# C8051F410/1/2/3

## 2.0 V, 32/16 kB Flash, smaRTClock, 12-bit ADC

### Analog Peripherals
- **12-Bit ADC**
  - ±1 LSB INL; no missing codes
  - Programmable throughput up to 200 ksps
  - Up to 24 external inputs
  - Data dependent windowed interrupt generator
  - Built-in temperature sensor (±3 °C)
- **Two 12-Bit Current Mode DACs**
- **Two Comparators**
  - Programmable hysteresis and response time
  - Configurable as wake-up or reset source
- **POR/Brownout Detector**
- **Voltage Reference—1.5, 2.2 V (programmable)**

### 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

### Supply Voltage 2.0 to 5.25 V
- Built-in LDO regulator: 2.1 or 2.5 V

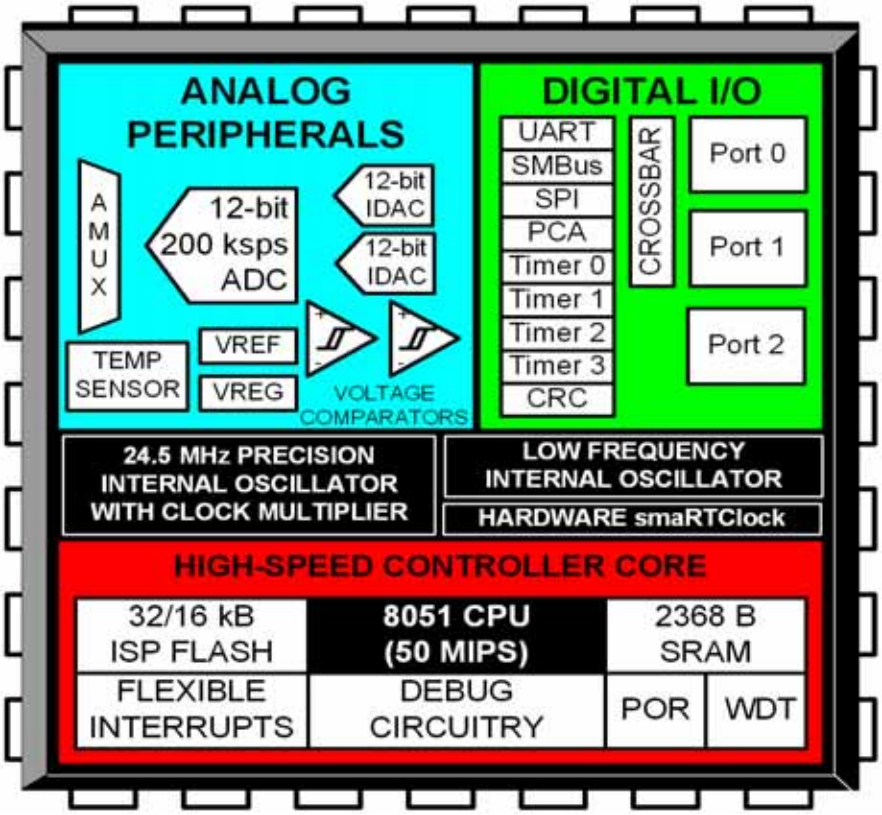
### High Speed 8051 µC Core
- Pipelined instruction architecture; executes 70% of instructions in 1 or 2 system clocks
- Up to **50 MIPS** throughput with 50 MHz system clock
- Expanded interrupt handler

### Memory
- 2304 bytes internal data RAM (256 + 2048)
- 32/16 kB Flash; In-system programmable in 512 byte sectors
- 64 bytes battery-backed RAM (smaRTClock)

### Digital Peripherals
- 24 port I/O; push-pull or open-drain, up to 5.25 V tolerance
- Hardware SMBus™ (I2C™ Compatible), SPI™, and UART serial ports available concurrently
- Four general purpose 16-bit counter/timers
- Programmable 16-bit counter/timer array with six capture/compare modules, WDT
- Hardware smaRTClock operates down to 1 V with 64 bytes battery-backed RAM and backup voltage regulator

### Clock Sources
- Internal oscillators: 24.5 MHz 2% accuracy supports UART operation; clock multiplier up to 50 MHz
- External oscillator: Crystal, RC, C, or Clock (1 or 2 pin modes)
- smaRTClock oscillator: 32 kHz Crystal or self-resonant oscillator
- Can switch between clock sources on-the-fly

**32-Pin LQFP or 28-Pin 5 x 5 QFN**
**Temperature Range: –40 to +85 °C**





Copyright © 2008 by Silicon Laboratories

## NOTES:

## Table of Contents

**1. System Overview ... 19**
1.1. CIP-51™ Microcontroller ... 25
1.1.1. Fully 8051 Compatible Instruction Set ... 25
1.1.2. Improved Throughput ... 25
1.1.3. Additional Features ... 25
1.2. On-Chip Debug Circuitry ... 26
1.3. On-Chip Memory ... 27
1.4. Operating Modes ... 28
1.5. 12-Bit Analog to Digital Converter ... 29
1.6. Two 12-bit Current-Mode DACs ... 29
1.7. Programmable Comparators ... 30
1.8. Cyclic Redundancy Check Unit ... 31
1.9. Voltage Regulator ... 31
1.10. Serial Ports ... 31
1.11. smaRTClock (Real Time Clock) ... 32
1.12. Port Input/Output ... 33
1.13. Programmable Counter Array ... 34
**2. Absolute Maximum Ratings ... 35**
**3. Global DC Electrical Characteristics ... 36**
**4. Pinout and Package Definitions ... 41**
**5. 12-Bit ADC (ADC0) ... 51**
5.1. Analog Multiplexer ... 51
5.2. Temperature Sensor ... 52
5.3. ADC0 Operation ... 52
5.3.1. Starting a Conversion ... 53
5.3.2. Tracking Modes ... 53
5.3.3. Timing ... 54
5.3.4. Burst Mode ... 56
5.3.5. Output Conversion Code ... 57
5.3.6. Settling Time Requirements ... 58
5.4. Programmable Window Detector ... 63
5.4.1. Window Detector In Single-Ended Mode ... 66
**6. 12-Bit Current Mode DACs (IDA0 and IDA1) ... 69**
6.1. IDAC Output Scheduling ... 69
6.1.1. Update Output On-Demand ... 69
6.1.2. Update Output Based on Timer Overflow ... 70
6.1.3. Update Output Based on CNVSTR Edge ... 70
6.2. IDAC Output Mapping ... 70
6.3. IDAC External Pin Connections ... 73
**7. Voltage Reference ... 77**
**8. Voltage Regulator (REG0) ... 81**
**9. Comparators ... 83**

**10. CIP-51 Microcontroller ..... 93**
10.1. Instruction Set ..... 94
10.1.1. Instruction and CPU Timing ..... 94
10.1.2. MOVX Instruction and Program Memory ..... 95
10.2. Register Descriptions ..... 98
10.3. Power Management Modes ..... 101
10.3.1. Idle Mode ..... 102
10.3.2. Stop Mode ..... 102
10.3.3. Suspend Mode ..... 102
**11. Memory Organization and SFRs ..... 103**
11.1. Program Memory ..... 103
11.2. Data Memory ..... 104
11.3. General Purpose Registers ..... 104
11.4. Bit Addressable Locations ..... 104
11.5. Stack ..... 104
11.6. Special Function Registers ..... 105
**12. Interrupt Handler ..... 110**
12.1. MCU Interrupt Sources and Vectors ..... 110
12.2. Interrupt Priorities ..... 110
12.3. Interrupt Latency ..... 110
12.4. Interrupt Register Descriptions ..... 112
12.5. External Interrupts ..... 117
**13. Prefetch Engine ..... 119**
**14. Cyclic Redundancy Check Unit (CRC0) ..... 121**
14.1. 16-bit CRC Algorithm ..... 121
14.2. 32-bit CRC Algorithm ..... 123
14.3. Preparing for a CRC Calculation ..... 124
14.4. Performing a CRC Calculation ..... 124
14.5. Accessing the CRC0 Result ..... 124
14.6. CRC0 Bit Reverse Feature ..... 124
**15. Reset Sources ..... 127**
15.1. Power-On Reset ..... 128
15.2. Power-Fail Reset / VDD Monitor ..... 129
15.3. External Reset ..... 130
15.4. Missing Clock Detector Reset ..... 130
15.5. Comparator0 Reset ..... 130
15.6. PCA Watchdog Timer Reset ..... 131
15.7. Flash Error Reset ..... 131
15.8. smaRTClock (Real Time Clock) Reset ..... 132
15.9. Software Reset ..... 132
**16. Flash Memory ..... 135**
16.1. Programming The Flash Memory ..... 135
16.1.1. Flash Lock and Key Functions ..... 135
16.1.2. Flash Erase Procedure ..... 135
16.1.3. Flash Write Procedure ..... 136

16.2.Non-volatile Data Storage ..... 137
16.3.Security Options ..... 137
16.4.Flash Write and Erase Guidelines ..... 139
16.4.1.VDD Maintenance and the VDD Monitor ..... 139
16.4.2.16.4.2 PSWE Maintenance ..... 140
16.4.3.System Clock ..... 140
16.5.Flash Read Timing ..... 142
**17. External RAM ..... 145**
**18. Port Input/Output ..... 147**
18.1.Priority Crossbar Decoder ..... 149
18.2.Port I/O Initialization ..... 151
18.3.General Purpose Port I/O ..... 154
**19. Oscillators ..... 165**
19.1.Programmable Internal Oscillator ..... 165
19.1.1.Internal Oscillator Suspend Mode ..... 166
19.2.External Oscillator Drive Circuit ..... 168
19.2.1.Clocking Timers Directly Through the External Oscillator ..... 168
19.2.2.External Crystal Example ..... 168
19.2.3.External RC Example ..... 170
19.2.4.External Capacitor Example ..... 170
19.3.Clock Multiplier ..... 172
19.4.System Clock Selection ..... 174
**20. smaRTClock (Real Time Clock) ..... 177**
20.1.smaRTClock Interface ..... 178
20.1.1.smaRTClock Lock and Key Functions ..... 178
20.1.2.Using RTC0ADR and RTC0DAT to Access smaRTClock Internal Registers ..... 178
20.1.3.smaRTClock Interface Autoread Feature ..... 178
20.1.4.RTC0ADR Autoincrement Feature ..... 179
20.2.smaRTClock Clocking Sources ..... 182
20.2.1.Using the smaRTClock Oscillator in Crystal Mode ..... 182
20.2.2.Using the smaRTClock Oscillator in Self-Oscillate Mode ..... 182
20.2.3.Automatic Gain Control (Crystal Mode Only) ..... 183
20.2.4.smaRTClock Bias Doubling ..... 183
20.2.5.smaRTClock Missing Clock Detector ..... 183
20.3.smaRTClock Timer and Alarm Function ..... 185
20.3.1.Setting and Reading the smaRTClock Timer Value ..... 185
20.3.2.Setting a smaRTClock Alarm ..... 186
20.4.Backup Regulator and RAM ..... 187
**21. SMBus ..... 191**
21.1.Supporting Documents ..... 192
21.2.SMBus Configuration ..... 192
21.3.SMBus Operation ..... 192
21.3.1.Arbitration ..... 193
21.3.2.Clock Low Extension ..... 193

21.3.3.SCL Low Timeout ... 194
21.3.4.SCL High (SMBus Free) Timeout ... 194
21.4.Using the SMBus ... 194
21.4.1.SMBus Configuration Register ... 195
21.4.2.SMB0CN Control Register ... 198
21.4.3.Data Register ... 201
21.5.SMBus Transfer Modes ... 201
21.5.1.Master Transmitter Mode ... 201
21.5.2.Master Receiver Mode ... 202
21.5.3.Slave Receiver Mode ... 203
21.5.4.Slave Transmitter Mode ... 204
21.6.SMBus Status Decoding ... 204
**22. UART0 ... 207**
22.1.Enhanced Baud Rate Generation ... 208
22.2.Operational Modes ... 209
22.2.1.8-Bit UART ... 209
22.2.2.9-Bit UART ... 210
22.3.Multiprocessor Communications ... 210
**23. Enhanced Serial Peripheral Interface (SPI0) ... 217**
23.1.Signal Descriptions ... 218
23.1.1.Master Out, Slave In (MOSI) ... 218
23.1.2.Master In, Slave Out (MISO) ... 218
23.1.3.Serial Clock (SCK) ... 218
23.1.4.Slave Select (NSS) ... 218
23.2.SPI0 Master Mode Operation ... 219
23.3.SPI0 Slave Mode Operation ... 220
23.4.SPI0 Interrupt Sources ... 221
23.5.Serial Clock Timing ... 221
23.6.SPI Special Function Registers ... 222
**24. Timers ... 231**
24.1.Timer 0 and Timer 1 ... 231
24.1.1.Mode 0: 13-bit Counter/Timer ... 231
24.1.2.Mode 1: 16-bit Counter/Timer ... 233
24.1.3.Mode 2: 8-bit Counter/Timer with Auto-Reload ... 233
24.1.4.Mode 3: Two 8-bit Counter/Timers (Timer 0 Only) ... 234
24.2.Timer 2 ... 239
24.2.1.16-bit Timer with Auto-Reload ... 239
24.2.2.8-bit Timers with Auto-Reload ... 240
24.2.3.External/smaRTClock Capture Mode ... 241
24.3.Timer 3 ... 244
24.3.1.16-bit Timer with Auto-Reload ... 244
24.3.2.8-bit Timers with Auto-Reload ... 245
24.3.3.External/smaRTClock Capture Mode ... 246
**25. Programmable Counter Array (PCA0) ... 249**
25.1.PCA Counter/Timer ... 250

25.2.Capture/Compare Modules ..... 251
25.2.1.Edge-triggered Capture Mode..... 252
25.2.2.Software Timer (Compare) Mode..... 253
25.2.3.High Speed Output Mode..... 254
25.2.4.Frequency Output Mode ..... 255
25.2.5.8-Bit Pulse Width Modulator Mode..... 256
25.2.6.16-Bit Pulse Width Modulator Mode..... 257
25.3.Watchdog Timer Mode ..... 257
25.3.1.Watchdog Timer Operation..... 258
25.3.2.Watchdog Timer Usage ..... 259
25.4.Register Descriptions for PCA..... 261
**26. C2 Interface ..... 265**
26.1.C2 Interface Registers..... 265
26.2.C2 Pin Sharing ..... 267

## NOTES:

## List of Figures

**1. System Overview**
Figure 1.1. C8051F410 Block Diagram ..... 21
Figure 1.2. C8051F411 Block Diagram ..... 22
Figure 1.3. C8051F412 Block Diagram ..... 23
Figure 1.4. C8051F413 Block Diagram ..... 24
Figure 1.5. Development/In-System Debug Diagram ..... 26
Figure 1.6. Memory Map ..... 27
Figure 1.7. 12-Bit ADC Block Diagram ..... 29
Figure 1.8. IDAC Block Diagram ..... 30
Figure 1.9. Comparators Block Diagram ..... 31
Figure 1.10. smaRTClock Block Diagram ..... 32
Figure 1.11. Port I/O Functional Block Diagram ..... 33
Figure 1.12. PCA Block Diagram ..... 34
**2. Absolute Maximum Ratings**
**3. Global DC Electrical Characteristics**
**4. Pinout and Package Definitions**
Figure 4.1. LQFP-32 Pinout Diagram (Top View) ..... 44
Figure 4.2. QFN-28 Pinout Diagram (Top View) ..... 45
Figure 4.3. LQFP-32 Package Diagram ..... 46
Figure 4.4. LQFP-32 Recommended PCB Land Pattern ..... 47
Figure 4.5. QFN-28 Package Drawing ..... 48
Figure 4.6. QFN-28 Recommended PCB Land Pattern ..... 49
**5. 12-Bit ADC (ADC0)**
Figure 5.1. ADC0 Functional Block Diagram ..... 51
Figure 5.2. Typical Temperature Sensor Transfer Function ..... 52
Figure 5.3. ADC0 Tracking Modes ..... 54
Figure 5.4. 12-Bit ADC Tracking Mode Example ..... 55
Figure 5.5. 12-Bit ADC Burst Mode Example with Repeat Count Set to 4 ..... 56
Figure 5.6. ADC0 Equivalent Input Circuits ..... 58
Figure 5.7. ADC Window Compare Example: Right-Justified Single-Ended Data ... 66
Figure 5.8. ADC Window Compare Example: Left-Justified Single-Ended Data ..... 66
**6. 12-Bit Current Mode DACs (IDA0 and IDA1)**
Figure 6.1. IDAC Functional Block Diagram ..... 69
Figure 6.2. IDAC Data Word Mapping ..... 70
Figure 6.3. IDAC Pin Connections ..... 74
**7. Voltage Reference**
Figure 7.1. Voltage Reference Functional Block Diagram ..... 77
**8. Voltage Regulator (REG0)**
Figure 8.1. External Capacitors for Voltage Regulator Input/Output ..... 81
Figure 8.2. External Capacitors for Voltage Regulator Input/Output ..... 81
**9. Comparators**
Figure 9.1. Comparator0 Functional Block Diagram ..... 83
Figure 9.2. Comparator1 Functional Block Diagram ..... 84

Figure 9.3. Comparator Hysteresis Plot ..... 85
**10. CIP-51 Microcontroller**
Figure 10.1. CIP-51 Block Diagram ..... 93
**11. Memory Organization and SFRs**
Figure 11.1. Memory Map ..... 103
**12. Interrupt Handler**
**13. Prefetch Engine**
**14. Cyclic Redundancy Check Unit (CRC0)**
Figure 14.1. CRC0 Block Diagram ..... 121
Figure 14.2. Bit Reverse Register ..... 124
**15. Reset Sources**
Figure 15.1. Reset Sources ..... 127
Figure 15.2. Power-On and VDD Monitor Reset Timing ..... 128
**16. Flash Memory**
Figure 16.1. Flash Program Memory Map ..... 137
**17. External RAM**
**18. Port Input/Output**
Figure 18.1. Port I/O Functional Block Diagram ..... 147
Figure 18.2. Port I/O Cell Block Diagram ..... 148
Figure 18.3. Crossbar Priority Decoder with No Pins Skipped ..... 149
Figure 18.4. Crossbar Priority Decoder with Crystal Pins Skipped ..... 150
Figure 18.5. Port 0 Input Overdrive Current Range ..... 152
**19. Oscillators**
Figure 19.1. Oscillator Diagram ..... 165
Figure 19.2. 32.768 kHz External Crystal Example ..... 169
Figure 19.3. Example Clock Multiplier Output ..... 172
**20. smaRTClock (Real Time Clock)**
Figure 20.1. smaRTClock Block Diagram ..... 177
**21. SMBus**
Figure 21.1. SMBus Block Diagram ..... 191
Figure 21.2. Typical SMBus Configuration ..... 192
Figure 21.3. SMBus Transaction ..... 193
Figure 21.4. Typical SMBus SCL Generation ..... 196
Figure 21.5. Typical Master Transmitter Sequence ..... 202
Figure 21.6. Typical Master Receiver Sequence ..... 202
Figure 21.7. Typical Slave Receiver Sequence ..... 203
Figure 21.8. Typical Slave Transmitter Sequence ..... 204
**22. UART0**
Figure 22.1. UART0 Block Diagram ..... 207
Figure 22.2. UART0 Baud Rate Logic ..... 208
Figure 22.3. UART Interconnect Diagram ..... 209
Figure 22.4. 8-Bit UART Timing Diagram ..... 209
Figure 22.5. 9-Bit UART Timing Diagram ..... 210
Figure 22.6. UART Multi-Processor Mode Interconnect Diagram ..... 211

