may be further adjusted by programming the Frequency Offset field fo[9:0]in "Register 73h. Frequency Offset 1" and "Register 74h. Frequency Offset 2", as discussed in “Frequency Control” on page 248.

The crystal oscillator frequency is divided down internally and may be output to the microcontroller through one of the GPIO pins for use as the System Clock. In this fashion, only one crystal oscillator is required for the entire system and the BOM cost is reduced. The available clock frequencies and GPIO configuration are discussed further in “Output Clock” on page 273.

The transceiver may also be driven with an external 30 MHz clock signal through the XOUT pin. When driving with an external reference or using a TCXO, the XTAL load capacitance register should be set to 0.

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 09 | R/W | Crystal Oscillator Load Capacitance | xtalshift | xlc[6] | xlc[5] | xlc[4] | xlc[3] | xlc[2] | xlc[1] | xlc[0] | 7Fh |

### 23.5.9. Regulators

There are a total of six regulators integrated onto the transceiver. With the exception of the digital regulator, all regulators are designed to operate with only internal decoupling. The digital regulator requires an external 1 µF decoupling capacitor. All regulators are designed to operate with an input supply voltage from +1.8 to +3.6 V. The output stage of the of PA is not connected internally to a regulator and is connected directly to the battery voltage.

A supply voltage should only be connected to the VDD pins. No voltage should be forced on the digital regulator output.

## 23.6. Data Handling and Packet Handler

The internal modem is designed to operate with a packet including a 010101... preamble structure. To configure the modem to operate with packet formats without a preamble or other legacy packet structures contact customer support.

### 23.6.1. RX and TX FIFOs

Two 64 byte FIFOs are integrated into the chip, one for RX and one for TX, as shown in Figure 23.11. "Register 7Fh. FIFO Access" is used to access both FIFOs. A burst write to address 7Fh will write data to the TX FIFO. A burst read from address 7Fh will read data from the RX FIFO.

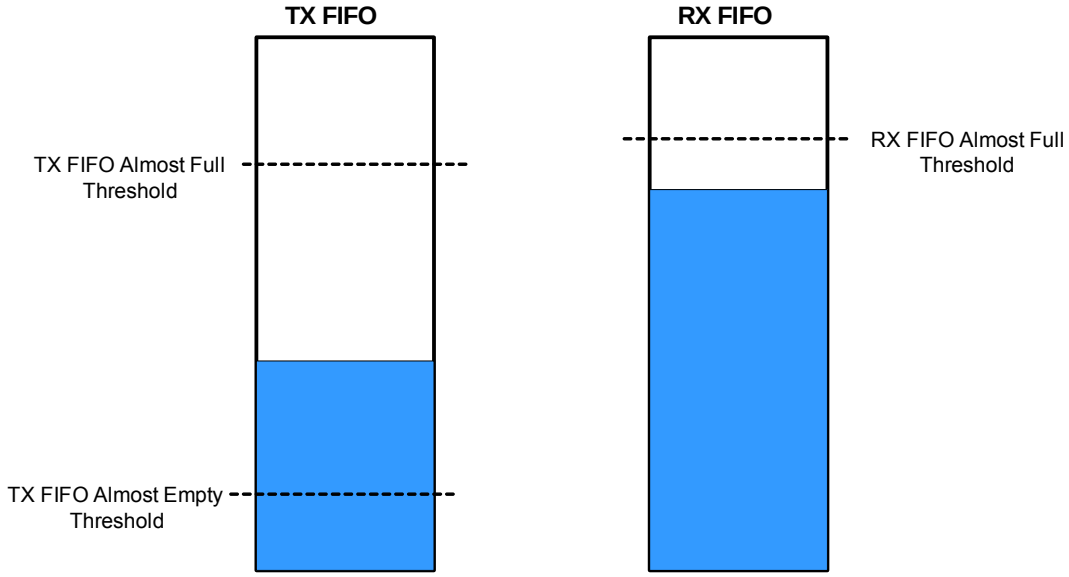



**Figure 23.11. FIFO Thresholds**

The TX FIFO has two programmable thresholds. An interrupt event occurs when the data in the TX FIFO reaches these thresholds. The first threshold is the FIFO almost full threshold, txafthr[5:0]. The value in this register corresponds to the desired threshold value in number of bytes. When the data being filled into the TX FIFO crosses this threshold limit, an interrupt to the microcontroller is generated so the chip can enter TX mode to transmit the contents of the TX FIFO. The second threshold for TX is the FIFO almost empty threshold, txaethr[5:0]. When the data being shifted out of the TX FIFO drops below the almost empty threshold an interrupt will be generated. If more data is not loaded into the FIFO then the chip automatically exits the TX State after the ipksent interrupt occurs. The chip will return to the mode selected by the remaining bits in SPI Register 07h. For example, the chip may be placed into TX mode by setting the txon bit, but with the xton bit additionally set. For this condition, the chip will transmit all of the contents of the FIFO and the ipksent interrupt will occur. When this interrupt event occurs, the chip will clear the txon bit and return to READY mode, as indicated by the set state of the xton bit. If the pllon bit D1 is set when entering TX mode (i.e., SPI Register 07h = 0Ah), the chip will exit from TX mode after sending the packet and return to TUNE mode.

However, the chip will not automatically return to STANDBY mode upon exit from the TX state, in the event the TX packet is initiated by setting SPI Register 07h = 08h (i.e., setting only txon bit D3). The chip will instead return to READY mode, with the crystal oscillator remaining enabled. This is intentional; the system may be configured such that the host MCU derives its clock from the MCU_CLK output of the RFIC (through GPIO2), and this clock signal must not be shut down without allowing the host MCU time to process any interrupt signals that may have occurred. The host MCU must subsequently perform a WRITE to SPI Register 07h = 00h to enter STANDBY mode and obtain minimum current consumption.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 08 | R/W | Operating & Function Control 2 | antdiv[2] | antdiv[1] | antdiv[0] | rxmpk | autotx | enldm | ffclrrx | ffclrtx | 00h |
| 7C | R/W | TX FIFO Control 1 | Reserved | Reserved | txafthr[5] | txafthr[4] | txafthr[3] | txafthr[2] | txafthr[1] | txafthr[0] | 37h |
| 7D | R/W | TX FIFO Control 2 | Reserved | Reserved | txaethr[5] | txaethr[4] | txaethr[3] | txaethr[2] | txaethr[1] | txaethr[0] | 04h |

The RX FIFO has one programmable threshold called the FIFO Almost Full Threshold, rxafthr[5:0]. When the incoming RX data crosses the Almost Full Threshold an interrupt will be generated to the microcontroller via the nIRQ pin. The microcontroller will then need to read the data from the RX FIFO.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7E | R/W | RX FIFO Control | Reserved | Reserved | rxafthr[5] | rxafthr[4] | rxafthr[3] | rxafthr[2] | rxafthr[1] | rxafthr[0] | 37h |

Both the TX and RX FIFOs may be cleared or reset with the ffclrtx and ffclrrx bits. All interrupts may be enabled by setting the Interrupt Enabled bits in "Register 05h. Interrupt Enable 1" and “Register 06h. Interrupt Enable 2.” If the interrupts are not enabled the function will not generate an interrupt on the nIRQ pin but the bits will still be read correctly in the Interrupt Status registers.

### 23.6.2. Packet Configuration

When using the FIFOs, automatic packet handling may be enabled for TX mode, RX mode, or both. "Register 30h. Data Access Control" through “Register 4Bh. Received Packet Length” control the configuration, status, and decoded RX packet data for Packet Handling. The usual fields for network communication (such as preamble, synchronization word, headers, packet length, and CRC) can be configured to be automatically added to the data payload. The fields needed for packet generation normally change infrequently and can therefore be stored in registers. Automatically adding these fields to the data payload greatly reduces the amount of communication between the microcontroller and the transceiver.

The general packet structure is shown in Figure 23.12. The length of each field is shown below the field. The preamble pattern is always a series of alternating ones and zeros, starting with a zero. All the fields have programmable lengths to accommodate different applications. The most common CRC polynominals are available for selection.

**Figure 23.12. Packet Structure**

An overview of the packet handler configuration registers is shown in Table 23.4.

### 23.6.3. Packet Handler TX Mode

If the TX packet length is set the packet handler will send the number of bytes in the packet length field before returning to IDLE mode and asserting the packet sent interrupt. To resume sending data from the FIFO the microcontroller needs to command the chip to re-enter TX mode. Figure 23.13 provides an example transaction where the packet length is set to three bytes.

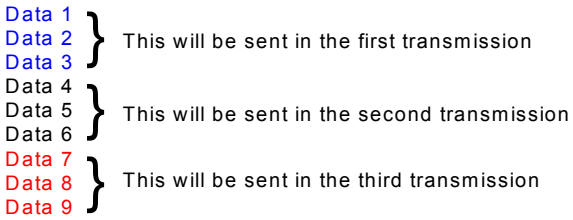



**Figure 23.13. Multiple Packets in TX Packet Handler**

### 23.6.4. Packet Handler RX Mode

#### 23.6.4.1. Packet Handler Disabled

When the packet handler is disabled certain fields in the received packet are still required. Proper modem operation requires preamble and sync when the FIFO is being used, as shown in Figure 23.14. Bits after sync will be treated as raw data with no qualification. This mode allows for the creation of a custom packet handler when the automatic qualification parameters are not sufficient. Manchester encoding is supported but data whitening, CRC, and header checks are not.

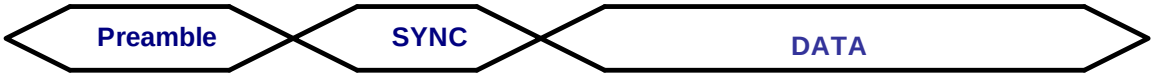



**Figure 23.14. Required RX Packet Structure with Packet Handler Disabled**

#### 23.6.4.2. Packet Handler Enabled

When the packet handler is enabled, all the fields of the packet structure need to be configured. Register contents are used to construct the header field and length information encoded into the transmitted packet when transmitting. The receive FIFO can be configured to handle packets of fixed or variable length with or without a header. If multiple packets are desired to be stored in the FIFO, then there are options available for the different fields that will be stored into the FIFO. Figure 23.15 demonstrates the options and settings available when multiple packets are enabled. Figure 23.16 demonstrates the operation of fixed packet length and correct/incorrect packets.

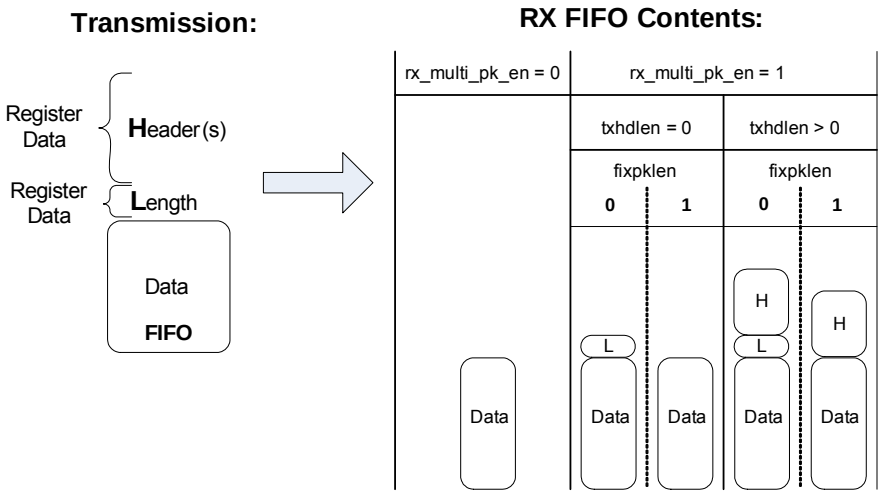



**Figure 23.15. Multiple Packets in RX Packet Handler**

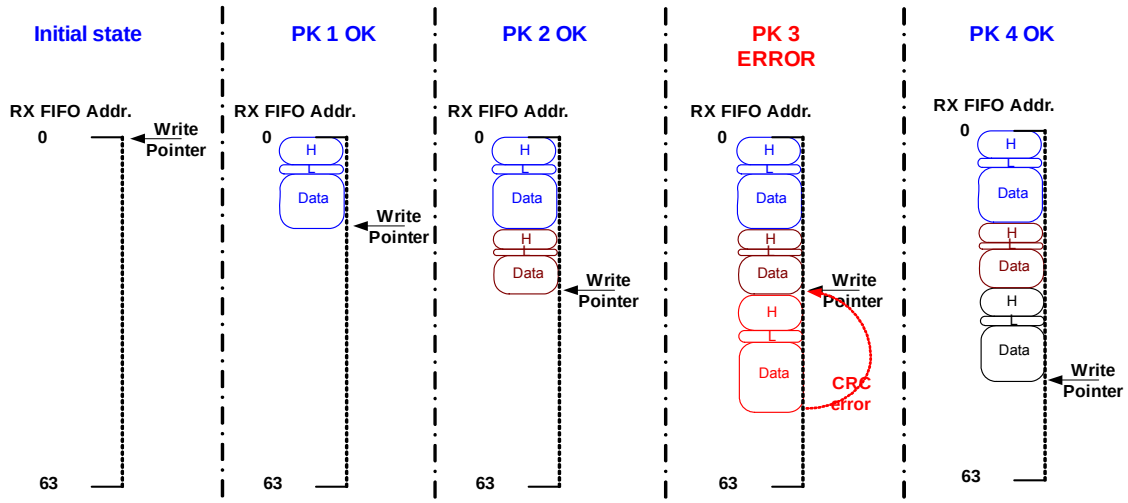



**Figure 23.16. Multiple Packets in RX with CRC or Header Error**

**Table 23.4. Packet Handler Registers**

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | R/W | Data Access Control | enpacrx | lsbfrst | crcdonly | skip2ph | enpactx | encrc | crc[1] | crc[0] | 8Dh |
| 31 | R | EzMAC status | 0 | rxcrc1 | pksrch | pkrx | pkvalid | crcerror | pktx | pksent | — |
| 32 | R/W | Header Control 1 | bcen[3:0] | | | | hdch[3:0] | | | | 0Ch |
| 33 | R/W | Header Control 2 | skipsyn | hdlen[2] | hdlen[1] | hdlen[0] | fixpklen | synclen[1] | synclen[0] | prealen[8] | 22h |
| 34 | R/W | Preamble Length | prealen[7] | prealen[6] | prealen[5] | prealen[4] | prealen[3] | prealen[2] | prealen[1] | prealen[0] | 08h |
| 35 | R/W | Preamble Detection Control | preath[4] | preath[3] | preath[2] | preath[1] | preath[0] | rssi_off[2] | rssi_off[1] | rssi_off[0] | 2Ah |
| 36 | R/W | Sync Word 3 | sync[31] | sync[30] | sync[29] | sync[28] | sync[27] | sync[26] | sync[25] | sync[24] | 2Dh |
| 37 | R/W | Sync Word 2 | sync[23] | sync[22] | sync[21] | sync[20] | sync[19] | sync[18] | sync[17] | sync[16] | D4h |
| 38 | R/W | Sync Word 1 | sync[15] | sync[14] | sync[13] | sync[12] | sync[11] | sync[10] | sync[9] | sync[8] | 00h |
| 39 | R/W | Sync Word 0 | sync[7] | sync[6] | sync[5] | sync[4] | sync[3] | sync[2] | sync[1] | sync[0] | 00h |
| 3A | R/W | Transmit Header 3 | txhd[31] | txhd[30] | txhd[29] | txhd[28] | txhd[27] | txhd[26] | txhd[25] | txhd[24] | 00h |
| 3B | R/W | Transmit Header 2 | txhd[23] | txhd[22] | txhd[21] | txhd[20] | txhd[19] | txhd[18] | txhd[17] | txhd[16] | 00h |
| 3C | R/W | Transmit Header 1 | txhd[15] | txhd[14] | txhd[13] | txhd[12] | txhd[11] | txhd[10] | txhd[9] | txhd[8] | 00h |
| 3D | R/W | Transmit Header 0 | txhd[7] | txhd[6] | txhd[5] | txhd[4] | txhd[3] | txhd[2] | txhd[1] | txhd[0] | 00h |
| 3E | R/W | Transmit Packet Length | pklen[7] | pklen[6] | pklen[5] | pklen[4] | pklen[3] | pklen[2] | pklen[1] | pklen[0] | 00h |
| 3F | R/W | Check Header 3 | chhd[31] | chhd[30] | chhd[29] | chhd[28] | chhd[27] | chhd[26] | chhd[25] | chhd[24] | 00h |
| 40 | R/W | Check Header 2 | chhd[23] | chhd[22] | chhd[21] | chhd[20] | chhd[19] | chhd[18] | chhd[17] | chhd[16] | 00h |
| 41 | R/W | Check Header 1 | chhd[15] | chhd[14] | chhd[13] | chhd[12] | chhd[11] | chhd[10] | chhd[9] | chhd[8] | 00h |
| 42 | R/W | Check Header 0 | chhd[7] | chhd[6] | chhd[5] | chhd[4] | chhd[3] | chhd[2] | chhd[1] | chhd[0] | 00h |
| 43 | R/W | Header Enable 3 | hden[31] | hden[30] | hden[29] | hden[28] | hden[27] | hden[26] | hden[25] | hden[24] | FFh |
| 44 | R/W | Header Enable 2 | hden[23] | hden[22] | hden[21] | hden[20] | hden[19] | hden[18] | hden[17] | hden[16] | FFh |
| 45 | R/W | Header Enable 1 | hden[15] | hden[14] | hden[13] | hden[12] | hden[11] | hden[10] | hden[9] | hden[8] | FFh |
| 46 | R/W | Header Enable 0 | hden[7] | hden[6] | hden[5] | hden[4] | hden[3] | hden[2] | hden[1] | hden[0] | FFh |
| 47 | R | Received Header 3 | rxhd[31] | rxhd[30] | rxhd[29] | rxhd[28] | rxhd[27] | rxhd[26] | rxhd[25] | rxhd[24] | — |
| 48 | R | Received Header 2 | rxhd[23] | rxhd[22] | rxhd[21] | rxhd[20] | rxhd[19] | rxhd[18] | rxhd[17] | rxhd[16] | — |
| 49 | R | Received Header 1 | rxhd[15] | rxhd[14] | rxhd[13] | rxhd[12] | rxhd[11] | rxhd[10] | rxhd[9] | rxhd[8] | — |
| 4A | R | Received Header 0 | rxhd[7] | rxhd[6] | rxhd[5] | rxhd[4] | rxhd[3] | rxhd[2] | rxhd[1] | rxhd[0] | — |
| 4B | R | Received Packet Length | rxplen[7] | rxplen[6] | rxplen[5] | rxplen[4] | rxplen[3] | rxplen[2] | rxplen[1] | rxplen[0] | — |

### 23.6.5. Data Whitening, Manchester Encoding, and CRC

Data whitening can be used to avoid extended sequences of 0s or 1s in the transmitted data stream to achieve a more uniform spectrum. When enabled, the payload data bits are XORed with a pseudorandom sequence output from the built-in PN9 generator. The generator is initialized at the beginning of the payload. The receiver recovers the original data by repeating this operation. Manchester encoding can be used to ensure a dc-free transmission and good synchronization properties. When Manchester encoding is used, the effective datarate is unchanged but the actual datarate (preamble length, etc.) is doubled due to the nature of the encoding. The effective datarate when using Manchester encoding is limited to 128 kbps. The implementation of Manchester encoding is shown in Figure 23.18. Data whitening and Manchester encoding can be selected with "Register 70h. Modulation Mode Control 1". The CRC is configured via "Register 30h. Data Access Control". Figure 23.17 demonstrates the portions of the packet which have Manchester encoding, data whitening, and CRC applied. CRC can be applied to only the data portion of the packet or to the data, packet length and header fields. Figure 23.18 provides an example of how the Manchester encoding is done and also the use of the Manchester invert (enmaniv) function.

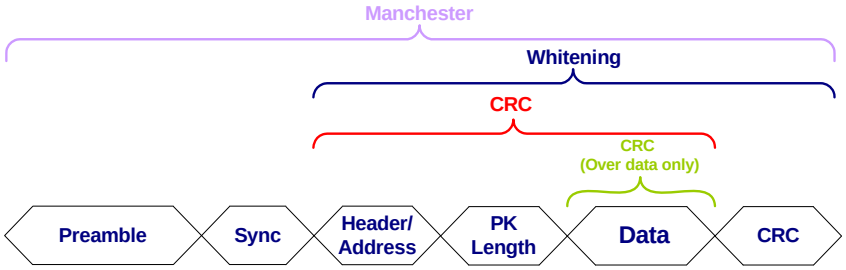



**Figure 23.17. Operation of Data Whitening, Manchester Encoding, and CRC**

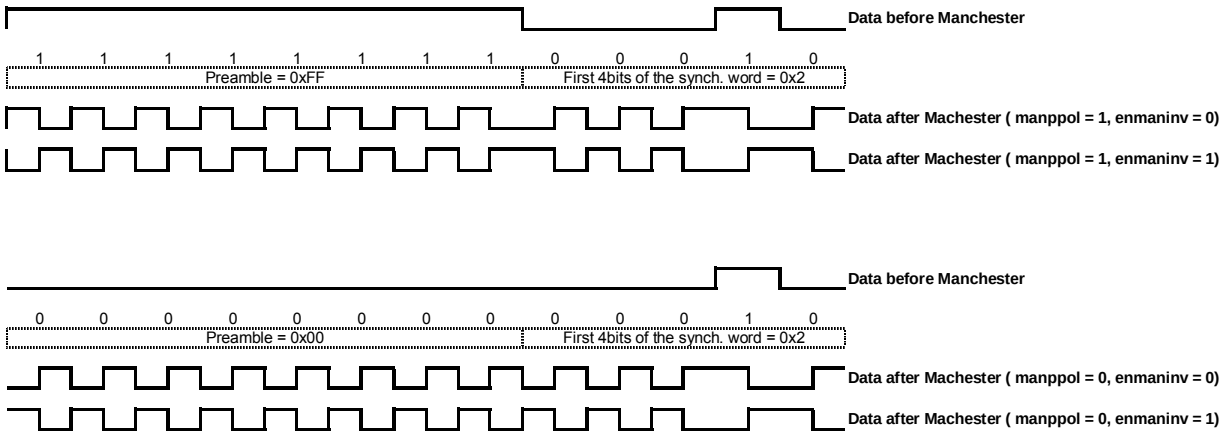



**Figure 23.18. Manchester Coding Example**

### 23.6.6. Preamble Detector

The EZRadioPRO transceiver has integrated automatic preamble detection. The preamble length is configurable from 1–255 bytes using the prealen[7:0] field in "Register 33h. Header Control 2" and "Register 34h. Preamble Length", as described in “23.6.2. Packet Configuration” . The preamble detection threshold, preath[4:0] as set in "Register 35h. Preamble Detection Control 1", is in units of 4 bits. The preamble detector searches for a preamble pattern with a length of preath[4:0].

If a false preamble detect occurs, the receiver will continuing searching for the preamble when no sync word is detected. Once preamble is detected (false or real) then the part will then start searching for sync. If no sync occurs then a timeout will occur and the device will initiate search for preamble again. The timeout period is defined as the sync word length plus four bits and will start after a non-preamble pattern is recognized after a valid preamble detection. The preamble detector output may be programmed onto one of the GPIO or read in the interrupt status registers.

### 23.6.7. Preamble Length

The preamble detection threshold determines the number of valid preamble bits the radio must receive to qualify a valid preamble. The preamble threshold should be adjusted depending on the nature of the application. The required preamble length threshold will depend on when receive mode is entered in relation to the start of the transmitted packet and the length of the transmit preamble. With a shorter than recommended preamble detection threshold the probability of false detection is directly related to how long the receiver operates on noise before the transmit preamble is received. False detection on noise may cause the actual packet to be missed. The preamble detection threshold is programmed in register 35h. For most applications with a preamble length longer than 32 bits the default value of 20 is recommended for the pre-

amble detection threshold. A shorter Preamble Detection Threshold may be chosen if occasional false detections may be tolerated. When antenna diversity is enabled a 20-bit preamble detection threshold is recommended. When the receiver is synchronously enabled just before the start of the packet, a shorter preamble detection threshold may be used. Table 23.5 demonstrates the recommended preamble detection threshold and preamble length for various modes.

It is possible to use the transceiver in a raw mode without the requirement for a 010101... preamble. Contact customer support for further details.

**Table 23.5. Minimum Receiver Settling Time**

| Mode | Approximate Receiver Settling Time | Recommended Preamble Length with 8-Bit Detection Threshold | Recommended Preamble Length with 20-Bit Detection Threshold |
|---|---|---|---|
| (G)FSK AFC Disabled | 1 byte | 20 bits | 32 bits |
| (G)FSK AFC Enabled | 2 byte | 28 bits | 40 bits |
| (G)FSK AFC Disabled +Antenna Diversity Enabled | 1 byte | — | 64 bits |
| (G)FSK AFC Enabled +Antenna Diversity Enabled | 2 byte | — | 8 byte |
| OOK | 2 byte | 3 byte | 4 byte |
| OOK + Antenna Diversity Enabled | 8 byte | — | 8 byte |

**Note:** The recommended preamble length and preamble detection threshold listed above are to achieve 0% PER. They may be shortened when occasional packet errors are tolerable.

### 23.6.8. Invalid Preamble Detector

When scanning channels in a frequency hopping system it is desirable to determine if a channel is valid in the minimum amount of time. The preamble detector can output an invalid preamble detect signal. which can be used to identify the channel as invalid. After a configurable time set in Register 60h[7:4], an invalid preamble detect signal is asserted indicating an invalid channel. The period for evaluating the signal for invalid preamble is defined as (inv_pre_th[3:0] x 4) x Bit Rate Period. The preamble detect and invalid preamble detect signals are available in "Register 03h. Interrupt/Status 1" and “Register 04h. Interrupt/Status 2.”

### 23.6.9. Synchronization Word Configuration

The synchronization word length for both TX and RX can be configured in Reg 33h, synclen[1:0]. The expected or transmitted sync word can be configured from 1 to 4 bytes as defined below:

- synclen[1:0] = 00—Expected/Transmitted Synchronization Word (sync word) 3.
- synclen[1:0] = 01—Expected/Transmitted Synchronization Word 3 first, followed by sync word 2.
- synclen[1:0] = 10—Expected/Transmitted Synchronization Word 3 first, followed by sync word 2, followed by sync word 1.
- synclen[1:0] = 1—Send/Expect Synchronization Word 3 first, followed by sync word 2, followed by sync word 1, followed by sync word 0.

The sync is transmitted or expected in the following sequence: sync 3→sync 2→sync 1→sync 0. The sync word values can be programmed in Registers 36h–39h. After preamble detection the part will search for sync for a fixed period of time. If a sync is not recognized in this period then a timeout will occur and the search for preamble will be re-initiated. The timeout period after preamble detections is defined as the value programmed into the sync word length plus four additional bits.

### 23.6.10. Receive Header Check

The header check is designed to support 1–4 bytes and broadcast headers. The header length needs to be set in register 33h, hdlen[2:0]. The headers to be checked need to be set in register 32h, hdch[3:0]. For instance, there can be four bytes of header in the packet structure but only one byte of the header is set to be checked (i.e., header 3). For the headers that are set to be checked, the expected value of the header should be programmed in chhd[31:0] in Registers 3F–42. The individual bits within the selected bytes to be checked can be enabled or disabled with the header enables, hden[31:0] in Registers 43–46. For example, if you want to check all bits in header 3 then hden[31:24] should be set to FF but if only the last 4 bits are desired to be checked then it should be set to 00001111 (0F). Broadcast headers can also be programmed by setting bcen[3:0] in Register 32h. For broadcast header check the value may be either “FFh” or the value stored in the Check Header register. A logic equivalent of the header check for Header 3 is shown in Figure 23.19. A similar logic check will be done for Header 2, Header 1, and Header 0 if enabled.

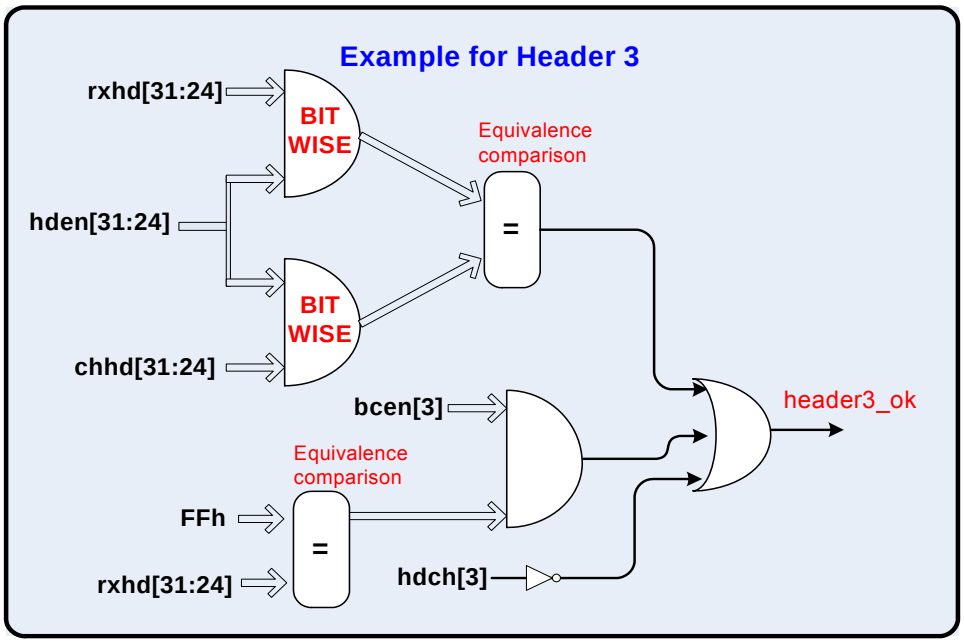



**Figure 23.19. Header**

### 23.6.11. TX Retransmission and Auto TX

The transceiver is capable of automatically retransmitting the last packet loaded in the TX FIFO. Automatic retransmission is set by entering the TX state with the txon bit without reloading the TX FIFO. This feature is useful for beacon transmission or when retransmission is required due to the absence of a valid acknowledgment. Only packets that fit completely in the TX FIFO can be automatically retransmitted.

An automatic transmission function is available, allowing the radio to automatically start or stop a transmission depending on the amount of data in the TX FIFO.

When autotx is set in “Register 08. Operating & Function Control 2", the transceiver will automatically enter the TX state when the TX FIFO almost full threshold is exceeded. Packets will be transmitted according to the configured packet length. To stop transmitting, clear the packet sent or TX FIFO almost empty interrupts must be cleared by reading register.

## 23.7. RX Modem Configuration

A Microsoft Excel parameter calculator or Wireless Development Suite (WDS) calculator is provided to determine the proper settings for the modem. The calculator can be found on www.silabs.com or on the CD provided with the demo kits. An application note is available to describe how to use the calculator and to provide advanced descriptions of the modem settings and calculations.

### 23.7.1. Modem Settings for FSK and GFSK

The modem performs channel selection and demodulation in the digital domain. The channel filter bandwidth is configurable from 2.6 to 620 kHz. The receiver data-rate, modulation index, and bandwidth are set via registers 1C–25h. The modulation index is equal to 2 times the peak deviation divided by the data rate (Rb).

When Manchester coding is disabled, the required channel filter bandwidth is calculated as BW = 2Fd + Rb where Fd is the frequency deviation and Rb is the data rate.

## 23.8. Auxiliary Functions

The EZRadioPRO has some auxiliary functions that duplicate the directly accessible MCU peripherals: ADC, temperature sensor, and 32 kHz oscillator. These auxiliary functions are retained primarily for compatibility with the Si4430/1/2. The directly accessed MCU peripherals typically provide lower system current consumption and better analog performance. However some of these EZRadioPRO auxiliary functions offer features not directly duplicated in the MCU directly accessed peripherals, such as the Low Duty Cycle Mode operation.

### 23.8.1. Smart Reset

The EZRadioPRO transceiver contains an enhanced integrated SMART RESET or POR circuit. The POR circuit contains both a classic level threshold reset as well as a slope detector POR. This reset circuit was designed to produce a reliable reset signal under any circumstances. Reset will be initiated if any of the following conditions occur:

- Initial power on, $V_{DD}$ starts from gnd: reset is active till $V_{DD}$ reaches $V_{RR}$ (see table);
- When $V_{DD}$ decreases below $V_{LD}$ for any reason: reset is active till $V_{DD}$ reaches $V_{RR}$;
- A software reset via “Register 08h. Operating Mode and Function Control 2”: reset is active for time $T_{SWRST}$
- On the rising edge of a $V_{DD}$ glitch when the supply voltage exceeds the following time functioned limit:

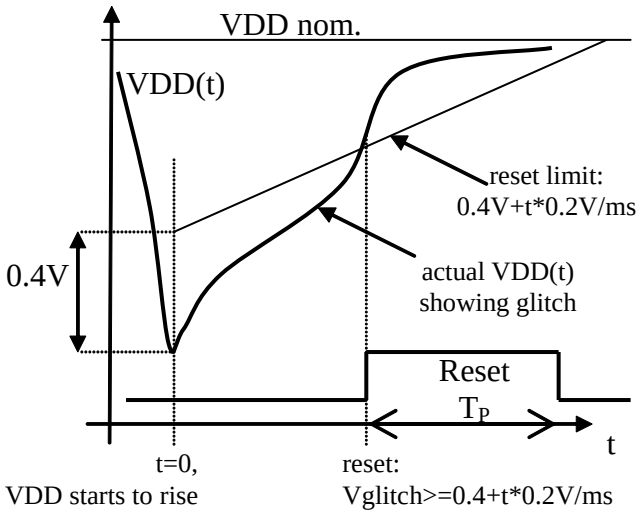



**Figure 23.20. POR Glitch Parameters**

**Table 23.6. POR Parameters**

| Parameter | Symbol | Comment | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Release Reset Voltage | VRR | | 0.85 | 1.3 | 1.75 | V |
| Power-On $V_{DD}$ Slope | SVDD | tested $V_{DD}$ slope region | 0.03 | — | 300 | V/ms |
| Low $V_{DD}$ Limit | VLD | VLD<VRR is guaranteed | 0.7 | 1 | 1.3 | V |
| Software Reset Pulse | TSWRST | | 50 | — | 470 | us |
| Threshold Voltage | VTSD | | — | 0.4 | — | V |
| Reference Slope | k | | — | 0.2 | — | V/ms |
| $V_{DD}$ Glitch Reset Pulse | TP | Also occurs after SDN, and initial power on | POR Reset Time $t_{POR}$ is specified in Table 4.20 on page 70. | | | |

The reset will initialize all registers to their default values. The reset signal is also available for output and use by the microcontroller by using the default setting for GPIO_0. The inverted reset signal is available by default on GPIO_1.

### 23.8.2. Output Clock

The 30 MHz crystal oscillator frequency is divided down internally and may be output on GPIO2. This feature is useful to lower BOM cost by using only one crystal in the system. The output clock on GPIO2 may be routed to the XTAL2 input to provide a synchronized clock source between the MCU and the EZRadioPRO peripheral. The output clock frequency is selectable from one of 8 options, as shown below. Except for the 32.768 kHz option, all other frequencies are derived by dividing the crystal oscillator frequency. The 32.768 kHz clock signal is derived from an internal RC oscillator or an external 32 kHz crystal. The default setting for GPIO2 is to output the clock signal with a frequency of 1 MHz.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0A | R/W | Output Clock | | | clkt[1] | clkt[0] | enlfc | mclk[2] | mclk[1] | mclk[0] | 06h |

| mclk[2:0] | Modulation Source |
|---|---|
| 000 | 30 MHz |
| 001 | 15 MHz |
| 010 | 10 MHz |
| 011 | 4 MHz |
| 100 | 3 MHz |
| 101 | 2 MHz |
| 110 | 1 MHz |
| 111 | 32.768 kHz |

Since the crystal oscillator is disabled in SLEEP mode in order to save current, the low-power 32.768 kHz clock can be automatically switched to become the output clock. This feature is called enable low frequency clock and is enabled by the enlfc bit in “Register 0Ah. Microcontroller Output Clock." When enlfc = 1 and the chip is in SLEEP mode then the 32.768 kHz clock will be provided regardless of the setting of mclk[2:0]. For example, if mclk[2:0] = 000, 30 MHz will be provided through the GPIO output pin in all IDLE,

TX, or RX states. When the chip enters SLEEP mode, the output clock will automatically switch to 32.768 kHz from the RC oscillator or 32.768 XTAL.

Another available feature for the output clock is the clock tail, clkt[1:0] in “Register 0Ah. Microcontroller Output Clock." If the low frequency clock feature is not enabled (enlfc = 0), then the output is disabled in SLEEP mode. Setting the clkt[1:0] field will provide additional cycles of the output clock before it shuts off.

| clkt[1:0] | Modulation Source |
|---|---|
| 00 | 0 cycles |
| 01 | 128 cycles |
| 10 | 256 cycles |
| 11 | 512 cycles |

If an interrupt is triggered, the output clock will remain enabled regardless of the selected mode. As soon as the interrupt is read the state machine will then move to the selected mode. The minimum current consumption will not be achieved until the interrupt is read. For instance, if the EZRadioPRO peripheral is commanded to SLEEP mode but an interrupt has occurred the 30 MHz XTAL will not be disabled until the interrupt has been cleared.

### 23.8.3. General Purpose ADC

The EZRadioPRO peripheral includes an 8-bit SAR ADC independent of ADC0. It may be used for general purpose analog sampling, as well as for digitizing the EZRadioPRO temperature sensor reading. In most cases, the ADC0 subsystem directly accessible from the MCU will be preferred over the ADC embedded inside the EZRadioPRO peripheral. Registers 0Fh "ADC Configuration", 10h "Sensor Offset" and 4Fh "Amplifier Offset" can be used to configure the ADC operation. Details of these registers are in “AN440: EZRadioPRO Detailed Register Descriptions.”

Every time an ADC conversion is desired, bit 7 "adcstart/adcdone" in Register 0Fh “ADC Configuration” must be set to 1. The conversion time for the ADC is 350 µs. After the ADC conversion is done and the adcdone signal is showing 1, then the ADC value may be read out of “Register 11h: ADC Value." When the ADC is doing its conversion, the adcstart/adcdone bit will read 0. When the ADC has finished its conversion, the bit will be set to 1. A new ADC conversion can be initiated by writing a 1 to the adcstart/adcdone bit.

The architecture of the ADC is shown in Figure 23.21. The signal and reference inputs of the ADC are selected by adcsel[2:0] and adcref[1:0] in register 0Fh “ADC Configuration”, respectively. The default setting is to read out the temperature sensor using the bandgap voltage (VBG) as reference. With the VBG reference the input range of the ADC is from 0–1.02 V with an LSB resolution of 4 mV (1.02/255). Changing the ADC reference will change the LSB resolution accordingly.

A differential multiplexer and amplifier are provided for interfacing external bridge sensors. The gain of the amplifier is selectable by adcgain[1:0] in Register 0Fh. The majority of sensor bridges have supply voltage (VDD) dependent gain and offset. The reference voltage of the ADC can be changed to either $V_{DD}/2$ or $V_{DD}/3$. A programmable $V_{DD}$ dependent offset voltage can be added using soffs[3:0] in register 10h.

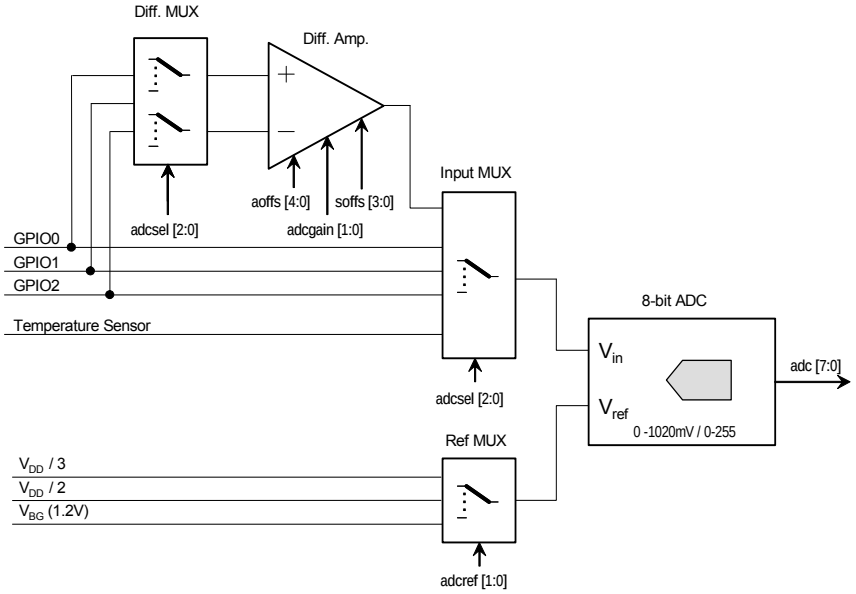



**Figure 23.21. General Purpose ADC Architecture**

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0F | R/W | ADC Configuration | adcstart/adcdone | adcsel[2] | adcsel[1] | adcsel[0] | adcref[1] | adcref[0] | adcgain[1] | adcgain[0] | 00h |
| 10 | R/W | Sensor Offset | | | | | soffs[3] | soffs[2] | soffs[1] | soffs[0] | 00h |
| 11 | R | ADC Value | adc[7] | adc[6] | adc[5] | adc[4] | adc[3] | adc[2] | adc[1] | adc[0] | — |

### 23.8.4. Temperature Sensor

The EZRadioPRO peripheral includes an integrated on-chip analog temperature sensor independent of the temperature sensor associated with ADC0. The temperature sensor will be automatically enabled when the temperature sensor is selected as the input of the EZRadioPRO ADC or when the analog temp voltage is selected on the analog test bus. The temperature sensor value may be digitized using the EZRadioPRO general-purpose ADC and read out through "Register 10h. ADC Sensor Amplifier Offset." The range of the temperature sensor is configurable. Table 23.7 lists the settings for the different temperature ranges and performance.

To use the Temp Sensor:

1. Set the input for ADC to the temperature sensor, "Register 0Fh. ADC Configuration"—adcsel[2:0] = 000
2. Set the reference for ADC, "Register 0Fh. ADC Configuration"—adcref[1:0] = 00
3. Set the temperature range for ADC, "Register 12h. Temperature Sensor Calibration"—tsrange[1:0]
4. Set entsoffs = 1, "Register 12h. Temperature Sensor Calibration"
5. Trigger ADC reading, "Register 0Fh. ADC Configuration"—adcstart = 1
6. Read temperature value—Read contents of "Register 11h. ADC Value"

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | R/W | Temperature Sensor Control | tsrange[1] | tsrange[0] | entsoffs | entstrim | tstrim[3] | tstrim[2] | vbgtrim[1] | vbgtrim[0] | 20h |
| 13 | R/W | Temperature Value Offset | tvoffs[7] | tvoffs[6] | tvoffs[5] | tvoffs[4] | tvoffs[3] | tvoffs[2] | tvoffs[1] | tvoffs[0] | 00h |

**Table 23.7. Temperature Sensor Range**

| entoff | tsrange[1] | tsrange[0] | Temp. range | Unit | Slope | ADC8 LSB |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | –64 … 64 | °C | 8 mV/°C | 0.5 °C |
| 1 | 0 | 1 | –64 … 192 | °C | 4 mV/°C | 1 °C |
| 1 | 1 | 0 | 0 … 128 | °C | 8 mV/°C | 0.5 °C |
| 1 | 1 | 1 | –40 … 216 | °F | 4 mV/°F | 1 °F |
| 0* | 1 | 0 | 0 … 341 | °K | 3 mV/°K | 1.333 °K |

**Note:** Absolute temperature mode, no temperature shift. This mode is only for test purposes. POR value of EN_TOFF is 1.

The slope of the temperature sensor is very linear and monotonic. For absolute accuracy better than 10 °C calibration is necessary. The temperature sensor may be calibrated by setting entsoffs = 1 in “Register 12h. Temperature Sensor Control” and setting the offset with the tvoffs[7:0] bits in “Register 13h. Temperature Value Offset.” This method adds a positive offset digitally to the ADC value that is read in “Register 11h. ADC Value.” The other method of calibration is to use the tstrim which compensates the analog circuit. This is done by setting entstrim = 1 and using the tstrim[2:0] bits to offset the temperature in “Register 12h. Temperature Sensor Control.” With this method of calibration, a negative offset may be achieved. With both methods of calibration better than ±3 °C absolute accuracy may be achieved.

The different ranges for the temperature sensor and ADC8 are demonstrated in Figure 23.22. The value of the ADC8 may be translated to a temperature reading by ADC8Value x ADC8 LSB + Lowest Temperature in Temp Range. For instance for a tsrange = 00, Temp = ADC8Value x 0.5 – 64.

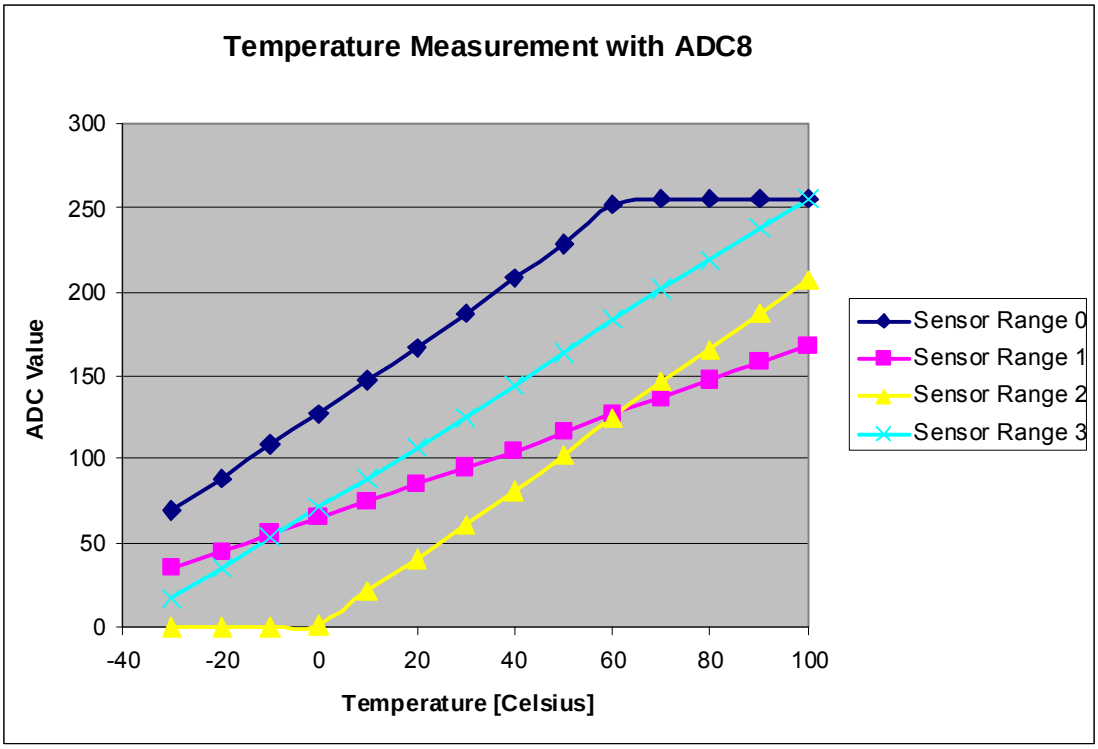



**Figure 23.22. Temperature Ranges using ADC8**

### 23.8.5. Low Battery Detector

The Low Battery Detector (LBD) feature of the EZRadioPRO peripheral is not supported in the Si1000/1/2/3/4/5. Use ADC0 instead. Refer to Section Section “5. 10-Bit SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode” on page 74 for details.

### 23.8.6. Wake-Up Timer and 32 kHz Clock Source

The EZRadioPRO peripheral contains an integrated wake-up timer independent of the SmaRTClock which can be used to periodically wake the chip from SLEEP mode using the interrupt pin. The wake-up timer runs from the internal 32.768 kHz RC Oscillator. The wake-up timer can be configured to run when in SLEEP mode. If enwt = 1 in "Register 07h. Operating Mode and Function Control 1" when entering SLEEP mode, the wake-up timer will count for a time specified defined in Registers 14–16h, "Wake Up Timer Period". At the expiration of this period an interrupt will be generated on the nIRQ pin if this interrupt is enabled. The software will then need to verify the interrupt by reading the Registers 03h–04h, "Interrupt Status 1 & 2". The wake-up timer value may be read at any time by the wtv[15:0] read only registers 17h–18h.

The formula for calculating the Wake-Up Period is the following:

$$WUT = \frac{32 \times M \times 2^{R}}{32.768} ms$$

| WUT Register | Description |
|---|---|
| wtr[4:0] | R Value in Formula |
| wtm[15:0] | M Value in Formula |

Use of the D variable in the formula is only necessary if finer resolution is required than can be achieved by using the R value.

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | R/W | Wake-Up Timer Period 1 | | | | wtr[4] | wtr[3] | wtr[2] | wtr[1] | wtr[0] | 03h |
| 15 | R/W | Wake-Up Timer Period 2 | wtm[15] | wtm[14] | wtm[13] | wtm[12] | wtm[11] | wtm[10] | wtm[9] | wtm[8] | 00h |
| 16 | R/W | Wake-Up Timer Period 3 | wtm[7] | wtm[6] | wtm[5] | wtm[4] | wtm[3] | wtm[2] | wtm[1] | wtm[0] | 00h |
| 17 | R | Wake-Up Timer Value 1 | wtv[15] | wtv[14] | wtv[13] | wtv[12] | wtv[11] | wtv[10] | wtv[9] | wtv[8] | — |
| 18 | R | Wake-Up Timer Value 2 | wtv[7] | wtv[6] | wtv[5] | wtv[4] | wtv[3] | wtv[2] | wtv[1] | wtv[0] | — |

There are two different methods for utilizing the wake-up timer (WUT) depending on if the WUT interrupt is enabled in “Register 06h. Interrupt Enable 2.” If the WUT interrupt is enabled then nIRQ pin will go low when the timer expires. The chip will also change state so that the 30 MHz XTAL is enabled so that the microcontroller clock output is available for the microcontroller to use to process the interrupt. The other method of use is to not enable the WUT interrupt and use the WUT GPIO setting. In this mode of operation the chip will not change state until commanded by the microcontroller. The different modes of operating the WUT and the current consumption impacts are demonstrated in Figure 23.23.

A 32 kHz XTAL may also be used for better timing accuracy. By setting the x32 ksel bit in Register 07h "Operating & Function Control 1", GPIO0 is automatically reconfigured so that an external 32 kHz XTAL may be connected to this pin. In this mode, the GPIO0 is extremely sensitive to parasitic capacitance, so only the XTAL should be connected to this pin with the XTAL physically located as close to the pin as possible. Once the x32 ksel bit is set, all internal functions such as WUT, microcontroller clock, and LDC mode will use the 32 kHz XTAL and not the 32 kHz RC oscillator.

The 32 kHz XTAL accuracy is comprised of both the XTAL parameters and the internal circuit. The XTAL accuracy can be defined as the XTAL initial error + XTAL aging + XTAL temperature drift + detuning from the internal oscillator circuit. The error caused by the internal circuit is typically less than 10 ppm.

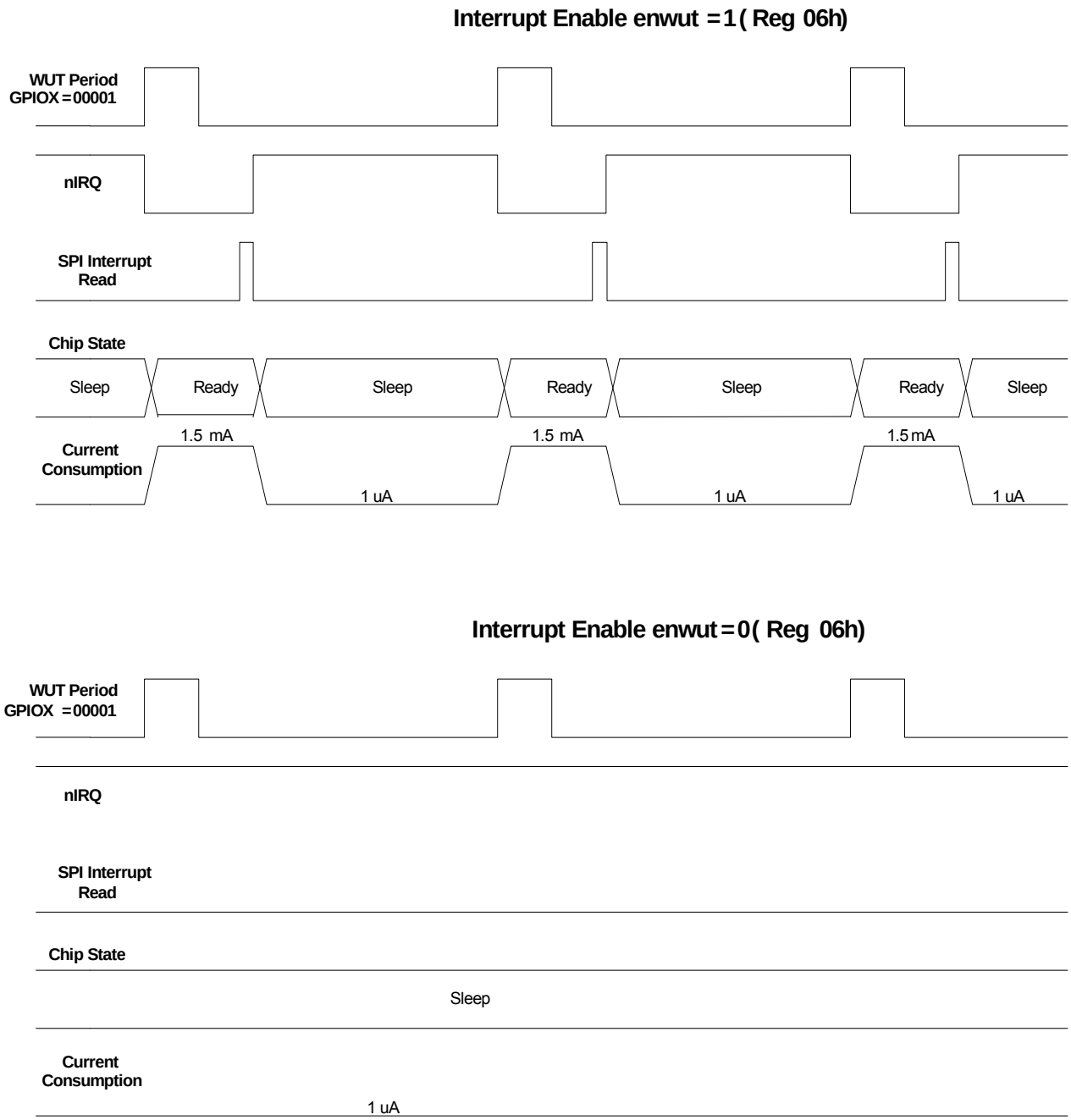



**Figure 23.23. WUT Interrupt and WUT Operation**

### 23.8.7. Low Duty Cycle Mode

The Low Duty Cycle Mode is available to automatically wake-up the receiver to check if a valid signal is available. The basic operation of the low duty cycle mode is demonstrated in the figure below. If a valid preamble or sync word is not detected the chip will return to sleep mode until the beginning of a new WUT period. If a valid preamble and sync are detected the receiver on period will be extended for the low duty cycle mode duration (TLDC) to receive all of the packet. The WUT period must be set in conjunction with the low duty cycle mode duration. The R value (“Register 14h. Wake-up Timer Period 1”) is shared between the WUT and the TLDC. The ldc[7:0] bits are located in “Register 19h. Low Duty Cycle Mode Duration.” The time of the TLDC is determined by the formula below:

$$TLDC = ldc[7:0] \times \frac{4 \times 2^{R}}{32.768}\ ms$$

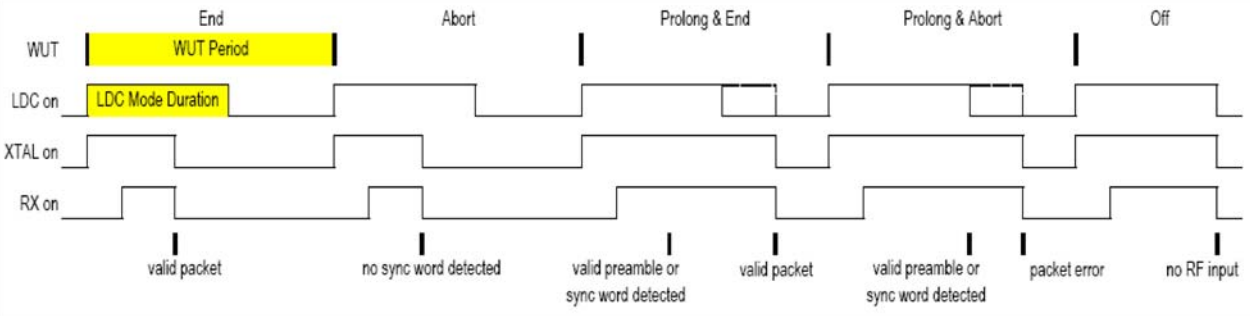



**Figure 23.24. Low Duty Cycle Mode**

### 23.8.8. GPIO Configuration

Three general purpose IOs (GPIOs) are available. Numerous functions such as specific interrupts, TRSW control, etc. can be routed to the GPIO pins as shown in the tables below. When in Shutdown mode all the GPIO pads are pulled low.

**Note:** The ADC should not be selected as an input to the GPIO in standby or sleep modes and will cause excess current consumption.

| Add | R/W | Function/ Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0B | R/W | GPIO0 Configuration | gpio0drv[1] | gpio0drv[0] | pup0 | gpio0[4] | gpio0[3] | gpio0[2] | gpio0[1] | gpio0[0] | 00h |
| 0C | R/W | GPIO1 Configuration | gpio1drv[1] | gpio1drv[0] | pup1 | gpio1[4] | gpio1[3] | gpio1[2] | gpio1[1] | gpio1[0] | 00h |
| 0D | R/W | GPIO2 Configuration | gpio2drv[1] | gpio2drv[0] | pup2 | gpio2[4] | gpio2[3] | gpio2[2] | gpio2[1] | gpio2[0] | 00h |
| 0E | R/W | I/O Port Configuration | | extitst[2] | extitst[1] | extitst[0] | itsdo | dio2 | dio1 | dio0 | 00h |

The GPIO settings for GPIO1 and GPIO2 are the same as for GPIO0 with the exception of the 00000 default setting. The default settings for each GPIO are listed below:

| GPIO | 00000—Default Setting |
|---|---|
| GPIO0 | POR |
| GPIO1 | POR Inverted |
| GPIO2 | Output Clock |

For a complete list of the available GPIOs see “AN440: EZRadioPRO Detailed Register Descriptions”.

The GPIO drive strength may be adjusted with the gpioXdrv[1:0] bits. Setting a higher value will increase the drive strength and current capability of the GPIO by changing the driver size. Special care should be taken in setting the drive strength and loading on GPIO2 when the microcontroller clock is used. Excess loading or inadequate drive may contribute to increased spurious emissions.

Pin 6, ANT_A may be used as an alternate to control a TR switch. Pin 6 is a hardwired version of GPIO setting 11000, Antenna 2 Switch used for antenna diversity. It can be manually controlled by the antdiv[2:0] bits in register 08h if antenna diversity is not used. See AN440, register 08h for more details.

### 23.8.9. Antenna Diversity

To mitigate the problem of frequency-selective fading due to multi-path propagation, some transceiver systems use a scheme known as antenna diversity. In this scheme, two antennas are used. Each time the transceiver enters RX mode the receive signal strength from each antenna is evaluated. This evaluation process takes place during the preamble portion of the packet. The antenna with the strongest received signal is then used for the remainder of that RX packet. The same antenna will also be used for the next corresponding TX packet.

This chip fully supports antenna diversity with an integrated antenna diversity control algorithm. The required signals needed to control an external SPDT RF switch (such as PIN diode or GaAs switch) are available on the GPIOx pins. The operation of these GPIO signals is programmable to allow for different antenna diversity architectures and configurations. The antdiv[2:0] bits are found in register 08h “Operating & Function Control 2.” The GPIO pins are capable of sourcing up to 5 mA of current, so it may be used directly to forward-bias a PIN diode if desired.

The antenna diversity algorithm will automatically toggle back and forth between the antennas until the packet starts to arrive. The recommended preamble length for optimal antenna selection is 8 bytes. A special antenna diversity algorithm (antdiv[2:0] = 110 or 111) is included that allows for shorter preamble lengths for beacon mode in TDMA-like systems where the arrival of the packet is synchronous to the receiver enable. The recommended preamble length to obtain optimal antenna selection for synchronous mode is 4 bytes.

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 08 | R/W | Operating & Function Control 2 | antdiv[2] | antdiv[1] | antdiv[0] | rxmpk | autotx | enldm | ffclrrx | ffclrtx | 00h |

**Table 23.8. Antenna Diversity Control**

| antdiv[2:0] | RX/TX State | | Non RX/TX State | |
|---|---|---|---|---|
| | GPIO Ant1 | GPIO Ant2 | GPIO Ant1 | GPIO Ant2 |
| 000 | 0 | 1 | 0 | 0 |
| 001 | 1 | 0 | 0 | 0 |
| 010 | 0 | 1 | 1 | 1 |
| 011 | 1 | 0 | 1 | 1 |
| 100 | Antenna Diversity Algorithm | | 0 | 0 |
| 101 | Antenna Diversity Algorithm | | 1 | 1 |
| 110 | Antenna Diversity Algorithm in Beacon Mode | | 0 | 0 |
| 111 | Antenna Diversity Algorithm in Beacon Mode | | 1 | 1 |

### 23.8.10. RSSI and Clear Channel Assessment

Received signal strength indicator (RSSI) is an estimate of the signal strength in the channel to which the receiver is tuned. The RSSI value can be read from an 8-bit register with 0.5 dB resolution per bit. Figure 23.25 demonstrates the relationship between input power level and RSSI value. The absolute value of the RSSI will change slightly depending on the modem settings. The RSSI may be read at anytime, but an incorrect error may rarely occur. The RSSI value may be incorrect if read during the update period. The update period is approximately 10 ns every 4 Tb. For 10 kbps, this would result in a 1 in 40,000 probability that the RSSI may be read incorrectly. This probability is extremely low, but to avoid this, one of the following options is recommended: majority polling, reading the RSSI value within 1 Tb of the RSSI interrupt, or using the RSSI threshold described in the next paragraph for Clear Channel Assessment (CCA).

| Add | R/W | Function/Description | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | POR Def. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 26 | R | Received Signal Strength Indicator | rssi[7] | rssi[6] | rssi[5] | rssi[4] | rssi[3] | rssi[2] | rssi[1] | rssi[0] | — |
| 27 | R/W | RSSI Threshold for Clear Channel Indicator | rssith[7] | rssith[6] | rssith[5] | rssith[4] | rssith[3] | rssith[2] | rssith[1] | rssith[0] | 00h |

For CCA, threshold is programmed into rssith[7:0] in "Register 27h. RSSI Threshold for Clear Channel Indicator." After the RSSI is evaluated in the preamble, a decision is made if the signal strength on this channel is above or below the threshold. If the signal strength is above the programmed threshold then the RSSI status bit, irssi, in "Register 04h. Interrupt/Status 2" will be set to 1. The RSSI status can also be routed to a GPIO line by configuring the GPIO configuration register to GPIOx[3:0] = 1110.