**23. Enhanced Serial Peripheral Interface (SPI0)**
Figure 23.1. SPI Block Diagram ..... 217
Figure 23.2. Multiple-Master Mode Connection Diagram ..... 220
Figure 23.3. 3-Wire Single Master and Slave Mode Connection Diagram ..... 220
Figure 23.4. 4-Wire Single Master and Slave Mode Connection Diagram ..... 220
Figure 23.5. Data/Clock Timing Relationship ..... 222
Figure 23.6. SPI Master Timing (CKPHA = 0) ..... 227
Figure 23.7. SPI Master Timing (CKPHA = 1) ..... 227
Figure 23.8. SPI Slave Timing (CKPHA = 0) ..... 228
Figure 23.9. SPI Slave Timing (CKPHA = 1) ..... 228
**24. Timers**
Figure 24.1. T0 Mode 0 Block Diagram ..... 232
Figure 24.2. T0 Mode 2 Block Diagram ..... 233
Figure 24.3. T0 Mode 3 Block Diagram ..... 234
Figure 24.4. Timer 2 16-Bit Mode Block Diagram ..... 239
Figure 24.5. Timer 2 8-Bit Mode Block Diagram ..... 240
Figure 24.6. Timer 2 Capture Mode Block Diagram ..... 241
Figure 24.7. Timer 3 16-Bit Mode Block Diagram ..... 244
Figure 24.8. Timer 3 8-Bit Mode Block Diagram ..... 245
Figure 24.9. Timer 3 Capture Mode Block Diagram ..... 246
**25. Programmable Counter Array (PCA0)**
Figure 25.1. PCA Block Diagram ..... 249
Figure 25.2. PCA Counter/Timer Block Diagram ..... 250
Figure 25.3. PCA Interrupt Block Diagram ..... 251
Figure 25.4. PCA Capture Mode Diagram ..... 252
Figure 25.5. PCA Software Timer Mode Diagram ..... 253
Figure 25.6. PCA High-Speed Output Mode Diagram ..... 254
Figure 25.7. PCA Frequency Output Mode ..... 255
Figure 25.8. PCA 8-Bit PWM Mode Diagram ..... 256
Figure 25.9. PCA 16-Bit PWM Mode ..... 257
Figure 25.10. PCA Module 5 with Watchdog Timer Enabled ..... 258
**26. C2 Interface**
Figure 26.1. Typical C2 Pin Sharing ..... 267

## NOTES:

## List of Tables

**1. System Overview**
Table 1.1. Product Selection Guide ..... 20
Table 1.2. Operating Modes Summary ..... 28
**2. Absolute Maximum Ratings**
Table 2.1.Absolute Maximum Ratings ..... 35
**3. Global DC Electrical Characteristics**
Table 3.1.Global DC Electrical Characteristics. ..... 36
Table 3.2. Index to Electrical Characteristics Tables ..... 39
**4. Pinout and Package Definitions**
Table 4.1. Pin Definitions for the C8051F41x ..... 41
Table 4.2. LQFP-32 Package Dimensions ..... 46
Table 4.3. LQFP-32 PCB Land Pattern Dimensions ..... 47
Table 4.4. QFN-28 Package Dimensions ..... 48
Table 4.5. QFN-28 PCB Land Pattern Dimensions ..... 49
**5. 12-Bit ADC (ADC0)**
Table 5.1. ADC0 Examples of Right- and Left-Justified Samples ..... 57
Table 5.2. ADC0 Repeat Count Examples at Various Input Voltages ..... 57
Table 5.3.ADC0 Electrical Characteristics ($V_{DD}$ = 2.5 V, $V_{REF}$ = 2.2 V) ..... 67
Table 5.4.ADC0 Electrical Characteristics ($V_{DD}$ = 2.1 V, $V_{REF}$ = 1.5 V) ..... 68
**6. 12-Bit Current Mode DACs (IDA0 and IDA1)**
Table 6.1.IDAC Electrical Characteristics. ..... 75
**7. Voltage Reference**
Table 7.1.Voltage Reference Electrical Characteristics ..... 79
**8. Voltage Regulator (REG0)**
Table 8.1.Voltage Regulator Electrical Specifications ..... 82
**9. Comparators**
Table 9.1.Comparator Electrical Characteristics ..... 92
**10. CIP-51 Microcontroller**
Table 10.1. CIP-51 Instruction Set Summary ..... 95
**11. Memory Organization and SFRs**
Table 11.1. Special Function Register (SFR) Memory Map ..... 105
Table 11.2. Special Function Registers ..... 106
**12. Interrupt Handler**
Table 12.1. Interrupt Summary ..... 111
**13. Prefetch Engine**
**14. Cyclic Redundancy Check Unit (CRC0)**
Table 14.1. Example 16-bit CRC Outputs ..... 122
Table 14.2. Example 32-bit CRC Outputs ..... 124
**15. Reset Sources**
Table 15.1.Reset Electrical Characteristics ..... 134
**16. Flash Memory**
Table 16.1. Flash Security Summary ..... 138
Table 16.2.Flash Electrical Characteristics ..... 143

**17. External RAM**
**18. Port Input/Output**
Table 18.1.Port I/O DC Electrical Characteristics. . . . . 163
**19. Oscillators**
Table 19.1.Oscillator Electrical Characteristics . . . . . 175
**20. smaRTClock (Real Time Clock)**
Table 20.1. smaRTClock Internal Registers ..... 179
**21. SMBus**
Table 21.1. SMBus Clock Source Selection ..... 195
Table 21.2. Minimum SDA Setup and Hold Times ..... 196
Table 21.3. Sources for Hardware Changes to SMB0CN ..... 200
Table 21.4. SMBus Status Decoding ..... 205
**22. UART0**
Table 22.1. Timer Settings for Standard Baud Rates Using the Internal Oscillator ..... 214
Table 22.2. Timer Settings for Standard Baud Rates Using an External 25.0 MHz Oscillator ..... 214
Table 22.3. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator ..... 215
Table 22.4. Timer Settings for Standard Baud Rates Using an External 18.432 MHz Oscillator ..... 215
Table 22.5. Timer Settings for Standard Baud Rates Using an External 11.0592 MHz Oscillator ..... 216
Table 22.6. Timer Settings for Standard Baud Rates Using an External 3.6864 MHz Oscillator ..... 216
**23. Enhanced Serial Peripheral Interface (SPI0)**
Table 23.1.SPI Slave Timing Parameters . . . . . 229
**24. Timers**
**25. Programmable Counter Array (PCA0)**
Table 25.1. PCA Timebase Input Options ..... 250
Table 25.2. PCA0CPM Register Settings for PCA Capture/Compare Modules .... 251
Table 25.3. Watchdog Timer Timeout Intervals ..... 260
**26. C2 Interface**

## List of Registers

SFR Definition 5.1. ADC0MX: ADC0 Channel Select . . . . . 59
SFR Definition 5.2. ADC0CF: ADC0 Configuration . . . . . 60
SFR Definition 5.3. ADC0H: ADC0 Data Word MSB . . . . . 61
SFR Definition 5.4. ADC0L: ADC0 Data Word LSB . . . . . 61
SFR Definition 5.5. ADC0CN: ADC0 Control . . . . . 62
SFR Definition 5.6. ADC0TK: ADC0 Tracking Mode Select . . . . . 63
SFR Definition 5.7. ADC0GTH: ADC0 Greater-Than Data High Byte . . . . . 64
SFR Definition 5.8. ADC0GTL: ADC0 Greater-Than Data Low Byte . . . . . 64
SFR Definition 5.9. ADC0LTH: ADC0 Less-Than Data High Byte . . . . . 65
SFR Definition 5.10. ADC0LTL: ADC0 Less-Than Data Low Byte . . . . . 65
SFR Definition 6.1. IDA0CN: IDA0 Control . . . . . 71
SFR Definition 6.2. IDA0H: IDA0 Data High Byte . . . . . 71
SFR Definition 6.3. IDA0L: IDA0 Data Low Byte . . . . . 72
SFR Definition 6.4. IDA1CN: IDA1 Control . . . . . 72
SFR Definition 6.5. IDA1H: IDA0 Data High Byte . . . . . 73
SFR Definition 6.6. IDA1L: IDA1 Data Low Byte . . . . . 73
SFR Definition 7.1. REF0CN: Reference Control . . . . . 78
SFR Definition 8.1. REG0CN: Regulator Control . . . . . 82
SFR Definition 9.1. CPT0CN: Comparator0 Control . . . . . 86
SFR Definition 9.2. CPT0MX: Comparator0 MUX Selection . . . . . 87
SFR Definition 9.3. CPT0MD: Comparator0 Mode Selection . . . . . 88
SFR Definition 9.4. CPT1MX: Comparator1 MUX Selection . . . . . 89
SFR Definition 9.5. CPT1MD: Comparator1 Mode Selection . . . . . 90
SFR Definition 9.6. CPT1CN: Comparator1 Control . . . . . 91
SFR Definition 10.1. SP: Stack Pointer . . . . . 98
SFR Definition 10.2. DPL: Data Pointer Low Byte . . . . . 99
SFR Definition 10.3. DPH: Data Pointer High Byte . . . . . 99
SFR Definition 10.4. PSW: Program Status Word . . . . . 100
SFR Definition 10.5. ACC: Accumulator . . . . . 101
SFR Definition 10.6. B: B Register . . . . . 101
SFR Definition 10.7. PCON: Power Control . . . . . 102
SFR Definition 12.1. IE: Interrupt Enable . . . . . 112
SFR Definition 12.2. IP: Interrupt Priority . . . . . 113
SFR Definition 12.3. EIE1: Extended Interrupt Enable 1 . . . . . 114
SFR Definition 12.4. EIP1: Extended Interrupt Priority 1 . . . . . 115
SFR Definition 12.5. EIE2: Extended Interrupt Enable 2 . . . . . 116
SFR Definition 12.6. EIP2: Extended Interrupt Priority 2 . . . . . 116
SFR Definition 12.7. IT01CF: INT0/INT1 Configuration . . . . . 118
SFR Definition 13.1. PFE0CN: Prefetch Engine Control . . . . . 119
SFR Definition 14.1. CRC0CN: CRC0 Control . . . . . 125
SFR Definition 14.2. CRC0IN: CRC0 Data Input . . . . . 125
SFR Definition 14.3. CRC0DAT: CRC0 Data Output . . . . . 126
SFR Definition 14.4. CRC0FLIP: CRC0 Bit Flip . . . . . 126

SFR Definition 15.1. VDM0CN: VDD Monitor Control . . . . . 130
SFR Definition 15.2. RSTSRC: Reset Source . . . . . 133
SFR Definition 16.1. PSCTL: Program Store R/W Control . . . . . 141
SFR Definition 16.2. FLKEY: Flash Lock and Key . . . . . 141
SFR Definition 16.3. FLSCL: Flash Scale . . . . . 142
SFR Definition 16.4. ONESHOT: Flash Oneshot Period . . . . . 143
SFR Definition 17.1. EMI0CN: External Memory Interface Control . . . . . 145
SFR Definition 18.1. XBR0: Port I/O Crossbar Register 0 . . . . . 153
SFR Definition 18.2. XBR1: Port I/O Crossbar Register 1 . . . . . 154
SFR Definition 18.3. P0: Port0 . . . . . 155
SFR Definition 18.4. P0MDIN: Port0 Input Mode . . . . . 155
SFR Definition 18.5. P0MDOUT: Port0 Output Mode . . . . . 156
SFR Definition 18.6. P0SKIP: Port0 Skip . . . . . 156
SFR Definition 18.7. P0MAT: Port0 Match . . . . . 157
SFR Definition 18.8. P0MASK: Port0 Mask . . . . . 157
SFR Definition 18.9. P0ODEN: Port0 Overdrive Mode . . . . . 157
SFR Definition 18.10. P1: Port1 . . . . . 158
SFR Definition 18.11. P1MDIN: Port1 Input Mode . . . . . 158
SFR Definition 18.12. P1MDOUT: Port1 Output Mode . . . . . 159
SFR Definition 18.13. P1SKIP: Port1 Skip . . . . . 159
SFR Definition 18.14. P1MAT: Port1 Match . . . . . 160
SFR Definition 18.15. P1MASK: Port1 Mask . . . . . 160
SFR Definition 18.16. P2: Port2 . . . . . 161
SFR Definition 18.17. P2MDIN: Port2 Input Mode . . . . . 161
SFR Definition 18.18. P2MDOUT: Port2 Output Mode . . . . . 162
SFR Definition 18.19. P2SKIP: Port2 Skip . . . . . 162
SFR Definition 19.1. OSCICN: Internal Oscillator Control . . . . . 167
SFR Definition 19.2. OSCICL: Internal Oscillator Calibration . . . . . 167
SFR Definition 19.3. OSCXCN: External Oscillator Control . . . . . 171
SFR Definition 19.4. CLKMUL: Clock Multiplier Control . . . . . 173
SFR Definition 19.5. CLKSEL: Clock Select . . . . . 174
SFR Definition 20.1. RTC0KEY: smaRTClock Lock and Key . . . . . 180
SFR Definition 20.2. RTC0ADR: smaRTClock Address . . . . . 181
SFR Definition 20.3. RTC0DAT: smaRTClock Data . . . . . 182
Internal Register Definition 20.4. RTC0CN: smaRTClock Control . . . . . 184
Internal Register Definition 20.5. RTC0XCN: smaRTClock Oscillator Control . . . . . 185
Internal Register Definition 20.6. CAPTUREn: smaRTClock Timer Capture . . . . . 186
Internal Register Definition 20.7. ALARMn: smaRTClock Alarm . . . . . 187
Internal Register Definition 20.8. RAMADDR: smaRTClock Backup RAM Address . . 187
Internal Register Definition 20.9. RAMDATA: smaRTClock Backup RAM Data . . . . . 188
SFR Definition 21.1. SMB0CF: SMBus Clock/Configuration . . . . . 197
SFR Definition 21.2. SMB0CN: SMBus Control . . . . . 199
SFR Definition 21.3. SMB0DAT: SMBus Data . . . . . 201
SFR Definition 22.1. SCON0: Serial Port 0 Control . . . . . 212
SFR Definition 22.2. SBUF0: Serial (UART0) Port Data Buffer . . . . . 213

SFR Definition 23.1. SPI0CFG: SPI0 Configuration . . . . . 223
SFR Definition 23.2. SPI0CN: SPI0 Control . . . . . 224
SFR Definition 23.3. SPI0CKR: SPI0 Clock Rate . . . . . 225
SFR Definition 23.4. SPI0DAT: SPI0 Data . . . . . 226
SFR Definition 24.1. TCON: Timer Control . . . . . 235
SFR Definition 24.2. TMOD: Timer Mode . . . . . 236
SFR Definition 24.3. CKCON: Clock Control . . . . . 237
SFR Definition 24.4. TL0: Timer 0 Low Byte . . . . . 238
SFR Definition 24.5. TL1: Timer 1 Low Byte . . . . . 238
SFR Definition 24.6. TH0: Timer 0 High Byte . . . . . 238
SFR Definition 24.7. TH1: Timer 1 High Byte . . . . . 238
SFR Definition 24.8. TMR2CN: Timer 2 Control . . . . . 242
SFR Definition 24.9. TMR2RLL: Timer 2 Reload Register Low Byte . . . . . 243
SFR Definition 24.10. TMR2RLH: Timer 2 Reload Register High Byte . . . . . 243
SFR Definition 24.11. TMR2L: Timer 2 Low Byte . . . . . 243
SFR Definition 24.12. TMR2H Timer 2 High Byte . . . . . 243
SFR Definition 24.13. TMR3CN: Timer 3 Control . . . . . 247
SFR Definition 24.14. TMR3RLL: Timer 3 Reload Register Low Byte . . . . . 248
SFR Definition 24.15. TMR3RLH: Timer 3 Reload Register High Byte . . . . . 248
SFR Definition 24.16. TMR3L: Timer 3 Low Byte . . . . . 248
SFR Definition 24.17. TMR3H Timer 3 High Byte . . . . . 248
SFR Definition 25.1. PCA0CN: PCA Control . . . . . 261
SFR Definition 25.2. PCA0MD: PCA Mode . . . . . 262
SFR Definition 25.3. PCA0CPMn: PCA Capture/Compare Mode . . . . . 263
SFR Definition 25.4. PCA0L: PCA Counter/Timer Low Byte . . . . . 264
SFR Definition 25.5. PCA0H: PCA Counter/Timer High Byte . . . . . 264
SFR Definition 25.6. PCA0CPLn: PCA Capture Module Low Byte . . . . . 264
SFR Definition 25.7. PCA0CPHn: PCA Capture Module High Byte . . . . . 264
C2 Register Definition 26.1. C2ADD: C2 Address . . . . . 265
C2 Register Definition 26.2. DEVICEID: C2 Device ID . . . . . 265
C2 Register Definition 26.3. REVID: C2 Revision ID . . . . . 266
C2 Register Definition 26.4. FPCTL: C2 Flash Programming Control . . . . . 266
C2 Register Definition 26.5. FPDAT: C2 Flash Programming Data . . . . . 266

## NOTES:

# 1. System Overview

C8051F41x devices are fully integrated, low power, mixed-signal system-on-a-chip MCUs. Highlighted features are listed below. Refer to Table 1.1 for specific product feature selection.

- High-speed pipelined 8051-compatible microcontroller core (up to 50 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- True 12-bit 200 ksps ADC with analog multiplexer and 24 analog inputs
- Two 12-bit Current Output DACs
- Precision programmable 24.5 MHz internal oscillator
- Up to 32 kB bytes of on-chip Flash memory
- 2304 bytes of on-chip RAM
- SMBus/I2C, Enhanced UART, and SPI serial interfaces implemented in hardware
- Four general-purpose 16-bit timers
- Programmable Counter/Timer Array (PCA) with six capture/compare modules and Watchdog Timer function
- Hardware smaRTClock (Real Time Clock) operates down to 1 V with 64 bytes of Backup RAM and a Backup Voltage Regulator
- Hardware CRC Engine
- On-chip Power-On Reset, $V_{DD}$ Monitor, and Temperature Sensor
- On-chip Voltage Comparators
- Up to 24 Port I/O

With on-chip Power-On Reset, $V_{DD}$ monitor, Watchdog Timer, and clock oscillator, the C8051F41x 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 Laboratories 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, 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 programming and debugging without occupying package pins.

Each device is specified for 2.0-to-2.75 V operation (supply voltage can be up to 5.25 V using on-chip regulator) over the industrial temperature range (–45 to +85 °C). The C8051F41x are available in 28-pin QFN (also referred to as MLP or MLF) or 32-pin LQFP packages.