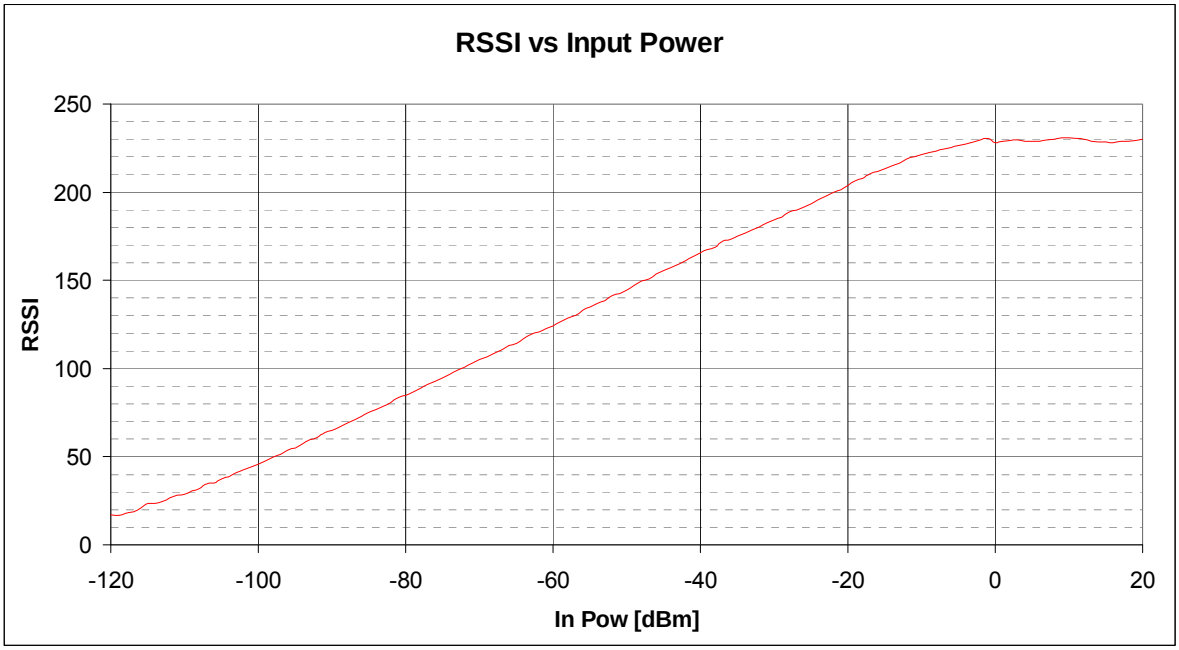
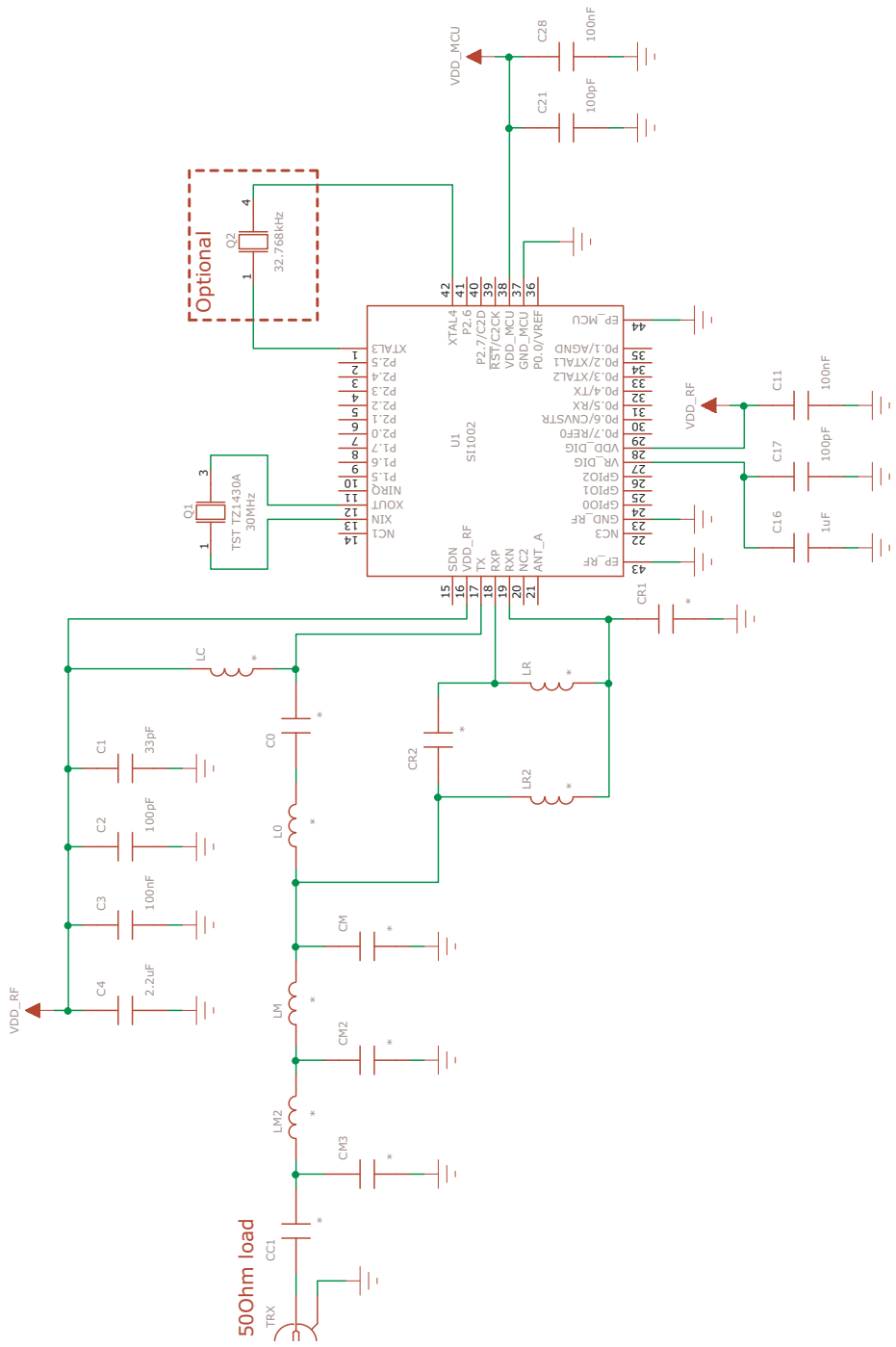



**Figure 23.25. RSSI Value vs. Input Power**

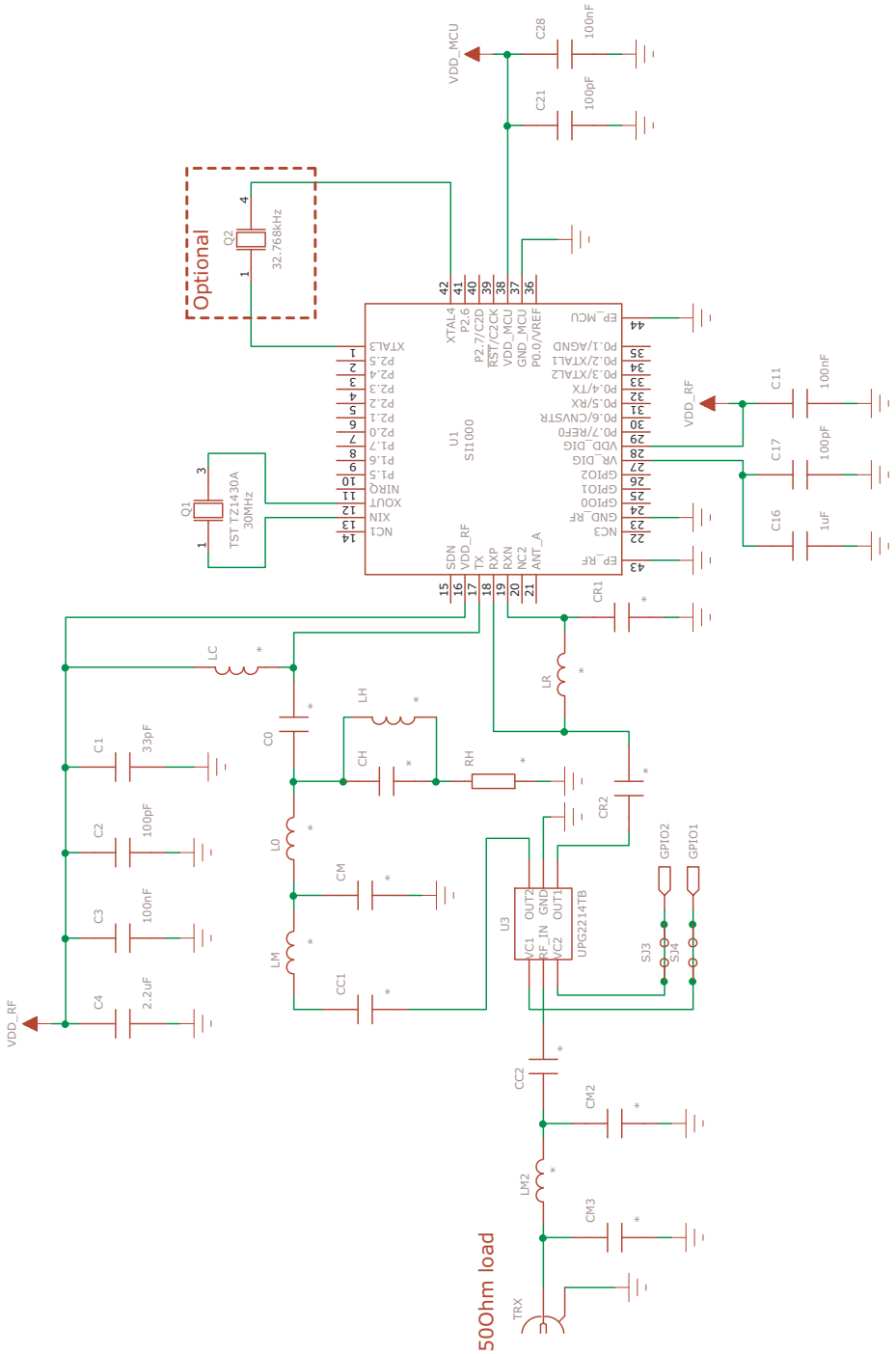
## 23.9. Reference Design

Reference designs are available at www.silabs.com for many common applications which include recommended schematics, BOM, and layout. TX matching component values for the different frequency bands can be found in the application notes “AN435: Si4032/4432 PA Matching” and “AN436: Si4030/4031/4430/4431 PA Matching.” RX matching component values for different frequency bands can be found in “AN427: EZRadioPRO Si433x and Si443x RX LNA Matching.”




Refer to "AN436: Si4030/4031/4430/4431 PA Matching" for component values.

**Figure 23.26. Si1002 Split RF TX/RX Direct-Tie Reference Design—Schematic**




Refer to "AN427: EZRadioPRO™ Si433X & Si443X RX LNA Matching" and "AN435: Si4032/4432 PA Matching" for component values.

**Figure 23.27. Si1000 Switch Matching Reference Design—Schematic**

## 23.10. Application Notes and Reference Designs

A comprehensive set of application notes and reference designs are available to assist with the development of a radio system. A partial list of applications notes is given below.

For the complete list of application notes, latest reference designs and demos visit the Silicon Labs website.

- AN361: Wireless MBUS Implementation using EZRadioPRO Devices
- AN379: Antenna Diversity with EZRadioPRO
- AN414: EZRadioPRO Layout Design Guide
- AN415: EZRadioPRO Programming Guide
- AN417: Si4x3x Family Crystal Oscillators
- AN419: ARIB STD-T67 Narrow-Band 426/429 MHz Measured on the Si4431-A0
- AN427: EZRadioPRO Si433x and Si443x RX LNA Matching
- AN429: Using the DC-DC Converter on the F9xx Series MCU for Single Battery Operation with the EZRadioPRO RF Devices
- AN432: RX BER Measurement on EZRadioPRO with a Looped PN Sequence
- AN435: Si4032/4432 PA Matching
- AN436: Si4030/4031/4430/4431 PA Matching
- AN437: 915 MHz Measurement Results and FCC Compliance
- AN439: EZRadioPRO Quick Start Guide
- AN440: Si4430/31/32 Register Descriptions
- AN445: Si4431 RF Performance and ETSI Compliance Test Results
- AN451: Wireless M-BUS Software Implementation
- AN459: 950 MHz Measurement Results and ARIB Compliance
- AN460: 470 MHz Measurement Results for China
- AN463: Support for Non-Standard Packet Structures and RAW Mode
- AN466: Si4030/31/32 Register Descriptions
- AN467: Si4330 Register Descriptions
- AN514: Using the EZLink Reference Design to Create a Two-Channel PWM Motor Control Circuit
- AN539: EZMacPRO Overview

## 23.11. Customer Support

Technical support for the complete family of Silicon Labs wireless products is available by accessing the wireless section of the Silicon Labs' website at www.silabs.com/wireless. For MCU support, please visit www.silabs.com/mcu.

For answers to common questions please visit the wireless and mcu knowledge base at www.silabs.com/support/knowledgebase.

## 23.12. Register Table and Descriptions

**Table 23.9. EZRadioPRO Internal Register Descriptions***

| Add | R/W | Function/Desc | Data | | | | | | | | POR Default |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| 00 | R | Device Type | 0 | 0 | 0 | dt[4] | dt[3] | dt[2] | dt[1] | dt[0] | 01000 |
| 01 | R | Device Version | 0 | 0 | 0 | vc[4] | vc[3] | vc[2] | vc[1] | vc[0] | 07h |
| 02 | R | Device Status | ffovfl | ffunfl | rxffem | headerr | reserved | reserved | cps[1] | cps[0] | — |
| 03 | R | Interrupt Status 1 | ifferr | itxffafull | itxffaem | irxffafull | iext | ipksent | ipkvalid | icrcerror | — |
| 04 | R | Interrupt Status 2 | iswdet | ipreaval | ipreainval | irssi | iwut | ilbd | ichiprdy | ipor | — |
| 05 | R/W | Interrupt Enable 1 | enfferr | entxffafull | entxffaem | enrxffafull | enext | enpksent | enpkvalid | encrcerror | 00h |
| 06 | R/W | Interrupt Enable 2 | enswdet | enpreaval | enpreainval | enrssi | enwut | enlbd | enchiprdy | enpor | 03h |
| 07 | R/W | Operating & Function Control 1 | swres | enlbd | enwt | x32ksel | txon | rxon | pllon | xton | 01h |
| 08 | R/W | Operating & Function Control 2 | antdiv[2] | antdiv[1] | antdiv[0] | rxmpk | autotx | enldm | ffclrrx | ffclrtx | 00h |
| 09 | R/W | Crystal Oscillator Load Capacitance | xtalshft | xlc[6] | xlc[5] | xlc[4] | xlc[3] | xlc[2] | xlc[1] | xlc[0] | 7Fh |
| 0A | R/W | Microcontroller Output Clock | Reserved | Reserved | clkt[1] | clkt[0] | enlfc | mclk[2] | mclk[1] | mclk[0] | 06h |
| 0B | R/W | GPIO0 Configuration | gpio0drv[1] | gpio0drv[0] | pup0 | gpio0[4] | gpio0[3] | gpio0[2] | gpio0[1] | gpio0[0] | 00h |
| 0C | R/W | GPIO1 Configuration | gpio1drv[1] | gpio1drv[0] | pup1 | gpio1[4] | gpio1[3] | gpio1[2] | gpio1[1] | gpio1[0] | 00h |
| 0D | R/W | GPIO2 Configuration | gpio2drv[1] | gpio2drv[0] | pup2 | gpio2[4] | gpio2[3] | gpio2[2] | gpio2[1] | gpio2[0] | 00h |
| 0E | R/W | I/O Port Configuration | Reserved | extitst[2] | extitst[1] | extitst[0] | itsdo | dio2 | dio1 | dio0 | 00h |
| 0F | R/W | ADC Configuration | adcstart/*adcdone* | adcsel[2] | adcsel[1] | adcsel[0] | adcref[1] | adcref[0] | adcgain[1] | adcgain[0] | 00h |
| 10 | R/W | ADC Sensor Amplifier Offset | Reserved | Reserved | Reserved | Reserved | adcoffs[3] | adcoffs[2] | adcoffs[1] | adcoffs[0] | 00h |
| 11 | R | ADC Value | adc[7] | adc[6] | adc[5] | adc[4] | adc[3] | adc[2] | adc[1] | adc[0] | — |
| 12 | R/W | Temperature Sensor Control | tsrange[1] | tsrange[0] | entsoffs | entstrim | tstrim[3] | tstrim[2] | tstrim[1] | tstrim[0] | 20h |
| 13 | R/W | Temperature Value Offset | tvoffs[7] | tvoffs[6] | tvoffs[5] | tvoffs[4] | tvoffs[3] | tvoffs[2] | tvoffs[1] | tvoffs[0] | 00h |
| 14 | R/W | Wake-Up Timer Period 1 | Reserved | Reserved | Reserved | wtr[4] | wtr[3] | wtr[2] | wtr[1] | wtr[0] | 03h |
| 15 | R/W | Wake-Up Timer Period 2 | wtm[15] | wtm[14] | wtm[13] | wtm[12] | wtm[11] | wtm[10] | wtm[9] | wtm[8] | 00h |
| 16 | R/W | Wake-Up Timer Period 3 | wtm[7] | wtm[6] | wtm[5] | wtm[4] | wtm[3] | wtm[2] | wtm[1] | wtm[0] | 01h |
| 17 | R | Wake-Up Timer Value 1 | wtv[15] | wtv[14] | wtv[13] | wtv[12] | wtv[11] | wtv[10] | wtv[9] | wtv[8] | — |
| 18 | R | Wake-Up Timer Value 2 | wtv[7] | wtv[6] | wtv[5] | wtv[4] | wtv[3] | wtv[2] | wtv[1] | wtv[0] | — |
| 19 | R/W | Low-Duty Cycle Mode Duration | ldc[7] | ldc[6] | ldc[5] | ldc[4] | ldc[3] | ldc[2] | ldc[1] | ldc[0] | 00h |
| 1A | R/W | Low Battery Detector Threshold | Reserved | Reserved | Reserved | lbdt[4] | lbdt[3] | lbdt[2] | lbdt[1] | lbdt[0] | 14h |
| 1B | R | Battery Voltage Level | 0 | 0 | 0 | vbat[4] | vbat[3] | vbat[2] | vbat[1] | vbat[0] | — |
| 1C | R/W | IF Filter Bandwidth | dwn3_bypass | ndec[2] | ndec[1] | ndec[0] | filset[3] | filset[2] | filset[1] | filset[0] | 01h |
| 1D | R/W | AFC Loop Gearshift Override | afcbd | enafc | afcgearh[2] | afcgearh[1] | afcgearh[0] | 1p5 bypass | matap | ph0size | 40h |
| 1E | R/W | AFC Timing Control | swait_timer[1] | swait_timer[0] | shwait[2] | shwait[1] | shwait[0] | anwait[2] | anwait[1] | anwait[0] | 0Ah |
| 1F | R/W | Clock Recovery Gearshift Override | Reserved | Reserved | crfast[2] | crfast[1] | crfast[0] | crslow[2] | crslow[1] | crslow[0] | 03h |
| 20 | R/W | Clock Recovery Oversampling Ratio | rxosr[7] | rxosr[6] | rxosr[5] | rxosr[4] | rxosr[3] | rxosr[2] | rxosr[1] | rxosr[0] | 64h |
| 21 | R/W | Clock Recovery Offset 2 | rxosr[10] | rxosr[9] | rxosr[8] | stallctrl | ncoff[19] | ncoff[18] | ncoff[17] | ncoff[16] | 01h |
| 22 | R/W | Clock Recovery Offset 1 | ncoff[15] | ncoff[14] | ncoff[13] | ncoff[12] | ncoff[11] | ncoff[10] | ncoff[9] | ncoff[8] | 47h |

***Note:** Detailed register descriptions are available in “AN440: Si4430/31/32 Register Descriptions”.

## Table 23.9. EZRadioPRO Internal Register Descriptions* (Continued)

| Add | R/W | Function/Desc | Data | | | | | | | | POR Default |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| 23 | R/W | Clock Recovery Offset 0 | ncoff[7] | ncoff[6] | ncoff[5] | ncoff[4] | ncoff[3] | ncoff[2] | ncoff[1] | ncoff[0] | AEh |
| 24 | R/W | Clock Recovery Timing Loop Gain 1 | Reserved | Reserved | Reserved | rxncocomp | crgain2x | crgain[10] | crgain[9] | crgain[8] | 02h |
| 25 | R/W | Clock Recovery Timing Loop Gain 0 | crgain[7] | crgain[6] | crgain[5] | crgain[4] | crgain[3] | crgain[2] | crgain[1] | crgain[0] | 8Fh |
| 26 | R | Received Signal Strength Indicator | rssi[7] | rssi[6] | rssi[5] | rssi[4] | rssi[3] | rssi[2] | rssi[1] | rssi[0] | — |
| 27 | R/W | RSSI Threshold for Clear Channel Indicator | rssith[7] | rssith[6] | rssith[5] | rssith[4] | rssith[3] | rssith[2] | rssith[1] | rssith[0] | 1Eh |
| 28 | R | Antenna Diversity Register 1 | adrssi1[7] | adrssia[6] | adrssia[5] | adrssia[4] | adrssia[3] | adrssia[2] | adrssia[1] | adrssia[0] | — |
| 29 | R | Antenna Diversity Register 2 | adrssib[7] | adrssib[6] | adrssib[5] | adrssib[4] | adrssib[3] | adrssib[2] | adrssib[1] | adrssib[0] | — |
| 2A | R/W | AFC Limiter | Afclim[7] | Afclim[6] | Afclim[5] | Afclim[4] | Afclim[3] | Afclim[2] | Afclim[1] | Afclim[0] | 00h |
| 2B | R | AFC Correction Read | afc_corr[9] | afc_corr[8] | afc_corr[7] | afc_corr[6] | afc_corr[5] | afc_corr[4] | afc_corr[3] | afc_corr[2] | 00h |
| 2C | R/W | OOK Counter Value 1 | afc_corr[9] | afc_corr[9] | ookfrzen | peakdeten | madeten | ookcnt[10] | ookcnt[9] | ookcnt[8] | 18h |
| 2D | R/W | OOK Counter Value 2 | ookcnt[7] | ookcnt[6] | ookcnt[5] | ookcnt[4] | ookcnt[3] | ookcnt[2] | ookcnt[1] | ookcnt[0] | BCh |
| 2E | R/W | Slicer Peak Hold | Reserved | attack[2] | attack[1] | attack[0] | decay[3] | decay[2] | decay[1] | decay[0] | 26h |
| 2F | Reserved | | | | | | | | | | |
| 30 | R/W | Data Access Control | enpacrx | lsbfrst | crcdonly | skip2ph | enpactx | encrc | crc[1] | crc[0] | 8Dh |
| 31 | R | EzMAC status | 0 | rxcrc1 | pksrch | pkrx | pkvalid | crcerror | pktx | pksent | — |
| 32 | R/W | Header Control 1 | bcen[3:0] | | | | hdch[3:0] | | | | 0Ch |
| 33 | R/W | Header Control 2 | skipsyn | hdlen[2] | hdlen[1] | hdlen[0] | fixpklen | synclen[1] | synclen[0] | prealen[8] | 22h |
| 34 | R/W | Preamble Length | prealen[7] | prealen[6] | prealen[5] | prealen[4] | prealen[3] | prealen[2] | prealen[1] | prealen[0] | 08h |
| 35 | R/W | Preamble Detection Control | preath[4] | preath[3] | preath[2] | preath[1] | preath[0] | rssi_off[2] | rssi_off[1] | rssi_off[0] | 2Ah |
| 36 | R/W | Sync Word 3 | sync[31] | sync[30] | sync[29] | sync[28] | sync[27] | sync[26] | sync[25] | sync[24] | 2Dh |
| 37 | R/W | Sync Word 2 | sync[23] | sync[22] | sync[21] | sync[20] | sync[19] | sync[18] | sync[17] | sync[16] | D4h |
| 38 | R/W | Sync Word 1 | sync[15] | sync[14] | sync[13] | sync[12] | sync[11] | sync[10] | sync[9] | sync[8] | 00h |
| 39 | R/W | Sync Word 0 | sync[7] | sync[6] | sync[5] | sync[4] | sync[3] | sync[2] | sync[1] | sync[0] | 00h |
| 3A | R/W | Transmit Header 3 | txhd[31] | txhd[30] | txhd[29] | txhd[28] | txhd[27] | txhd[26] | txhd[25] | txhd[24] | 00h |
| 3B | R/W | Transmit Header 2 | txhd[23] | txhd[22] | txhd[21] | txhd[20] | txhd[19] | txhd[18] | txhd[17] | txhd[16] | 00h |
| 3C | R/W | Transmit Header 1 | txhd[15] | txhd[14] | txhd[13] | txhd[12] | txhd[11] | txhd[10] | txhd[9] | txhd[8] | 00h |
| 3D | R/W | Transmit Header 0 | txhd[7] | txhd[6] | txhd[5] | txhd[4] | txhd[3] | txhd[2] | txhd[1] | txhd[0] | 00h |
| 3E | R/W | Transmit Packet Length | pklen[7] | pklen[6] | pklen[5] | pklen[4] | pklen[3] | pklen[2] | pklen[1] | pklen[0] | 00h |
| 3F | R/W | Check Header 3 | chhd[31] | chhd[30] | chhd[29] | chhd[28] | chhd[27] | chhd[26] | chhd[25] | chhd[24] | 00h |
| 40 | R/W | Check Header 2 | chhd[23] | chhd[22] | chhd[21] | chhd[20] | chhd[19] | chhd[18] | chhd[17] | chhd[16] | 00h |
| 41 | R/W | Check Header 1 | chhd[15] | chhd[14] | chhd[13] | chhd[12] | chhd[11] | chhd[10] | chhd[9] | chhd[8] | 00h |
| 42 | R/W | Check Header 0 | chhd[7] | chhd[6] | chhd[5] | chhd[4] | chhd[3] | chhd[2] | chhd[1] | chhd[0] | 00h |
| 43 | R/W | Header Enable 3 | hden[31] | hden[30] | hden[29] | hden[28] | hden[27] | hden[26] | hden[25] | hden[24] | FFh |
| 44 | R/W | Header Enable 2 | hden[23] | hden[22] | hden[21] | hden[20] | hden[19] | hden[18] | hden[17] | hden[16] | FFh |
| 45 | R/W | Header Enable 1 | hden[15] | hden[14] | hden[13] | hden[12] | hden[11] | hden[10] | hden[9] | hden[8] | FFh |
| 46 | R/W | Header Enable 0 | hden[7] | hden[6] | hden[5] | hden[4] | hden[3] | hden[2] | hden[1] | hden[0] | FFh |
| 47 | R | Received Header 3 | rxhd[31] | rxhd[30] | rxhd[29] | rxhd[28] | rxhd[27] | rxhd[26] | rxhd[25] | rxhd[24] | — |
| 48 | R | Received Header 2 | rxhd[23] | rxhd[22] | rxhd[21] | rxhd[20] | rxhd[19] | rxhd[18] | rxhd[17] | rxhd[16] | — |
| 49 | R | Received Header 1 | rxhd[15] | rxhd[14] | rxhd[13] | rxhd[12] | rxhd[11] | rxhd[10] | rxhd[9] | rxhd[8] | — |
| 4A | R | Received Header 0 | rxhd[7] | rxhd[6] | rxhd[5] | rxhd[4] | rxhd[3] | rxhd[2] | rxhd[1] | rxhd[0] | — |

***Note:** Detailed register descriptions are available in “AN440: Si4430/31/32 Register Descriptions”.

**Table 23.9. EZRadioPRO Internal Register Descriptions* (Continued)**

| Add | R/W | Function/Desc | Data | | | | | | | | POR Default |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| 4B | R | Received Packet Length | rxplen[7] | rxplen[6] | rxplen[5] | rxplen[4] | rxplen[3] | rxplen[2] | rxplen[1] | rxplen[0] | — |
| 4C-4E | Reserved | | | | | | | | | | |
| 4F | R/W | ADC8 Control | Reserved | Reserved | adc8[5] | adc8[4] | adc8[3] | adc8[2] | adc8[1] | adc8[0] | 10h |
| 50-5F | Reserved | | | | | | | | | | |
| 60 | R/W | Channel Filter Coefficient Address | Inv_pre_th[3] | Inv_pre_th[2] | Inv_pre_th[1] | Inv_pre_th[0] | chfiladd[3] | chfiladd[2] | chfiladd[1] | chfiladd[0] | 00h |
| 61 | Reserved | | | | | | | | | | |
| 62 | R/W | Crystal Oscillator/ Control Test | pwst[2] | pwst[1] | pwst[0] | clkhyst | enbias2x | enamp2x | bufovr | enbuf | 24h |
| 63-6C | Reserved | | | | | | | | | | |
| 6D | R/W | TX Power | Reserved | Reserved | Reserved | Reserved | lna_sw | txpow[2] | txpow[1] | txpow[0] | 18h |
| 6E | R/W | TX Data Rate 1 | txdr[15] | txdr[14] | txdr[13] | txdr[12] | txdr[11] | txdr[10] | txdr[9] | txdr[8] | 0Ah |
| 6F | R/W | TX Data Rate 0 | txdr[7] | txdr[6] | txdr[5] | txdr[4] | txdr[3] | txdr[2] | txdr[1] | txdr[0] | 3Dh |
| 70 | R/W | Modulation Mode Control 1 | Reserved | Reserved | txdtrtscale | enphpwdn | manppol | enmaninv | enmanch | enwhite | 0Ch |
| 71 | R/W | Modulation Mode Control 2 | trclk[1] | trclk[0] | dtmod[1] | dtmod[0] | eninv | fd[8] | modtyp[1] | modtyp[0] | 00h |
| 72 | R/W | Frequency Deviation | fd[7] | fd[6] | fd[5] | fd[4] | fd[3] | fd[2] | fd[1] | fd[0] | 20h |
| 73 | R/W | Frequency Offset 1 | fo[7] | fo[6] | fo[5] | fo[4] | fo[3] | fo[2] | fo[1] | fo[0] | 00h |
| 74 | R/W | Frequency Offset 2 | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | fo[9] | fo[8] | 00h |
| 75 | R/W | Frequency Band Select | Reserved | sbsel | hbsel | fb[4] | fb[3] | fb[2] | fb[1] | fb[0] | 75h |
| 76 | R/W | Nominal Carrier Frequency 1 | fc[15] | fc[14] | fc[13] | fc[12] | fc[11] | fc[10] | fc[9] | fc[8] | BBh |
| 77 | R/W | Nominal Carrier Frequency 0 | fc[7] | fc[6] | fc[5] | fc[4] | fc[3] | fc[2] | fc[1] | fc[0] | 80h |
| 78 | Reserved | | | | | | | | | | |
| 79 | R/W | Frequency Hopping Channel Select | fhch[7] | fhch[6] | fhch[5] | fhch[4] | fhch[3] | fhch[2] | fhch[1] | fhch[0] | 00h |
| 7A | R/W | Frequency Hopping Step Size | fhs[7] | fhs[6] | fhs[5] | fhs[4] | fhs[3] | fhs[2] | fhs[1] | fhs[0] | 00h |
| 7B | Reserved | | | | | | | | | | |
| 7C | R/W | TX FIFO Control 1 | Reserved | Reserved | txafthr[5] | txafthr[4] | txafthr[3] | txafthr[2] | txafthr[1] | txafthr[0] | 37h |
| 7D | R/W | TX FIFO Control 2 | Reserved | Reserved | txaethr[5] | txaethr[4] | txaethr[3] | txaethr[2] | txaethr[1] | txaethr[0] | 04h |
| 7E | R/W | RX FIFO Control | Reserved | Reserved | rxafthr[5] | rxafthr[4] | rxafthr[3] | rxafthr[2] | rxafthr[1] | rxafthr[0] | 37h |
| 7F | R/W | FIFO Access | fifod[7] | fifod[6] | fifod[5] | fifod[4] | fifod[3] | fifod[2] | fifod[1] | fifod[0] | — |

***Note:** Detailed register descriptions are available in “AN440: Si4430/31/32 Register Descriptions”.

## 23.13. Required Changes to Default Register Values

The following register writes should be performed during device initialization.

Register modifications are required for operation in the frequency bands between 240–320 MHz and 480–640 MHz with a temperature above 60 °C. In these cases, write 03h to register 59h and 02h to register 5Ah.

## 24. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the $I^2C$ serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/20th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. The SMBus peripheral can be fully driven by software (i.e., software accepts/rejects slave addresses, and generates ACKs), or hardware slave address recognition and automatic ACK generation can be enabled to minimize software overhead. A block diagram of the SMBus peripheral and the associated SFRs is shown in Figure 24.1.

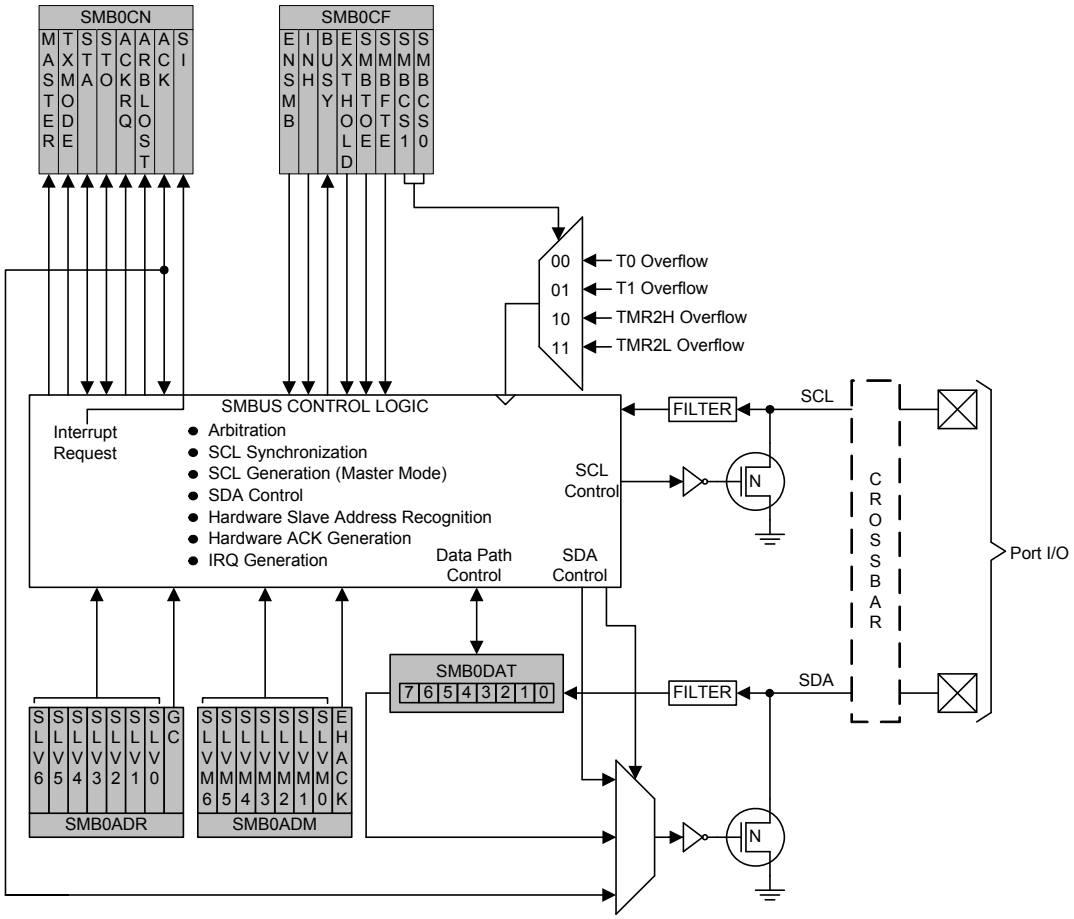



**Figure 24.1. SMBus Block Diagram**

## 24.1. Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

1. The $I^2C$-Bus and How to Use It (including specifications), Philips Semiconductor.
2. The $I^2C$-Bus Specification—Version 2.0, Philips Semiconductor.
3. System Management Bus Specification—Version 1.1, SBS Implementers Forum.

## 24.2. SMBus Configuration

Figure 24.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

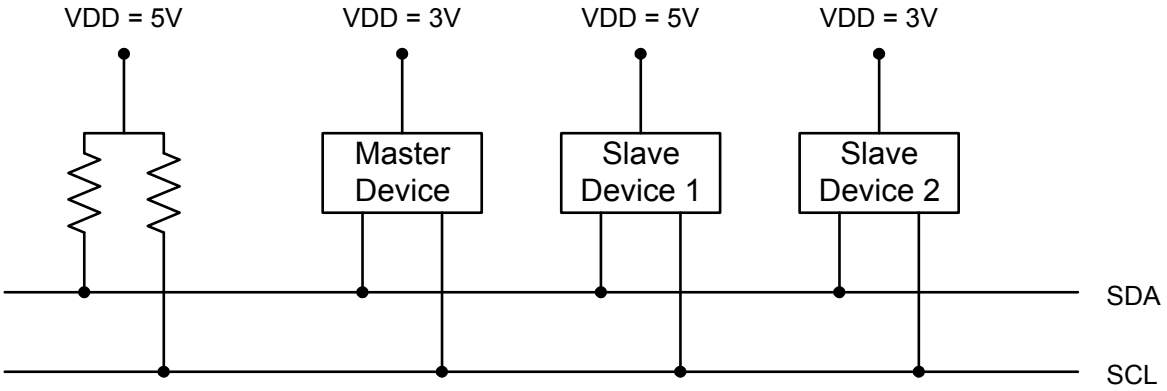



**Figure 24.2. Typical SMBus Configuration**

## 24.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. Note that it is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 24.3). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 24.3 illustrates a typical SMBus transaction.

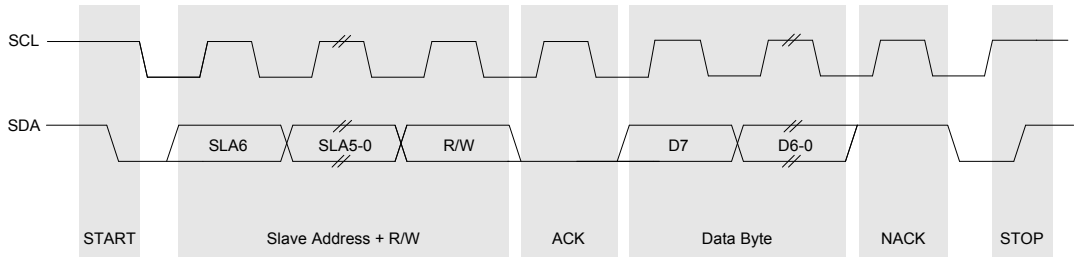



**Figure 24.3. SMBus Transaction**

### 24.3.1. Transmitter vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

### 24.3.2. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see Section “24.3.5. SCL High (SMBus Free) Timeout” on page 294). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

### 24.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to $I^2C$, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

### 24.3.4. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

When the SMBTOE bit in SMB0CF is set, Timer 3 is used to detect SCL low timeouts. Timer 3 is forced to reload when SCL is high, and allowed to count when SCL is low. With Timer 3 enabled and configured to overflow after 25 ms (and SMBTOE set), the Timer 3 interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

### 24.3.5. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMBFTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. Note that a clock source is required for free timeout detection, even in a slave-only implementation.

## 24.4. Using the SMBus

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgement is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e. sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e. receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgement is enabled, these interrupts are always generated after the ACK cycle. See Section 24.5 for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. The SMB0CN register is described in Section 24.4.2; Table 24.5 provides a quick SMB0CN decoding reference.

### 24.4.1. SMBus Configuration Register

The SMBus Configuration register (SMB0CF) is used to enable the SMBus Master and/or Slave modes, select the SMBus clock source, and select the SMBus timing and timeout options. When the ENSMB bit is set, the SMBus is enabled for all master and slave events. Slave events may be disabled by setting the INH bit. With slave events inhibited, the SMBus interface will still monitor the SCL and SDA pins; however, the interface will NACK all received addresses and will not generate any slave interrupts. When the INH bit is set, all slave events will be inhibited following the next START (interrupts will continue for the duration of the current transfer).

**Table 24.1. SMBus Clock Source Selection**

| SMBCS1 | SMBCS0 | SMBus Clock Source |
|---|---|---|
| 0 | 0 | Timer 0 Overflow |
| 0 | 1 | Timer 1 Overflow |
| 1 | 0 | Timer 2 High Byte Overflow |
| 1 | 1 | Timer 2 Low Byte Overflow |

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 24.1. Note that the selected clock source may be shared by other peripherals so long as the timer is left running at all times. For example, Timer 1 overflows may generate the SMBus and UART baud rates simultaneously. Timer configuration is covered in Section “27. Timers” on page 335.

$$T_{HighMin} = T_{LowMin} = \frac{1}{f_{ClockSourceOverflow}}$$

**Equation 24.1. Minimum SCL High and Low Times**

The selected clock source should be configured to establish the minimum SCL High and Low times as per Equation 24.1. When the interface is operating as a master (and SCL is not driven or extended by any other devices on the bus), the typical SMBus bit rate is approximated by Equation 24.2.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

**Equation 24.2. Typical SMBus Bit Rate**

Figure 24.4 shows the typical SCL generation described by Equation 24.2. Notice that $T_{HIGH}$ is typically twice as large as $T_{LOW}$. The actual SCL output may vary due to other devices on the bus (SCL may be extended low by slower slave devices, or driven low by contending master devices). The bit rate when operating as a master will never exceed the limits defined by equation Equation 24.1.

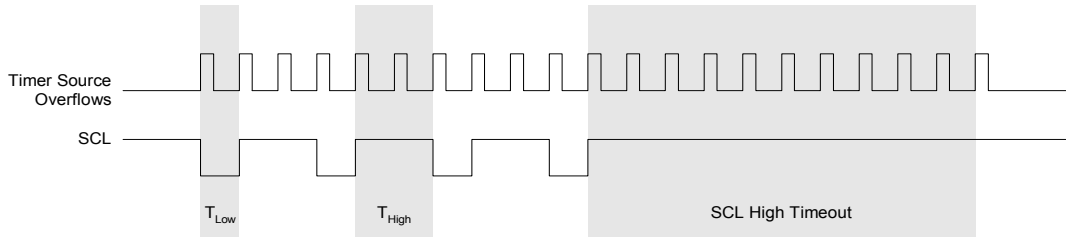



**Figure 24.4. Typical SMBus SCL Generation**

Setting the EXTHOLD bit extends the minimum setup and hold times for the SDA line. The minimum SDA setup time defines the absolute minimum time that SDA is stable before SCL transitions from low-to-high. The minimum SDA hold time defines the absolute minimum time that the current SDA value remains stable after SCL transitions from high-to-low. EXTHOLD should be set so that the minimum setup and hold times meet the SMBus Specification requirements of 250 ns and 300 ns, respectively. Table 24.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

**Table 24.2. Minimum SDA Setup and Hold Times**

| EXTHOLD | Minimum SDA Setup Time | Minimum SDA Hold Time |
|---|---|---|
| 0 | $T_{low}$ – 4 system clocks<br>or<br>1 system clock + s/w delay* | 3 system clocks |
| 1 | 11 system clocks | 12 system clocks |
| **Note:** Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgement, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero. | | |

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “24.3.4. SCL Low Timeout” on page 294). The SMBus interface will force Timer 3 to reload while SCL is high, and allow Timer 3 to count when SCL is low. The Timer 3 interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 24.4).

## SFR Definition 24.1. SMB0CF: SMBus Clock/Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ENSMB | INH | BUSY | EXTHOLD | SMBTOE | SMBFTE | SMBCS[1:0] | |
| Type | R/W | R/W | R | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC1

| Bit | Name | Function |
|---|---|---|
| 7 | ENSMB | **SMBus Enable.**<br>This bit enables the SMBus interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins. |
| 6 | INH | **SMBus Slave Inhibit.**<br>When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected. |
| 5 | BUSY | **SMBus Busy Indicator.**<br>This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed. |
| 4 | EXTHOLD | **SMBus Setup and Hold Time Extension Enable.**<br>This bit controls the SDA setup and hold times according to Table 24.2.<br>0: SDA Extended Setup and Hold Times disabled.<br>1: SDA Extended Setup and Hold Times enabled. |
| 3 | SMBTOE | **SMBus SCL Timeout Detection Enable.**<br>This bit enables SCL low timeout detection. If set to logic 1, the SMBus forces Timer 3 to reload while SCL is high and allows Timer 3 to count when SCL goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication. |
| 2 | SMBFTE | **SMBus Free Timeout Detection Enable.**<br>When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods. |
| 1:0 | SMBCS[1:0] | **SMBus Clock Source Selection.**<br>These two bits select the SMBus clock source, which is used to generate the SMBus bit rate. The selected device should be configured according to Equation 24.1.<br>00: Timer 0 Overflow<br>01: Timer 1 Overflow<br>10:Timer 2 High Byte Overflow<br>11: Timer 2 Low Byte Overflow |

### 24.4.2. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information (see SFR Definition 24.2). The higher four bits of SMB0CN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost; see Table 24.3 for more details.

**Important Note About the SI Bit:** The SMBus interface is stalled while SI is set; thus SCL is held low, and the bus is stalled until software clears SI.

#### 24.4.2.1. Software ACK Generation

When the EHACK bit in register SMB0ADM is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during the last ACK cycle. ACKRQ is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQ is set, software should write the desired outgoing value to the ACK bit before clearing SI. A NACK will be generated if software does not write the ACK bit before clearing SI. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however SCL will remain low until SI is cleared. If a received slave address is not acknowledged, further slave events will be ignored until the next START is detected.

#### 24.4.2.2. Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. More detail about automatic slave address recognition can be found in Section 24.4.3. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 24.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 24.5 for SMBus status decoding using the SMB0CN register.

Refer to “Limitations for Hardware Acknowledge Feature” on page 303 when using hardware ACK generation.

## SFR Definition 24.2. SMB0CN: SMBus Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | MASTER | TXMODE | STA | STO | ACKRQ | ARBLOST | ACK | SI |
| Type | R | R | R/W | R/W | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC0; Bit-Addressable

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7 | MASTER | **SMBus Master/Slave Indicator.** This read-only bit indicates when the SMBus is operating as a master. | 0: SMBus operating in slave mode.<br>1: SMBus operating in master mode. | N/A |
| 6 | TXMODE | **SMBus Transmit Mode Indicator.** This read-only bit indicates when the SMBus is operating as a transmitter. | 0: SMBus in Receiver Mode.<br>1: SMBus in Transmitter Mode. | N/A |
| 5 | STA | **SMBus Start Flag.** | 0: No Start or repeated Start detected.<br>1: Start or repeated Start detected. | 0: No Start generated.<br>1: When Configured as a Master, initiates a START or repeated START. |
| 4 | STO | **SMBus Stop Flag.** | 0: No Stop condition detected.<br>1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode). | 0: No STOP condition is transmitted.<br>1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle.<br>Cleared by Hardware. |
| 3 | ACKRQ | **SMBus Acknowledge Request.** | 0: No Ack requested<br>1: ACK requested | N/A |
| 2 | ARBLOST | **SMBus Arbitration Lost Indicator.** | 0: No arbitration error.<br>1: Arbitration Lost | N/A |
| 1 | ACK | **SMBus Acknowledge.** | 0: NACK received.<br>1: ACK received. | 0: Send NACK<br>1: Send ACK |
| 0 | SI | **SMBus Interrupt Flag.**<br>This bit is set by hardware under the conditions listed in Table 15.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled. | **0: No interrupt pending**<br>1: Interrupt Pending | 0: Clear interrupt, and initiate next state machine event.<br>1: Force interrupt. |

**Table 24.3. Sources for Hardware Changes to SMB0CN**

| Bit | Set by Hardware When: | Cleared by Hardware When: |
|---|---|---|
| MASTER | ■ A START is generated. | ■ A STOP is generated.<br>■ Arbitration is lost. |
| TXMODE | ■ START is generated.<br>■ SMB0DAT is written before the start of an SMBus frame. | ■ A START is detected.<br>■ Arbitration is lost.<br>■ SMB0DAT is not written before the start of an SMBus frame. |
| STA | ■ A START followed by an address byte is received. | ■ Must be cleared by software. |
| STO | ■ A STOP is detected while addressed as a slave.<br>■ Arbitration is lost due to a detected STOP. | ■ A pending STOP is generated. |
| ACKRQ | ■ A byte has been received and an ACK response value is needed (only when hardware ACK is not enabled). | ■ After each ACK cycle. |
| ARBLOST | ■ A repeated START is detected as a MASTER when STA is low (unwanted repeated START).<br>■ SCL is sensed low while attempting to generate a STOP or repeated START condition.<br>■ SDA is sensed low while transmitting a 1 (excluding ACK bits). | ■ Each time SI is cleared. |
| ACK | ■ The incoming ACK value is low (ACKNOWLEDGE). | ■ The incoming ACK value is high (NOT ACKNOWLEDGE). |
| SI | ■ A START has been generated.<br>■ Lost arbitration.<br>■ A byte has been transmitted and an ACK/NACK received.<br>■ A byte has been received.<br>■ A START or repeated START followed by a slave address + R/W has been received.<br>■ A STOP has been received. | ■ Must be cleared by software. |

### 24.4.3. Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave). More detail on automatic hardware ACK generation can be found in Section 24.4.2.2.

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register (SFR Definition 24.3) and the SMBus Slave Address Mask register (SFR Definition 24.4). A single address or range of addresses (including the General Call Address 0x00) can be specified using these two registers. The most-significant seven bits of the two registers are used to define which addresses will be ACKed. A 1 in bit positions of the slave address mask SLVM[6:0] enable a comparison between the received slave address and the hardware's slave address SLV[6:0] for those bits. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address. Additionally, if the GC bit in register SMB0ADR is set to 1, hardware will recognize the General Call Address (0x00). Table 24.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions. Refer to "Limitations for Hardware Acknowledge Feature" on page 303 when using hardware slave address recognition.

**Table 24.4. Hardware Address Recognition Examples (EHACK = 1)**

| Hardware Slave Address SLV[6:0] | Slave Address Mask SLVM[6:0] | GC bit | Slave Addresses Recognized by Hardware |
|---|---|---|---|
| 0x34 | 0x7F | 0 | 0x34 |
| 0x34 | 0x7F | 1 | 0x34, 0x00 (General Call) |
| 0x34 | 0x7E | 0 | 0x34, 0x35 |
| 0x34 | 0x7E | 1 | 0x34, 0x35, 0x00 (General Call) |
| 0x70 | 0x73 | 0 | 0x70, 0x74, 0x78, 0x7C |

## SFR Definition 24.3. SMB0ADR: SMBus Slave Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SLV[6:0] | | | | | | | GC |
| Type | R/W | | | | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xF4

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLV[6:0] | **SMBus Hardware Slave Address.**<br>Defines the SMBus Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM[6:0] are checked against the incoming address. This allows multiple addresses to be recognized. |
| 0 | GC | **General Call Address Enable.**<br>When hardware address recognition is enabled (EHACK = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware.<br>0: General Call Address is ignored.<br>1: General Call Address is recognized. |

## SFR Definition 24.4. SMB0ADM: SMBus Slave Address Mask

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SLVM[6:0] | | | | | | | EHACK |
| Type | R/W | | | | | | | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

SFR Page = 0x0; SFR Address = 0xF5

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLVM[6:0] | **SMBus Slave Address Mask.**<br>Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM[6:0] enables comparisons with the corresponding bit in SLV[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address). |
| 0 | EHACK | **Hardware Acknowledge Enable.**<br>Enables hardware acknowledgement of slave address and received data bytes.<br>0: Firmware must manually acknowledge all incoming address and data bytes.<br>1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled. |

### 24.4.4. Limitations for Hardware Acknowledge Feature

In some system management bus (SMBus) configurations, the Hardware Acknowledge mechanism of the SMBus peripheral can cause incorrect or undesired behavior. The Hardware Acknowledge mechanism is enabled when the EHACK bit (SMB0ADM.0) is set to logic 1.

The configurations to which these limitations do not apply are as follows:

a. All SMBus configurations when Hardware Acknowledge is disabled.
b. All **single-master/single-slave** SMBus configurations when Hardware Acknowledge is enabled and the MCU is operating as a **master or slave**.
c. All **multi-master/single-slave** SMBus configurations when Hardware Acknowledge is enabled and the MCU is operating as a **slave**.
d. All **single-master/multi-slave** SMBus configurations when Hardware Acknowledge is enabled and the MCU is operating as a **master**.

These limitations only apply to the following configurations:

a. All **multi-slave** SMBus configurations when Hardware Acknowledge is enabled and the MCU is operating as a **slave**.
b. All **multi-master** SMBus configurations when Hardware Acknowledge is enabled and the MCU is operating as a **master**.

The following issues are present when operating as a **slave** in a **multi-slave** SMBus configuration:

a. When Hardware Acknowledge is enabled and SDA setup and hold times are not extended (EXTHOLD = 0 in the SMB0CF register), the SMBus hardware will always generate an SMBus interrupt following the ACK/NACK cycle of any slave address transmission on the bus, whether or not the address matches the conditions of SMB0ADR and SMB0MASK. The expected behavior is that an interrupt is only generated when the address matches.
b. When Hardware Acknowledge is enabled and SDA setup and hold times are extended (EXTHOLD = 1 in the SMB0CF register), the SMBus hardware will only generate an SMBus interrupt as expected when the slave address transmission on the bus matches the conditions of SMB0ADR and SMB0MASK. However, in this mode, the Start bit (STA) will be incorrectly cleared on reception of a slave address before software vectors to the interrupt service routine.
c. When Hardware Acknowledge is enabled and the ACK bit (SMB0CN.1) is set to 1, an unaddressed slave may cause interference on the SMBus by driving SDA low during an ACK cycle. The ACK bit of the unaddressed slave may be set to 1 if any device on the bus generates an ACK.

**Impact**:

a. Once the CPU enters the interrupt service routine, SCL will be asserted low until SI is cleared, causing the clock to be stretched when the MCU is not being addressed. This may limit the maximum speed of the SMBus if the master supports SCL clock stretching. Incompliant SMBus masters that do not support SCL clock stretching will not recognize that the clock is being stretched. If the CPU issues a write to SMB0DAT, it will have no effect on the bus. No data collisions will occur.
b. Once the hardware has matched an address and entered the interrupt service routine, the firmware will not be able to use the Start bit to distinguish between the reception of an address byte versus the reception of a data byte. However, the hardware will still correctly acknowledge the address byte (SLA+R/W).
c. The SMBus master and the addressed slave are prevented from generating a NACK by the unaddressed slave because it is holding SDA low during the ACK cycle. There is a potential for the SMBus to lock up.

**Workarounds**:

a. The SMBus interrupt service routine should verify an address when it is received and clear SI as soon as possible if the address does not match to minimize clock stretching. To prevent clock stretching when not being addressed, enable setup and hold time extensions (EXTHOLD = 1).

b. Detection of Initial Start:

   To distinguish between the reception of an address byte **at the beginning** of a transfer versus the reception of a data byte when setup and hold time extensions are enabled (EXTHOLD = 1), software should maintain a status bit to determine whether it is currently inside or outside a transfer. Once hardware detects a matching slave address and interrupts the MCU, software should assume a start condition and set the software bit to indicate that it is currently inside a transfer. A transfer ends any time the STO bit is set or on an error condition (e.g., SCL Low Timeout).

   Detection of Repeated Start:

   To detect the reception of an address byte **in the middle** of a transfer when setup and hold time extensions are enabled (EXTHOLD = 1), disable setup and hold time extensions (EXTHOLD = 0) upon entry into a transfer and re-enable setup and hold time extensions (EXHOLD = 1) at the end of a transfer.

c. Schedule a timer interrupt to clear the ACK bit at an interval shorter than 7 bit periods when the slave is not being addressed. For example, on a 400 kHz SMBus, the ACK bit should be cleared every 17.5 µs (or at 1/7 the bus frequency, 57 kHz). As soon as a matching slave address is detected (a transfer is started), the timer which clears the ACK bit should be stopped and its interrupt flag cleared. The timer should be re-started once a stop or error condition is detected (the transfer has ended).

A code example demonstrating these workarounds can be found in the SMBus examples folder with the following default location:

C:\SiLabs\MCU\Examples\C8051F93x_92x\SMBus\F93x_SMBus_Slave_Multibyte_HWACK.c

The SMBus examples folder, along with examples for many additional peripherals, is created when the Silicon Laboratories IDE is installed. The latest version of the IDE may be downloaded from the software downloads page www.silabs.com/MCUDownloads on the Silicon Laboratories website.

The following issue is present when operating as a **master** in a **multi-master** SMBus configuration:

If the SMBus master loses arbitration in a multi-master system, it may cause interference on the SMBus by driving SDA low during the ACK cycle of transfers which it is not participating. This will occur regardless of the state of the ACK bit (SMB0CN.1).

**Impact**:

The SMBus master and slave participating in the transfer are prevented from generating a NACK by the MCU because it is holding SDA low during the ACK cycle. There is a potential for the SMBus to lock up.

**Workaround**:

Disable Hardware Acknowledge (EHACK = 0) when the MCU is operating as a master in a multi-master SMBus configuration.

### 24.4.5. Data Register

The SMBus Data register SMB0DAT holds a byte of serial data to be transmitted or one that has just been received. Software may safely read or write to the data register when the SI flag is set. Software should not attempt to access the SMB0DAT register when the SMBus is enabled and the SI flag is cleared to logic 0, as the interface may be in the process of shifting a byte of data into or out of the register.

Data in SMB0DAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMB0DAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMB0DAT always contains the last data byte present on the bus. In the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data or address in SMB0DAT.

## SFR Definition 24.5. SMB0DAT: SMBus Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SMB0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC2

| Bit | Name | Function |
|---|---|---|
| 7:0 | SMB0DAT[7:0] | **SMBus Data.**<br>The SMB0DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can read from or write to this register whenever the SI serial interrupt flag (SMB0CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register. |

## 24.5. SMBus Transfer Modes

The SMBus interface may be configured to operate as master and/or slave. At any particular time, it will be operating in one of the following four modes: Master Transmitter, Master Receiver, Slave Transmitter, or Slave Receiver. The SMBus interface enters Master Mode any time a START is generated, and remains in Master Mode until it loses an arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames. Note that the position of the ACK interrupt when operating as a receiver depends on whether hardware ACK generation is enabled. As a receiver, the interrupt for an ACK occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled. As a transmitter, interrupts occur **after** the ACK, regardless of whether hardware ACK generation is enabled or not.

### 24.5.1. Write Sequence (Master)

During a write sequence, an SMBus master writes data to a slave device. The master in this transfer will be a transmitter during the address byte, and a transmitter during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 0 (WRITE). The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. The transfer is ended when the STO bit is set and a STOP is generated. Note that the interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 24.5 shows a typical master write sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that all of the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

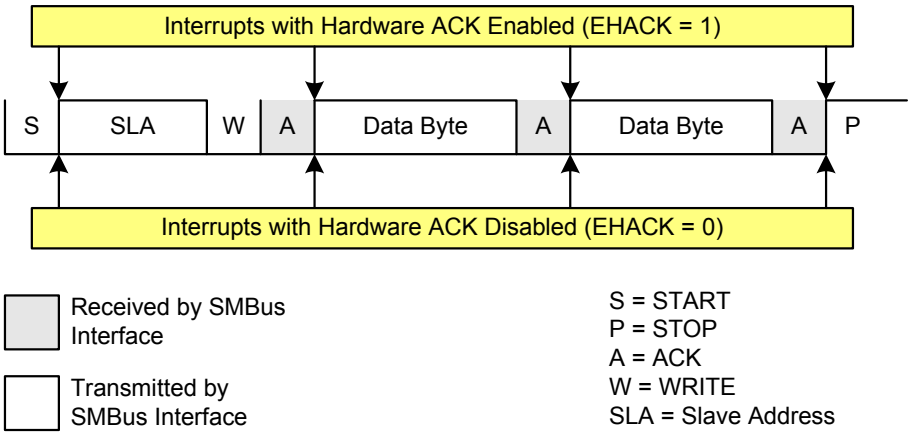



**Figure 24.5. Typical Master Write Sequence**

### 24.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

Writing a 1 to the ACK bit generates an ACK; writing a 0 generates a NACK. Software should write a 0 to the ACK bit for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 24.6 shows a typical master read sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the 'data byte transferred' interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

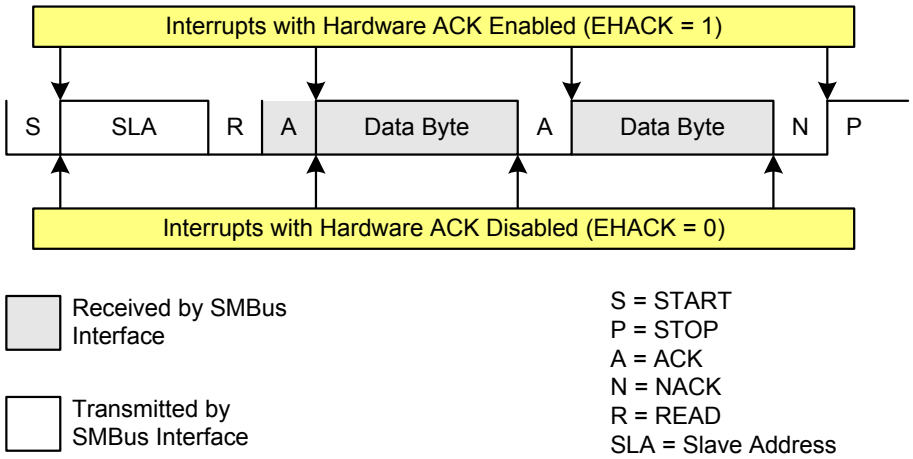



**Figure 24.6. Typical Master Read Sequence**

### 24.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 24.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice

that the “data byte transferred” interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

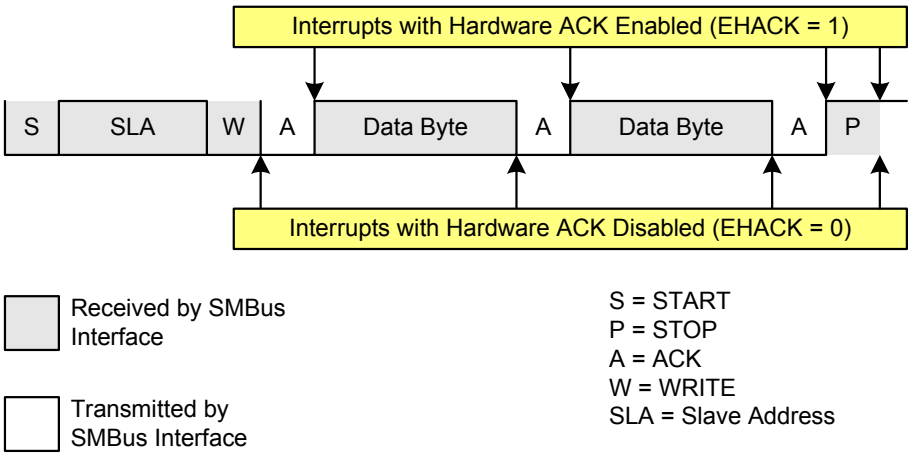



**Figure 24.7. Typical Slave Write Sequence**

### 24.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters Slave Transmitter Mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (Note: an error condition may be generated if SMB0DAT is written following a received NACK while in Slave Transmitter Mode). The interface exits Slave Transmitter Mode after receiving a STOP. Note that the interface will switch to Slave Receiver Mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 24.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

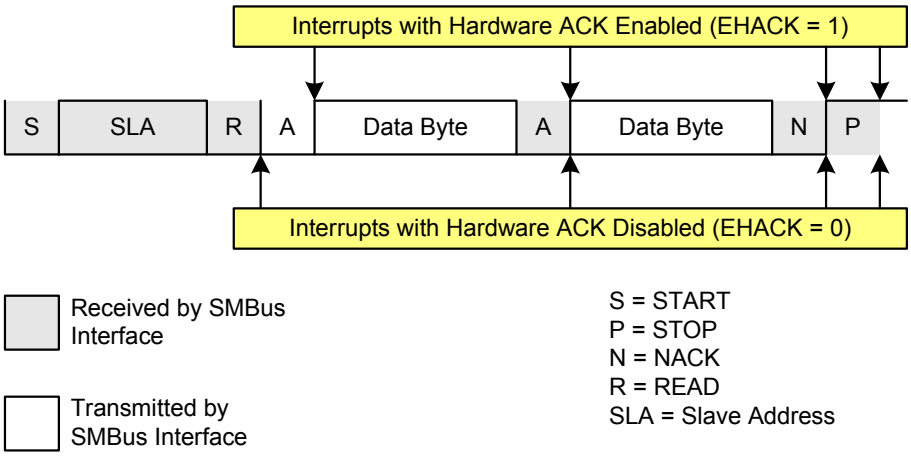



**Figure 24.8. Typical Slave Read Sequence**

## 24.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. The appropriate actions to take in response to an SMBus event depend on whether hardware slave address recognition and ACK generation is enabled or disabled. Table 24.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 24.6 describes the typical actions when hardware slave address recognition and ACK generation is enabled. In the tables, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

**Table 24.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Master Transmitter | 1110 | 0 | 0 | X | A master START was generated. | Load slave address + R/W into SMB0DAT. | 0 | 0 | X | 1100 |
| | 1100 | 0 | 0 | 0 | A master data or address byte was transmitted; NACK received. | Set STA to restart transfer. | 1 | 0 | X | 1110 |
| | | | | | | Abort transfer. | 0 | 1 | X | - |
| | | 0 | 0 | 1 | A master data or address byte was transmitted; ACK received. | Load next data byte into SMB0DAT. | 0 | 0 | X | 1100 |
| | | | | | | End transfer with STOP. | 0 | 1 | X | - |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X | - |
| | | | | | | Send repeated START. | 1 | 0 | X | 1110 |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). | 0 | 0 | X | 1000 |
| Master Receiver | 1000 | 1 | 0 | X | A master data byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 | 1000 |
| | | | | | | Send NACK to indicate last byte, and send STOP. | 0 | 1 | 0 | - |
| | | | | | | Send NACK to indicate last byte, and send STOP followed by START. | 1 | 1 | 0 | 1110 |
| | | | | | | Send ACK followed by repeated START. | 1 | 0 | 1 | 1110 |
| | | | | | | Send NACK to indicate last byte, and send repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 1 | 1100 |
| | | | | | | Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 0 | 1100 |

**Table 24.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) (Continued)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Slave Transmitter | 0100 | 0 | 0 | 0 | A slave byte was transmitted; NACK received. | No action required (expecting STOP condition). | 0 | 0 | X | 0001 |
| | | 0 | 0 | 1 | A slave byte was transmitted; ACK received. | Load SMB0DAT with next data byte to transmit. | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | A Slave byte was transmitted; error detected. | No action required (expecting Master to end transfer). | 0 | 0 | X | 0001 |
| | 0101 | 0 | X | X | An illegal STOP or bus error was detected while a Slave Transmission was in progress. | Clear STO. | 0 | 0 | X | - |
| Slave Receiver | 0010 | 1 | 0 | X | A slave address + R/W was received; ACK requested. | If Write, Acknowledge received address | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte; ACK received address | 0 | 0 | 1 | 0100 |
| | | | | | | NACK received address. | 0 | 0 | 0 | - |
| | | 1 | 1 | X | Lost arbitration as master; slave address + R/W received; ACK requested. | If Write, Acknowledge received address | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte; ACK received address | 0 | 0 | 1 | 0100 |
| | | | | | | NACK received address. | 0 | 0 | 0 | - |
| | | | | | | Reschedule failed transfer; NACK received address. | 1 | 0 | 0 | 1110 |
| | 0001 | 0 | 0 | X | A STOP was detected while addressed as a Slave Transmitter or Slave Receiver. | Clear STO. | 0 | 0 | X | - |
| | | 1 | 1 | X | Lost arbitration while attempting a STOP. | No action required (transfer complete/aborted). | 0 | 0 | 0 | - |
| | 0000 | 1 | 0 | X | A slave byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 | 0000 |
| | | | | | | NACK received byte. | 0 | 0 | 0 | - |
| Bus Error Condition | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0001 | 0 | 1 | X | Lost arbitration due to a detected STOP. | Abort failed transfer. | 0 | 0 | X | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0000 | 1 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | 0 | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | 0 | 1110 |

**Table 24.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Master Transmitter | 1110 | 0 | 0 | X | A master START was generated. | Load slave address + R/W into SMB0DAT. | 0 | 0 | X | 1100 |
| | 1100 | 0 | 0 | 0 | A master data or address byte was transmitted; NACK received. | Set STA to restart transfer. | 1 | 0 | X | 1110 |
| | | | | | | Abort transfer. | 0 | 1 | X | - |
| | | 0 | 0 | 1 | A master data or address byte was transmitted; ACK received. | Load next data byte into SMB0DAT. | 0 | 0 | X | 1100 |
| | | | | | | End transfer with STOP. | 0 | 1 | X | - |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X | - |
| | | | | | | Send repeated START. | 1 | 0 | X | 1110 |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). Set ACK for initial data byte. | 0 | 0 | 1 | 1000 |
| Master Receiver | 1000 | 0 | 0 | 1 | A master data byte was received; ACK sent. | Set ACK for next data byte; Read SMB0DAT. | 0 | 0 | 1 | 1000 |
| | | | | | | Set NACK to indicate next data byte as the last data byte; Read SMB0DAT. | 0 | 0 | 0 | 1000 |
| | | | | | | Initiate repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | X | 1100 |
| | | 0 | 0 | 0 | A master data byte was received; NACK sent (last byte). | Read SMB0DAT; send STOP. | 0 | 1 | 0 | - |
| | | | | | | Read SMB0DAT; Send STOP followed by START. | 1 | 1 | 0 | 1110 |
| | | | | | | Initiate repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | X | 1100 |

**Table 24.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) (Continued)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Slave Transmitter | 0100 | 0 | 0 | 0 | A slave byte was transmitted; NACK received. | No action required (expecting STOP condition). | 0 | 0 | X | 0001 |
| | | 0 | 0 | 1 | A slave byte was transmitted; ACK received. | Load SMB0DAT with next data byte to transmit. | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | A Slave byte was transmitted; error detected. | No action required (expecting Master to end transfer). | 0 | 0 | X | 0001 |
| | 0101 | 0 | X | X | An illegal STOP or bus error was detected while a Slave Transmission was in progress. | Clear STO. | 0 | 0 | X | - |
| Slave Receiver | 0010 | 0 | 0 | X | A slave address + R/W was received; ACK sent. | If Write, Set ACK for first data byte. | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | Lost arbitration as master; slave address + R/W received; ACK sent. | If Write, Set ACK for first data byte. | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte | 0 | 0 | X | 0100 |
| | | | | | | Reschedule failed transfer | 1 | 0 | X | 1110 |
| | 0001 | 0 | 0 | X | A STOP was detected while addressed as a Slave Transmitter or Slave Receiver. | Clear STO. | 0 | 0 | X | - |
| | | 0 | 1 | X | Lost arbitration while attempting a STOP. | No action required (transfer complete/aborted). | 0 | 0 | 0 | - |
| | 0000 | 0 | 0 | X | A slave byte was received. | Set ACK for next data byte; Read SMB0DAT. | 0 | 0 | 1 | 0000 |
| | | | | | | Set NACK for next data byte; Read SMB0DAT. | 0 | 0 | 0 | 0000 |
| Bus Error Condition | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0001 | 0 | 1 | X | Lost arbitration due to a detected STOP. | Abort failed transfer. | 0 | 0 | X | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0000 | 0 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | X | - |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |

## 25. UART0

UART0 is an asynchronous, full duplex serial port offering modes 1 and 3 of the standard 8051 UART. Enhanced baud rate support allows a wide range of clock sources to generate standard baud rates (details in Section “25.1. Enhanced Baud Rate Generation” on page 315). Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART0 has two associated SFRs: Serial Control Register 0 (SCON0) and Serial Data Buffer 0 (SBUF0). The single SBUF0 location provides access to both transmit and receive registers. **Writes to SBUF0 always access the Transmit register. Reads of SBUF0 always access the buffered Receive register; it is not possible to read data from the Transmit register.**

With UART0 interrupts enabled, an interrupt is generated each time a transmit is completed (TI0 is set in SCON0), or a data byte has been received (RI0 is set in SCON0). The UART0 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART0 interrupt (transmit complete or receive complete).