## Table 1.1. Product Selection Guide

| Ordering Part Number | MIPS (Peak) | Flash Memory | RAM | Calibrated Internal 24.5 MHz Oscillator | Clock Multiplier | SMBus/I2C | SPI | UART | Timers (16-bit) | Programmable Counter Array | Port I/Os | 12-bit ADC ±1 LSB INL | smaRTClock (Real Time Clock) | Two 12-bit Current Output DACs | Internal Voltage Reference | Temperature Sensor | Analog Comparators | Lead-Free (RoHS compliant) | Package |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C8051F410-GQ | 50 | 32 kB | 2368 | ✓ | ✓ | ✓ | ✓ | ✓ | 4 | ✓ | 24 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | LQFP-32 |
| C8051F411-GM | 50 | 32 kB | 2368 | ✓ | ✓ | ✓ | ✓ | ✓ | 4 | ✓ | 20 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | QFN-28 |
| C8051F412-GQ | 50 | 16 kB | 2368 | ✓ | ✓ | ✓ | ✓ | ✓ | 4 | ✓ | 24 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | LQFP-32 |
| C8051F413-GM | 50 | 16 kB | 2368 | ✓ | ✓ | ✓ | ✓ | ✓ | 4 | ✓ | 20 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | QFN-28 |

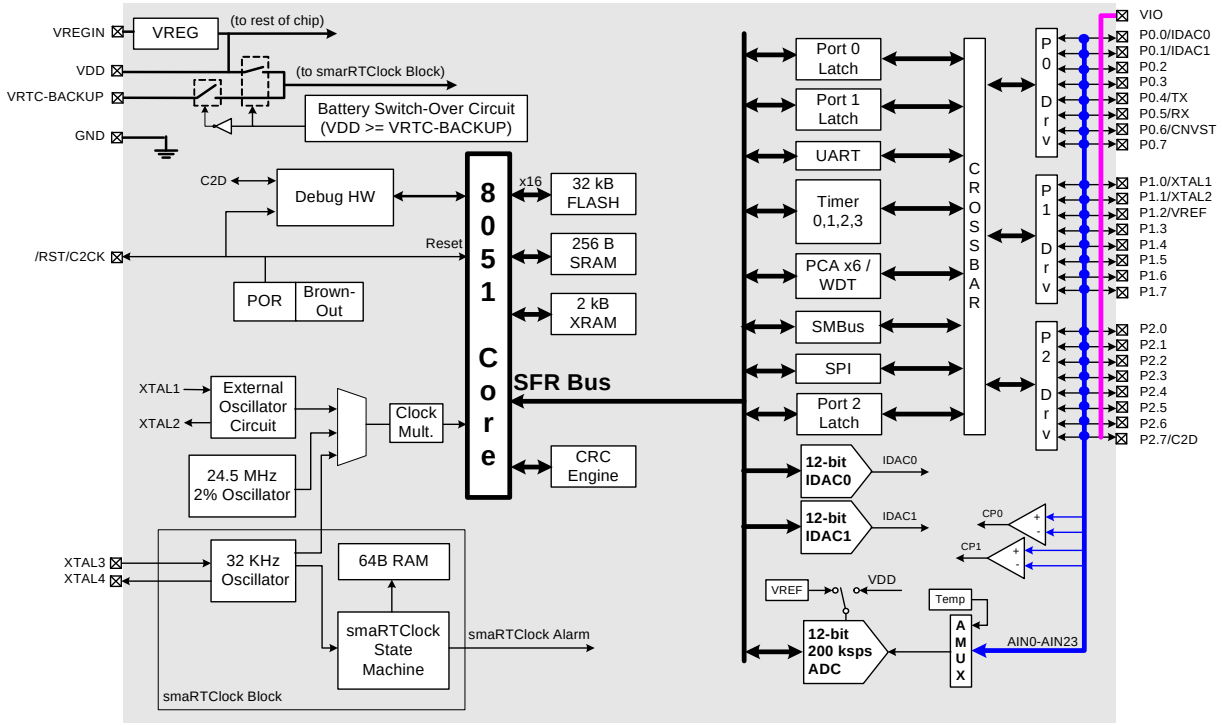



**Figure 1.1. C8051F410 Block Diagram**

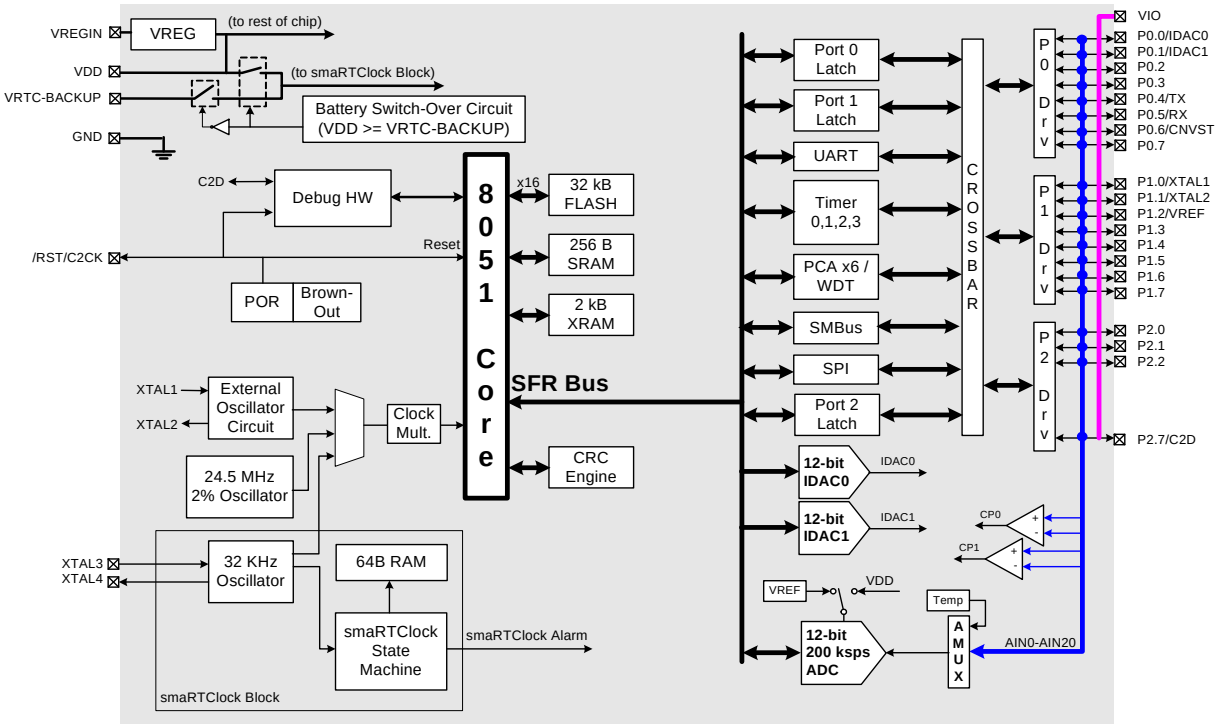



**Figure 1.2. C8051F411 Block Diagram**

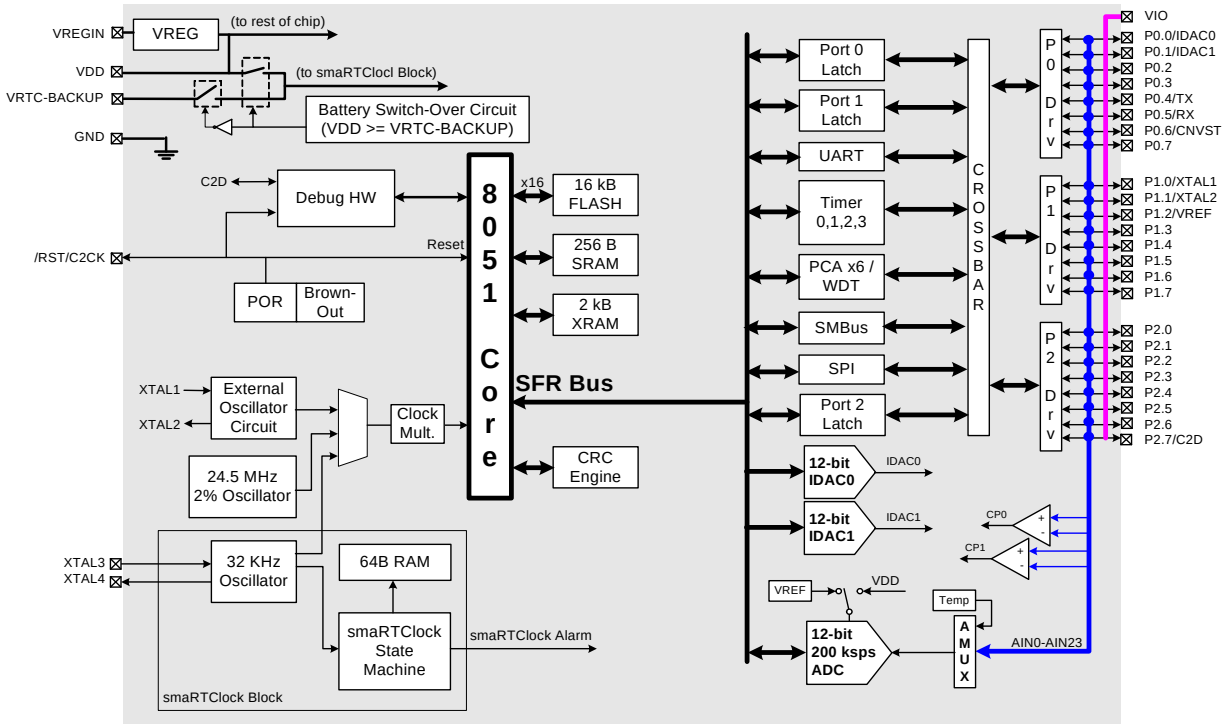



**Figure 1.3. C8051F412 Block Diagram**

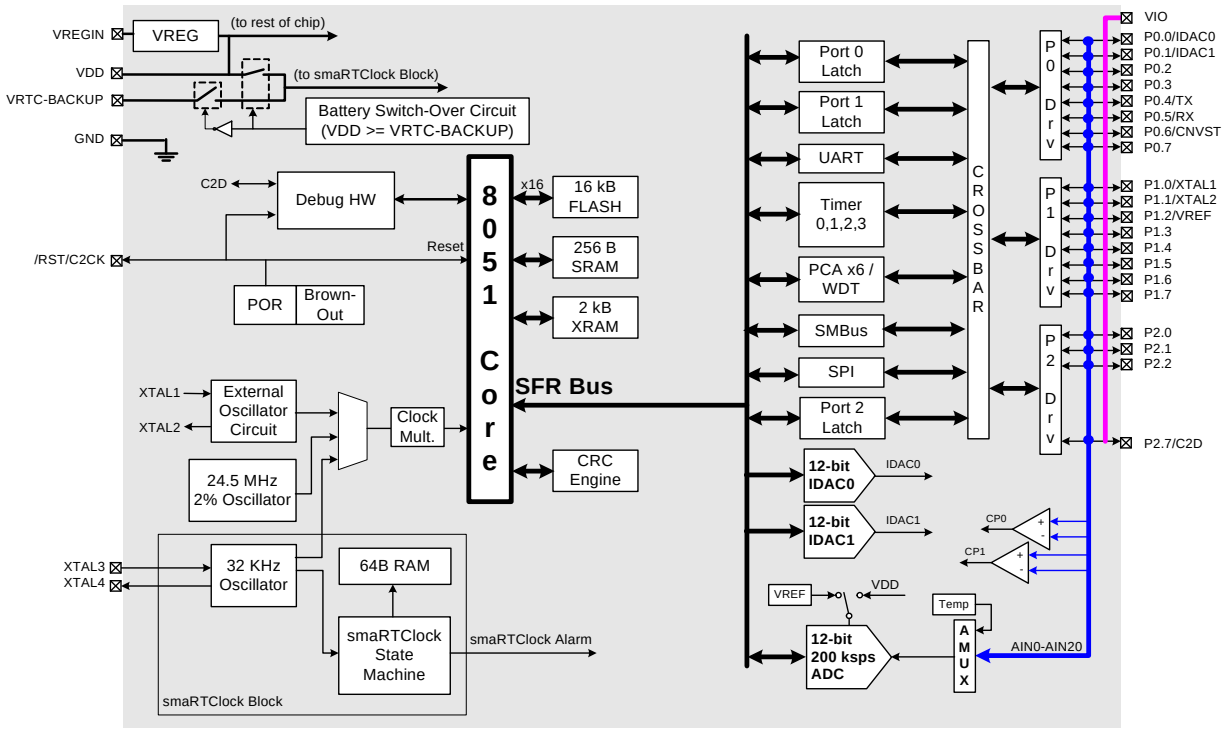



**Figure 1.4. C8051F413 Block Diagram**

## 1.1. CIP-51™ Microcontroller

### 1.1.1. Fully 8051 Compatible Instruction Set

The C8051F41x devices use Silicon Laboratories' 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 C8051F41x family has a superset of all the peripherals included with a standard 8052.

### 1.1.2. 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, and usually have 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 no instructions taking more than eight system clock cycles.

With the CIP-51's system clock running at 50 MHz, it has a peak throughput of 50 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 | 2/4 | 3 | 3/5 | 4 | 5 | 4/6 | 6 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of Instructions | 26 | 50 | 5 | 10 | 7 | 5 | 2 | 1 | 2 | 1 |

### 1.1.3. Additional Features

The C8051F41x SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

An extended interrupt handler allows the numerous analog and digital peripherals to operate independently of the controller core and interrupt the controller only when necessary. By requiring less intervention from the microcontroller core, an interrupt-driven system is more efficient and allows for easier implementation of multi-tasking, real-time systems.

Eight reset sources are available: power-on reset circuitry (POR), an on-chip $V_{DD}$ monitor, a Watchdog Timer, a Missing Clock Detector, a voltage level detection from Comparator0, a smaRTClock alarm or missing smaRTClock clock detector reset, 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 enabled in software after a power-on reset during MCU initialization.

The internal oscillator is factory calibrated to 24.5 MHz ±2%. An external oscillator drive circuit is also included, allowing an external crystal, ceramic resonator, capacitor, RC, or CMOS clock source to generate the system clock. A clock multiplier allows for operation at up to 50 MHz. The dedicated smaRTClock oscillator can be extremely useful in low power applications, allowing the system to maintain accurate time while the MCU is not powered, or its internal oscillator is suspended. The MCU can be reset or have its oscillator awakened using the smaRTClock alarm function.

## 1.2. On-Chip Debug Circuitry

The C8051F41x devices include on-chip Silicon Laboratories 2-Wire (C2) debug circuitry that provides non-intrusive, full speed, in-circuit debugging of the production part *installed in the end application.*

Silicon Laboratories' debugging system supports inspection and modification of memory and registers, breakpoints, and single stepping. No additional target RAM, program memory, timers, or communications channels are required. All the digital and analog peripherals are functional and work correctly while debugging. All the peripherals (except for the ADC and SMBus) are stalled when the MCU is halted, during single stepping, or at a breakpoint in order to keep them synchronized.

The C8051F410DK development kit provides all the hardware and software necessary to develop application code and perform in-circuit debugging with the C8051F41x MCUs. The kit includes software with a developer's studio and debugger, a USB debug adapter, a target application board with the associated MCU installed, and the required cables and wall-mount power supply. The development kit requires a computer with Windows®98 SE or later installed. As shown in Figure 1.5, the PC is connected to the USB debug adapter. A six-inch ribbon cable connects the USB debug adapter to the user's application board, picking up the two C2 pins and GND.

The Silicon Laboratories IDE interface is a vastly superior developing and debugging configuration, compared to standard MCU emulators that use on-board "ICE Chips" and require the MCU in the application board to be socketed. Silicon Laboratories' debug paradigm increases ease of use and preserves the performance of the precision analog peripherals.

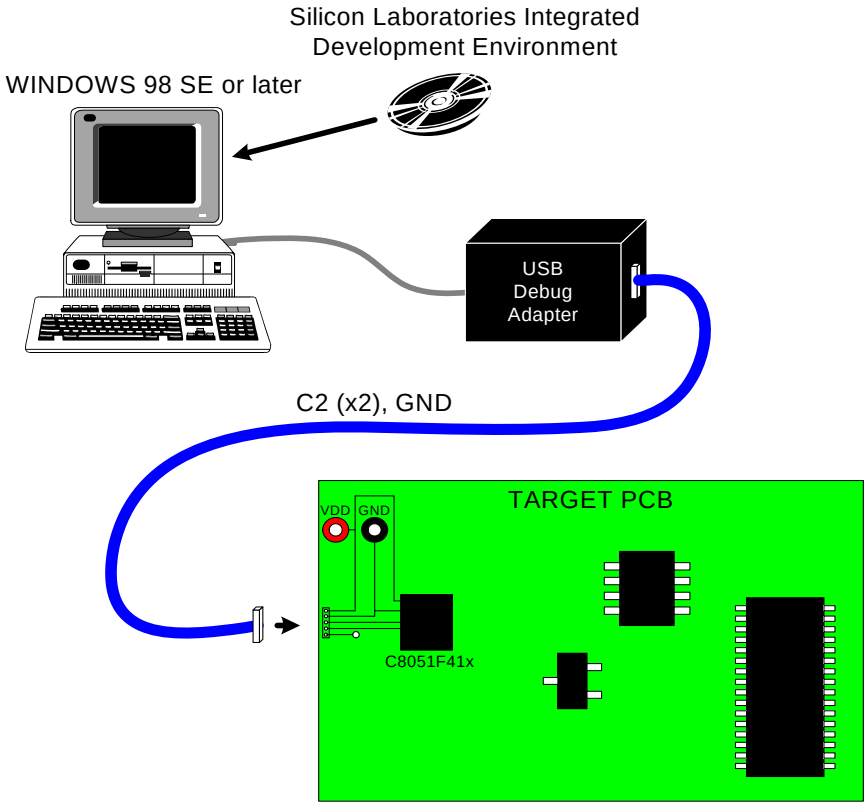



**Figure 1.5. Development/In-System Debug Diagram**

## 1.3. On-Chip Memory

The CIP-51 has a standard 8051 program and data address configuration. It includes 256 bytes of data RAM, with the upper 128 bytes dual-mapped. Indirect addressing accesses the upper 128 bytes of general purpose RAM, and direct addressing accesses the 128-byte SFR address space. The lower 128 bytes of RAM are accessible via direct and indirect addressing. The first 32 bytes are addressable as four banks of general purpose registers, and the next 16 bytes can be byte addressable or bit addressable.

Program memory consists of 32 kB (‘F410/1) or 16 kB (‘F412/3) of Flash. This memory may be reprogrammed in-system in 512 byte sectors and requires no special off-chip programming voltage.

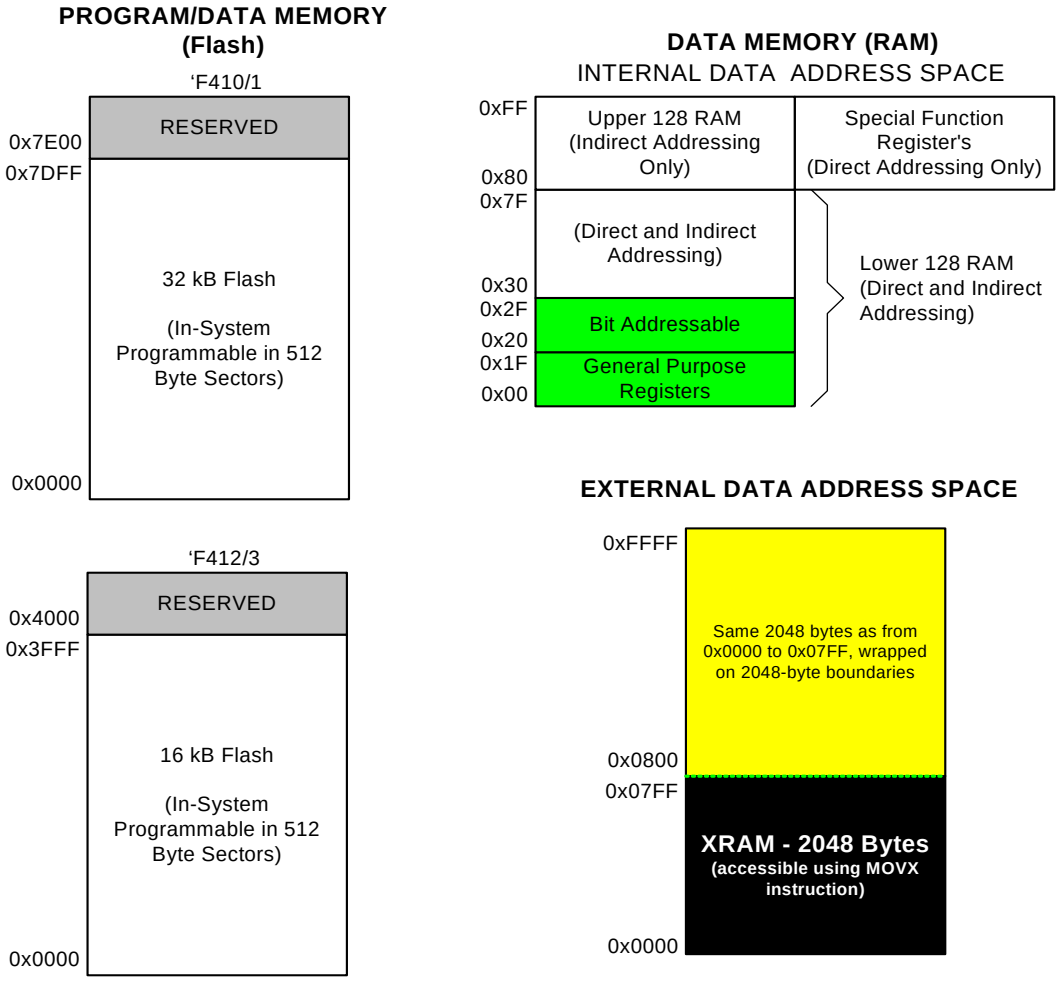



**Figure 1.6. Memory Map**

## 1.4. Operating Modes

The C8051F41x devices have four operating modes: Active (Normal), Idle, Suspend, and Stop. Active mode occurs during normal operation when the oscillator and peripherals are active. Idle mode halts the CPU while leaving the peripherals and internal clocks active. Suspend mode halts SYSCLK until a wakening event occurs, which also halts all peripherals using SYSCLK. In Stop mode, the CPU is halted, all interrupts and timers are inactive, and the internal oscillator is stopped. The various operating modes are described in Table 1.2 below:

**Table 1.2. Operating Modes Summary**

| | Properties | Power Consumption | How Entered? | How Exited? |
|---|---|---|---|---|
| Active | • SYSCLK active<br>• CPU active (accessing Flash)<br>• Peripherals active or inactive depending on user settings<br>• smaRTClock active or inactive | Full | — | — |
| Idle | • SYSCLK active<br>• CPU inactive (not accessing Flash)<br>• Peripherals active or inactive depending on user settings<br>• smaRTClock active or inactive | Less than Full | IDLE (PCON.0) | Any enabled interrupt or device reset |
| Suspend | • SYSCLK inactive<br>• CPU inactive (not accessing Flash)<br>• Peripherals enabled (but not operating) or disabled depending on user settings<br>• smaRTClock active or inactive | Low | SUSPEND (OSCICN.5) | Wakening event or external/MCD reset |
| Stop | • SYSCLK inactive<br>• CPU inactive (not accessing Flash)<br>• Digital peripherals inactive; analog peripherals enabled (but not operating) or disabled depending on user settings<br>• smaRTClock inactive | Very low | STOP (PCON.1) | External or MCD reset |

See Section **“10.3. Power Management Modes” on page 101** for Idle and Stop mode details. See **Section “19.1.1. Internal Oscillator Suspend Mode” on page 166** for more information on Suspend mode.