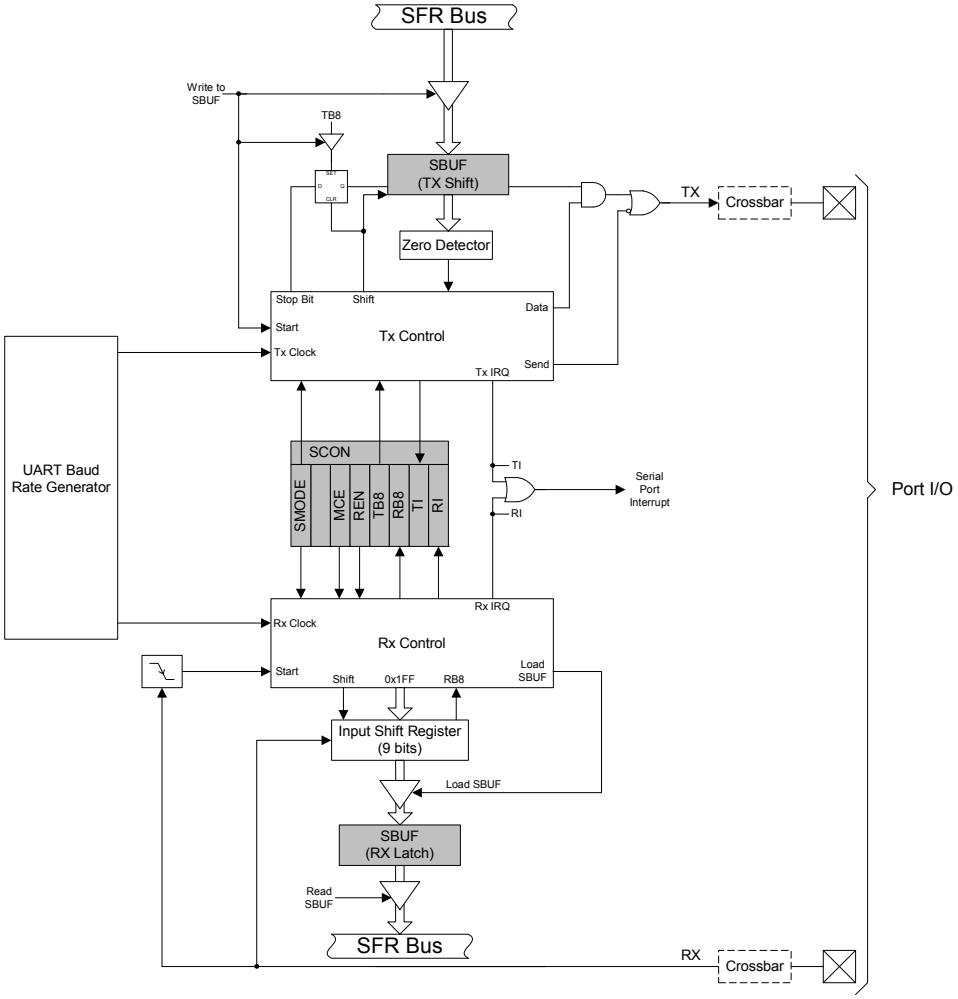



**Figure 25.1. UART0 Block Diagram**

## 25.1. Enhanced Baud Rate Generation

The UART0 baud rate is generated by Timer 1 in 8-bit auto-reload mode. The TX clock is generated by TL1; the RX clock is generated by a copy of TL1 (shown as RX Timer in Figure 25.2), which is not user-accessible. Both TX and RX Timer overflows are divided by two to generate the TX and RX baud rates. The RX Timer runs when Timer 1 is enabled, and uses the same reload value (TH1). However, an RX Timer reload is forced when a START condition is detected on the RX pin. This allows a receive to begin any time a START is detected, independent of the TX Timer state.

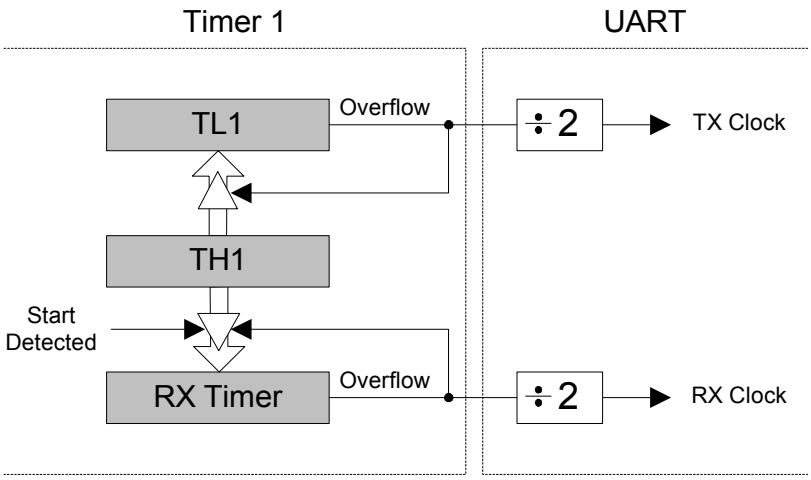



**Figure 25.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see Section "27.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload" on page 338). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Note that Timer 1 may be clocked by one of six sources: SYSCLK, SYSCLK / 4, SYSCLK / 12, SYSCLK / 48, the external oscillator clock / 8, or an external input T1. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 25.1-A and Equation 25.1-B.

A) $$\text{UartBaudRate} = \frac{1}{2} \times \text{T1_Overflow_Rate}$$

B) $$\text{T1_Overflow_Rate} = \frac{\text{T1}_{\text{CLK}}}{256 - \text{TH1}}$$

**Equation 25.1. UART0 Baud Rate**

Where $T1_{CLK}$ is the frequency of the clock supplied to Timer 1, and *T1H* is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in Section "27.1. Timer 0 and Timer 1" on page 337. A quick reference for typical baud rates and system clock frequencies is given in Table 25.1 through Table 25.2. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

## 25.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit (SCON0.7). Typical UART connection options are shown below.

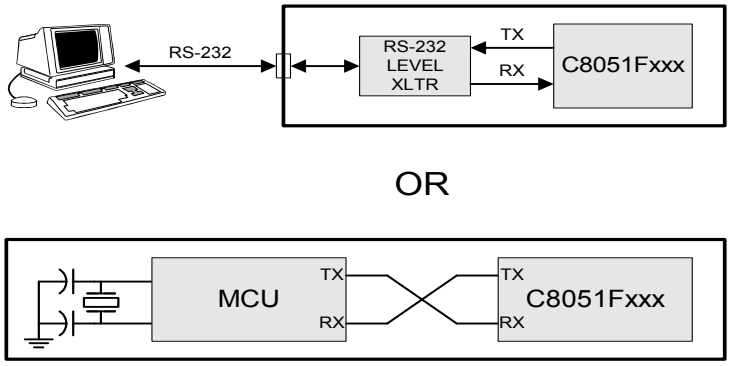



**Figure 25.3. UART Interconnect Diagram**

### 25.2.1. 8-Bit UART

8-Bit UART mode uses a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit. Data are transmitted LSB first from the TX0 pin and received at the RX0 pin. On receive, the eight data bits are stored in SBUF0 and the stop bit goes into RB80 (SCON0.2).

Data transmission begins when software writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: RI0 must be logic 0, and if MCE0 is logic 1, the stop bit must be logic 1. In the event of a receive data overrun, the first received 8 bits are latched into the SBUF0 receive register and the following overrun data bits are lost.

If these conditions are met, the eight bits of data is stored in SBUF0, the stop bit is stored in RB80 and the RI0 flag is set. If these conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set. An interrupt will occur if enabled when either TI0 or RI0 is set.

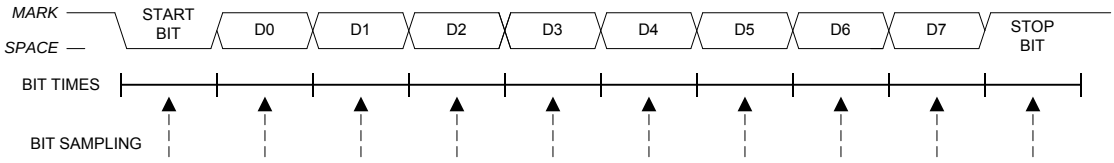



**Figure 25.4. 8-Bit UART Timing Diagram**

### 25.2.2. 9-Bit UART

9-bit UART mode uses a total of eleven bits per data byte: a start bit, 8 data bits (LSB first), a programmable ninth data bit, and a stop bit. The state of the ninth transmit data bit is determined by the value in TB80 (SCON0.3), which is assigned by user software. It can be assigned the value of the parity flag (bit P in register PSW) for error detection, or used in multiprocessor communications. On receive, the ninth data bit goes into RB80 (SCON0.2) and the stop bit is ignored.

Data transmission begins when an instruction writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: (1) RI0 must be logic 0, and (2) if MCE0 is logic 1, the 9th bit must be logic 1 (when MCE0 is logic 0, the state of the ninth data bit is unimportant). If these conditions are met, the eight bits of data are stored in SBUF0, the ninth bit is stored in RB80, and the RI0 flag is set to 1. If the above conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set to 1. A UART0 interrupt will occur if enabled when either TI0 or RI0 is set to 1.

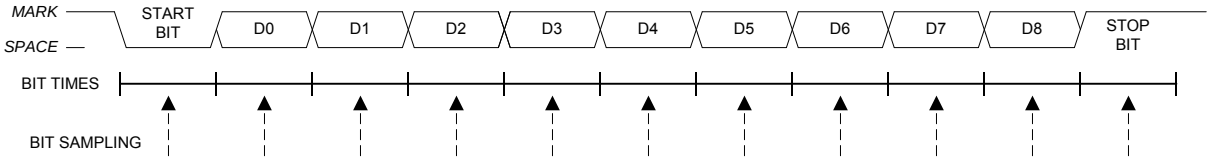



**Figure 25.5. 9-Bit UART Timing Diagram**

## 25.3. Multiprocessor Communications

9-Bit UART mode supports multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic 1; in a data byte, the ninth bit is always set to logic 0.

Setting the MCE0 bit (SCON0.5) of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic 1 (RB80 = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its MCE0 bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE0 bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE0 bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

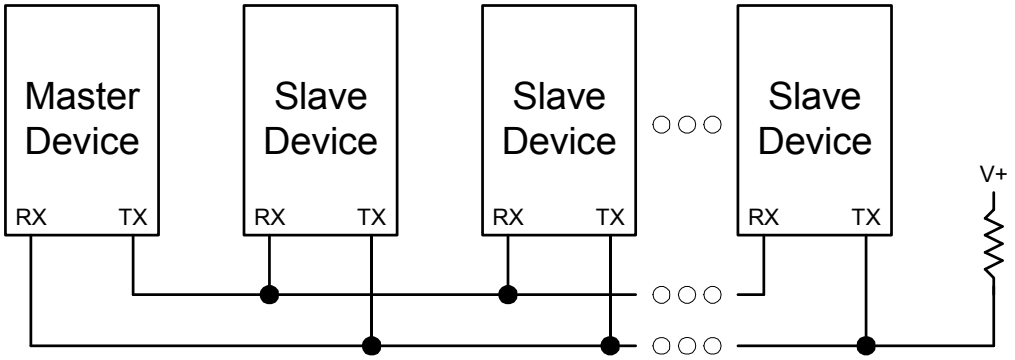



**Figure 25.6. UART Multi-Processor Mode Interconnect Diagram**

## SFR Definition 25.1. SCON0: Serial Port 0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | S0MODE | | MCE0 | REN0 | TB80 | RB80 | TI0 | RI0 |
| Type | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x98; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | S0MODE | **Serial Port 0 Operation Mode.**<br>Selects the UART0 Operation Mode.<br>0: 8-bit UART with Variable Baud Rate.<br>1: 9-bit UART with Variable Baud Rate. |
| 6 | Unused | Read = 1b. Write = Don't Care. |
| 5 | MCE0 | **Multiprocessor Communication Enable.**<br>**For Mode 0 (8-bit UART): Checks for valid stop bit.**<br>0: Logic level of stop bit is ignored.<br>1: RI0 will only be activated if stop bit is logic level 1.<br>**For Mode 1 (9-bit UART): Multiprocessor Communications Enable.**<br>0: Logic level of ninth bit is ignored.<br>1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1. |
| 4 | REN0 | **Receive Enable.**<br>0: UART0 reception disabled.<br>1: UART0 reception enabled. |
| 3 | TB80 | **Ninth Transmission Bit.**<br>The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0). |
| 2 | RB80 | **Ninth Receive Bit.**<br>RB80 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1. |
| 1 | TI0 | **Transmit Interrupt Flag.**<br>Set by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software. |
| 0 | RI0 | **Receive Interrupt Flag.**<br>Set to 1 by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). When the UART0 interrupt is enabled, setting this bit to 1 causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software. |

## SFR Definition 25.2. SBUF0: Serial (UART0) Port Data Buffer

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SBUF0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x99

| Bit | Name | Function |
|---|---|---|
| 7:0 | SBUF0 | **Serial Data Buffer Bits 7:0 (MSB–LSB)**<br>This SFR accesses two registers; a transmit shift register and a receive latch register. When data is written to SBUF0, it goes to the transmit shift register and is held for serial transmission. Writing a byte to SBUF0 initiates the transmission. A read of SBUF0 returns the contents of the receive latch. |

**Table 25.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator**

| | Frequency: 24.5 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscillator Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from Internal Osc. | 230400 | –0.32% | 106 | SYSCLK | XX[2] | 1 | 0xCB |
| | 115200 | –0.32% | 212 | SYSCLK | XX | 1 | 0x96 |
| | 57600 | 0.15% | 426 | SYSCLK | XX | 1 | 0x2B |
| | 28800 | –0.32% | 848 | SYSCLK/4 | 01 | 0 | 0x96 |
| | 14400 | 0.15% | 1704 | SYSCLK/12 | 00 | 0 | 0xB9 |
| | 9600 | –0.32% | 2544 | SYSCLK/12 | 00 | 0 | 0x96 |
| | 2400 | –0.32% | 10176 | SYSCLK/48 | 10 | 0 | 0x96 |
| | 1200 | 0.15% | 20448 | SYSCLK/48 | 10 | 0 | 0x2B |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in Section 27.1.
2. X = Don't care.

**Table 25.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator**

| | Frequency: 22.1184 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscillator Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 96 | SYSCLK | XX[2] | 1 | 0xD0 |
| | 115200 | 0.00% | 192 | SYSCLK | XX | 1 | 0xA0 |
| | 57600 | 0.00% | 384 | SYSCLK | XX | 1 | 0x40 |
| | 28800 | 0.00% | 768 | SYSCLK / 12 | 00 | 0 | 0xE0 |
| | 14400 | 0.00% | 1536 | SYSCLK / 12 | 00 | 0 | 0xC0 |
| | 9600 | 0.00% | 2304 | SYSCLK / 12 | 00 | 0 | 0xA0 |
| | 2400 | 0.00% | 9216 | SYSCLK / 48 | 10 | 0 | 0xA0 |
| | 1200 | 0.00% | 18432 | SYSCLK / 48 | 10 | 0 | 0x40 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 96 | EXTCLK / 8 | 11 | 0 | 0xFA |
| | 115200 | 0.00% | 192 | EXTCLK / 8 | 11 | 0 | 0xF4 |
| | 57600 | 0.00% | 384 | EXTCLK / 8 | 11 | 0 | 0xE8 |
| | 28800 | 0.00% | 768 | EXTCLK / 8 | 11 | 0 | 0xD0 |
| | 14400 | 0.00% | 1536 | EXTCLK / 8 | 11 | 0 | 0xA0 |
| | 9600 | 0.00% | 2304 | EXTCLK / 8 | 11 | 0 | 0x70 |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in Section 27.1.
2. X = Don't care.

## 26. Enhanced Serial Peripheral Interface (SPI0)

The Enhanced Serial Peripheral Interface (SPI0) provides access to a flexible, full-duplex synchronous serial bus. SPI0 can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select SPI0 in slave mode, or to disable Master Mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a chip-select output in master mode, or disabled for 3-wire operation. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

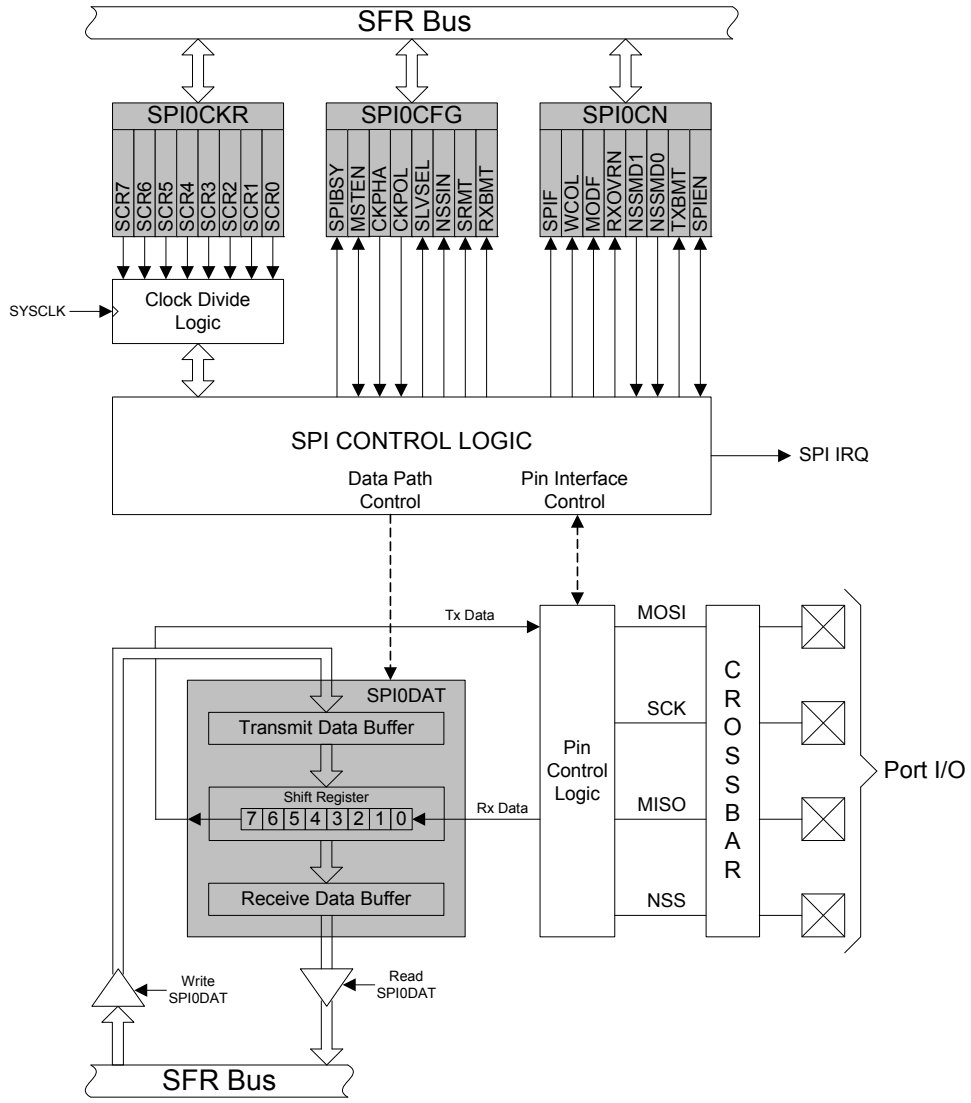



**Figure 26.1. SPI Block Diagram**

## 26.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

### 26.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output when SPI0 is operating as a master and an input when SPI0 is operating as a slave. Data is transferred most-significant bit first. When configured as a master, MOSI is driven by the MSB of the shift register in both 3- and 4-wire mode.

### 26.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. This signal is an input when SPI0 is operating as a master and an output when SPI0 is operating as a slave. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled and when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven by the MSB of the shift register.

### 26.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI0 generates this signal when operating as a master. The SCK signal is ignored by a SPI slave when the slave is not selected (NSS = 1) in 4-wire slave mode.

### 26.1.4. Slave Select (NSS)

The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD1 and NSSMD0 bits in the SPI0CN register. There are three possible modes that can be selected with these bits:

1. NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: SPI0 operates in 3-wire mode, and NSS is disabled. When operating as a slave device, SPI0 is always selected in 3-wire mode. Since no select signal is present, SPI0 must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and one slave.
2. NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an input. When operating as a slave, NSS selects the SPI0 device. When operating as a master, a 1-to-0 transition of the NSS signal disables the master function of SPI0 so that multiple master devices can be used on the same SPI bus.
3. NSSMD[1:0] = 1x: 4-Wire Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating SPI0 as a master device.

See Figure 26.2, Figure 26.3, and Figure 26.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “21. Port Input/Output” on page 210 for general purpose port I/O and crossbar information.

## 26.2. SPI0 Master Mode Operation

A SPI master device initiates all data transfers on a SPI bus. SPI0 is placed in master mode by setting the Master Enable flag (MSTEN, SPI0CN.6). Writing a byte of data to the SPI0 data register (SPI0DAT) when in master mode writes to the transmit buffer. If the SPI shift register is empty, the byte in the transmit buffer is moved to the shift register, and a data transfer begins. The SPI0 master immediately shifts out the data serially on the MOSI line while providing the serial clock on SCK. The SPIF (SPI0CN.7) flag is set to logic 1 at the end of the transfer. If interrupts are enabled, an interrupt request is generated when the SPIF flag is set. While the SPI0 master transfers data to a slave on the MOSI line, the addressed SPI slave device simultaneously transfers the contents of its shift register to the SPI master on the MISO line in a full-duplex

operation. Therefore, the SPIF flag serves as both a transmit-complete and receive-data-ready flag. The data byte received from the slave is transferred MSB-first into the master's shift register. When a byte is fully shifted into the register, it is moved to the receive buffer where it can be read by the processor by reading SPI0DAT.

When configured as a master, SPI0 can operate in one of three different modes: multi-master mode, 3-wire single-master mode, and 4-wire single-master mode. The default, multi-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In this mode, NSS is an input to the device, and is used to disable the master SPI0 when another master is accessing the bus. When NSS is pulled low in this mode, MSTEN (SPI0CN.6) and SPIEN (SPI0CN.0) are set to 0 to disable the SPI master device, and a Mode Fault is generated (MODF, SPI0CN.5 = 1). Mode Fault will generate an interrupt if enabled. SPI0 must be manually re-enabled in software under these circumstances. In multi-master systems, devices will typically default to being slave devices while they are not acting as the system master device. In multi-master mode, slave devices can be addressed individually (if needed) using general-purpose I/O pins. Figure 26.2 shows a connection diagram between two master devices in multiple-master mode.

3-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. In this mode, NSS is not used, and is not mapped to an external port pin through the crossbar. Any slave devices that must be addressed in this mode should be selected using general-purpose I/O pins. Figure 26.3 shows a connection diagram between a master device in 3-wire master mode and a slave device.

4-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 1. In this mode, NSS is configured as an output pin, and can be used as a slave-select signal for a single SPI device. In this mode, the output value of NSS is controlled (in software) with the bit NSSMD0 (SPI0CN.2). Additional slave devices can be addressed using general-purpose I/O pins. Figure 26.4 shows a connection diagram for a master device in 4-wire master mode and two slave devices.

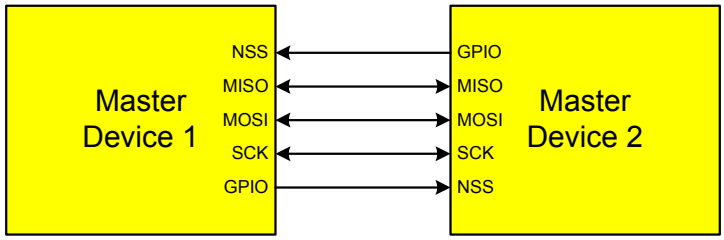



**Figure 26.2. Multiple-Master Mode Connection Diagram**

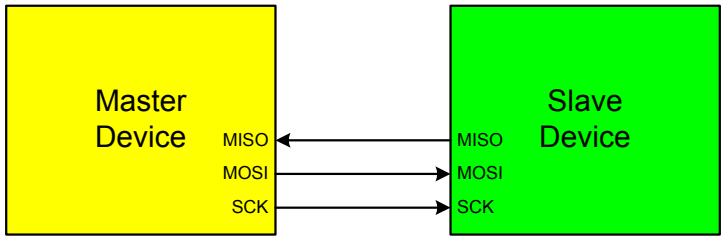



**Figure 26.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram**

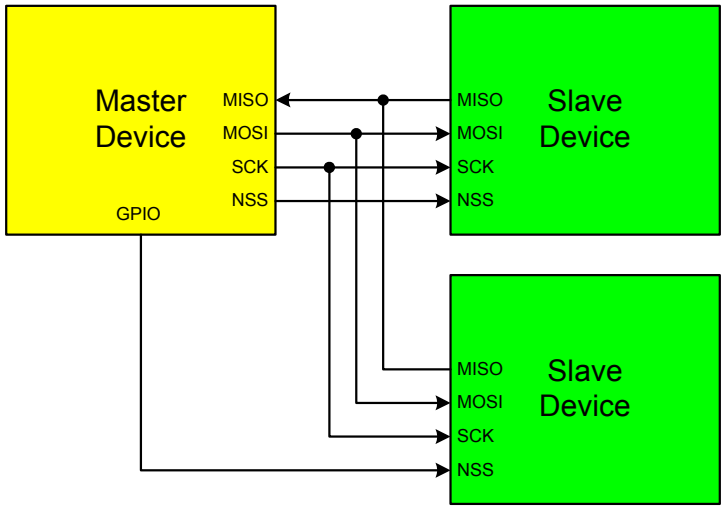



**Figure 26.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram**

## 26.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 26.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 26.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

## 26.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

All of the following bits must be cleared by software.

- The SPI Interrupt Flag, SPIF (SPI0CN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPI0 modes.
- The Write Collision Flag, WCOL (SPI0CN.6) is set to logic 1 if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPI0 modes.
- The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits in SPI0CN are set to logic 0 to disable SPI0 and allow another master device to access the bus.
- The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

## 26.5. Serial Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI0 Configuration Register (SPI0CFG). The CKPHA bit (SPI0CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI0CFG.4) selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. SPI0 should be disabled (by clearing the SPIEN bit, SPI0CN.0) when changing the clock phase or polarity. The clock and data line relationships for master mode are shown in Figure 26.5. For slave mode, the clock and data relationships are shown in Figure 26.6 and Figure 26.7. Note that CKPHA should be set to 0 on both the master and slave SPI when communicating between two Silicon Labs C8051 devices.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 26.9 controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock.

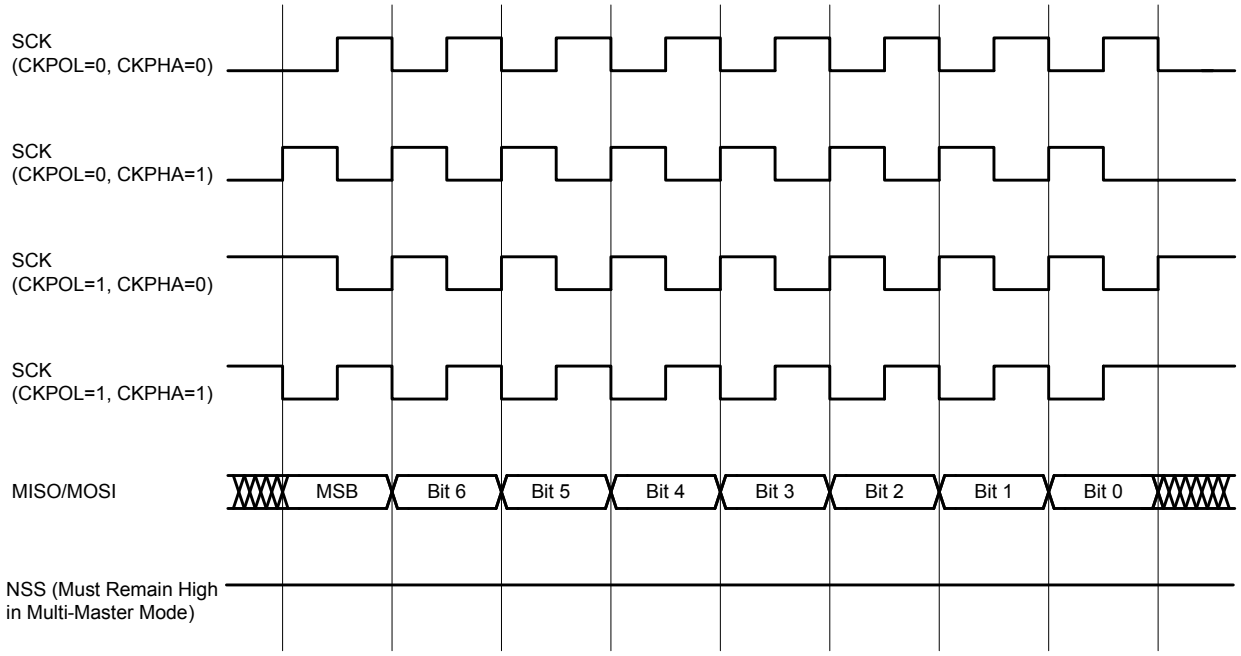



**Figure 26.5. Master Mode Data/Clock Timing**

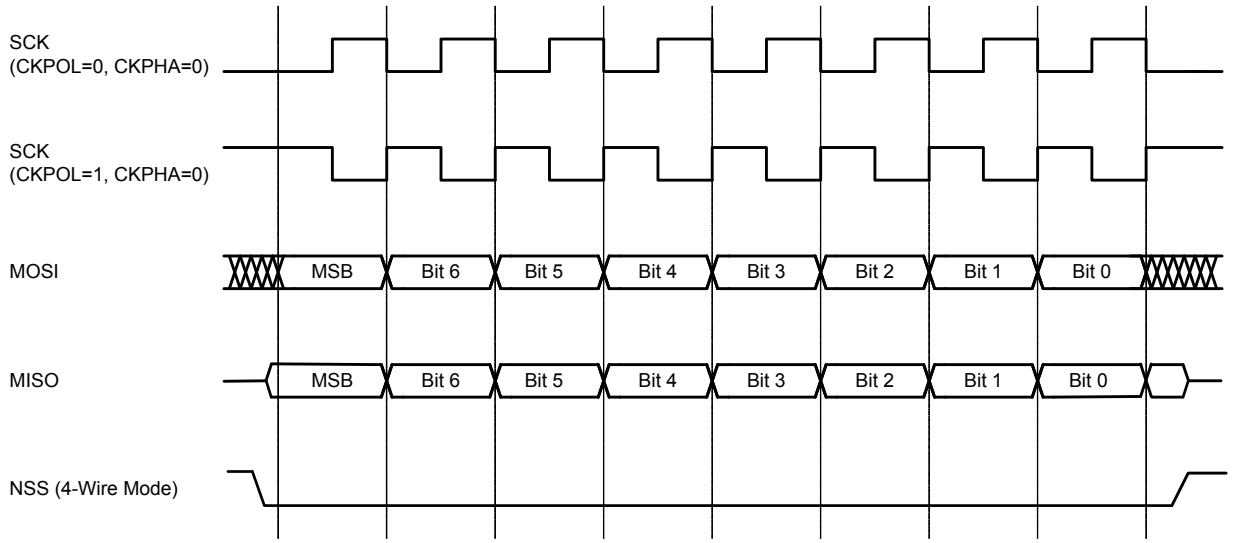



**Figure 26.6. Slave Mode Data/Clock Timing (CKPHA = 0)**

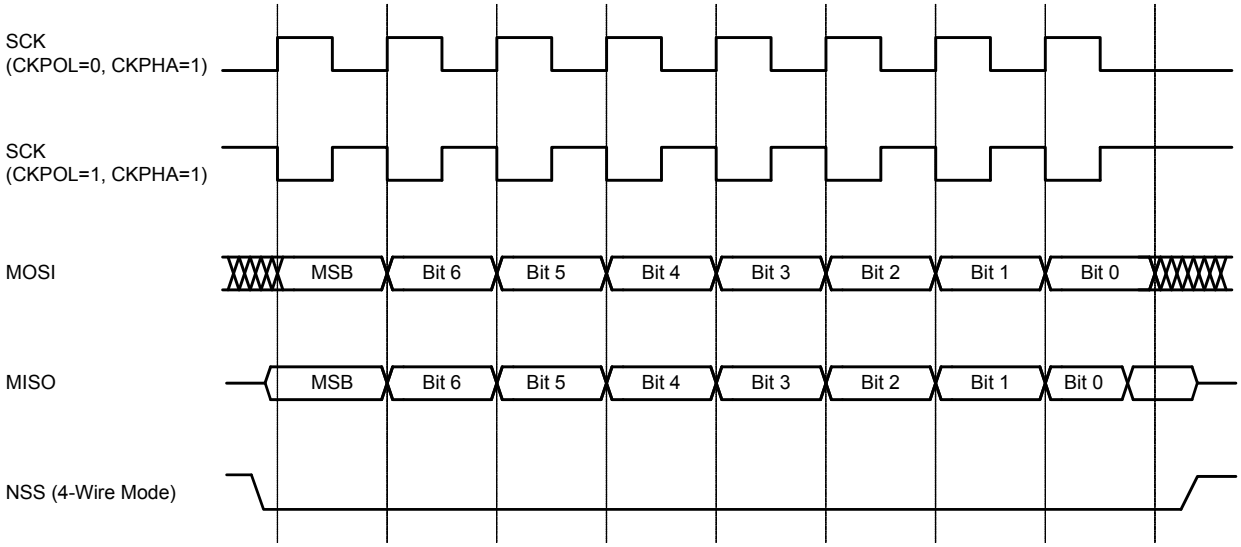



**Figure 26.7. Slave Mode Data/Clock Timing (CKPHA = 1)**

## 26.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.

## SFR Definition 26.7. SPI0CFG: SPI0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIBSY | MSTEN | CKPHA | CKPOL | SLVSEL | NSSIN | SRMT | RXBMT |
| Type | R | R/W | R/W | R/W | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xA1

| Bit | Name | Function |
|---|---|---|
| 7 | SPIBSY | **SPI Busy.**<br>This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode). |
| 6 | MSTEN | **Master Mode Enable.**<br>0: Disable master mode. Operate in slave mode.<br>1: Enable master mode. Operate as a master. |
| 5 | CKPHA | **SPI0 Clock Phase.**<br>0: Data centered on first edge of SCK period.*<br>1: Data centered on second edge of SCK period.* |
| 4 | CKPOL | **SPI0 Clock Polarity.**<br>0: SCK line low in idle state.<br>1: SCK line high in idle state. |
| 3 | SLVSEL | **Slave Selected Flag.**<br>This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input. |
| 2 | NSSIN | **NSS Instantaneous Pin Input.**<br>This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched. |
| 1 | SRMT | **Shift Register Empty (valid in slave mode only).**<br>This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode. |
| 0 | RXBMT | **Receive Buffer Empty (valid in slave mode only).**<br>This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode. |

**Note:** In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 26.1 for timing parameters.

## SFR Definition 26.8. SPI0CN: SPI0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIF | WCOL | MODF | RXOVRN | NSSMD[1:0] | | TXBMT | SPIEN |
| Type | R/W | R/W | R/W | R/W | R/W | | R | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