## 1.5. 12-Bit Analog to Digital Converter

The C8051F41x devices include an on-chip 12-bit SAR ADC with a 27-channel single-ended input multiplexer and a maximum throughput of 200 ksps. The ADC system includes a configurable analog multiplexer that selects the positive ADC input, which is measured with respect to GND. Ports 0–2 are available as ADC inputs; additionally, the on-chip Temperature Sensor output and the core supply voltage ($V_{DD}$) are available as ADC inputs. User firmware may shut down the ADC or use it in **Burst Mode** to save power.

Conversions can be started in four ways: a software command, an overflow of Timer 2 or 3, or an external convert start signal. This flexibility allows the start of conversion to be triggered by software events, a periodic signal (timer overflows), or external HW signals. Conversion completions are indicated by a status bit and an interrupt (if enabled) and occur after 1, 4, 8, or 16 samples have been accumulated by a hardware accumulator. The resulting data word is latched into the ADC data SFRs upon completion of a conversion. When the system clock is slow, Burst Mode allows ADC0 to automatically wake from a low power shutdown state, acquire and accumulate samples, then re-enter the low power shutdown state without CPU intervention.

Window compare registers for the ADC data can be configured to interrupt the controller when ADC data is either within or outside of a specified range. The ADC can monitor a key voltage continuously in background mode, but not interrupt the controller unless the converted data is within/outside the specified range.

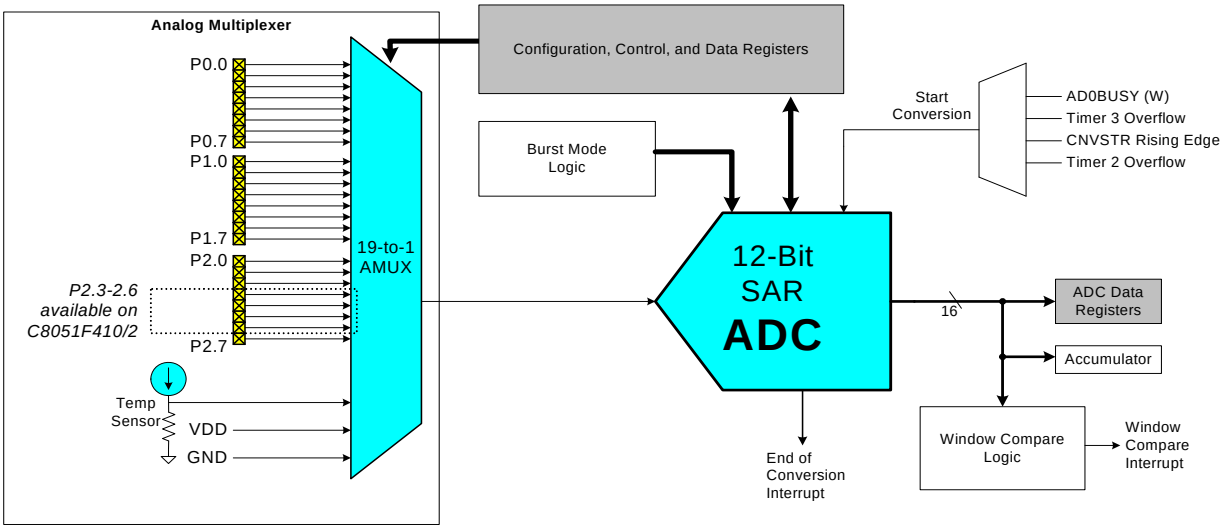



**Figure 1.7. 12-Bit ADC Block Diagram**

## 1.6. Two 12-bit Current-Mode DACs

The C8051F41x devices include two 12-bit current-mode Digital-to-Analog Converters (IDACs). The maximum current output of the IDACs can be adjusted for four different current settings; 0.25 mA, 0.5 mA, 1 mA, and 2 mA. A flexible output update mechanism allows for seamless full-scale changes, and supports jitter-free updates for waveform generation. The IDAC outputs can be merged onto a single port I/O pin for increased full-scale current output or increased resolution. IDAC updates can be performed on-demand, scheduled on a Timer overflow, or synchronized with an external signal. Figure 1.8 shows a block diagram of the IDAC circuitry.

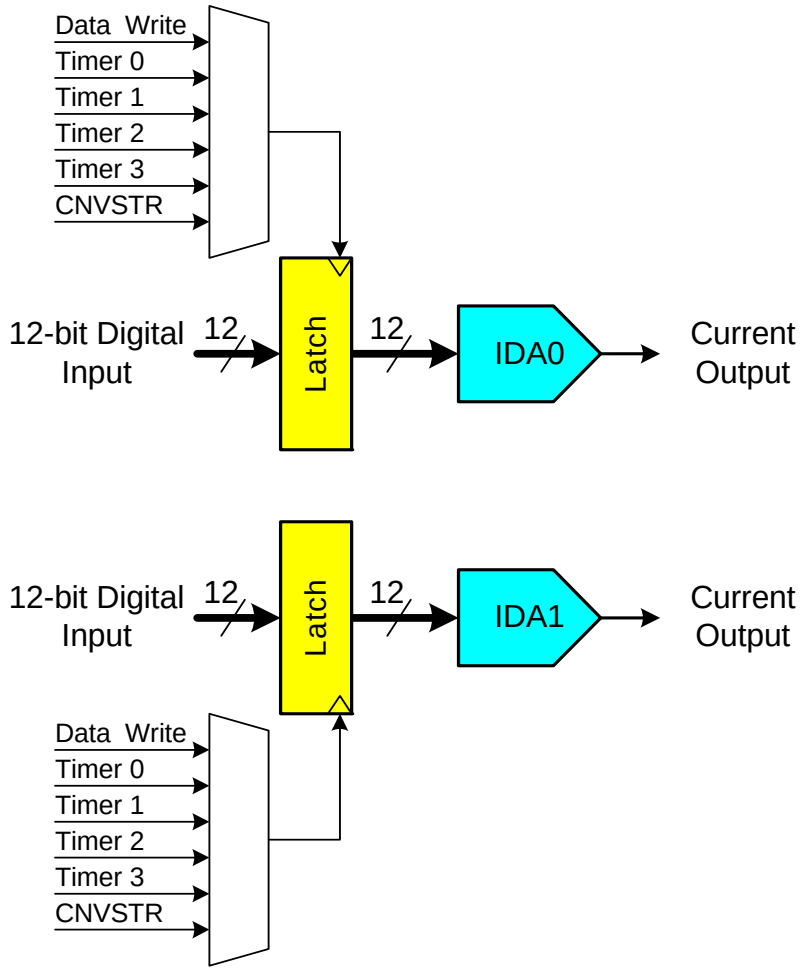



**Figure 1.8. IDAC Block Diagram**

## 1.7. Programmable Comparators

C8051F41x devices include two software-configurable voltage comparators with an input multiplexer. Each comparator offers programmable response time and hysteresis and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0 and CP1), or an asynchronous “raw” output (CP0A and CP1A). Comparator interrupts may be generated on rising, falling, or both edges. When in IDLE or SUSPEND mode, these interrupts may be used as a “wake-up” source for the processor. Comparator0 may also be configured as a reset source. A block diagram of the comparator is shown in Figure 1.9.

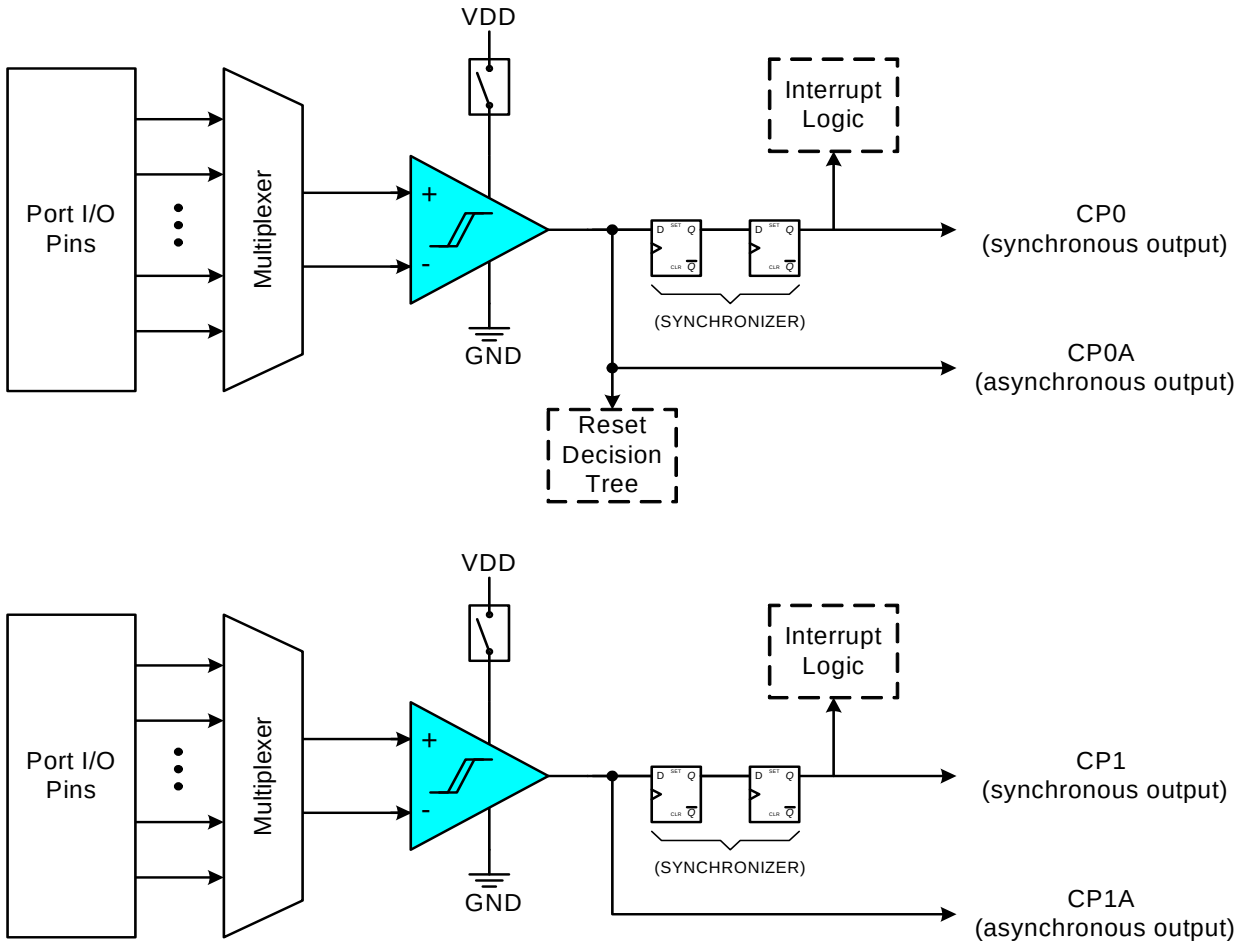



**Figure 1.9. Comparators Block Diagram**

## 1.8. Cyclic Redundancy Check Unit

C8051F41x 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 and outputs a 16-bit or 32-bit result. CRC0 also has a hardware bit reverse feature for quick data manipulation.

## 1.9. Voltage Regulator

C8051F41x devices include an on-chip low dropout voltage regulator (REG0). The input to REG0 at the $V_{REGIN}$ pin can be as high as 5.25 V. The output can be selected by software to 2.0 V or 2.5 V. When enabled, the output of REG0 powers the device and drives the $V_{DD}$ pin. The voltage regulator can be used to power external devices connected to $V_{DD}$.

## 1.10. Serial Ports

The C8051F41x Family includes an SMBus/I2C 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.

## 1.11. smaRTClock (Real Time Clock)

C8051F41x devices include a smaRTClock Peripheral (Real Time Clock). The smaRTClock has a dedicated 32 kHz oscillator that can be configured for use with or without a crystal, a 47-bit smaRTClock timer with alarm, a backup supply regulator, and 64 bytes of backup SRAM. When the backup supply voltage ($V_{RTC-BACKUP}$) is powered, the smaRTClock peripheral remains fully functional even if the core supply voltage ($V_{DD}$) is lost.

The smaRTClock allows a maximum of 137 year 47-bit independent time-keeping when used with a 32.768 kHz Watch Crystal and backup supply voltage of at least 1 V. The switchover logic powers smaRTClock from the backup supply when the voltage at $V_{RTC-BACKUP}$ is greater than $V_{DD}$. The smaRTClock alarm and missing clock detector can interrupt the CIP-51, wake the internal oscillator from SUSPEND mode, or generate a device reset if the smaRTClock timer reaches a pre-set value or the oscillator stops.

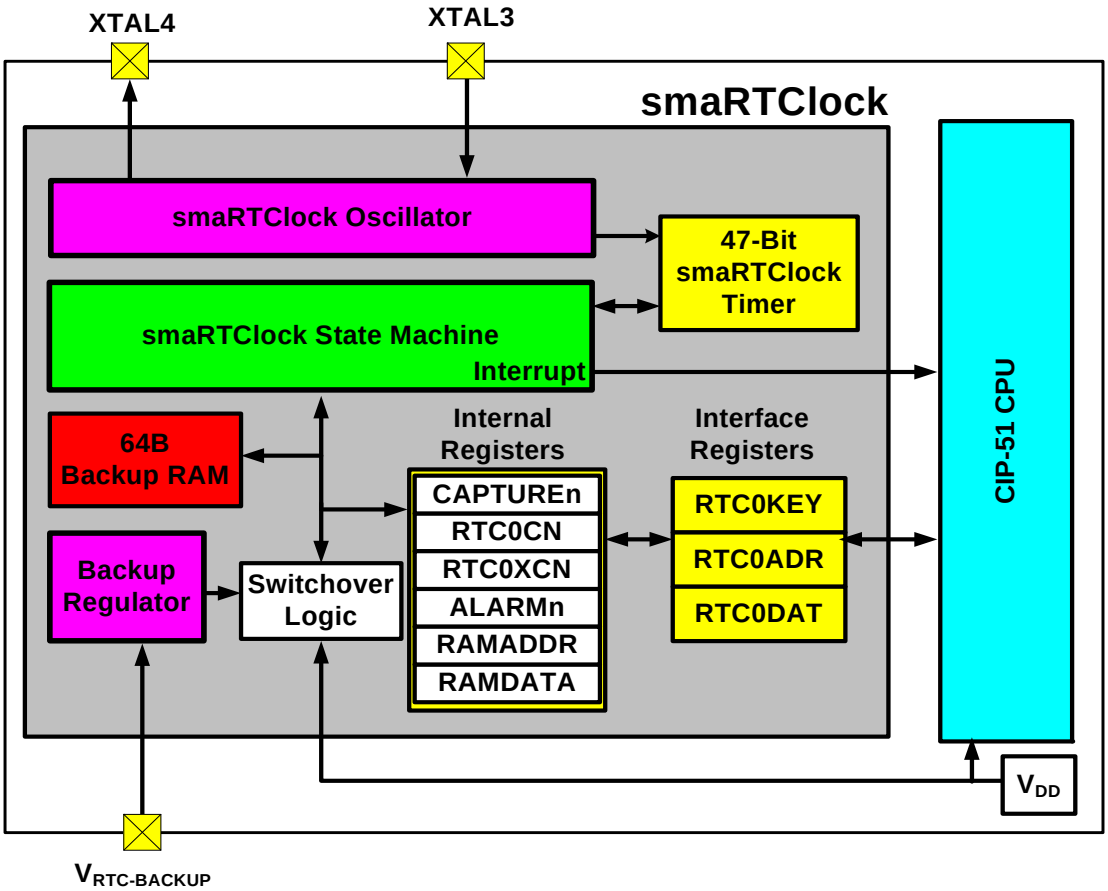



**Figure 1.10. smaRTClock Block Diagram**

## 1.12. Port Input/Output

C8051F41x devices include up to 24 I/O pins. Port pins are organized as three byte-wide ports. The port pins behave like typical 8051 ports with a few enhancements. Each port pin can be configured as a digital or analog I/O pin. Pins selected as digital I/O can be configured for push-pull or open-drain operation. The "weak pullups" that are fixed on typical 8051 devices may be individually or globally disabled to save power.

The Digital Crossbar allows mapping of internal digital system resources to port I/O pins. On-chip counter/timers, serial buses, hardware interrupts, and other digital signals can be configured to appear on the port pins using the Crossbar control registers. This allows the user to select the exact mix of general-purpose port I/O, digital, and analog resources needed for the application.