SFR Page = 0x0; SFR Address = 0xF8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | SPIF | **SPI0 Interrupt Flag.**<br>This bit is set to logic 1 by hardware at the end of a data transfer. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 6 | WCOL | **Write Collision Flag.**<br>This bit is set to logic 1 if a write to SPI0DAT is attempted when TXBMT is 0. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 5 | MODF | **Mode Fault Flag.**<br>This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 4 | RXOVRN | **Receive Overrun Flag (valid in slave mode only).**<br>This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 3:2 | NSSMD[1:0] | **Slave Select Mode.**<br>Selects between the following NSS operation modes:<br>(See Section 26.2 and Section 26.3).<br>00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin.<br>01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device.<br>1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0. |
| 1 | TXBMT | **Transmit Buffer Empty.**<br>This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer. |
| 0 | SPIEN | **SPI0 Enable.**<br>0: SPI disabled.<br>1: SPI enabled. |

## SFR Definition 26.9. SPI0CKR: SPI0 Clock Rate

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SCR[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA2

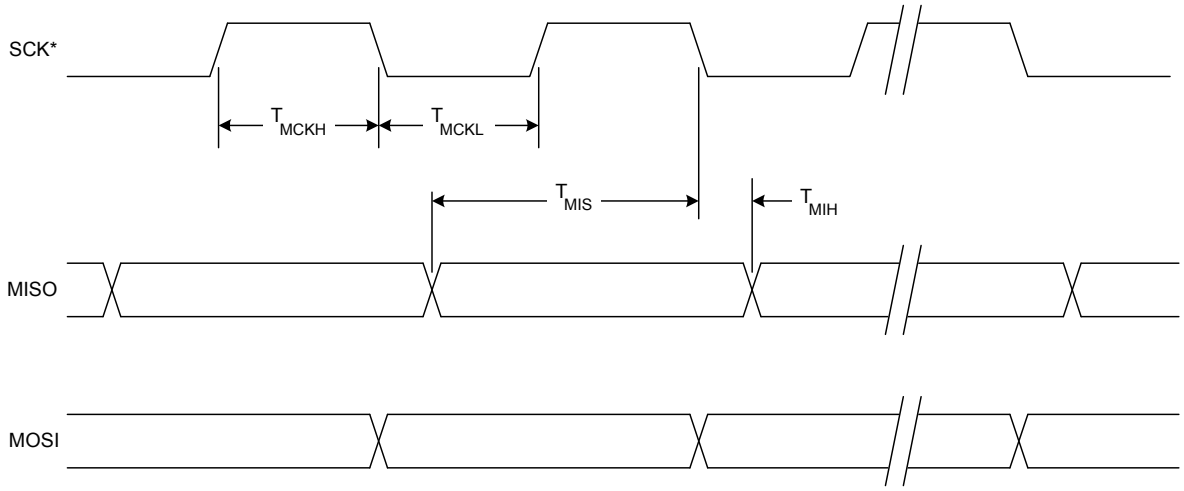
| Bit | Name | Function |
|---|---|---|
| 7:0 | SCR[7:0] | **SPI0 Clock Rate.**<br>These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI0CKR* is the 8-bit value held in the SPI0CKR register.<br>$f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR[7:0] + 1)}$<br>for 0 <= SPI0CKR <= 255<br><br>Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04,<br>$f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$<br>$f_{SCK} = 200kHz$ |

## SFR Definition 26.10. SPI0DAT: SPI0 Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPI0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

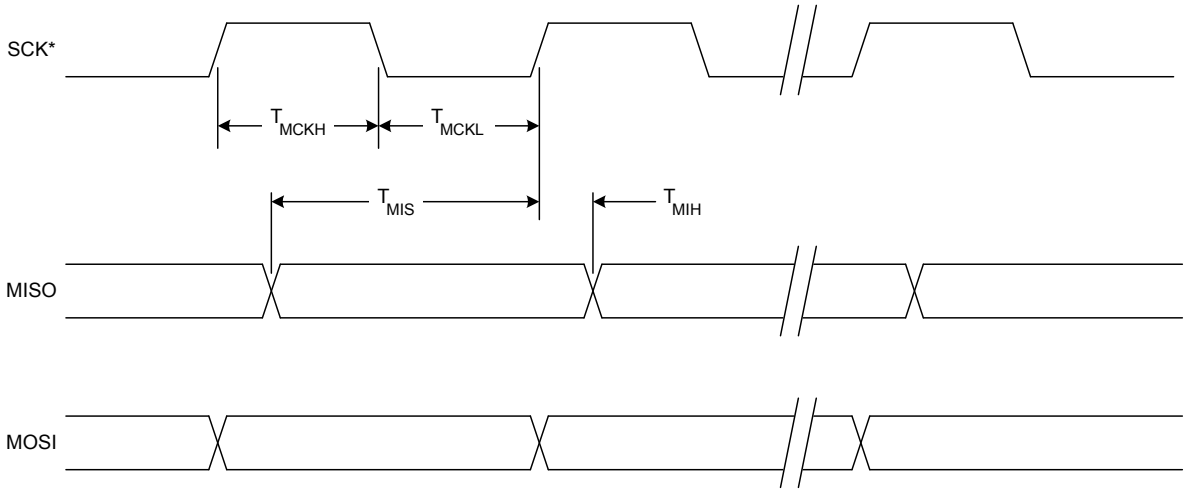
SFR Page = 0x0; SFR Address = 0xA3

| Bit | Name | Function |
|---|---|---|
| 7:0 | SPI0DAT[7:0] | **SPI0 Transmit and Receive Data.**<br>The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI0DAT returns the contents of the receive buffer. |



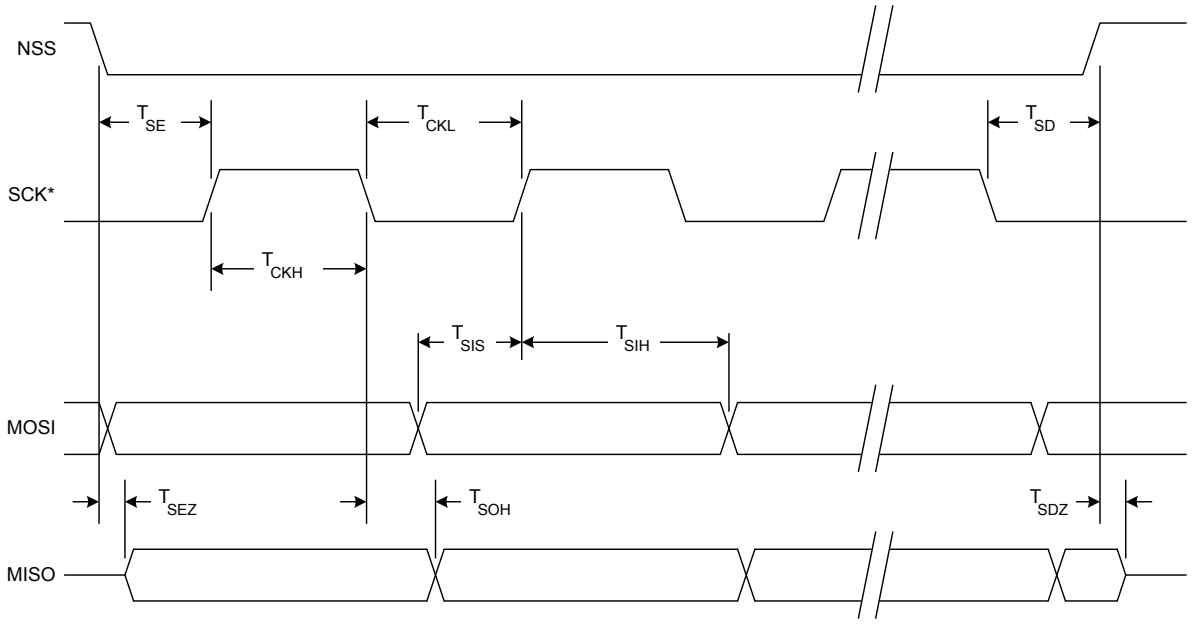

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 26.8. SPI Master Timing (CKPHA = 0)**



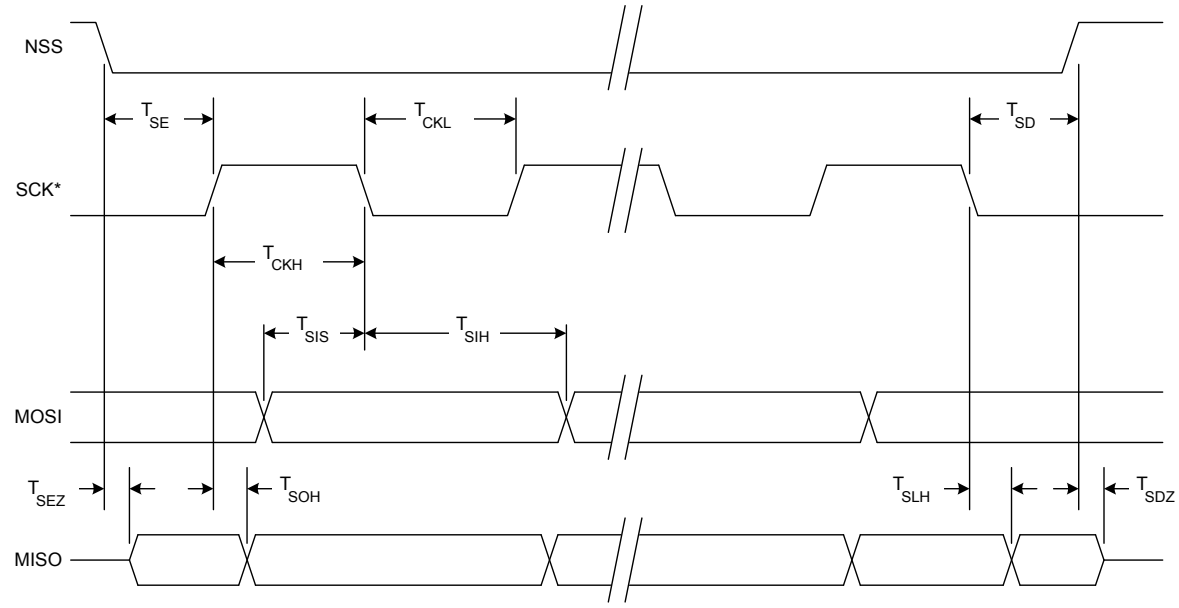

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 26.9. SPI Master Timing (CKPHA = 1)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 26.10. SPI Slave Timing (CKPHA = 0)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 26.11. SPI Slave Timing (CKPHA = 1)**

**Table 26.1. SPI Slave Timing Parameters**

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing** (See Figure 26.8 and Figure 26.9) | | | | |
| $T_{MCKH}$ | SCK High Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MCKL}$ | SCK Low Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MIS}$ | MISO Valid to SCK Shift Edge | 1 x $T_{SYSCLK}$ + 20 | — | ns |
| $T_{MIH}$ | SCK Shift Edge to MISO Change | 0 | — | ns |
| **Slave Mode Timing** (See Figure 26.10 and Figure 26.11) | | | | |
| $T_{SE}$ | NSS Falling to First SCK Edge | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SD}$ | Last SCK Edge to NSS Rising | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SEZ}$ | NSS Falling to MISO Valid | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{SDZ}$ | NSS Rising to MISO High-Z | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{CKH}$ | SCK High Time | 5 x $T_{SYSCLK}$ | — | ns |
| $T_{CKL}$ | SCK Low Time | 5 x $T_{SYSCLK}$ | — | ns |
| $T_{SIS}$ | MOSI Valid to SCK Sample Edge | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SIH}$ | SCK Sample Edge to MOSI Change | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SOH}$ | SCK Shift Edge to MISO Change | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{SLH}$ | Last SCK Edge to MISO Change (CKPHA = 1 ONLY) | 6 x $T_{SYSCLK}$ | 8 x $T_{SYSCLK}$ | ns |
| **Note:** $T_{SYSCLK}$ is equal to one period of the device system clock (SYSCLK). | | | | |

# 27. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two are 16-bit auto-reload timer for use with the ADC, SMBus, or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload. Additionally, Timer 2 and Timer 3 have a Capture Mode that can be used to measure the SmaRTClock or a Comparator period with respect to another oscillator. This is particularly useful when using Capacitive Touch Switches. See Application Note AN338 for details on Capacitive Touch Switch sensing.

| Timer 0 and Timer 1 Modes: | Timer 2 Modes: | Timer 3 Modes: |
|---|---|---|
| 13-bit counter/timer | 16-bit timer with auto-reload | 16-bit timer with auto-reload |
| 16-bit counter/timer | | |
| 8-bit counter/timer with auto-reload | Two 8-bit timers with auto-reload | Two 8-bit timers with auto-reload |
| Two 8-bit counter/timers (Timer 0 only) | | |

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked (See SFR Definition 27.1 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12. Timer 2 may additionally be clocked by the SmaRTClock divided by 8 or the Comparator0 output. Timer 3 may additionally be clocked by the external oscillator clock source divided by 8 or the Comparator1 output.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin (T0 or T1). Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

## SFR Definition 27.1. CKCON: Clock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | T3MH | T3ML | T2MH | T2ML | T1M | T0M | SCA[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8E

| Bit | Name | Function |
|---|---|---|
| 7 | T3MH | **Timer 3 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only).<br>0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 high byte uses the system clock. |
| 6 | T3ML | **Timer 3 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode.<br>0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 low byte uses the system clock. |
| 5 | T2MH | **Timer 2 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only).<br>0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 high byte uses the system clock. |
| 4 | T2ML | **Timer 2 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.<br>0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 low byte uses the system clock. |
| 3 | T1M | **Timer 1 Clock Select.**<br>Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1.<br>0: Timer 1 uses the clock defined by the prescale bits SCA[1:0].<br>1: Timer 1 uses the system clock. |
| 2 | T0M | **Timer 0 Clock Select.**<br>Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1.<br>0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0].<br>1: Counter/Timer 0 uses the system clock. |
| 1:0 | SCA[1:0] | **Timer 0/1 Prescale Bits.**<br>These bits control the Timer 0/1 Clock Prescaler:<br>00: System clock divided by 12<br>01: System clock divided by 4<br>10: System clock divided by 48<br>11: External clock divided by 8 (synchronized with the system clock) |

## 27.1. Timer 0 and Timer 1

Each timer is implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (Section “12.5. Interrupt Register Descriptions” on page 133); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (Section “12.5. Interrupt Register Descriptions” on page 133). Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently. Each operating mode is described below.

### 27.1.1. Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 (TCON.5) is set and an interrupt will occur if Timer 0 interrupts are enabled.

The C/T0 bit (TMOD.2) selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “21.3. Priority Crossbar Decoder” on page 214 for information on selecting and configuring external I/O pins). Clearing C/T selects the clock defined by the T0M bit (CKCON.3). When T0M is set, Timer 0 is clocked by the system clock. When T0M is cleared, Timer 0 is clocked by the source selected by the Clock Scale bits in CKCON (see SFR Definition 27.1).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7). Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{\text{INT0}}$ (see Section “12.5. Interrupt Register Descriptions” on page 133), facilitating pulse width measurements

**Table 27.1. Timer 0 Running Modes**

| TR0 | GATE0 | $\overline{\text{INT0}}$ | Counter/Timer |
|---|---|---|---|
| 0 | X | X | Disabled |
| 1 | 0 | X | Enabled |
| 1 | 1 | 0 | Disabled |
| 1 | 1 | 1 | Enabled |
| **Note:** X = Don't Care | | | |

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{\text{INT1}}$ is used with Timer 1; the $\overline{\text{INT1}}$ polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 12.7).

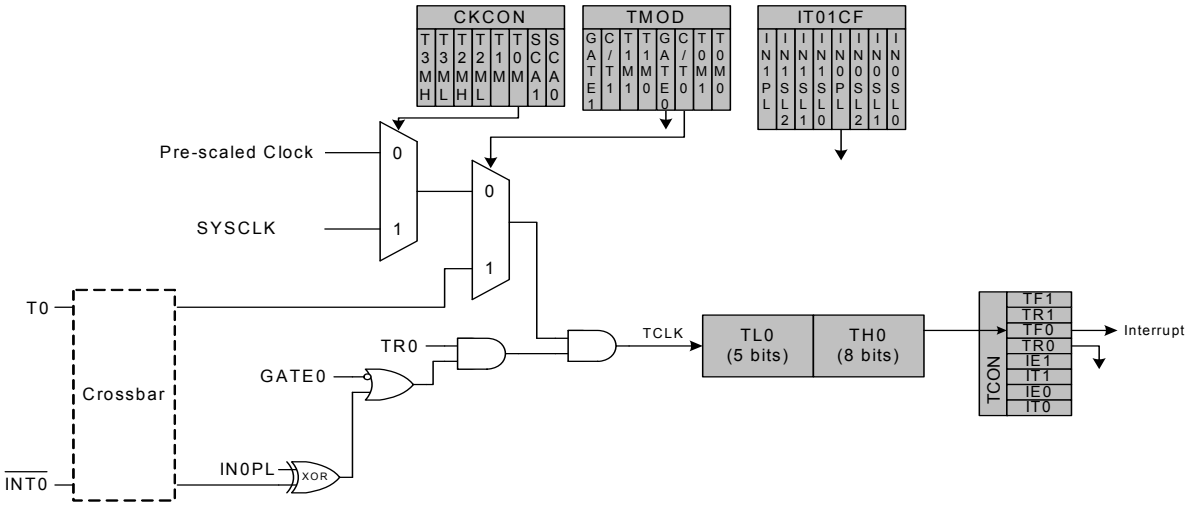



**Figure 27.1. T0 Mode 0 Block Diagram**

### 27.1.2. Mode 1: 16-bit Counter/Timer

Mode 1 operation is the same as Mode 0, except that the counter/timer registers use all 16 bits. The counter/timers are enabled and configured in Mode 1 in the same manner as for Mode 0.

### 27.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 (TCON.5) is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or when the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see Section “12.6. External Interrupts INT0 and INT1” on page 140 for details on the external input signals $\overline{INT0}$ and $\overline{INT1}$).

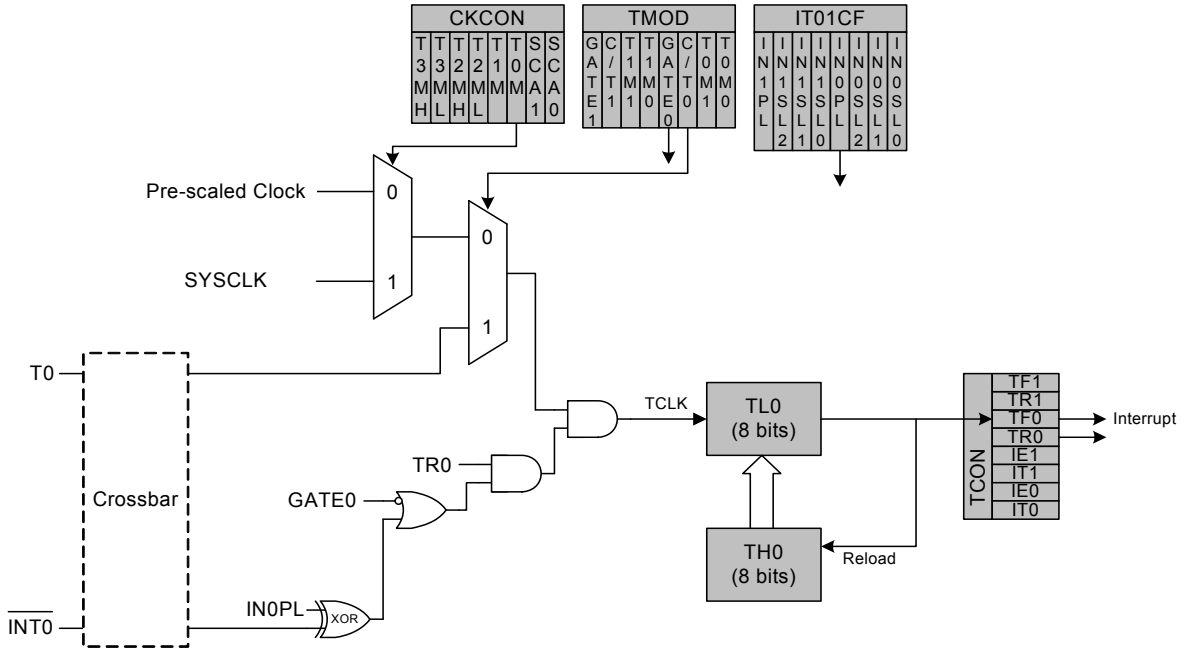



**Figure 27.2. T0 Mode 2 Block Diagram**

### 27.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

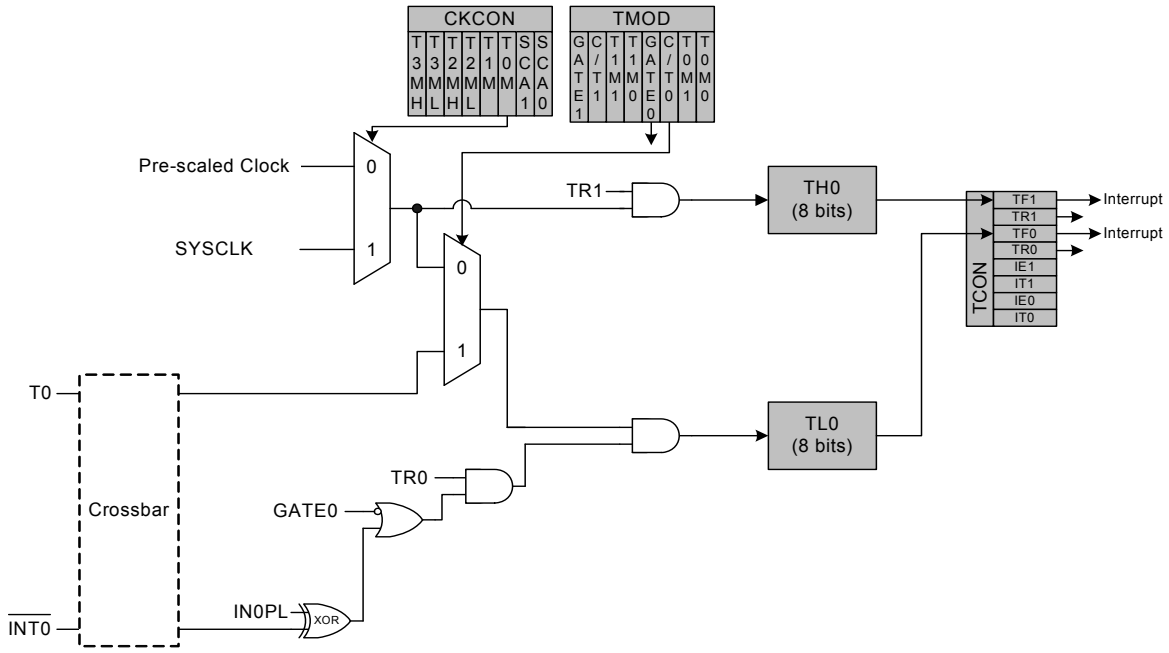



**Figure 27.3. T0 Mode 3 Block Diagram**

## SFR Definition 27.2. TCON: Timer Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x88; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | TF1 | **Timer 1 Overflow Flag.**<br>Set to 1 by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine. |
| 6 | TR1 | **Timer 1 Run Control.**<br>Timer 1 is enabled by setting this bit to 1. |
| 5 | TF0 | **Timer 0 Overflow Flag.**<br>Set to 1 by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine. |
| 4 | TR0 | **Timer 0 Run Control.**<br>Timer 0 is enabled by setting this bit to 1. |
| 3 | IE1 | **External Interrupt 1.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode. |
| 2 | IT1 | **Interrupt 1 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT1}}$ interrupt will be edge or level sensitive. $\overline{\text{INT1}}$ is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 12.7).<br>0: $\overline{\text{INT1}}$ is level triggered.<br>1: $\overline{\text{INT1}}$ is edge triggered. |
| 1 | IE0 | **External Interrupt 0.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode. |
| 0 | IT0 | **Interrupt 0 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 12.7).<br>0: $\overline{\text{INT0}}$ is level triggered.<br>1: $\overline{\text{INT0}}$ is edge triggered. |

## SFR Definition 27.3. TMOD: Timer Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GATE1 | C/T1 | T1M[1:0] | | GATE0 | C/T0 | T0M[1:0] | |
| Type | R/W | R/W | R/W | | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x89

| Bit | Name | Function |
|---|---|---|
| 7 | GATE1 | **Timer 1 Gate Control.**<br>0: Timer 1 enabled when TR1 = 1 irrespective of $\overline{\text{INT1}}$ logic level.<br>1: Timer 1 enabled only when TR1 = 1 AND $\overline{\text{INT1}}$ is active as defined by bit IN1PL in register IT01CF (see SFR Definition 12.7). |
| 6 | C/T1 | **Counter/Timer 1 Select.**<br>0: Timer: Timer 1 incremented by clock defined by T1M bit in register CKCON.<br>1: Counter: Timer 1 incremented by high-to-low transitions on external pin (T1). |
| 5:4 | T1M[1:0] | **Timer 1 Mode Select.**<br>These bits select the Timer 1 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Timer 1 Inactive |
| 3 | GATE0 | **Timer 0 Gate Control.**<br>0: Timer 0 enabled when TR0 = 1 irrespective of $\overline{\text{INT0}}$ logic level.<br>1: Timer 0 enabled only when TR0 = 1 AND $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7). |
| 2 | C/T0 | **Counter/Timer 0 Select.**<br>0: Timer: Timer 0 incremented by clock defined by T0M bit in register CKCON.<br>1: Counter: Timer 0 incremented by high-to-low transitions on external pin (T0). |
| 1:0 | T0M[1:0] | **Timer 0 Mode Select.**<br>These bits select the Timer 0 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Two 8-bit Counter/Timers |

## SFR Definition 27.4. TL0: Timer 0 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8A

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL0[7:0] | **Timer 0 Low Byte.**<br>The TL0 register is the low byte of the 16-bit Timer 0. |

## SFR Definition 27.5. TL1: Timer 1 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8B

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL1[7:0] | **Timer 1 Low Byte.**<br>The TL1 register is the low byte of the 16-bit Timer 1. |

## SFR Definition 27.6. TH0: Timer 0 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8C

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH0[7:0] | **Timer 0 High Byte.**<br>The TH0 register is the high byte of the 16-bit Timer 0. |

## SFR Definition 27.7. TH1: Timer 1 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8D

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH1[7:0] | **Timer 1 High Byte.**<br>The TH1 register is the high byte of the 16-bit Timer 1. |

## 27.2. Timer 2

Timer 2 is a 16-bit timer formed by two 8-bit SFRs: TMR2L (low byte) and TMR2H (high byte). Timer 2 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T2SPLIT bit (TMR2CN.3) defines the Timer 2 operation mode. Timer 2 can also be used in Capture Mode to measure the SmaRTClock or the Comparator 0 period with respect to another oscillator. The ability to measure the Comparator 0 period with respect to the system clock is makes using Touch Sense Switches very easy.

Timer 2 may be clocked by the system clock, the system clock divided by 12, SmaRTClock divided by 8, or Comparator 0 output. Note that the SmaRTClock divided by 8 and Comparator 0 output is synchronized with the system clock.

### 27.2.1. 16-bit Timer with Auto-Reload

When T2SPLIT (TMR2CN.3) is zero, Timer 2 operates as a 16-bit timer with auto-reload. Timer 2 can be clocked by SYSCLK, SYSCLK divided by 12, SmaRTClock divided by 8, or Comparator 0 output. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 2 reload registers (TMR2RLH and TMR2RLL) is loaded into the Timer 2 register as shown in Figure 27.4, and the Timer 2 High Byte Overflow Flag (TMR2CN.7) is set. If Timer 2 interrupts are enabled (if IE.5 is set), an interrupt will be generated on each Timer 2 overflow. Additionally, if Timer 2 interrupts are enabled and the TF2LEN bit is set (TMR2CN.5), an interrupt will be generated each time the lower 8 bits (TMR2L) overflow from 0xFF to 0x00.

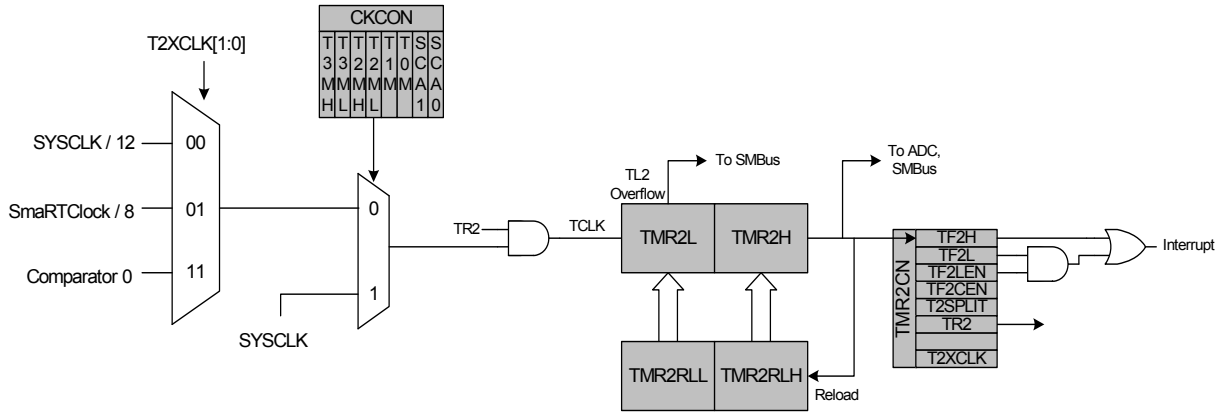



**Figure 27.4. Timer 2 16-Bit Mode Block Diagram**

### 27.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 27.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, SmaRTClock divided by 8 or Comparator 0 output. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bits (T2XCLK[1:0] in TMR2CN), as follows:

| T2MH | T2XCLK[1:0] | TMR2H Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock / 8 |
| 0 | 10 | Reserved |
| 0 | 11 | Comparator 0 |
| 1 | X | SYSCLK |

| T2ML | T2XCLK[1:0] | TMR2L Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock / 8 |
| 0 | 10 | Reserved |
| 0 | 11 | Comparator 0 |
| 1 | X | SYSCLK |

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

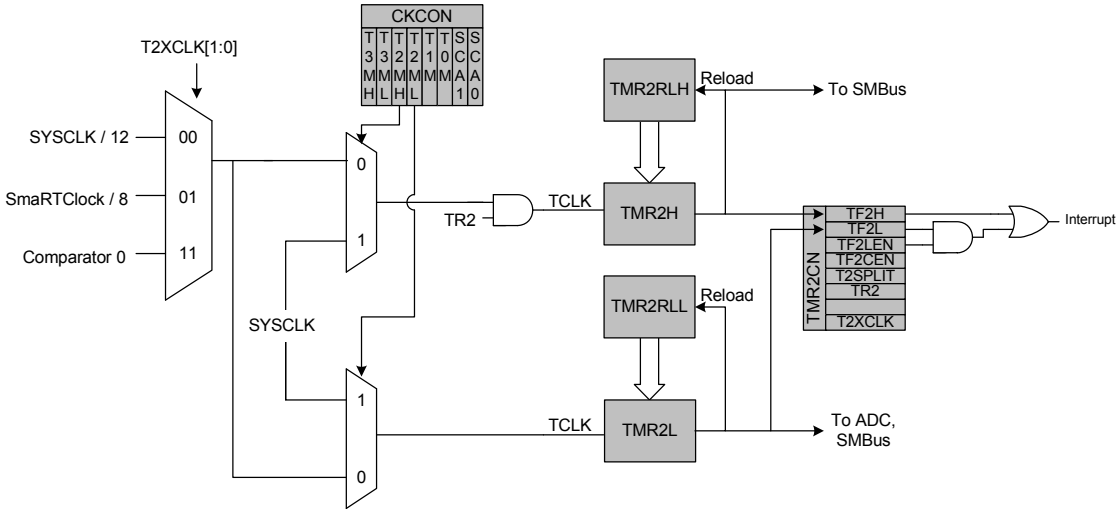



**Figure 27.5. Timer 2 8-Bit Mode Block Diagram**

### 27.2.3. Comparator 0/SmaRTClock Capture Mode

The Capture Mode in Timer 2 allows either Comparator 0 or the SmaRTClock period to be measured against the system clock or the system clock divided by 12. Comparator 0 and the SmaRTClock period can also be compared against each other. Timer 2 Capture Mode is enabled by setting TF2CEN to 1. Timer 2 should be in 16-bit auto-reload mode when using Capture Mode.

When Capture Mode is enabled, a capture event will be generated either every Comparator 0 rising edge or every 8 SmaRTClock clock cycles, depending on the T2XCLK1 setting. When the capture event occurs, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set (triggering an interrupt if Timer 2 interrupts are enabled). By recording the difference between two successive timer capture values, the Comparator 0 or SmaRTClock period can be determined with respect to the Timer 2 clock. The Timer 2 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T2ML = 1b, T2XCLK1 = 0b, and TF2CEN = 1b, Timer 2 will clock every SYSCLK and capture every SmaRTClock clock divided by 8. If the SYSCLK is 24.5 MHz and the difference between two successive captures is 5984, then the SmaRTClock clock is as follows:

24.5 MHz/(5984/8) = 0.032754 MHz or 32.754 kHz.

This mode allows software to determine the exact SmaRTClock frequency in self-oscillate mode and the time between consecutive Comparator 0 rising edges, which is useful for detecting changes in the capacitance of a Touch Sense Switch.

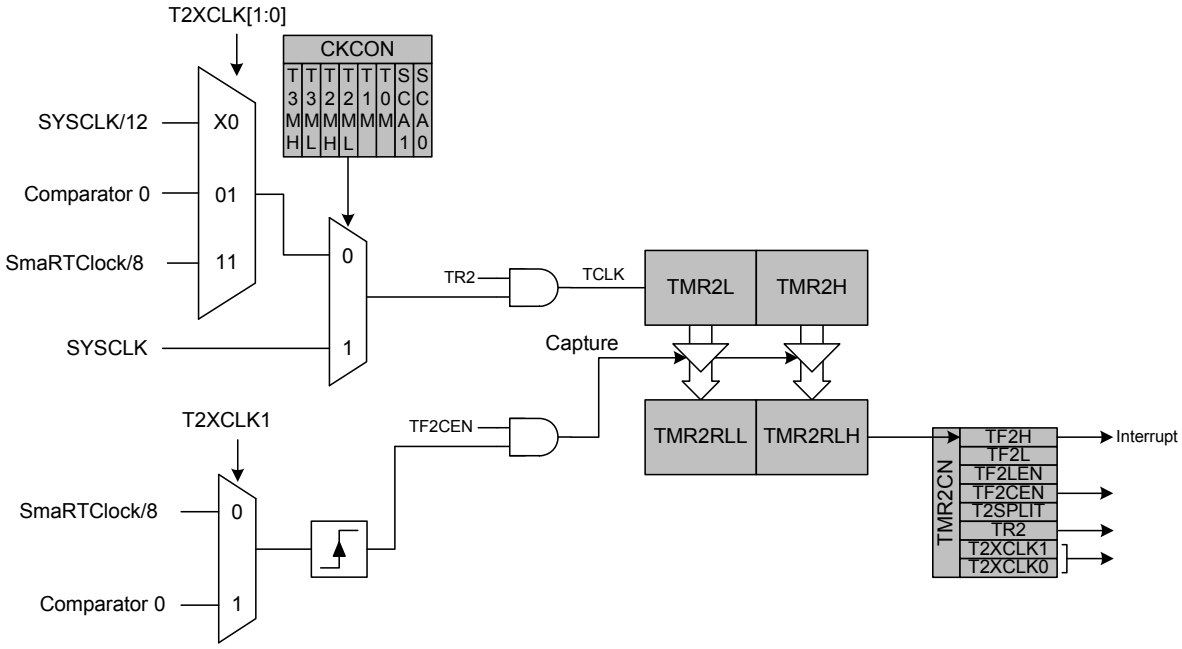



**Figure 27.6. Timer 2 Capture Mode Block Diagram**

## SFR Definition 27.8. TMR2CN: Timer 2 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF2H | TF2L | TF2LEN | TF2CEN | T2SPLIT | TR2 | T2XCLK[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | TF2H | **Timer 2 High Byte Overflow Flag.**<br>Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware. |
| 6 | TF2L | **Timer 2 Low Byte Overflow Flag.**<br>Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware. |
| 5 | TF2LEN | **Timer 2 Low Byte Interrupt Enable.**<br>When set to 1, this bit enables Timer 2 Low Byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows. |
| 4 | TF2CEN | **Timer 2 Capture Enable.**<br>When set to 1, this bit enables Timer 2 Capture Mode. |
| 3 | T2SPLIT | **Timer 2 Split Mode Enable.**<br>When set to 1, Timer 2 operates as two 8-bit timers with auto-reload. Otherwise, Timer 2 operates in 16-bit auto-reload mode. |
| 2 | TR2 | **Timer 2 Run Control.**<br>Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode. |
| 1:0 | T2XCLK[1:0] | **Timer 2 External Clock Select.**<br>This bit selects the “external” and “capture trigger” clock sources for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer.<br>Note: External clock sources are synchronized with the system clock.<br>00: External Clock is SYSCLK/12. Capture trigger is SmaRTClock/8.<br>01: External Clock is Comparator 0. Capture trigger is SmaRTClock/8.<br>10: External Clock is SYSCLK/12. Capture trigger is Comparator 0.<br>11: External Clock is SmaRTClock/8. Capture trigger is Comparator 0. |

## SFR Definition 27.9. TMR2RLL: Timer 2 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCA

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLL[7:0] | **Timer 2 Reload Register Low Byte.**<br>TMR2RLL holds the low byte of the reload value for Timer 2. |

## SFR Definition 27.10. TMR2RLH: Timer 2 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCB

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLH[7:0] | **Timer 2 Reload Register High Byte.**<br>TMR2RLH holds the high byte of the reload value for Timer 2. |

## SFR Definition 27.11. TMR2L: Timer 2 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCC

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2L[7:0] | **Timer 2 Low Byte.**<br>In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value. |

## SFR Definition 27.12. TMR2H Timer 2 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2H[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCD

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2H[7:0] | **Timer 2 Low Byte.**<br>In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value. |

## 27.3. Timer 3

Timer 3 is a 16-bit timer formed by two 8-bit SFRs: TMR3L (low byte) and TMR3H (high byte). Timer 3 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T3SPLIT bit (TMR3CN.3) defines the Timer 3 operation mode. Timer 3 can also be used in Capture Mode to measure the external oscillator source or the Comparator 1 period with respect to another oscillator. The ability to measure the Comparator 1 period with respect to the system clock is makes using Touch Sense Switches very easy.

Timer 3 may be clocked by the system clock, the system clock divided by 12, external oscillator source divided by 8, or Comparator 1 output. The external oscillator source divided by 8 and Comparator 1 output is synchronized with the system clock.

### 27.3.1. 16-bit Timer with Auto-Reload

When T3SPLIT (TMR3CN.3) is zero, Timer 3 operates as a 16-bit timer with auto-reload. Timer 3 can be clocked by SYSCLK, SYSCLK divided by 12, external oscillator clock source divided by 8, or Comparator 1 output. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 3 reload registers (TMR3RLH and TMR3RLL) is loaded into the Timer 3 register as shown in Figure 27.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 3 overflow. Additionally, if Timer 3 interrupts are enabled and the TF3LEN bit is set (TMR3CN.5), an interrupt will be generated each time the lower 8 bits (TMR3L) overflow from 0xFF to 0x00.

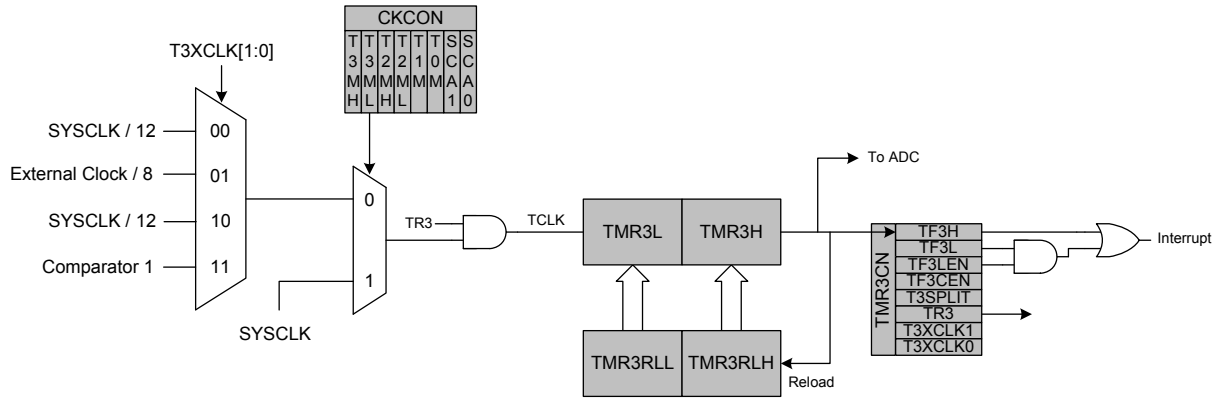



**Figure 27.7. Timer 3 16-Bit Mode Block Diagram**

### 27.3.2. 8-bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 27.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or Comparator 1. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

| T3MH | T3XCLK[1:0] | TMR3H Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | External Clock / 8 |
| 0 | 10 | SYSCLK / 12 |
| 0 | 11 | Comparator 1 |
| 1 | X | SYSCLK |

| T3ML | T3XCLK[1:0] | TMR3L Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | External Clock / 8 |
| 0 | 10 | SYSCLK / 12 |
| 0 | 11 | Comparator 1 |
| 1 | X | SYSCLK |

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

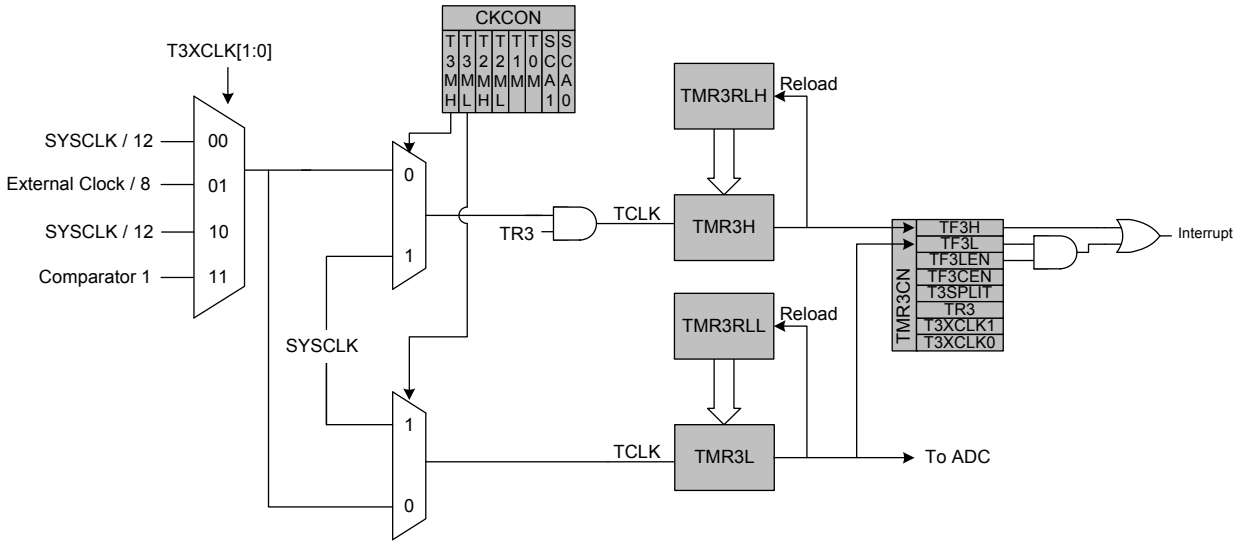



**Figure 27.8. Timer 3 8-Bit Mode Block Diagram.**

### 27.3.3. Comparator 1/External Oscillator Capture Mode

The Capture Mode in Timer 3 allows either Comparator 1 or the external oscillator period to be measured against the system clock or the system clock divided by 12. Comparator 1 and the external oscillator period can also be compared against each other.

Setting TF3CEN to 1 enables the Comparator 1/External Oscillator Capture Mode for Timer 3. In this mode, T3SPLIT should be set to 0, as the full 16-bit timer is used.

When Capture Mode is enabled, a capture event will be generated either every Comparator 1 rising edge or every 8 external clock cycles, depending on the T3XCLK1 setting. When the capture event occurs, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set (triggering an interrupt if Timer 3 interrupts are enabled). By recording the difference between two successive timer capture values, the Comparator 1 or external clock period can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T3ML = 1b, T3XCLK1 = 0b, and TF3CEN = 1b, Timer 3 will clock every SYSCLK and capture every Comparator 1 rising edge. If SYSCLK is 24.5 MHz and the difference between two successive captures is 350 counts, then the Comparator 1 period is:

350 x (1 / 24.5 MHz) = 14.2 µs.

This mode allows software to determine the exact frequency of the external oscillator in C and RC mode or the time between consecutive Comparator 0 rising edges, which is useful for detecting changes in the capacitance of a Touch Sense Switch.

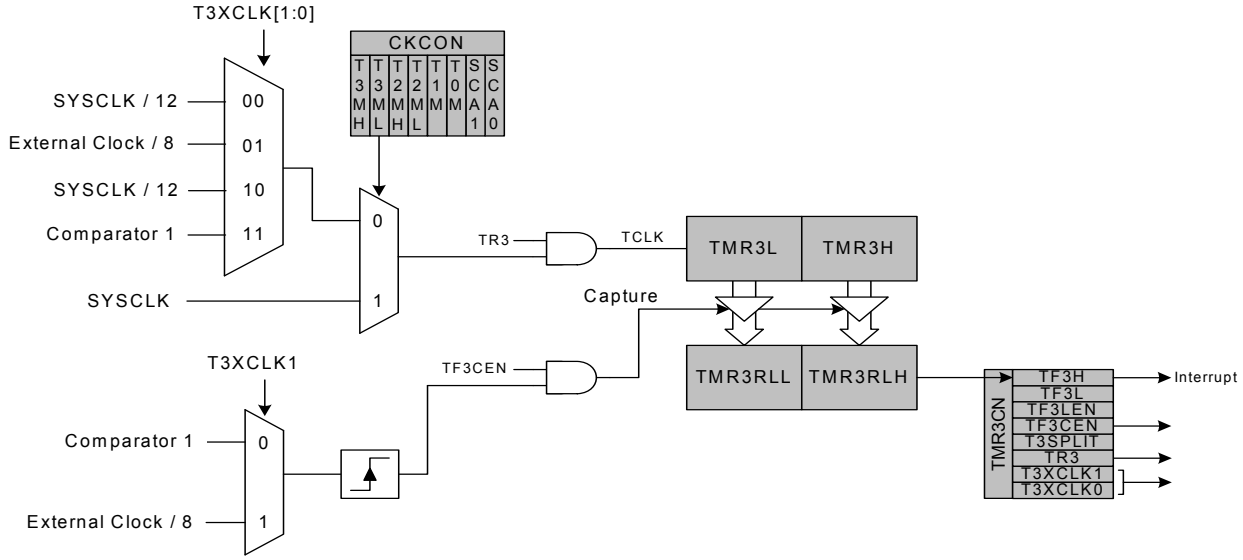



**Figure 27.9. Timer 3 Capture Mode Block Diagram**

## SFR Definition 27.13. TMR3CN: Timer 3 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF3H | TF3L | TF3LEN | TF3CEN | T3SPLIT | TR3 | T3XCLK[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x91

| Bit | Name | Function |
|---|---|---|
| 7 | TF3H | **Timer 3 High Byte Overflow Flag.**<br>Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware. |
| 6 | TF3L | **Timer 3 Low Byte Overflow Flag.**<br>Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware. |
| 5 | TF3LEN | **Timer 3 Low Byte Interrupt Enable.**<br>When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows. |
| 4 | TF3CEN | **Timer 3 Comparator 1/External Oscillator Capture Enable.**<br>When set to 1, this bit enables Timer 3 Capture Mode. |
| 3 | T3SPLIT | **Timer 3 Split Mode Enable.**<br>When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload.<br>0: Timer 3 operates in 16-bit auto-reload mode.<br>1: Timer 3 operates as two 8-bit auto-reload timers. |
| 2 | TR3 | **Timer 3 Run Control.**<br>Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode. |
| 1:0 | T3XCLK[1:0] | **Timer 3 External Clock Select.**<br>This bit selects the “external” and “capture trigger” clock sources for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer.<br>Note: External clock sources are synchronized with the system clock.<br>00: External Clock is SYSCLK /12. Capture trigger is Comparator 1.<br>01: External Clock is External Oscillator/8. Capture trigger is Comparator 1.<br>10: External Clock is SYSCLK/12. Capture trigger is External Oscillator/8.<br>11: External Clock is Comparator 1. Capture trigger is External Oscillator/8. |

## SFR Definition 27.14. TMR3RLL: Timer 3 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x92

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLL[7:0] | **Timer 3 Reload Register Low Byte.**<br>TMR3RLL holds the low byte of the reload value for Timer 3. |

## SFR Definition 27.15. TMR3RLH: Timer 3 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x93

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLH[7:0] | **Timer 3 Reload Register High Byte.**<br>TMR3RLH holds the high byte of the reload value for Timer 3. |

## SFR Definition 27.16. TMR3L: Timer 3 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x94

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3L[7:0] | **Timer 3 Low Byte.**<br>In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value. |

## SFR Definition 27.17. TMR3H Timer 3 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3H[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x95

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3H[7:0] | **Timer 3 High Byte.**<br>In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value. |

# 28. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and six 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between the following sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section "28.3. Capture/Compare Modules" on page 360). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 28.1

**Important Note:** The PCA Module 5 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. **Access to certain PCA registers is restricted while WDT mode is enabled**. See Section 28.4 for details.

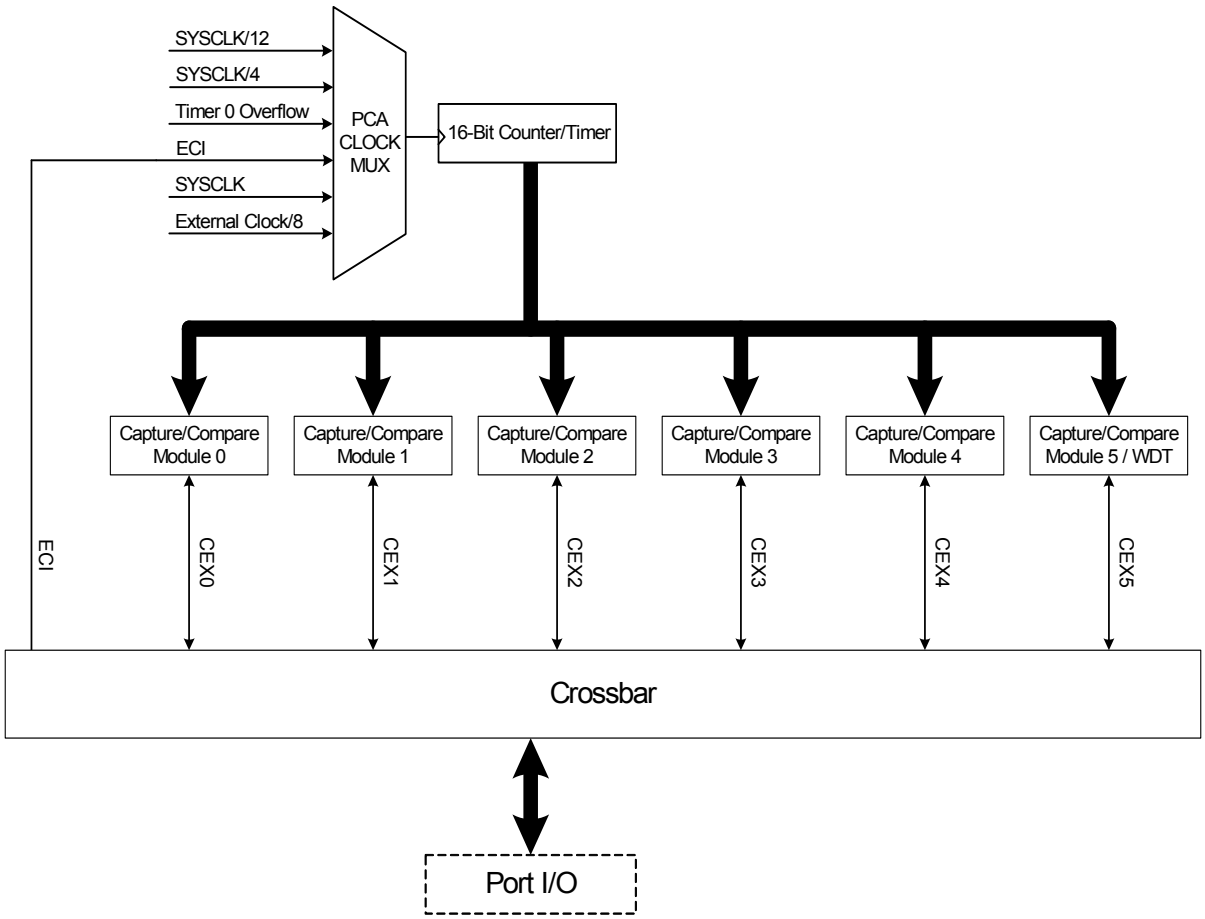



**Figure 28.1. PCA Block Diagram**

## 28.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter.** Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 28.1.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 28.1. PCA Timebase Input Options**

| CPS2 | CPS1 | CPS0 | Timebase |
|---|---|---|---|
| 0 | 0 | 0 | System clock divided by 12 |
| 0 | 0 | 1 | System clock divided by 4 |
| 0 | 1 | 0 | Timer 0 overflow |
| 0 | 1 | 1 | High-to-low transitions on ECI (max rate = system clock divided by 4) |
| 1 | 0 | 0 | System clock |
| 1 | 0 | 1 | External oscillator source divided by 8* |
| 1 | 1 | 0 | Reserved |
| 1 | 1 | 1 | Reserved |
| **Note:** External oscillator source divided by 8 is synchronized with the system clock. | | | |

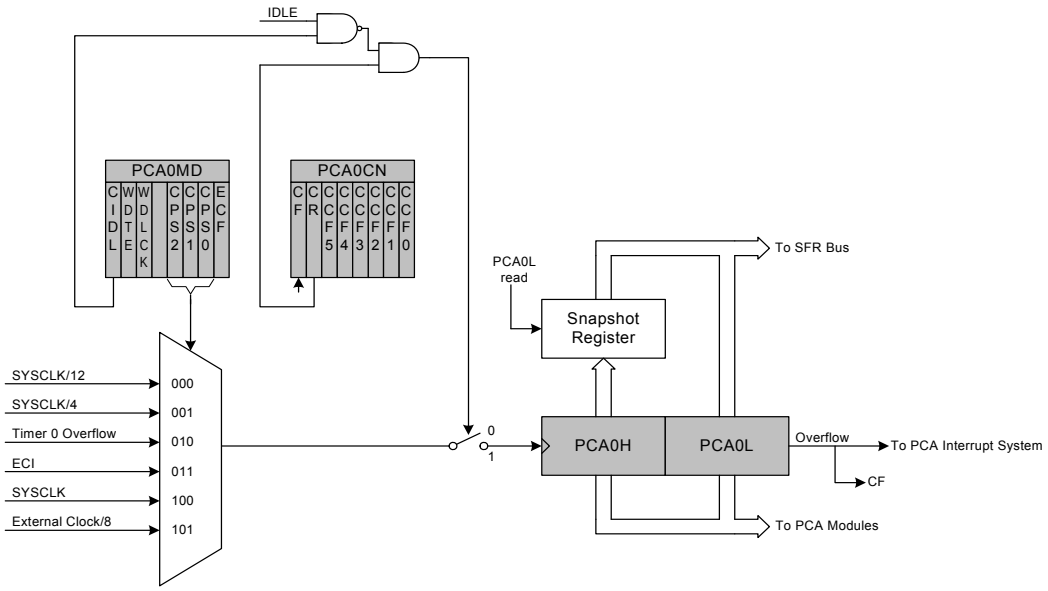



**Figure 28.2. PCA Counter/Timer Block Diagram**

## 28.2. PCA0 Interrupt Sources

Figure 28.3 shows a diagram of the PCA interrupt tree. There are eight independent event flags that can be used to generate a PCA0 interrupt. They are: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter, an intermediate overflow flag (COVF), which can be set on an overflow from the 8th, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCF0, CCF1, CCF2, CCF3, CCF4, and CCF5), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt, using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

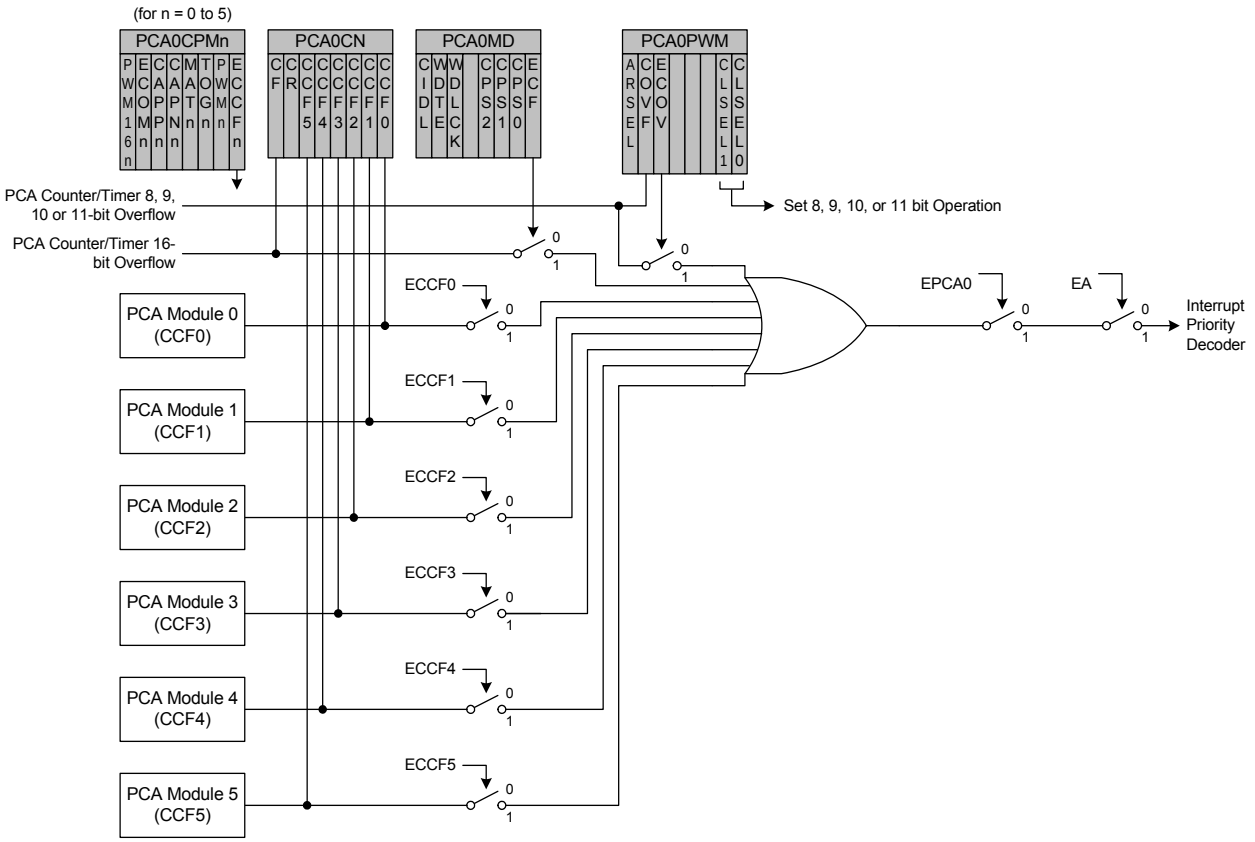



**Figure 28.3. PCA Interrupt Block Diagram**

## 28.3. Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: edge-triggered capture, software timer, high speed output, frequency output, 8 to 11-bit pulse width modulator, or 16-bit pulse width modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation. Table 28.2 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module's operating mode. Note that all modules set to use 8, 9, 10, or 11-bit PWM mode must use the same cycle length (8-11 bits). Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

**Table 28.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules**

| **Operational Mode** | **PCA0CPMn** | | | | | | | | **PCA0PWM** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit Number | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4-2 | 1–0 |
| Capture triggered by positive edge on CEXn | X | X | 1 | 0 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by negative edge on CEXn | X | X | 0 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by any transition on CEXn | X | X | 1 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Software Timer | X | C | 0 | 0 | 1 | 0 | 0 | A | 0 | X | B | XXX | XX |
| High Speed Output | X | C | 0 | 0 | 1 | 1 | 0 | A | 0 | X | B | XXX | XX |
| Frequency Output | X | C | 0 | 0 | 0 | 1 | 1 | A | 0 | X | B | XXX | XX |
| 8-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | 00 |
| 9-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 01 |
| 10-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 10 |
| 11-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 11 |
| 16-Bit Pulse Width Modulator | 1 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | XX |

**Notes:**

1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th, 9th, 10th or 11th bit overflow interrupt (Depends on setting of CLSEL[1:0]).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

### 28.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPPn and CAPNn bits are set to logic 1, then the state of the Port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

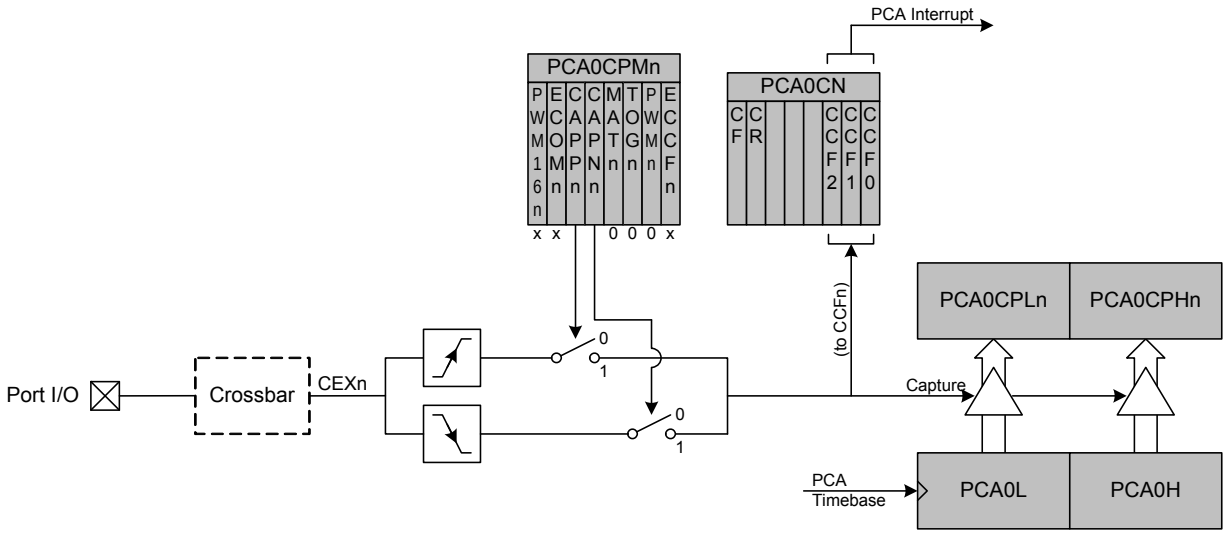



**Figure 28.4. PCA Capture Mode Diagram**

**Note:** The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

### 28.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

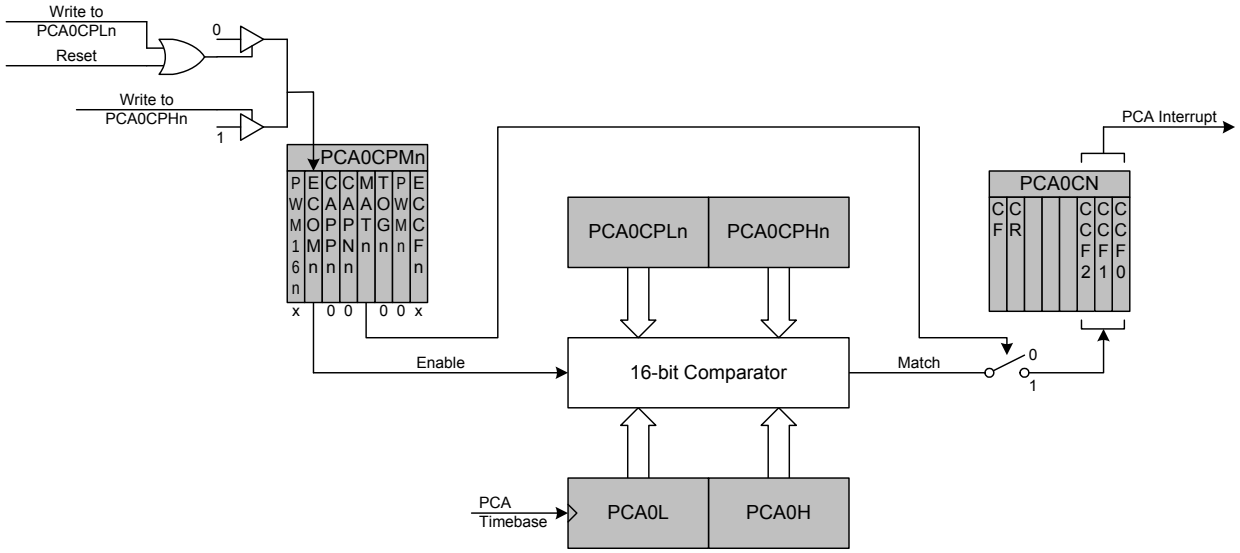



**Figure 28.5. PCA Software Timer Mode Diagram**

### 28.3.3. High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

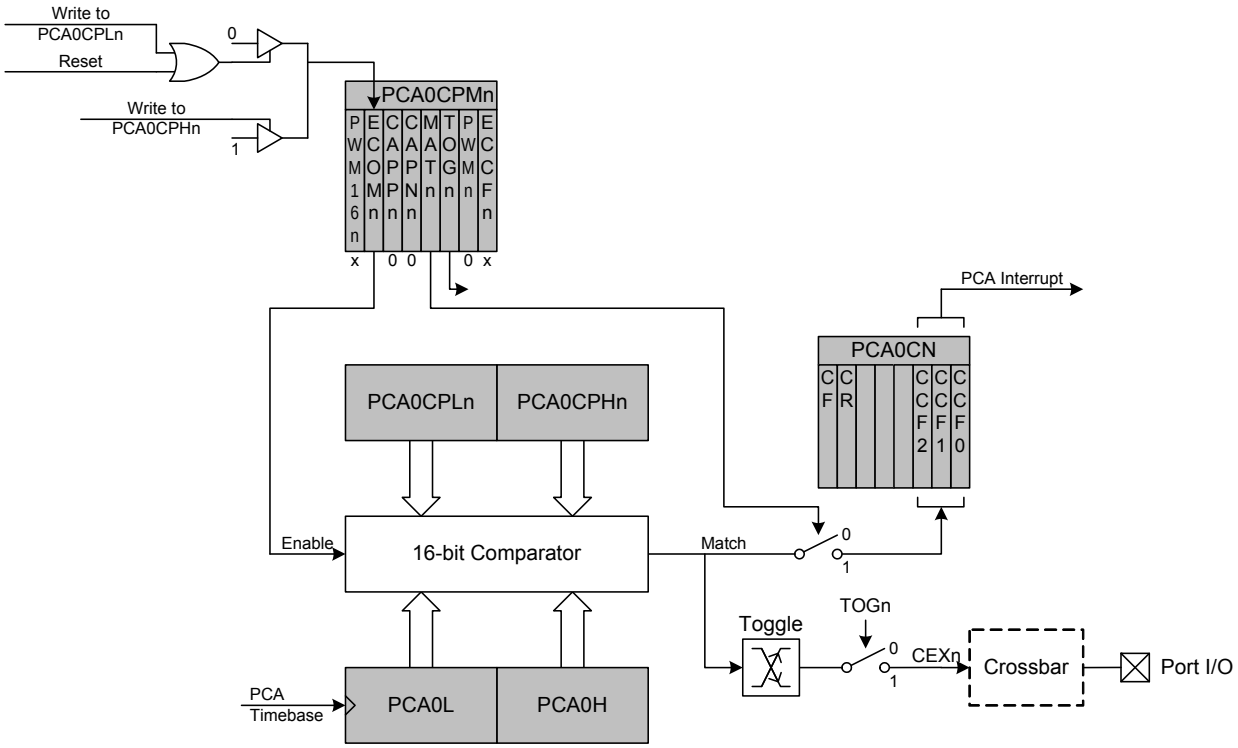



**Figure 28.6. PCA High-Speed Output Mode Diagram**

### 28.3.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 28.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

**Equation 28.1. Square Wave Frequency Output**

Where $F_{PCA}$ is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register. Note that the MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

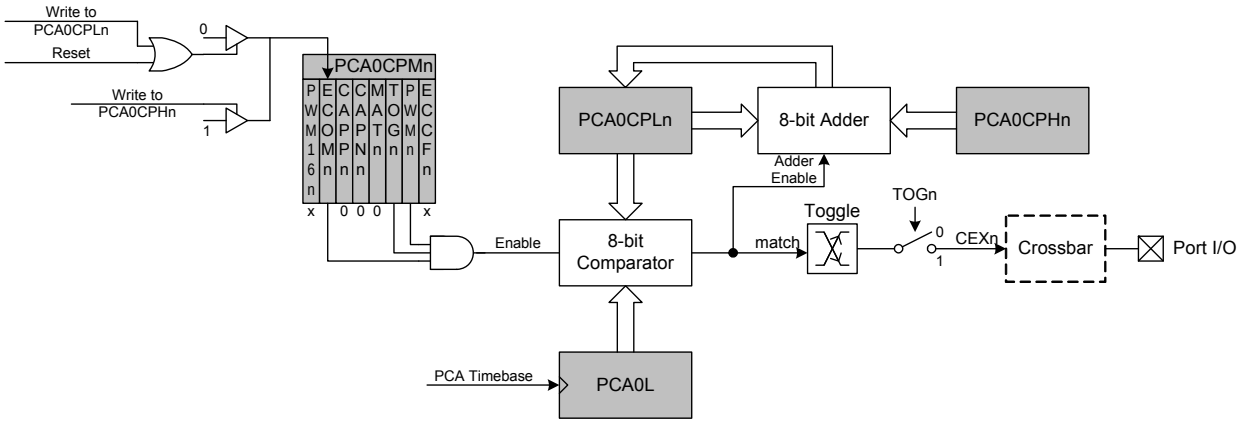



**Figure 28.7. PCA Frequency Output Mode**

### 28.3.5. 8-Bit, 9-Bit, 10-Bit and 11-Bit Pulse Width Modulator Modes

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer, and the setting of the PWM cycle length (8, 9, 10 or 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9, 10 and 11-bit PWM modes. **It is important to note that all channels configured for 8/9/10/11-bit PWM mode will use the same cycle length.** It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently.