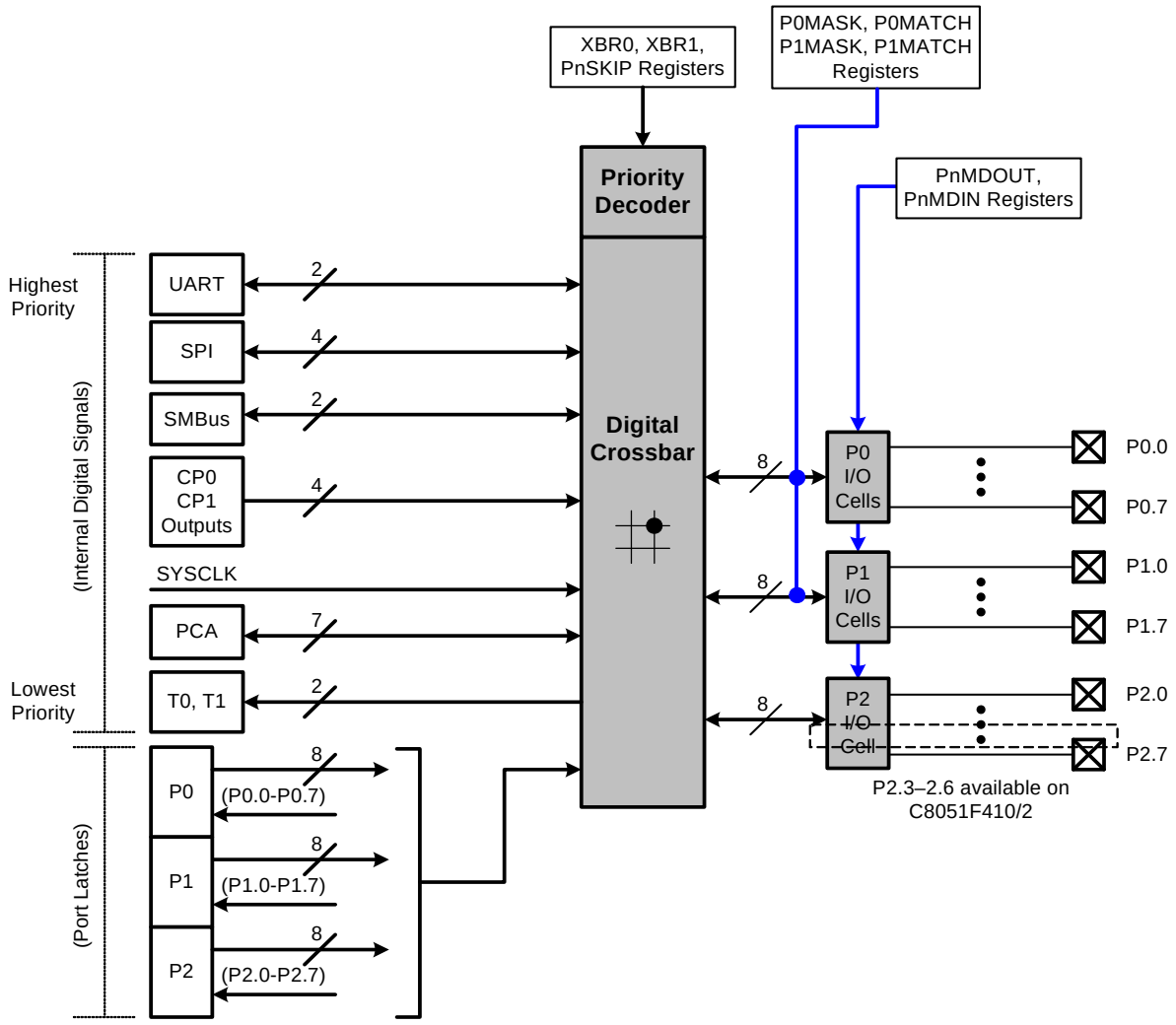



**Figure 1.11. Port I/O Functional Block Diagram**

## 1.13. 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. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, real-time clock source divided by 8, Timer 0 overflow, or an external clock signal on the External Clock Input (ECI) 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-Bit PWM, or 16-Bit PWM. Additionally, PCA Module 5 may be used as a watchdog timer (WDT), and is enabled in this mode following a system reset. The PCA Capture/Compare Module I/O and the External Clock Input may be routed to Port I/O using the digital crossbar.

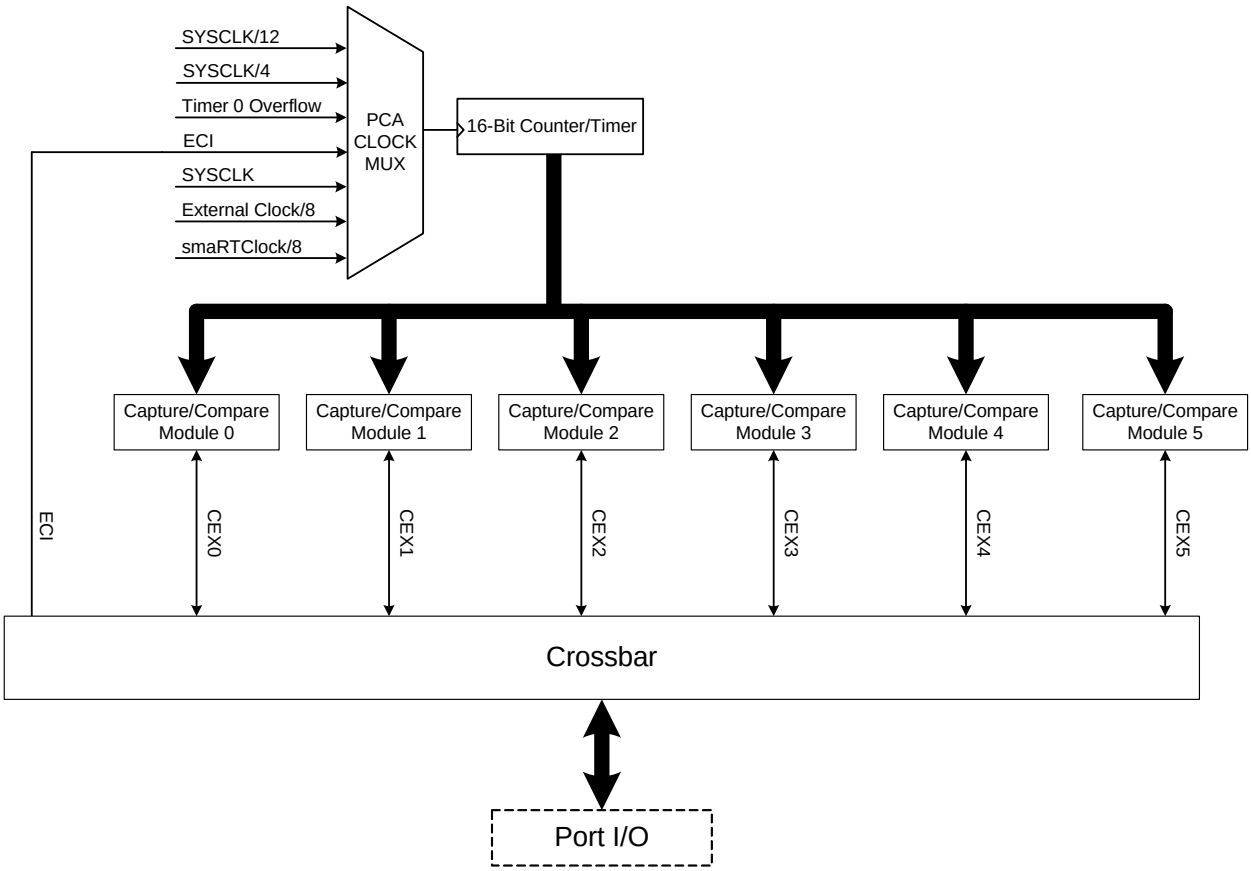



**Figure 1.12. PCA Block Diagram**

## 2. Absolute Maximum Ratings

### Table 2.1. Absolute Maximum Ratings*

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Ambient temperature under bias | | –55 | — | 125 | °C |
| Storage Temperature | | –65 | — | 150 | °C |
| Voltage on $V_{REGIN}$ with respect to GND | | –0.3 | — | 5.5 | V |
| Voltage on $V_{DD}$ with respect to GND | | –0.3 | — | 3.0 | V |
| Voltage on $V_{RTC\text{-}BACKUP}$ with respect to GND | | –0.3 | — | 5.5 | V |
| Voltage on XTAL1 with respect to GND | | –0.3 | — | $V_{DD}$+ 0.3 | V |
| Voltage on XTAL3 with respect to GND | | –0.3 | — | 5.5 | V |
| Voltage on any Port I/O Pin (except Port 0 pins) or $\overline{RST}$ with respect to GND | | –0.3 | — | $V_{IO}$ + 0.3 | V |
| Voltage on any Port 0 Pin with respect to GND | | 0.3 | — | 5.5 | V |
| Maximum output current sunk by any Port pin | | — | — | 100 | mA |
| Maximum output current sourced by any Port pin | | — | — | 100 | mA |
| Maximum Total current through $V_{DD}$, $V_{IO}$, $V_{RTC\text{-}BACKUP}$, $V_{REGIN}$, and GND | | — | — | 500 | mA |

***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.

# 3. Global DC Electrical Characteristics

## Table 3.1. Global DC Electrical Characteristics

–40 to +85 °C, 50 MHz System Clock unless otherwise specified. Typical values are given at 25 °C

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Supply Input Voltage ($V_{REGIN}$)[1] | Output Current = 1 mA | 2.15 | — | 5.25 | V |
| Core Supply Voltage ($V_{DD}$) | | 2.0 | — | 2.75 | V |
| I/O Supply Voltage ($V_{IO}$)[2] | | 2.0 | — | 5.25 | V |
| Backup Supply Voltage ($V_{RTC\text{-}BACKUP}$)[3] | | 1.0 | — | 5.25 | V |
| Backup Supply Current ($I_{RTC\text{-}BACKUP}$) ($V_{DD}$ = 0 V, smaRTClock clock = 32 kHz) | $V_{RTC\text{-}BACKUP}$ = 1.0 V: | | | | |
| | at –40 ºC | — | 0.65 | 1.5 | µA |
| | at 25 ºC | — | 0.9 | 1.8 | µA |
| | at 85 ºC | — | 1.4 | 2.5 | µA |
| | $V_{RTC\text{-}BACKUP}$ = 1.8 V: | | | | |
| | at –40 ºC | — | 0.7 | — | µA |
| | at 25 ºC | — | 0.92 | — | µA |
| | at 85 ºC | — | 1.45 | — | µA |
| | $V_{RTC\text{-}BACKUP}$ = 2.5 V: | | | | |
| | at –40 ºC | — | 0.72 | 1.6 | µA |
| | at 25 ºC | — | 0.95 | 1.85 | µA |
| | at 85 ºC | — | 1.5 | 2.6 | µA |
| Core Supply RAM Data Retention Voltage | | — | 1.5 | — | V |
| SYSCLK (System Clock)[4,5] | | 0 | — | 50 | MHz |
| Specified Operating Temperature Range | | –40 | — | +85 | °C |

**Notes:**

1. For more information on $V_{REGIN}$ characteristics, see Table 8.1 on page 82.
2. VIO must be equal to or greater than VDD.
3. The Backup Supply Voltage ($V_{RTC\text{-}BACKUP}$) is used to power the smaRTClock peripheral only.
4. SYSCLK is the internal device clock. For operational speeds in excess of 25 MHz, SYSCLK must be derived from the internal clock multiplier.
5. SYSCLK must be at least 32 kHz to enable debugging.
6. Based on device characterization data, not production tested.
7. Active and Inactive IDD at voltages and frequencies other than those specified can be calculated using the IDD Supply Sensitivity. For example, if the $V_{DD}$ is 2.2 V instead of 2.0 V at 25 MHz: $I_{DD}$ = 5.5 mA typical at 2.0 V and f = 25 MHz. From this, $I_{DD}$ = 5.5 mA + 1.14 x (2.2 V – 2.0 V) = 5.73 mA at 2.2 V and f = 25 MHz.
8. $I_{DD}$ can be estimated for frequencies < 15 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate $I_{DD}$ for > 15 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 20 MHz, $I_{DD}$ = 5.5 mA – (25 MHz – 20 MHz) x 0.16 mA/MHz = 4.7 mA.
9. Idle IDD can be estimated for frequencies < 1 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate Idle for > 1 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 5 MHz, Idle $I_{DD}$ = 2.8 mA – (25 MHz – 5 MHz) x 0.1 mA/MHz = 0.8 mA.

## Table 3.1. Global DC Electrical Characteristics (Continued)

–40 to +85 °C, 50 MHz System Clock unless otherwise specified. Typical values are given at 25 °C

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Digital Supply Current—CPU Active (Normal Mode, fetching instructions from Flash)** | | | | | |
| Core Supply Current ($I_{DD}$)[6] | **$V_{DD}$ = 2.0 V:** | | | | |
| | F = 32 kHz | — | 13 | 30 | µA |
| | F = 1 MHz | — | 0.30 | 0.5 | mA |
| | F = 25 MHz | — | 5.5 | 6.5 | mA |
| | F = 50 MHz | — | 9.5 | 12 | mA |
| | **$V_{DD}$ = 2.5 V:** | | | | |
| | F = 32 kHz | — | 17 | 40 | µA |
| | F = 1 MHz | — | 0.43 | 0.65 | mA |
| | F = 25 MHz | — | 8.3 | 9.5 | mA |
| | F = 50 MHz | — | 13.5 | 15 | mA |
| Supply Sensitivity ($I_{DD}$)[6,7] | F = 25 MHz | — | 114 | — | %/V |
| | F = 1 MHz | — | 100 | — | %/V |
| Frequency Sensitivity ($I_{DD}$)[6,8] | **$V_{DD}$ = 2.0 V:** | | | | |
| | F < 15 MHz, T = 25 ºC | — | 0.27 | — | mA/MHz |
| | F > 15 MHz, T = 25 ºC | — | 0.16 | — | mA/MHz |
| | **$V_{DD}$ = 2.5 V:** | | | | |
| | F < 15 MHz, T = 25 ºC | — | 0.39 | — | mA/MHz |
| | F > 15 MHz, T = 25 ºC | — | 0.2 | — | mA/MHz |

**Notes:**

1. For more information on $V_{REGIN}$ characteristics, see Table 8.1 on page 82.
2. VIO must be equal to or greater than VDD.
3. The Backup Supply Voltage ($V_{RTC-BACKUP}$) is used to power the smaRTClock peripheral only.
4. SYSCLK is the internal device clock. For operational speeds in excess of 25 MHz, SYSCLK must be derived from the internal clock multiplier.
5. SYSCLK must be at least 32 kHz to enable debugging.
6. Based on device characterization data, not production tested.
7. Active and Inactive IDD at voltages and frequencies other than those specified can be calculated using the IDD Supply Sensitivity. For example, if the $V_{DD}$ is 2.2 V instead of 2.0 V at 25 MHz: $I_{DD}$ = 5.5 mA typical at 2.0 V and f = 25 MHz. From this, $I_{DD}$ = 5.5 mA + 1.14 x (2.2 V – 2.0 V) = 5.73 mA at 2.2 V and f = 25 MHz.
8. $I_{DD}$ can be estimated for frequencies < 15 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate $I_{DD}$ for > 15 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 20 MHz, $I_{DD}$ = 5.5 mA – (25 MHz – 20 MHz) x 0.16 mA/MHz = 4.7 mA.
9. Idle IDD can be estimated for frequencies < 1 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate Idle for > 1 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 5 MHz, Idle $I_{DD}$ = 2.8 mA – (25 MHz – 5 MHz) x 0.1 mA/MHz = 0.8 mA.

## Table 3.1. Global DC Electrical Characteristics (Continued)

–40 to +85 °C, 50 MHz System Clock unless otherwise specified. Typical values are given at 25 °C

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Digital Supply Current—CPU Inactive (Idle Mode, not fetching instructions from Flash)** | | | | | |
| Core Supply Current ($I_{DD}$)[6] | **$V_{DD}$ = 2.0 V:** | | | | |
| | F = 32 kHz | — | 10 | 25 | µA |
| | F = 1 MHz | — | 0.15 | 0.25 | mA |
| | F = 25 MHz | — | 2.8 | 3.3 | mA |
| | F = 50 MHz | — | 5 | 11 | mA |
| | **$V_{DD}$ = 2.5 V:** | | | | |
| | F = 32 kHz | — | 11 | 30 | µA |
| | F = 1 MHz | — | 0.21 | 0.37 | mA |
| | F = 25 MHz | — | 3.8 | 4.3 | mA |
| | F = 50 MHz | — | 7.5 | 8.0 | mA |
| Supply Sensitivity ($I_{DD}$)[6,7] | F = 25 MHz | — | 75 | — | %/V |
| | F = 1 MHz | — | 68 | — | %/V |
| Frequency Sensitivity ($I_{DD}$)[6,9] | **$V_{DD}$ = 2.0 V:** | | | | |
| | F < 1 MHz, T = 25 ºC | — | 0.14 | — | mA/MHz |
| | F > 1 MHz, T = 25 ºC | — | 0.1 | — | mA/MHz |
| | **$V_{DD}$ = 2.5 V:** | | | | |
| | F < 1 MHz, T = 25 ºC | — | 0.19 | — | mA/MHz |
| | F > 1 MHz, T = 25 ºC | — | 0.13 | — | mA/MHz |
| Digital Supply Current (Suspend Mode) | Oscillator not running, VDD = 2.5 V | — | 0.15 | 50 | µA |
| Digital Supply Current (Stop Mode, shutdown) | Oscillator not running, VDD = 2.5 V | — | 0.15 | 50 | µA |

**Notes:**

1. For more information on $V_{REGIN}$ characteristics, see Table 8.1 on page 82.
2. VIO must be equal to or greater than VDD.
3. The Backup Supply Voltage ($V_{RTC-BACKUP}$) is used to power the smaRTClock peripheral only.
4. SYSCLK is the internal device clock. For operational speeds in excess of 25 MHz, SYSCLK must be derived from the internal clock multiplier.
5. SYSCLK must be at least 32 kHz to enable debugging.
6. Based on device characterization data, not production tested.
7. Active and Inactive IDD at voltages and frequencies other than those specified can be calculated using the IDD Supply Sensitivity. For example, if the $V_{DD}$ is 2.2 V instead of 2.0 V at 25 MHz: $I_{DD}$ = 5.5 mA typical at 2.0 V and f = 25 MHz. From this, $I_{DD}$ = 5.5 mA + 1.14 x (2.2 V – 2.0 V) = 5.73 mA at 2.2 V and f = 25 MHz.
8. $I_{DD}$ can be estimated for frequencies < 15 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate $I_{DD}$ for > 15 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 20 MHz, $I_{DD}$ = 5.5 mA – (25 MHz – 20 MHz) x 0.16 mA/MHz = 4.7 mA.
9. Idle IDD can be estimated for frequencies < 1 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range. When using these numbers to estimate Idle for > 1 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 2.0 V; F = 5 MHz, Idle $I_{DD}$ = 2.8 mA – (25 MHz – 5 MHz) x 0.1 mA/MHz = 0.8 mA.

**Table 3.2. Index to Electrical Characteristics Tables**

| Table Title | Page # |
|---|---|
| ADC0 Electrical Characteristics ($V_{DD}$ = 2.5 V, $V_{REF}$ = 2.2 V) | 67 |
| ADC0 Electrical Characteristics ($V_{DD}$ = 2.1 V, $V_{REF}$ = 1.5 V) | 68 |
| IDAC Electrical Characteristics | 75 |
| Voltage Reference Electrical Characteristics | 79 |
| Voltage Regulator Electrical Specifications | 82 |
| Comparator Electrical Characteristics | 92 |
| Reset Electrical Characteristics | 134 |
| Flash Electrical Characteristics | 143 |
| Port I/O DC Electrical Characteristics | 163 |
| Oscillator Electrical Characteristics | 175 |

## NOTES:

# 4. Pinout and Package Definitions

**Table 4.1. Pin Definitions for the C8051F41x**

| Name | Pin Numbers | | Type | Description |
|---|---|---|---|---|
| | ‘F410/2 | ‘F411/3 | | |
| $V_{DD}$ | 7 | 6 | | Core Supply Voltage. |
| $V_{IO}$ | 1 | 28 | | I/O Supply Voltage. |
| GND | 6 | 5 | | Ground. |
| $V_{RTC\text{-}BACKUP}$ | 3 | 2 | | smaRTClock Backup Supply Voltage. |
| $V_{REGIN}$ | 8 | 7 | | On-Chip Voltage Regulator Input. |
| $\overline{RST}$/<br>C2CK | 2 | 1 | D I/O<br>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 kΩ pullup to $V_{IO}$ is recommended. See Reset Sources Section for a complete description.<br>Clock signal for the C2 Debug Interface. |
| P2.7/<br>C2D | 32 | 27 | D I/O<br>D I/O | Port 2.7. See Port I/O Section for a complete description.<br>Bi-directional data signal for the C2 Debug Interface. |
| XTAL3 | 5 | 4 | A In | smaRTClock Oscillator Crystal Input.<br>See Section 20. "smaRTClock (Real Time Clock)" for a complete description. |
| XTAL4 | 4 | 3 | A Out | smaRTClock Oscillator Crystal Input.<br>See Section 20. "smaRTClock (Real Time Clock)" for a complete description. |
| P0.0/<br>IDAC0 | 17 | 16 | D I/O or A In<br>A Out | Port 0.0. See Port I/O Section for a complete description.<br>IDAC0 Output. See IDAC Section for complete description. |
| P0.1/<br>IDAC1 | 18 | 17 | D I/O or A In<br>A Out | Port 0.1. See Port I/O Section for a complete description.<br>IDAC1 Output. See IDAC Section for complete description. |
| P0.2 | 19 | 18 | D I/O or A In | Port 0.2. See Port I/O Section for a complete description. |
| P0.3 | 20 | 19 | D I/O or A In | Port 0.3. See Port I/O Section for a complete description. |

**Table 4.1. Pin Definitions for the C8051F41x (Continued)**

| Name | Pin Numbers | | Type | Description |
|---|---|---|---|---|
| | 'F410/2 | 'F411/3 | | |
| P0.4/ | 21 | 20 | 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 for a complete description. |
| P0.5/ | 22 | 21 | 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 for a complete description. |
| P0.6/ | 23 | 22 | 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, IDA0, and IDA1. See ADC0 or IDACs section for a complete description. |
| P0.7 | 24 | 23 | D I/O or A In | Port 0.7. See Port I/O Section for a complete description. |
| P1.0/ | 9 | 8 | D I/O or A In | Port 1.0. 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. |
| P1.1/ | 10 | 9 | D I/O or A In | Port 1.1. See Port I/O Section for a complete description. |
| XTAL2 | | | A O or D In | External Clock Output. This pin is the excitation driver for an external crystal or resonator, or an external clock input for CMOS, capacitor, or RC oscillator configurations. See Oscillator Section. |
| P1.2 | 11 | 10 | D I/O or A In | Port 1.2. See Port I/O Section for a complete description. |
| $V_{REF}$ | | | A In | External $V_{REF}$ Input. See $V_{REF}$ Section. |
| P1.3 | 12 | 11 | D I/O or A In | Port 1.3. See Port I/O Section for a complete description. |
| P1.4 | 13 | 12 | D I/O or A In | Port 1.4. See Port I/O Section for a complete description. |
| P1.5 | 14 | 13 | D I/O or A In | Port 1.5. See Port I/O Section for a complete description. |
| P1.6 | 15 | 14 | D I/O or A In | Port 1.6. See Port I/O Section for a complete description. |

**Table 4.1. Pin Definitions for the C8051F41x (Continued)**

| Name | Pin Numbers | | Type | Description |
|---|---|---|---|---|
| | ‘F410/2 | ‘F411/3 | | |
| P1.7 | 16 | 15 | D I/O or A In | Port 1.7. See Port I/O Section for a complete description. |
| P2.0 | 25 | 24 | D I/O or A In | Port 2.0. See Port I/O Section for a complete description. |
| P2.1 | 26 | 25 | D I/O or A In | Port 2.1. See Port I/O Section for a complete description. |
| P2.2 | 27 | 26 | D I/O or A In | Port 2.2. See Port I/O Section for a complete description. |
| P2.3* | 28 | | D I/O or A In | Port 2.3. See Port I/O Section for a complete description. |
| P2.4* | 29 | | D I/O or A In | Port 2.4. See Port I/O Section for a complete description. |
| P2.5* | 30 | | D I/O or A In | Port 2.5. See Port I/O Section for a complete description. |
| P2.6* | 31 | | D I/O or A In | Port 2.6. See Port I/O Section for a complete description. |
| ***Note:** Available only on the C8051F410/2. | | | | |

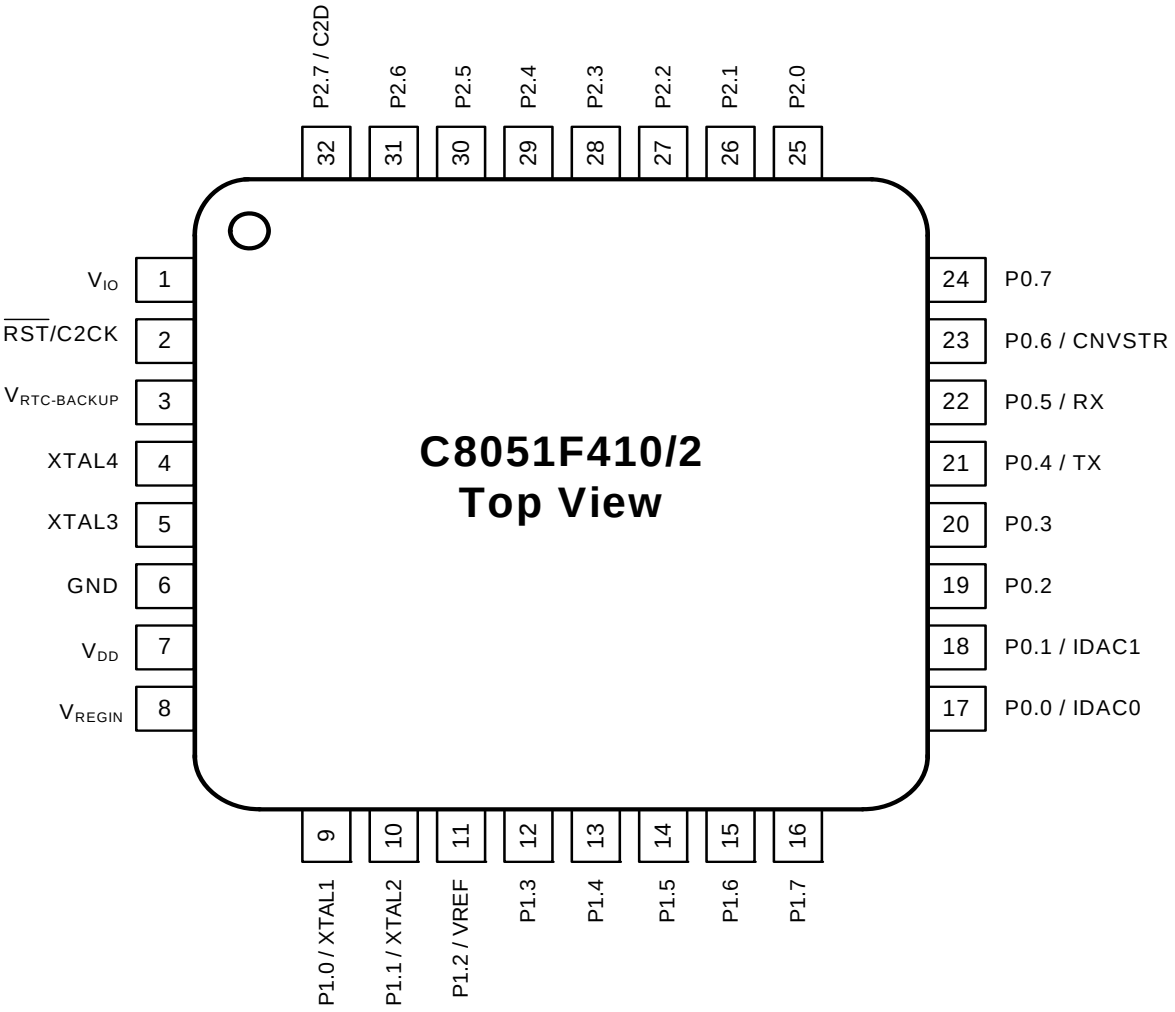



**Figure 4.1. LQFP-32 Pinout Diagram (Top View)**

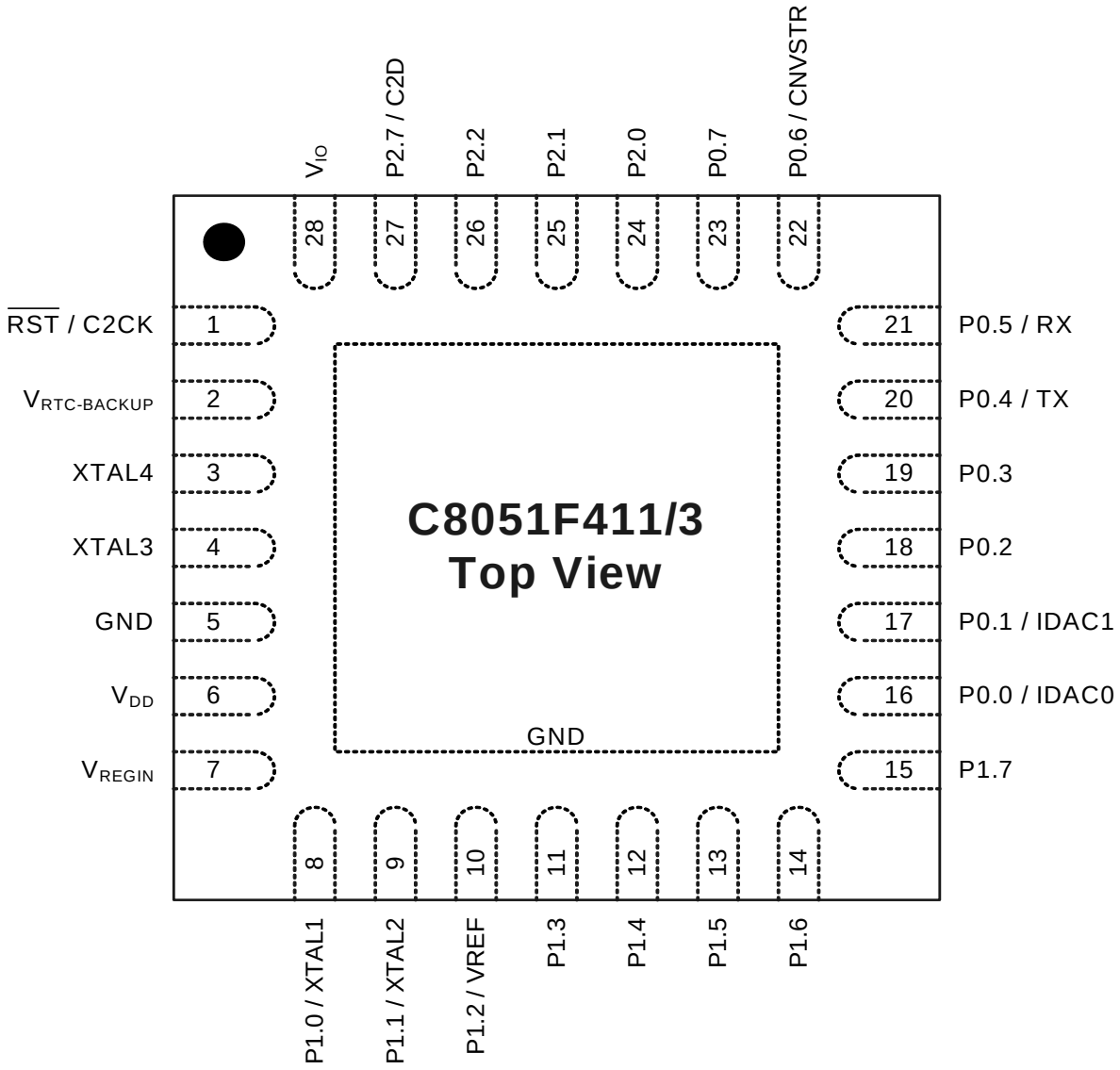



**Figure 4.2. QFN-28 Pinout Diagram (Top View)**

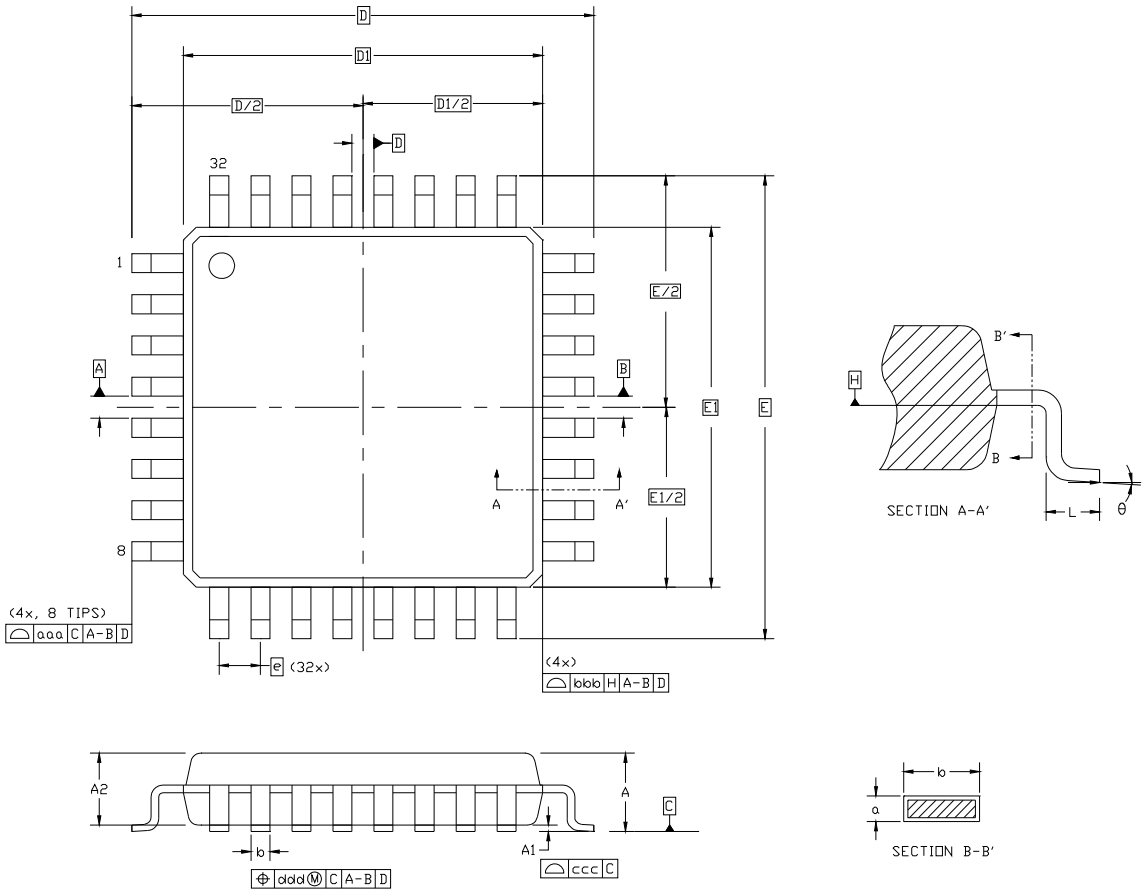



**Figure 4.3. LQFP-32 Package Diagram**

**Table 4.2. LQFP-32 Package Dimensions**

| | MM | | |
|---|---|---|---|
| | **MIN** | **TYP** | **MAX** |
| A | — | — | 1.60 |
| A1 | 0.05 | — | 0.15 |
| A2 | 1.35 | 1.40 | 1.45 |
| b | 0.30 | 0.37 | 0.45 |
| c | 0.09 | — | 0.20 |
| D | — | 9.00 | — |
| D1 | — | 7.00 | — |
| e | — | 0.80 | — |
| E | — | 9.00 | — |
| E1 | — | 7.00 | — |
| L | 0.45 | 0.60 | 0.75 |

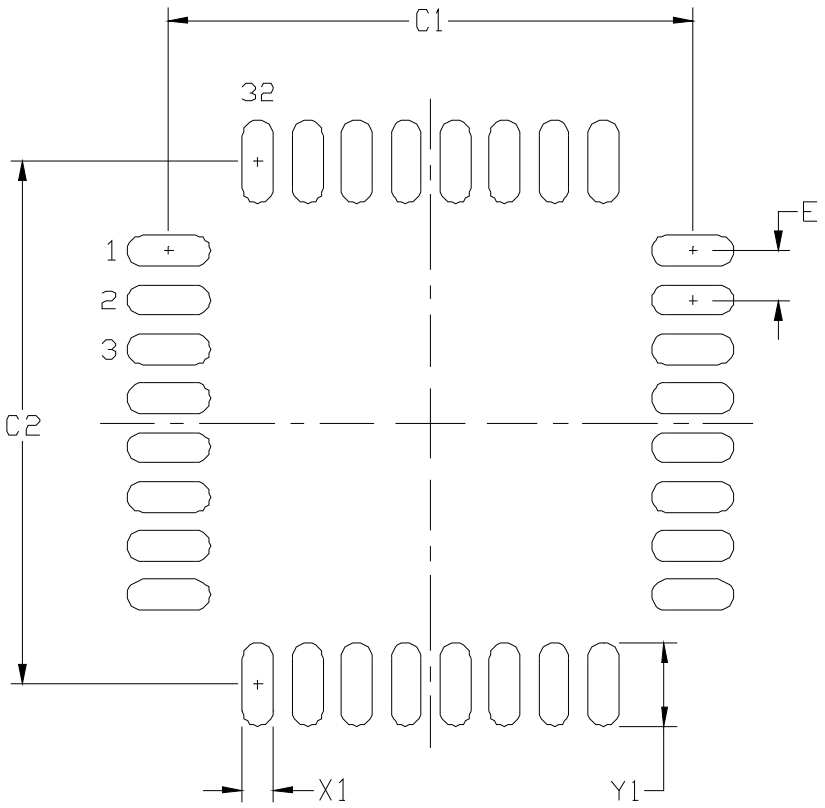



**Figure 4.4. LQFP-32 Recommended PCB Land Pattern**

**Table 4.3. LQFP-32 PCB Land Pattern Dimensions**

| Dimension | Min | Max |
|---|---|---|
| C1 | 8.40 | 8.50 |
| C2 | 8.40 | 8.50 |
| E | 0.80 BSC | |
| X1 | 0.40 | 0.50 |
| Y1 | 1.25 | 1.35 |

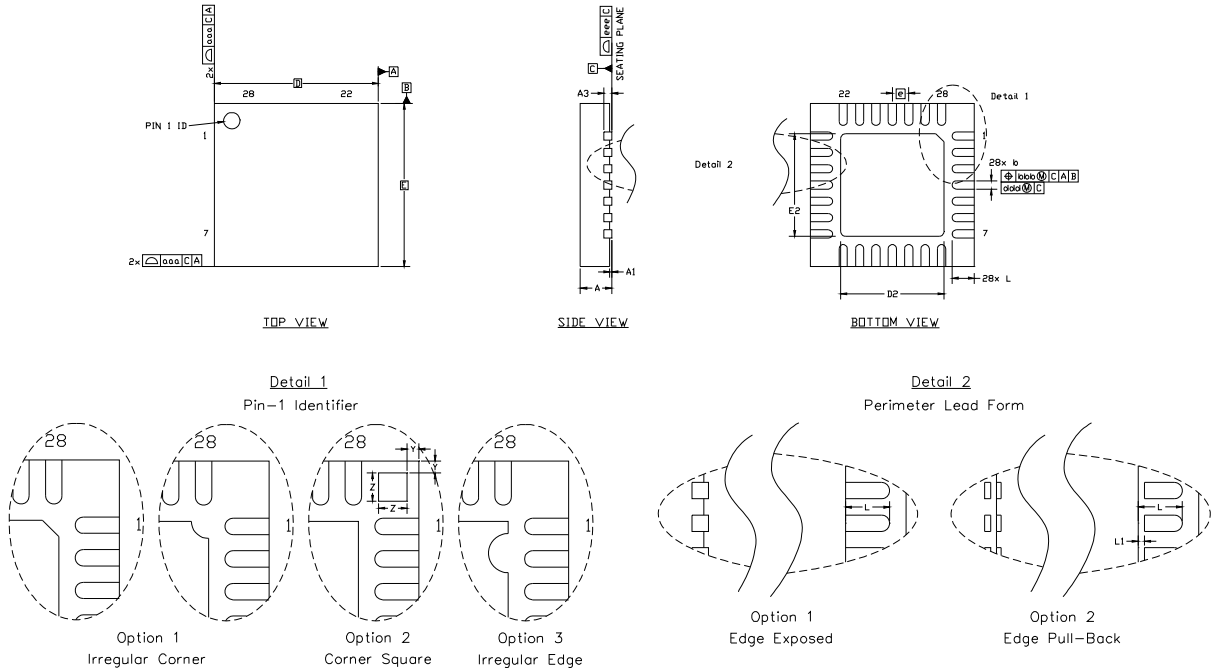



**Figure 4.5. QFN-28 Package Drawing**

**Table 4.4. QFN-28 Package Dimensions**

| Dimension | Min | Typ | Max | Dimension | Min | Typ | Max |
|---|---|---|---|---|---|---|---|
| A | 0.80 | 0.90 | 1.00 | L | 0.35 | 0.55 | 0.65 |
| A1 | 0.00 | 0.02 | 0.05 | L1 | 0.00 | — | 0.15 |
| A3 | | 0.25 REF | | aaa | | 0.15 | |
| b | 0.18 | 0.23 | 0.30 | bbb | | 0.10 | |
| D | | 5.00 BSC. | | ddd | | 0.05 | |
| D2 | 2.90 | 3.15 | 3.35 | eee | | 0.08 | |
| e | | 0.50 BSC. | | Z | | 0.44 | |
| E | | 5.00 BSC. | | Y | | 0.18 | |
| E2 | 2.90 | 3.15 | 3.35 | | | | |

**Notes:**

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This drawing conforms to the JEDEC Solid State Outline MO-220, variation VHHD except for custom features D2, E2, Z, Y, and L which are toleranced per supplier designation.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020C specification for Small Body Components.