#### 28.3.5.1. 8-Bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 8-bit PWM mode is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be set. When the count value in PCA0L overflows, the CEXn output will be reset (see Figure 28.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the module's capture/compare high byte (PCA0CPHn) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit Pulse Width Modulator mode. If the MATn bit is set to 1, the CCFn flag for the module will be set each time an 8-bit comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles. The duty cycle for 8-Bit PWM Mode is given in Equation 28.2.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

**Equation 28.2. 8-Bit PWM Duty Cycle**

Using Equation 28.2, the largest duty cycle is 100% (PCA0CPHn = 0), and the smallest duty cycle is 0.39% (PCA0CPHn = 0xFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

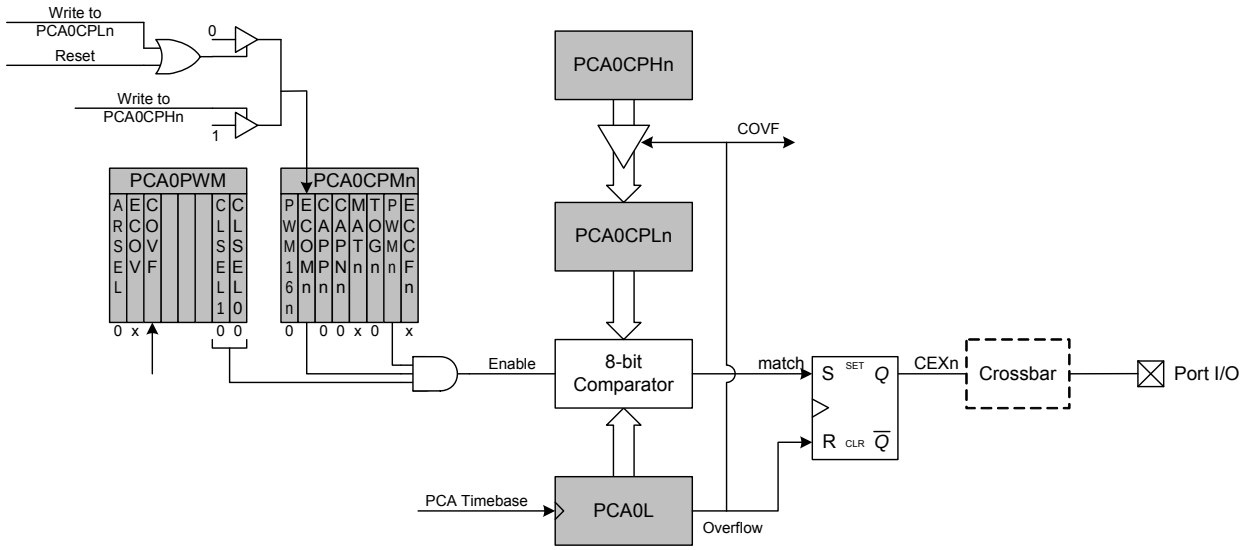



**Figure 28.8. PCA 8-Bit PWM Mode Diagram**

### 28.3.5.2. 9/10/11-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 9/10/11-bit PWM mode should be varied by writing to an "Auto-Reload" Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0.

When the least-significant N bits of the PCA0 counter match the value in the associated module's capture/compare register (PCA0CPn), the output on CEXn is asserted high. When the counter overflows from the Nth bit, CEXn is asserted low (see Figure 28.9). Upon an overflow from the Nth bit, the COVF flag is set, and the value stored in the module's auto-reload register is loaded into the capture/compare register. The value of N is determined by the CLSEL bits in register PCA0PWM.

The 9, 10 or 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module will be set each time a comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 512 (9-bit), 1024 (10-bit) or 2048 (11-bit) PCA clock cycles. The duty cycle for 9/10/11-Bit PWM Mode is given in Equation 28.2, where N is the number of bits in the PWM cycle.

**Important Note About PCA0CPHn and PCA0CPLn Registers**: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(2^N - PCA0CPn)}{2^N}$$

**Equation 28.3. 9, 10, and 11-Bit PWM Duty Cycle**

A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

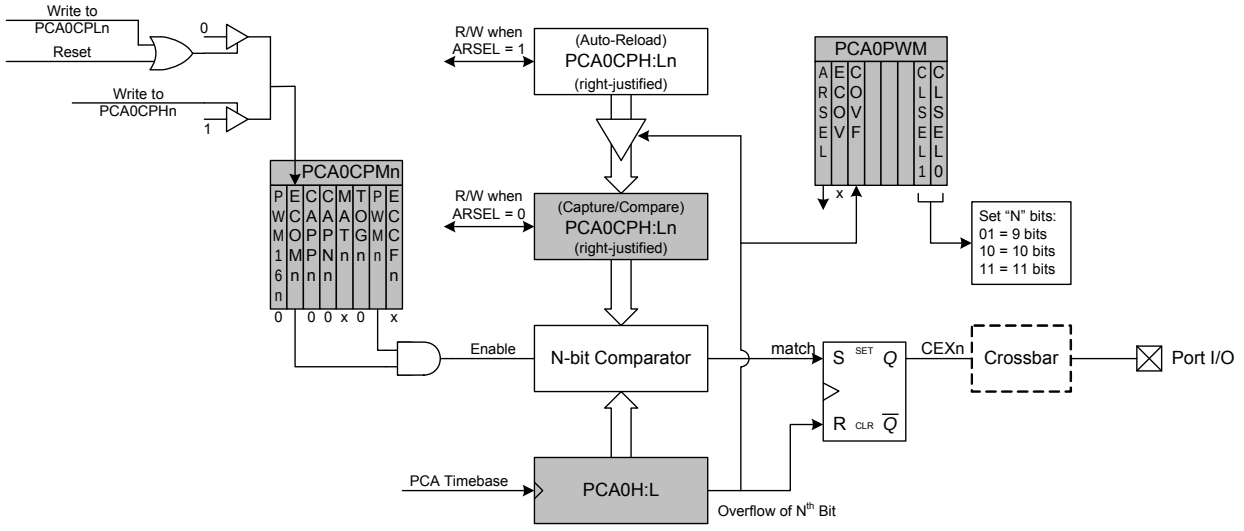



**Figure 28.9. PCA 9, 10 and 11-Bit PWM Mode Diagram**

### 28.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8/9/10/11-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. To output a varying duty cycle, new value writes should be synchronized with PCA CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 28.4.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

**Equation 28.4. 16-Bit PWM Duty Cycle**

Using Equation 28.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

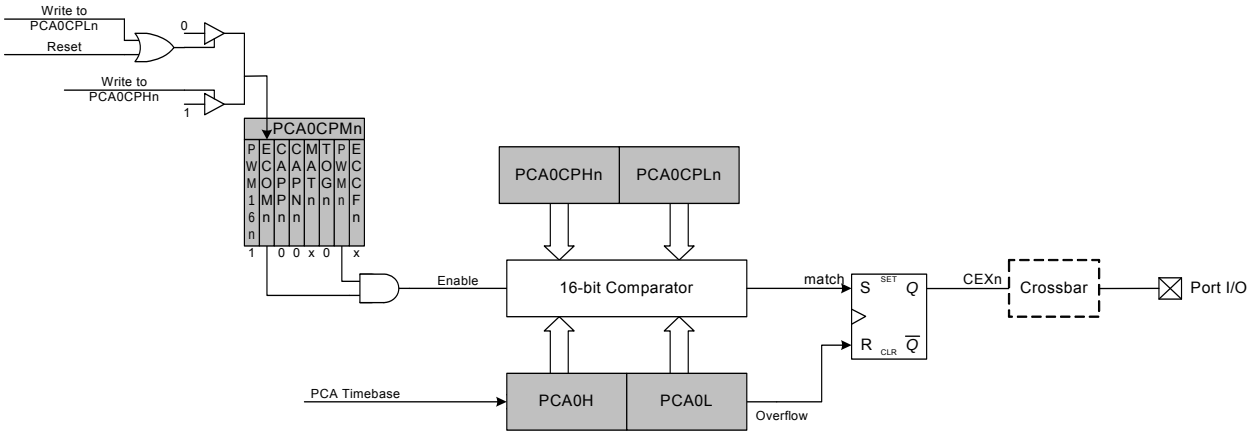



**Figure 28.10. PCA 16-Bit PWM Mode**

## 28.4. Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the PCA Module 5. The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH5) exceed a specified limit. The WDT can be configured and enabled/disabled as needed by software.

With the WDTE bit set in the PCA0MD register, Module 5 operates as a watchdog timer (WDT). The Module 5 high byte is compared to the PCA counter high byte; the Module 5 low byte holds the offset to be used when WDT updates are performed. **The Watchdog Timer is enabled on reset. Writes to some PCA registers are restricted while the Watchdog Timer is enabled.** The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

### 28.4.1. Watchdog Timer Operation

While the WDT is enabled:

- PCA counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA clock source bits (CPS2–CPS0) are frozen.
- PCA Idle control bit (CIDL) is frozen.
- Module 5 is forced into software timer mode.
- Writes to the Module 5 mode register (PCA0CPM5) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA counter state; the counter will run until the WDT is disabled. The PCA counter run control bit (CR) will read zero if the WDT is enabled but user software has not enabled the PCA counter. If a match occurs between PCA0CPH5 and PCA0H while the WDT is enabled, a reset will be generated. To prevent a WDT reset, the WDT may be updated with a write of any value to PCA0CPH5. Upon a PCA0CPH5 write, PCA0H plus the offset held in PCA0CPL5 is loaded into PCA0CPH5 (See Figure 28.11).

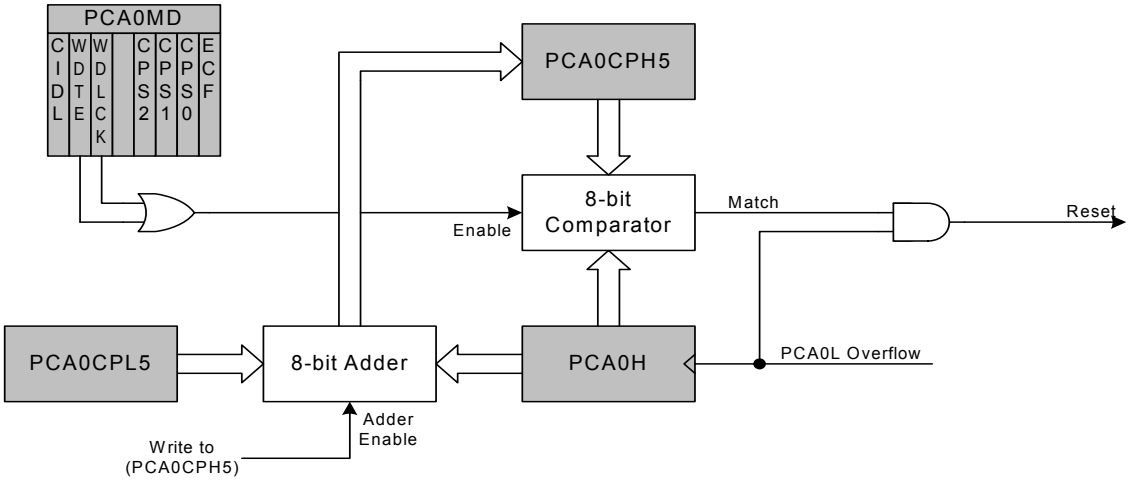



**Figure 28.11. PCA Module 5 with Watchdog Timer Enabled**

Note that the 8-bit offset held in PCA0CPH5 is compared to the upper byte of the 16-bit PCA counter. This offset value is the number of PCA0L overflows before a reset. Up to 256 PCA clocks may pass before the first PCA0L overflow occurs, depending on the value of the PCA0L when the update is performed. The total offset is then given (in PCA clocks) by Equation 28.5, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL5) + (256 - PCA0L)$$

**Equation 28.5. Watchdog Timer Offset in PCA Clocks**

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH5 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF5 flag (PCA0CN.5) while the WDT is enabled.

### 28.4.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

- Disable the WDT by writing a 0 to the WDTE bit.
- Select the desired PCA clock source (with the CPS2–CPS0 bits).
- Load PCA0CPL5 with the desired WDT update offset value.
- Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
- Enable the WDT by setting the WDTE bit to 1.
- Reset the WDT timer by writing to PCA0CPH5.

The PCA clock source and Idle mode select cannot be changed while the WDT is enabled. The watchdog timer is enabled by setting the WDTE or WDLCK bits in the PCA0MD register. When WDLCK is set, the WDT cannot be disabled until the next system reset. If WDLCK is not set, the WDT is disabled by clearing the WDTE bit.

The WDT is enabled following any reset. The PCA0 counter clock defaults to the system clock divided by 12, PCA0L defaults to 0x00, and PCA0CPL5 defaults to 0x00. Using Equation 28.5, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 28.3 lists some example timeout intervals for typical system clocks.

**Table 28.3. Watchdog Timer Timeout Intervals[1]**

| System Clock (Hz) | PCA0CPL5 | Timeout Interval (ms) |
|---|---|---|
| 24,500,000 | 255 | 32.1 |
| 24,500,000 | 128 | 16.2 |
| 24,500,000 | 32 | 4.1 |
| 3,062,500[2] | 255 | 257 |
| 3,062,500[2] | 128 | 129.5 |
| 3,062,500[2] | 32 | 33.1 |
| 32,000 | 255 | 24576 |
| 32,000 | 128 | 12384 |
| 32,000 | 32 | 3168 |

**Notes:**
1. Assumes SYSCLK/12 as the PCA clock source, and a PCA0L value of 0x00 at the update time.
2. Internal SYSCLK reset frequency = Internal Oscillator divided by 8.

## 28.5. Register Descriptions for PCA0

Following are detailed descriptions of the special function registers related to the operation of the PCA.

### SFR Definition 28.1. PCA0CN: PCA Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CF | CR | CCF5 | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | CF | **PCA Counter/Timer Overflow Flag.**<br>Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |
| 6 | CR | **PCA Counter/Timer Run Control.**<br>This bit enables/disables the PCA Counter/Timer.<br>0: PCA Counter/Timer disabled.<br>1: PCA Counter/Timer enabled. |
| 5:0 | CCF[5:0] | **PCA Module n Capture/Compare Flag.**<br>These bits are set by hardware when a match or capture occurs in the associated PCA Module n. When the CCFn interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |

## SFR Definition 28.2. PCA0MD: PCA Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CIDL | WDTE | WDLCK | | CPS2 | CPS1 | CPS0 | ECF |
| Type | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD9

| Bit | Name | Function |
|---|---|---|
| 7 | CIDL | **PCA Counter/Timer Idle Control.**<br>Specifies PCA behavior when CPU is in Idle Mode.<br>0: PCA continues to function normally while the system controller is in Idle Mode.<br>1: PCA operation is suspended while the system controller is in Idle Mode. |
| 6 | WDTE | **Watchdog Timer Enable.**<br>If this bit is set, PCA Module 2 is used as the watchdog timer.<br>0: Watchdog Timer disabled.<br>1: PCA Module 2 enabled as Watchdog Timer. |
| 5 | WDLCK | **Watchdog Timer Lock.**<br>This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.<br>0: Watchdog Timer Enable unlocked.<br>1: Watchdog Timer Enable locked. |
| 4 | Unused | Read = 0b, Write = don't care. |
| 3:1 | CPS[2:0] | **PCA Counter/Timer Pulse Select.**<br>These bits select the timebase source for the PCA counter<br>000: System clock divided by 12<br>001: System clock divided by 4<br>010: Timer 0 overflow<br>011: High-to-low transitions on ECI (max rate = system clock divided by 4)<br>100: System clock<br>101: External clock divided by 8 (synchronized with the system clock)<br>110: Reserved<br>111: Reserved |
| 0 | ECF | **PCA Counter/Timer Overflow Interrupt Enable.**<br>This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.<br>0: Disable the CF interrupt.<br>1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set. |
| **Note:** When the WDTE bit is set to 1, the other bits in the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled. | | |

## SFR Definition 28.3. PCA0PWM: PCA PWM Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ARSEL | ECOV | COVF | | | | CLSEL[1:0] | |
| Type | R/W | R/W | R/W | R | R | R | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xDF

| Bit | Name | Function |
|---|---|---|
| 7 | ARSEL | **Auto-Reload Register Select.**<br>This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function.<br>0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn.<br>1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn. |
| 6 | ECOV | **Cycle Overflow Interrupt Enable.**<br>This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt.<br>0: COVF will not generate PCA interrupts.<br>1: A PCA interrupt will be generated when COVF is set. |
| 5 | COVF | **Cycle Overflow Flag.**<br>This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software.<br>0: No overflow has occurred since the last time this bit was cleared.<br>1: An overflow has occurred since the last time this bit was cleared. |
| 4:2 | Unused | Read = 000b; Write = don't care. |
| 1:0 | CLSEL[1:0] | **Cycle Length Select.**<br>When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode.<br>00: 8 bits.<br>01: 9 bits.<br>10: 10 bits.<br>11: 11 bits. |

## SFR Definition 28.4. PCA0CPMn: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16n | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address, Page: PCA0CPM0 = 0xDA, 0x0; PCA0CPM1 = 0xDB, 0x0; PCA0CPM2 = 0xDC, 0x0
PCA0CPM3 = 0xDD, 0x0; PCA0CPM4 = 0xDE, 0x0; PCA0CPM5 = 0xCE, 0x0

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16n | **16-bit Pulse Width Modulation Enable.**<br>This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.<br>0: 8 to 11-bit PWM selected.<br>1: 16-bit PWM selected. |
| 6 | ECOMn | **Comparator Function Enable.**<br>This bit enables the comparator function for PCA module n when set to 1. |
| 5 | CAPPn | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture for PCA module n when set to 1. |
| 4 | CAPNn | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture for PCA module n when set to 1. |
| 3 | MATn | **Match Function Enable.**<br>This bit enables the match function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCFn bit in PCA0MD register to be set to logic 1. |
| 2 | TOGn | **Toggle Function Enable.**<br>This bit enables the toggle function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the logic level on the CEXn pin to toggle. If the PWMn bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWMn | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function for PCA module n when set to 1. When enabled, a pulse width modulated signal is output on the CEXn pin. 8 to 11-bit PWM is used if PWM16n is cleared; 16-bit mode is used if PWM16n is set to logic 1. If the TOGn bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCFn | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt.<br>0: Disable CCFn interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCFn is set. |

**Note:** When the WDTE bit is set to 1, the PCA0CPM5 register cannot be modified, and module 5 acts as the watchdog timer. To change the contents of the PCA0CPM5 register or the function of module 5, the Watchdog Timer must be disabled.

## SFR Definition 28.5. PCA0L: PCA Counter/Timer Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xF9

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[7:0] | **PCA Counter/Timer Low Byte.**<br>The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer. |

**Note:** When the WDTE bit is set to 1, the PCA0L register cannot be modified by software. To change the contents of the PCA0L register, the Watchdog Timer must first be disabled.

## SFR Definition 28.6. PCA0H: PCA Counter/Timer High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFA

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[15:8] | **PCA Counter/Timer High Byte.**<br>The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 28.1). |

**Note:** When the WDTE bit is set to 1, the PCA0H register cannot be modified by software. To change the contents of the PCA0H register, the Watchdog Timer must first be disabled.

## SFR Definition 28.7. PCA0CPLn: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPL0 = 0xFB, PCA0CPL1 = 0xE9, PCA0CPL2 = 0xEB,
PCA0CPL3 = 0xED, PCA0CPL4 = 0xFD, PCA0CPL5 = 0xD2

SFR Pages: PCA0CPL0 = 0x0, PCA0CPL1 = 0x0, PCA0CPL2 = 0x0,
PCA0CPL3 = 0x0, PCA0CPL4 = 0x0, PCA0CPL5 = 0x0

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[7:0] | **PCA Capture Module Low Byte.**<br>The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will clear the module's ECOMn bit to a 0. | | |

## SFR Definition 28.8. PCA0CPHn: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPH0 = 0xFC, PCA0CPH1 = 0xEA, PCA0CPH2 = 0xEC,
PCA0CPH3 = 0xEE, PCA0CPH4 = 0xFE, PCA0CPH5 = 0xD3

SFR Pages: PCA0CPH0 = 0x0, PCA0CPH1 = 0x0, PCA0CPH2 = 0x0,
PCA0CPH3 = 0x0, PCA0CPH4 = 0x0, PCA0CPH5 = 0x0

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[15:8] | **PCA Capture Module High Byte.**<br>The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will set the module's ECOMn bit to a 1. | | |

# 29. Device Specific Behavior

This chapter contains behavioral differences between the silicon revisions of Si100x devices. These differences do not affect the functionality or performance of most systems and are described below.

## 29.1. Device Identification

The Part Number Identifier on the top side of the device package can be used for decoding device information. The first character of the trace code identifies the silicon revision. On Si100x devices, the trace code will be the fifth letter on the second line. Figure 29.1 show how to find the part number on the top side of the device package.

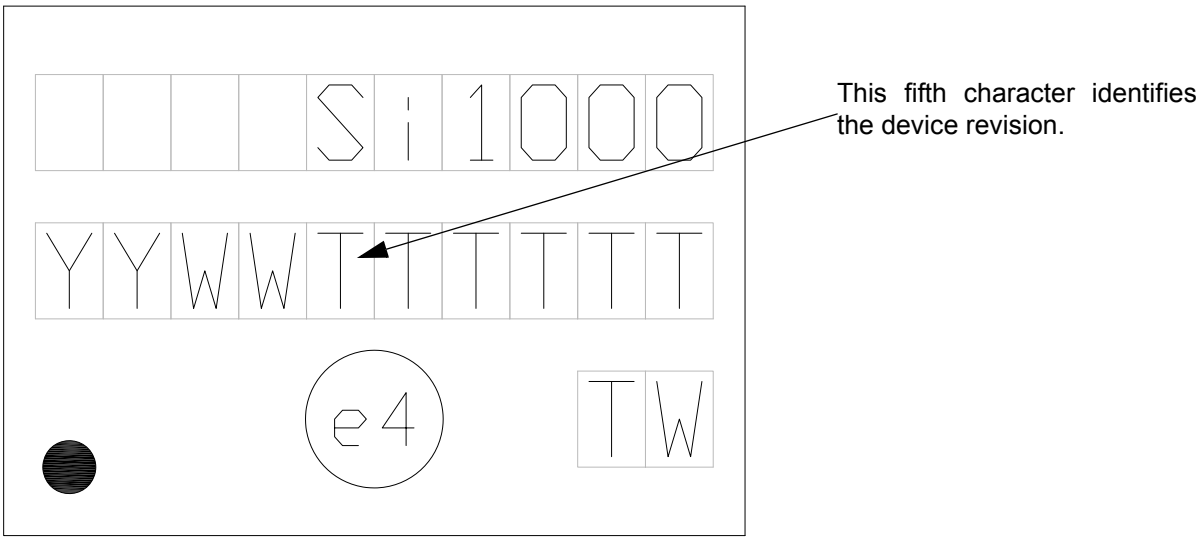



**Figure 29.1. Si100x Revision Information**

# 30. C2 Interface

Si1000/1/2/3/4/5 devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

## 30.1. C2 Interface Registers

The following describes the C2 registers necessary to perform flash programming through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

### C2 Register Definition 30.1. C2ADD: C2 Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2ADD[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | C2ADD[7:0] | **C2 Address.**<br>The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands. | |
| | | **Address** | **Description** |
| | | 0x00 | Selects the Device ID register for Data Read instructions |
| | | 0x01 | Selects the Revision ID register for Data Read instructions |
| | | 0x02 | Selects the C2 Flash Programming Control register for Data Read/Write instructions |
| | | 0xB4 | Selects the C2 Flash Programming Data register for Data Read/Write instructions |

## C2 Register Definition 30.2. DEVICEID: C2 Device ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DEVICEID[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

C2 Address: 0x00

| Bit | Name | Function |
|---|---|---|
| 7:0 | DEVICEID[7:0] | **Device ID.**<br>This read-only register returns the 8-bit device ID: 0x16 (Si1000/1/2/3/4/5). |

## C2 Register Definition 30.3. REVID: C2 Revision ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | REVID[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | Varies | Varies | Varies | Varies | Varies | Varies | Varies | Varies |

C2 Address: 0x01

| Bit | Name | Function |
|---|---|---|
| 7:0 | REVID[7:0] | **Revision ID.**<br>This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A. |

## C2 Register Definition 30.4. FPCTL: C2 Flash Programming Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FPCTL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C2 Address: 0x02

| Bit | Name | Function |
|---|---|---|
| 7:0 | FPCTL[7:0] | **Flash Programming Control Register.**<br>This register is used to enable flash programming via the C2 interface. To enable C2 flash programming, the following codes must be written in order: 0x02, 0x01. Note that once C2 flash programming is enabled, a system reset must be issued to resume normal operation. |

## C2 Register Definition 30.5. FPDAT: C2 Flash Programming Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FPDAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C2 Address: 0xB4

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | FPDAT[7:0] | **C2 Flash Programming Data Register.**<br>This register is used to pass flash commands, addresses, and data during C2 flash accesses. Valid commands are listed below. | |
| | | **Code** | **Command** |
| | | 0x06 | Flash Block Read |
| | | 0x07 | Flash Block Write |
| | | 0x08 | Flash Page Erase |
| | | 0x03 | Device Erase |

## 30.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and flash programming may be performed. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely “borrow” the C2CK ($\overline{\text{RST}}$) and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application. A typical isolation configuration is shown in Figure 30.1.

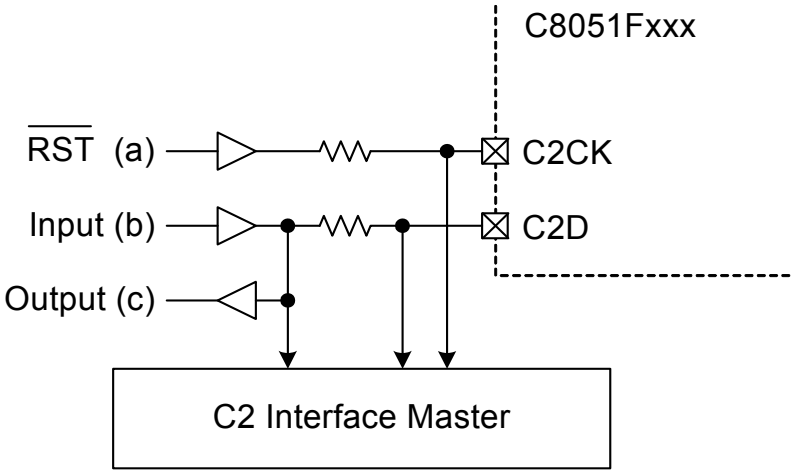



**Figure 30.1. Typical C2 Pin Sharing**

The configuration in Figure 30.1 assumes the following:

1. The user input (b) cannot change state while the target device is halted.
2. The $\overline{\text{RST}}$ pin on the target device is used as an input only.

Additional resistors may be necessary depending on the specific application.

# DOCUMENT CHANGE LIST

## Revision 0.2 to Revision 1.0

- Updated specification tables.
- Added Temperature Sensor Settling Time Specification.
- Updated power management section to indicate that the low power or precision oscillator must be selected when entering sleep or suspend mode. Also added a recommendation of executing two NOP instructions following the wake up from sleep mode.
- Updated ADC0 Burst Mode description.
- Updated DC0 inductor peak current equation.
- Added a note to the OSCICL register description.
- Updated Section 20.3 to indicate that when using Auto Reset, the Alarm match value should always be set to 2 counts less than the desired match value.
- Updated Port I/O chapter with additional clarification on 5 V and 3.3 V tolerance.
- Updated EZRadioPRO chapter.
- Updated QFN-42 landing diagram and stencil recommendations.

## Revision 1.0 to Revision 1.1

- Added revision E information to section “2. Ordering Information”
- Added LGA-42 package information to section “3. Pinout and Package Definitions” .
- Clarified conditions that apply to “VBAT Ramp Time for Power On” for one-cell mode vs. two-cell mode in Table 4.4.
- Added Si100x-E-GM POR Reset Time value to Table 4.20.
- Updated “5.2.3. Burst Mode” and Figure 5.3 to show difference in behavior between internal convert start signals and external CNVSTR signal.
- Added note about the need to ground the ADC mux before switching to the temperature sensor in “5.6. Temperature Sensor” and in SFR Definition 5.12. ADC0MX: ADC0 Input Channel Select
- Updated Table 8.1 on page 112 to correct number of clock cycles for “CJNE A, direct, rel”.
- Updated CPT0WK bit description in Register 14.1, “PMU0CF: Power Management Unit Configuration[1,2],” on page 157.
- Added section “15.2. 32-bit CRC Algorithm” to illustrate the 32-bit CRC algorithm.
- Updated links from section “19. Clocking Sources” to the proper Electrical Characteristics table.
- Changed the reset value of Register 00 in Table 23.9
- Updated Figure 23.6, “FSK vs. GFSK Spectrums,” on page 255 and Figure 23.7, “Direct Synchronous Mode Example,” on page 258
- Updated the Important Note in section “23.2. Interrupts ”
- Removed support for the Low Battery Detector in section “23.8.5. Low Battery Detector”
- Updated item 2 in section “23.13. Required Changes to Default Register Values”
- Corrected clock sources associated with T3XCLK settings in section “27.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload” , Figure 27.7, Figure 27.8, and Figure 27.9 to match the description in SFR Definition 27.13.
- Updated text in the first paragraph of the section “28. Programmable Counter Array” .
- Changed all instances of “ANT” to “ANT_A”.
- Updated the description of the XOUT and XIN pins.
- Updated QFN-42 stencil design and card assembly recommendations.
- Replaced incorrect PCA channel references from PCA0CPH2 to PCA0CPH5 in “28.4. Watchdog Timer Mode” and Figure 28.11, “PCA Module 5 with Watchdog Timer Enabled,” on page 368.
- Added section “29. Device Specific Behavior”

## Revision 1.1 to Revision 1.2

- Update Figure 3.3 to show asymmetry of center pad.
- Updated Figure 3.4 to show asymmetry of center pad.
- Updated Figure 3.5 dimensions and removed notes.
- Updated Table 3.4 dimensions.
- Updated Table 3.3 dimensions and removed Note 4.
- Added values for Si1004 and Si1005 to section 13.4.
- Updated Table 23.6 description of $V_{DD}$ Glitch Reset Pulse time.
- Updated Table 23.9 to show Device Version as 06h for Si100x-C-GM and 07h for Si100x-E-GM.

## Revision 1.2 to Revision 1.3

- Removed all references to Si100x-C-GM parts.
- Removed all references to Si100x-E-GM parts.
- Added ordering part numbers for Si100x-E-GM2 parts.
- Added package information for Si100x-E-GM2 parts.

## CONTACT INFORMATION

**Silicon Laboratories Inc.**

400 West Cesar Chavez
Austin, TX 78701

Please visit the Silicon Labs Technical Support web page:
https://www.silabs.com/support/pages/contacttechnicalsupport.aspx
and register to submit a technical support request.

**Patent Notice**

Silicon Labs invests in research and development to help our customers differentiate in the market with innovative low-power, small size, analog-intensive mixed-signal solutions. Silicon Labs' extensive patent portfolio is a testament to our unique approach and world-class engineering team.

The information in this document is believed to be accurate in all respects at the time of publication but is subject to change without notice. Silicon Laboratories assumes no responsibility for errors and omissions, and disclaims responsibility for any consequences resulting from the use of information included herein. Additionally, Silicon Laboratories assumes no responsibility for the functioning of undescribed features or parameters. Silicon Laboratories reserves the right to make changes without further notice. Silicon Laboratories makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Silicon Laboratories assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Silicon Laboratories products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Silicon Laboratories product could create a situation where personal injury or death may occur. Should Buyer purchase or use Silicon Laboratories products for any such unintended or unauthorized application, Buyer shall indemnify and hold Silicon Laboratories harmless against all claims and damages.

Silicon Laboratories and Silicon Labs are trademarks of Silicon Laboratories Inc.
Other products or brandnames mentioned herein are trademarks or registered trademarks of their respective holders.