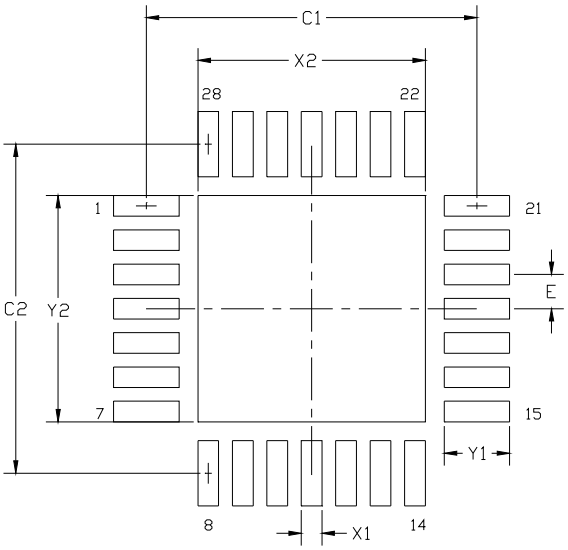



**Figure 4.6. QFN-28 Recommended PCB Land Pattern**

**Table 4.5. QFN-28 PCB Land Pattern Dimensions**

| Dimension | Min | Max | Dimension | Min | Max |
|---|---|---|---|---|---|
| C1 | 4.80 | | X2 | 3.20 | 3.30 |
| C2 | 4.80 | | Y1 | 0.85 | 0.95 |
| E | 0.50 | | Y2 | 3.20 | 3.30 |
| X1 | 0.20 | 0.30 | | | |

**Notes:**

General

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing is per the ANSI Y14.5M-1994 specification.
3. This Land Pattern Design is based on the IPC-7351 guidelines.

Solder Mask Design

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

Stencil Design

5. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
6. The stencil thickness should be 0.125mm (5 mils).
7. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pins.
8. A 3x3 array of 0.90mm openings on a 1.1mm pitch should be used for the center pad to assure the proper paste volume (67% Paste Coverage).

Card Assembly

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

## NOTES:

# 5. 12-Bit ADC (ADC0)

The ADC0 subsystem for the C8051F41x consists of an analog multiplexer (AMUX0) with 27 total input selections, and a 200 ksps, 12-bit successive-approximation-register ADC with integrated track-and-hold, programmable window detector, and hardware accumulator. The ADC0 subsystem has a special **Burst Mode** which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. The AMUX0, data conversion modes, and window detector are all configurable under software control via the Special Function Registers shown in Figure 5.1. ADC0 inputs are single-ended and may be configured to measure P0.0-P2.7, the Temperature Sensor output, $V_{DD}$, or GND with respect to GND. ADC0 is enabled when the AD0EN bit in the ADC0 Control register (ADC0CN) is set to logic 1, or when performing conversions in Burst Mode. ADC0 is in low power shutdown when AD0EN is logic 0 and no Burst Mode conversions are taking place.

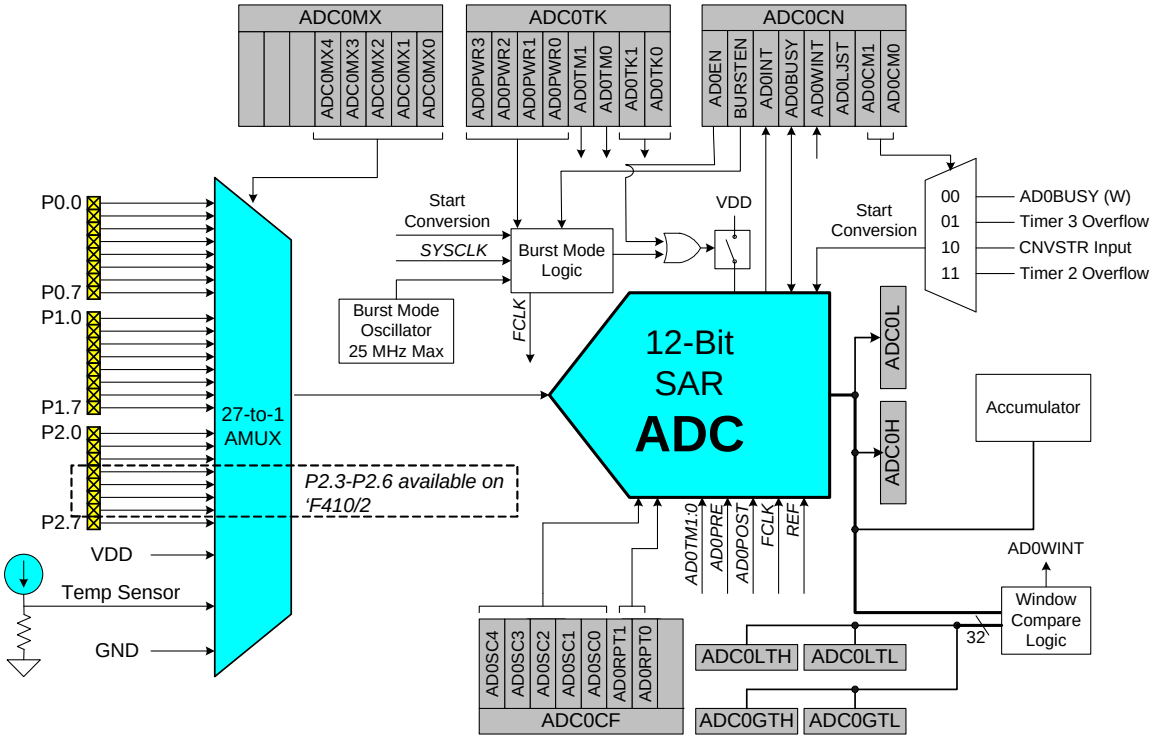



**Figure 5.1. ADC0 Functional Block Diagram**

## 5.1. Analog Multiplexer

AMUX0 selects the input channel to the ADC. Any of the following may be selected as an input: P0.0-P2.7, the on-chip temperature sensor, the core power supply ($V_{DD}$), or ground (GND). **ADC0 is single-ended and all signals measured are with respect to GND.** The ADC0 input channels are selected using the ADC0MX register as described in SFR Definition 5.1.

**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 (for n = 0,1,2) and write a '1' in the corresponding Port Latch register Pn (for n = 0,1,2). To force the Crossbar to skip a Port pin, set to '1' the corresponding bit in register PnSKIP (for n = 0,1,2). See **Section "18. Port Input/Output" on page 147** for more Port I/O configuration details.

## 5.2. Temperature Sensor

The typical temperature sensor transfer function is shown in Figure 5.2. The output voltage ($V_{TEMP}$) is the positive ADC input when the temperature sensor is selected by bits AD0MX4-0 in register ADC0MX.

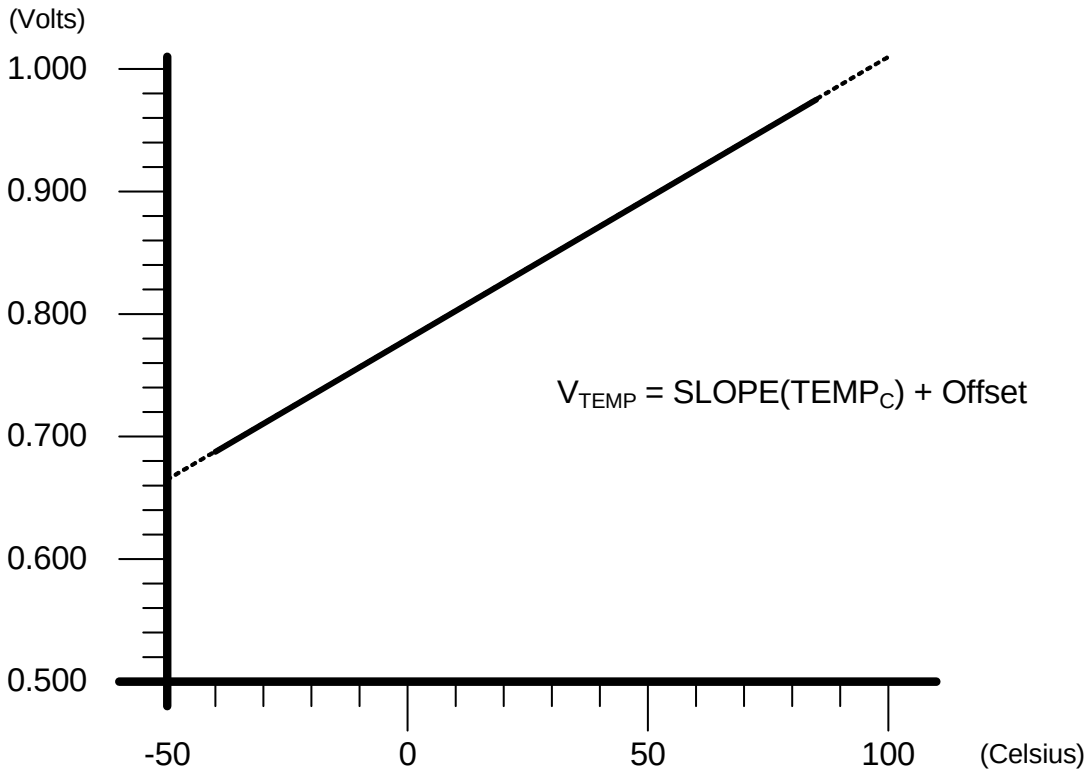



**Figure 5.2. Typical Temperature Sensor Transfer Function**

## 5.3. ADC0 Operation

In a typical system, ADC0 is configured using the following steps:

Step 1. Choose the start of conversion source.
Step 2. Choose Normal Mode or Burst Mode operation.
Step 3. If Burst Mode, choose the ADC0 Idle Power State and set the Power-Up Time.
Step 4. Choose the tracking mode. Note that Pre-Tracking Mode can only be used with Normal Mode.
Step 5. Calculate required settling time and set the post convert-start tracking time using the AD0TK bits.
Step 6. Choose the repeat count.
Step 7. Choose the output word justification (Right-Justified or Left-Justified).
Step 8. Enable or disable the End of Conversion and Window Comparator Interrupts.

### 5.3.1. Starting a Conversion

A conversion can be initiated in one of four ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM1-0) in register ADC0CN. Conversions may be initiated by one of the following:

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

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 **Section “24. Timers” on page 231** 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 bit 6 in the P0SKIP register to logic 1. See **Section “18. Port Input/Output” on page 147** for details on Port I/O configuration.

### 5.3.2. Tracking Modes

According to Table 5.3 and Table 5.4, each ADC0 conversion must be preceded by a minimum tracking time for the converted result to be accurate. ADC0 has three tracking modes: Pre-Tracking, Post-Tracking, and Dual-Tracking. Pre-Tracking Mode provides the minimum delay between the convert start signal and end of conversion by tracking continuously before the convert start signal. This mode requires software management in order to meet minimum tracking requirements. In Post-Tracking Mode, a programmable tracking time starts after the convert start signal and is managed by hardware. Dual-Tracking Mode maximizes tracking time by tracking before and after the convert start signal. Figure 5.3 shows examples of the three tracking modes.

Pre-Tracking Mode is selected when AD0TM is set to 10b. Conversions are started immediately following the convert start signal. ADC0 is tracking continuously when not performing a conversion. Software must allow at least the minimum tracking time between each end of conversion and the next convert start signal. The minimum tracking time must also be met prior to the first convert start signal after ADC0 is enabled.

Post-Tracking Mode is selected when AD0TM is set to 01b. A programmable tracking time based on AD0TK is started immediately following the convert start signal. Conversions are started after the programmed tracking time ends. After a conversion is complete, ADC0 does not track the input. Rather, the sampling capacitor remains disconnected from the input making the input pin high-impedance until the next convert start signal.

Dual-Tracking Mode is selected when AD0TM is set to 11b. A programmable tracking time based on AD0TK is started immediately following the convert start signal. Conversions are started after the programmed tracking time ends. After a conversion is complete, ADC0 tracks continuously until the next conversion is started.

Depending on the output connected to the ADC input, additional tracking time, more than is specified in Table 5.3 and Table 5.4, may be required after changing MUX settings. See the settling time requirements described in **Section "5.3.6. Settling Time Requirements" on page 58**.

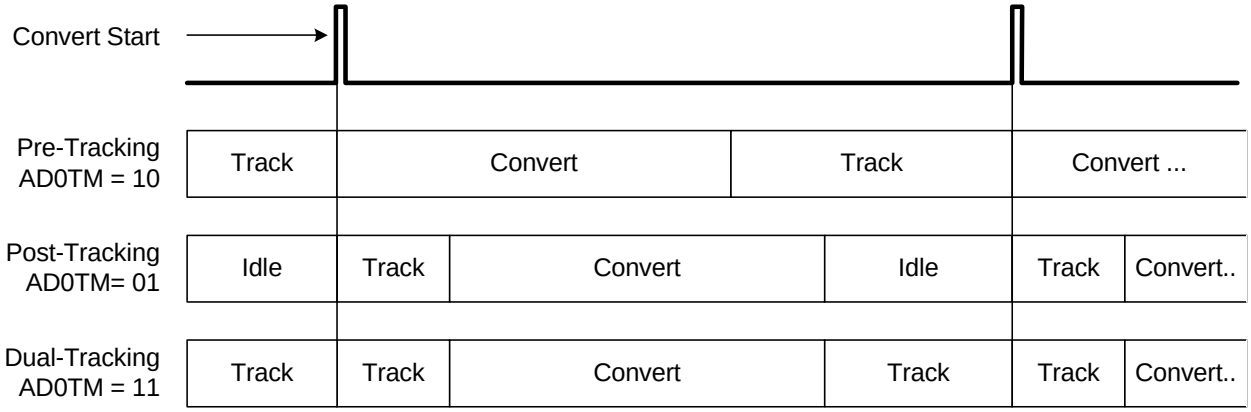



**Figure 5.3. ADC0 Tracking Modes**

### 5.3.3. Timing

ADC0 has a maximum conversion speed specified in Table 5.3 and Table 5.4. ADC0 is clocked from the ADC0 Subsystem Clock (FCLK). The source of FCLK is selected based on the BURSTEN bit. When BURSTEN is logic 0, FCLK is derived from the current system clock. When BURSTEN is logic 1, FCLK is derived from the Burst Mode Oscillator, an independent clock source with a maximum frequency of 25 MHz.

When ADC0 is performing a conversion, it requires a clock source that is typically slower than FCLK. The ADC0 SAR conversion clock (SAR clock) is a divided version of FCLK. The divide ratio can be configured using the AD0SC bits in the ADC0CF register. The maximum SAR clock frequency is listed in Table 5.3 and Table 5.4.

ADC0 can be in one of three states at any given time: tracking, converting, or idle. Tracking time depends on the tracking mode selected. For Pre-Tracking Mode, tracking is managed by software and ADC0 starts conversions immediately following the convert start signal. For Post-Tracking and Dual-Tracking Modes, the tracking time after the convert start signal is equal to the value determined by the AD0TK bits plus 2 FCLK cycles. Tracking is immediately followed by a conversion. The ADC0 conversion time is always 13 SAR clock cycles plus an additional 2 FCLK cycles to start and complete a conversion. Figure 5.4 shows timing diagrams for a conversion in Pre-Tracking Mode and tracking plus conversion in Post-Tracking or Dual-Tracking Mode. In this example, repeat count is set to one.

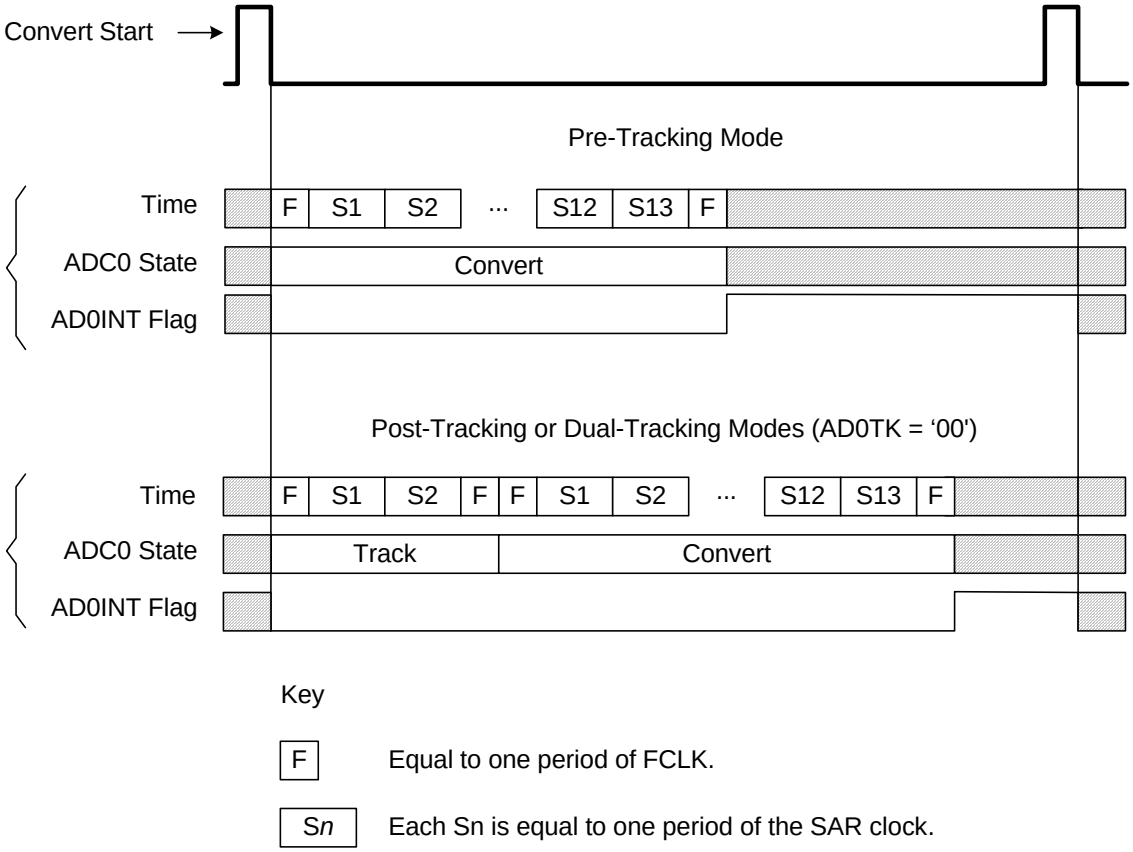



**Figure 5.4. 12-Bit ADC Tracking Mode Example**

### 5.3.4. 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, or 16 samples using an internal Burst Mode clock (approximately 25 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 5.5 shows an example of Burst Mode Operation with a slow system clock and a repeat count of 4.

**Important Note:** When Burst Mode is enabled, only Post-Tracking and Dual-Tracking modes can be used.

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.

**Note:** 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.

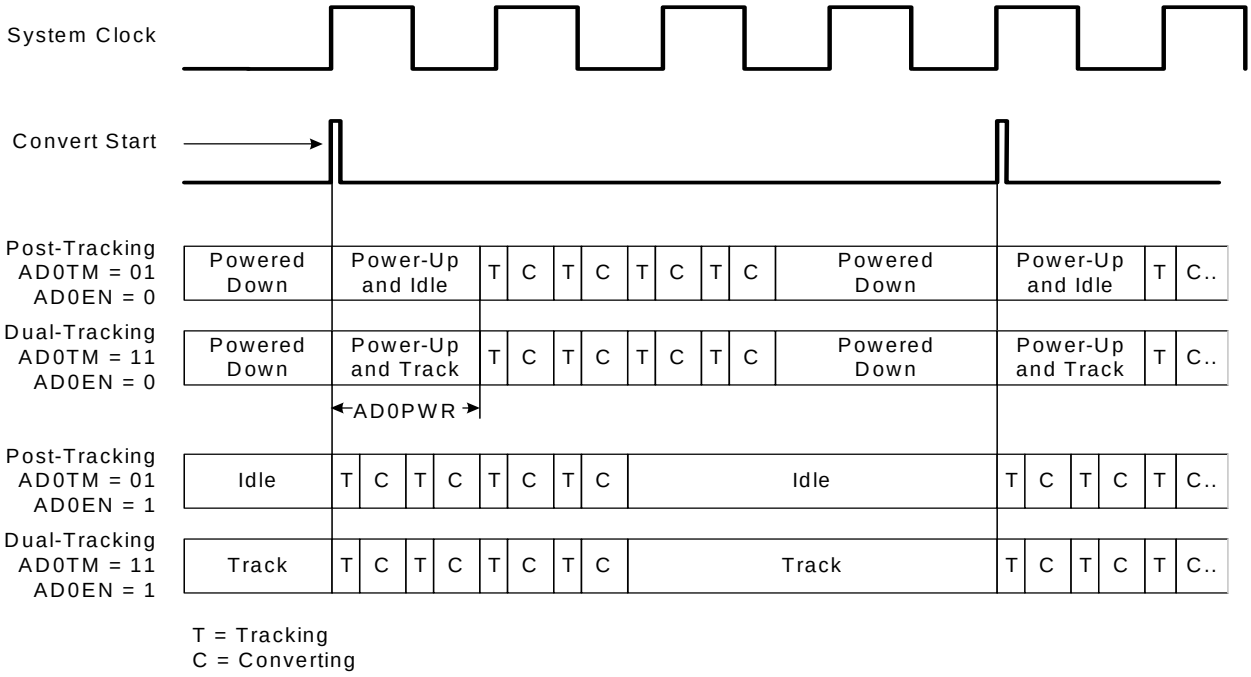



**Figure 5.5. 12-Bit ADC Burst Mode Example with Repeat Count Set to 4**

### 5.3.5. Output Conversion Code

The registers ADC0H and ADC0L contain the high and low bytes of the output conversion code. When the repeat count is set to 1, conversion codes are represented in 12-bit unsigned integer format and the output conversion code is updated after each conversion. Inputs are measured from ‘0’ to $V_{REF}$ x 4095/4096. Data can be right-justified or left-justified, depending on the setting of the AD0LJST bit (ADC0CN.2). Unused bits in the ADC0H and ADC0L registers are set to ‘0’. Example codes are shown in Table 5.1 for both right-justified and left-justified data.

**Table 5.1. ADC0 Examples of Right- and Left-Justified Samples**

| Input Voltage | Right-Justified ADC0H:ADC0L (AD0LJST = 0) | Left-Justified ADC0H:ADC0L (AD0LJST = 1) |
|---|---|---|
| $V_{REF}$ x 4095/4096 | 0x0FFF | 0xFFF0 |
| $V_{REF}$ x 2048/4096 | 0x0800 | 0x8000 |
| $V_{REF}$ x 2047/4096 | 0x07FF | 0x7FF0 |
| 0 | 0x0000 | 0x0000 |

When the ADC0 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, or 16 consecutive samples can be accumulated and represented in unsigned integer format. The repeat count can be selected using the AD0RPT bits in the ADC0CF register. The value must be right-justified (AD0LJST = “0”), and unused bits in the ADC0H and ADC0L registers are set to '0'. The example in Table 5.2 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.

**Table 5.2. ADC0 Repeat Count Examples at Various Input Voltages**

| Input Voltage | Repeat Count = 4 | Repeat Count = 8 | Repeat Count = 16 |
|---|---|---|---|
| $V_{REF}$ x 4095/4096 | 0x3FFC | 0x7FF8 | 0xFFF0 |
| $V_{REF}$ x 2048/4096 | 0x2000 | 0x4000 | 0x8000 |
| $V_{REF}$ x 2047/4096 | 0x1FFC | 0x3FF8 | 0x7FF0 |
| 0 | 0x0000 | 0x0000 | 0x0000 |

### 5.3.6. Settling Time Requirements

A minimum tracking time is required before an accurate conversion can be performed. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion.

Figure 5.6 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 $V_{DD}$ with respect to GND, $R_{TOTAL}$ reduces to $R_{MUX}$. See Table 5.3 and Table 5.4 for ADC0 minimum settling time requirements.

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

**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 (12).

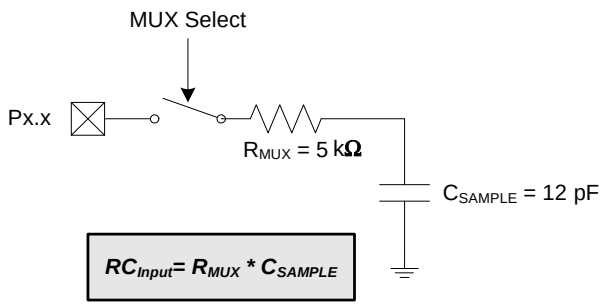



**Figure 5.6. ADC0 Equivalent Input Circuits**

## SFR Definition 5.1. ADC0MX: ADC0 Channel Select

| R | R | R | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | AD0MX | | | | | 00011111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xBB |

Bits7–5: UNUSED. Read = 000b; Write = don't care.
Bits4–0: AD0MX4–0: AMUX0 Positive Input Selection

| AD0MX4–0 | ADC0 Input Channel |
|---|---|
| 00000 | P0.0 |
| 00001 | P0.1 |
| 00010 | P0.2 |
| 00011 | P0.3 |
| 00100 | P0.4 |
| 00101 | P0.5 |
| 00110 | P0.6 |
| 00111 | P0.7 |
| 01000 | P1.0 |
| 01001 | P1.1 |
| 01010 | P1.2 |
| 01011 | P1.3 |
| 01100 | P1.4 |
| 01101 | P1.5 |
| 01110 | P1.6 |
| 01111 | P1.7 |
| 10000 | P2.0 |
| 10001 | P2.1 |
| 10010 | P2.2 |
| 10011 | P2.3* |
| 10100 | P2.4* |
| 10101 | P2.5* |
| 10110 | P2.6* |
| 10111 | P2.7 |
| 11000 | Temp Sensor |
| 11001 | $V_{DD}$ |
| 11010 - 11111 | GND |

***Note:** Only applies to C8051F410/2; selection RESERVED on C8051F411/3 devices.

## SFR Definition 5.2. ADC0CF: ADC0 Configuration

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| AD0SC | | | | | AD0RPT | | Reserved | 11111000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xBC |

Bits7–3: AD0SC4–0: ADC0 SAR Conversion Clock Period Bits.
SAR Conversion clock is derived from FCLK by the following equation, where *AD0SC* refers to the 5-bit value held in bits AD0SC4-0. SAR Conversion clock requirements are given in Table 5.3.
BURSTEN = 0: FCLK is the current system clock.
BURSTEN = 1: FCLK is a maximum of 25 MHz, independent of the current system clock.

$$AD0SC = \frac{FCLK}{CLK_{SAR}} - 1^{*} \quad \text{or} \quad CLK_{SAR} = \frac{FCLK}{AD0SC + 1}$$

***Note:** Round the result up.

Bits2–1: AD0RPT1–0: ADC0 Repeat Count.
Controls the number of conversions taken and accumulated between ADC0 End of Conversion (ADCINT) and ADC0 Window Comparator (ADCWINT) interrupts. A convert start is required for each conversion unless Burst Mode is enabled. In Burst Mode, a single convert start can initiate multiple self-timed conversions. Results in both modes are accumulated in the ADC0H:ADC0L register. **When AD0RPT1-0 are set to a value other than '00', the AD0LJST bit in the ADC0CN register must be set to '0' (right justified).**
00: 1 conversion is performed.
01: 4 conversions are performed and accumulated.
10: 8 conversions are performed and accumulated.
11: 16 conversions are performed and accumulated.

**Note:** The ADC0 output register is automatically reset to 0x0000 upon reaching the last conversion specified by the repeat counter. If the ADC is disabled during a conversion and re-enabled later, the ADC0H and ADC0L registers should be manually cleared to 0x00.

Bit0: RESERVED. Read = 0b; Must write 0b.

## SFR Definition 5.3. ADC0H: ADC0 Data Word MSB

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xBE |

Bits7-0: ADC0 Data Word High-Order Bits.
For AD0LJST = 0 and AD0RPT as follows:
00: Bits 3–0 are the upper 4 bits of the accumulated result. Bits 7–4 are 0000b.
01: Bits 5–0 are the upper 6 bits of the accumulated result. Bits 7–6 are 00b.
10: Bits 6–0 are the upper 7 bits of the accumulated result. Bit 7 is 0b.
11: Bits 7–0 are the upper 8 bits of the accumulated result.
For AD0LJST = 1 (AD0RPT must be '00'): Bits 7–0 are the most-significant bits of the ADC0 12-bit result.

## SFR Definition 5.4. ADC0L: ADC0 Data Word LSB

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xBD |

Bits7-0: ADC0 Data Word Low-Order Bits.
For AD0LJST = 0: Bits 7-0 are the lower 8 bits of the ADC0 accumulated result.
For AD0LJST = 1 (AD0RPT must be '00'): Bits 7-4 are the lower 4 bits of the 12-bit result. Bits 3-0 are 0000b.

## SFR Definition 5.5. ADC0CN: ADC0 Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| AD0EN | BURSTEN | AD0INT | AD0BUSY | AD0WINT | AD0LJST | AD0CM1 | AD0CM0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: |
| | | | | | | | (bit addressable) | 0xE8 |

Bit7: AD0EN: ADC0 Enable Bit.
0: ADC0 Disabled. ADC0 is in low-power shutdown.
1: ADC0 Enabled. ADC0 is active and ready for data conversions.

Bit6: BURSTEN: ADC0 Burst Mode Enable Bit.
0: ADC0 Burst Mode Disabled.
1: ADC0 Burst Mode Enabled.

Bit5: AD0INT: ADC0 Conversion Complete Interrupt Flag.
0: ADC0 has not completed a data conversion since the last time AD0INT was cleared.
1: ADC0 has completed a data conversion.

Bit4: AD0BUSY: ADC0 Busy Bit.
Read:
0: ADC0 conversion is complete or a conversion is not currently in progress. AD0INT is set to logic 1 on the falling edge of AD0BUSY.
1: ADC0 conversion is in progress.
Write:
0: No Effect.
1: Initiates ADC0 Conversion if AD0CM1-0 = 00b

Bit3: AD0WINT: ADC0 Window Compare Interrupt Flag.
This bit must be cleared by software.
0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared.
1: ADC0 Window Comparison Data match has occurred.

Bit2: AD0LJST: ADC0 Left Justify Select
0: Data in ADC0H:ADC0L registers is right justified.
1: Data in ADC0H:ADC0L registers is left justified. This option should not be used with a repeat count greater than 1 (when AD0RPT1-0 is 01b, 10b, or 11b).

Bits1-0: AD0CM1-0: ADC0 Start of Conversion Mode Select.
00: ADC0 conversion initiated on every write of ‘1’ to AD0BUSY.
01: ADC0 conversion initiated on overflow of Timer 3.
10: ADC0 conversion initiated on rising edge of external CNVSTR.
11: ADC0 conversion initiated on overflow of Timer 2.

**SFR Definition 5.6. ADC0TK: ADC0 Tracking Mode Select**

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| AD0PWR | | | | AD0TM | | AD0TK | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: |
| | | | | | | | (bit addressable) | 0xBA |

Bits7–4: AD0PWR3–0: ADC0 Burst Power-Up Time.
For BURSTEN = 0:
ADC0 power state controlled by AD0EN.
For BURSTEN = 1 and AD0EN = 1;
ADC0 remains enabled and does not enter the low power state.
For BURSTEN = 1 and AD0EN = 0:
ADC0 enters the low power state as specified in Table 5.3 and Table 5.4 and is enabled after each convert start signal. The Power Up time is programmed according to the following equation:

$$AD0PWR = \frac{Tstartup}{400ns} - 1 \quad \text{or} \quad Tstartup = (AD0PWR + 1)400ns$$

Bits3–2: AD0TM1–0: ADC0 Tracking Mode Select Bits.
00: Reserved.
01: ADC0 is configured to Post-Tracking Mode.
10: ADC0 is configured to Pre-Tracking Mode.
11: ADC0 is configured to Dual-Tracking Mode (default).

Bits1–0: AD0TK1–0: ADC0 Post-Track Time.
Post-Tracking time is controlled by AD0TK as follows:
00: Post-Tracking time is equal to 2 SAR clock cycles + 2 FCLK cycles.
01: Post-Tracking time is equal to 4 SAR clock cycles + 2 FCLK cycles.
10: Post-Tracking time is equal to 8 SAR clock cycles + 2 FCLK cycles.
11: Post-Tracking time is equal to 16 SAR clock cycles + 2 FCLK cycles.

## 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 5.7. ADC0GTH: ADC0 Greater-Than Data High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xC4 |

Bits7–0: High byte of ADC0 Greater-Than Data Word.

## SFR Definition 5.8. ADC0GTL: ADC0 Greater-Than Data Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xC3 |

Bits7–0: Low byte of ADC0 Greater-Than Data Word.

## SFR Definition 5.9. ADC0LTH: ADC0 Less-Than Data High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xC6 |

Bits7–0: High byte of ADC0 Less-Than Data Word.

## SFR Definition 5.10. ADC0LTL: ADC0 Less-Than Data Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xC5 |

Bits7–0: Low byte of ADC0 Less-Than Data Word.

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

Figure 5.7 shows two example window comparisons for right-justified data with ADC0LTH:ADC0LTL = 0x0200 (512d) and ADC0GTH:ADC0GTL = 0x0100 (256d). The input voltage can range from '0' to $V_{REF}$ x (4095/4096) with respect to GND, and is represented by a 12-bit unsigned integer value. The repeat count is set to one. 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 0x0100 < ADC0H:ADC0L < 0x0200). 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 < 0x0100 or ADC0H:ADC0L > 0x0200). Figure 5.8 shows an example using left-justified data with the same comparison values.

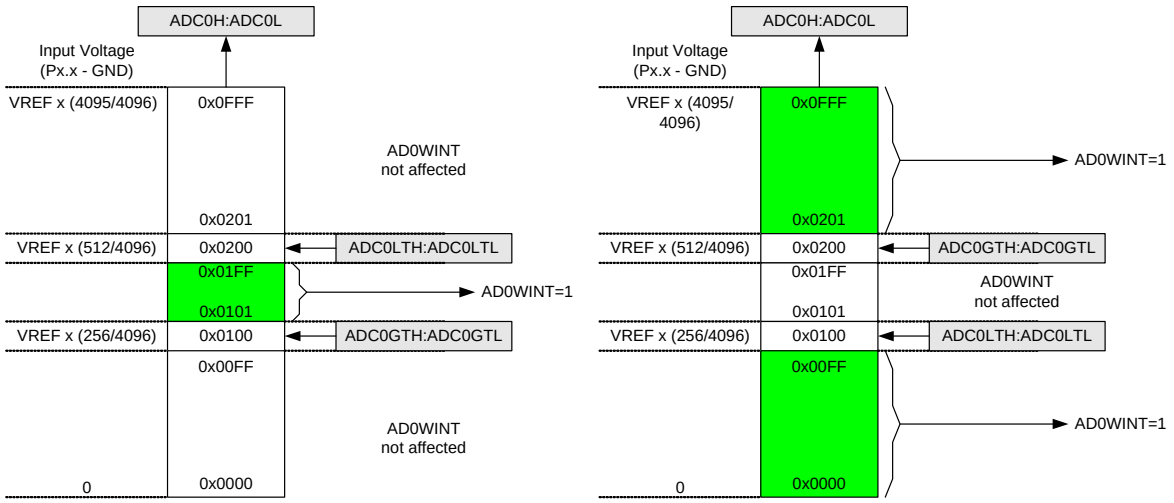



**Figure 5.7. ADC Window Compare Example: Right-Justified Single-Ended Data**

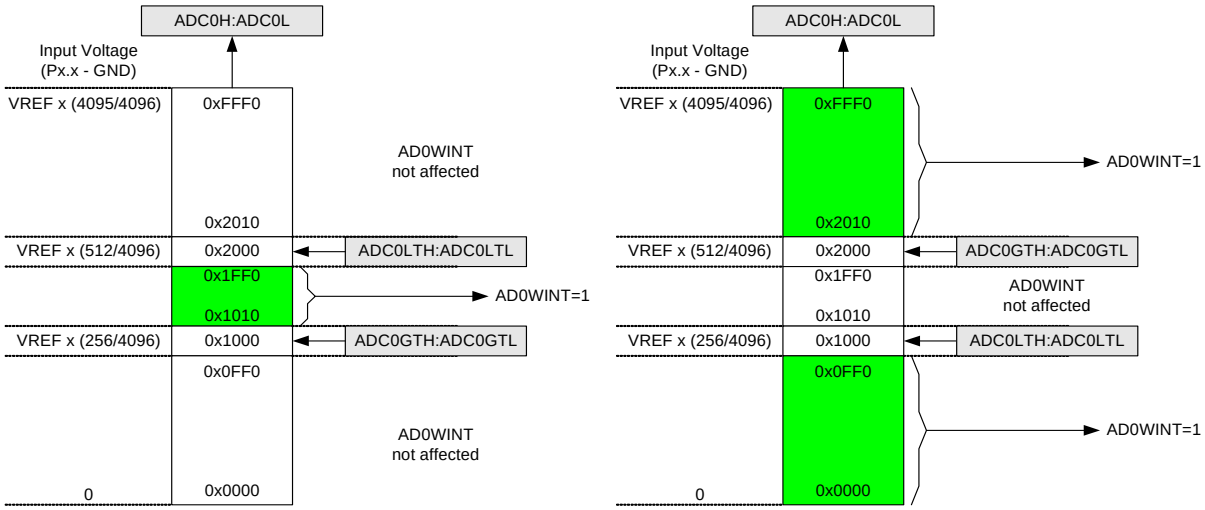



**Figure 5.8. ADC Window Compare Example: Left-Justified Single-Ended Data**

## Table 5.3. ADC0 Electrical Characteristics ($V_{DD}$ = 2.5 V, $V_{REF}$ = 2.2 V)

$V_{DD}$ = 2.5 V, $V_{REF}$ = 2.2 V (REFSL=0), –40 to +85 °C unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **DC Accuracy** | | | | | |
| Resolution | | | 12 | | bits |
| Integral Nonlinearity | | — | — | ±1 | LSB |
| Differential Nonlinearity | Guaranteed Monotonic | — | — | ±1 | LSB |
| Offset Error | | — | ±3 | ±10 | LSB |
| Full Scale Error | | — | ±3 | ±10 | LSB |
| **Dynamic Performance (10 kHz sine-wave Single-ended input, 0 to 1 dB below Full Scale, 200 ksps)** | | | | | |
| Signal-to-Noise Plus Distortion | Regular Mode (BURSTEN = '0')<br>Burst Mode (BURSTEN = '0') | 66<br>60 | 69<br>63 | —<br>— | dB |
| Total Harmonic Distortion | Up to the 5th harmonic | — | –77 | — | dB |
| Spurious-Free Dynamic Range | | — | –94 | — | dB |
| **Conversion Rate** | | | | | |
| SAR Conversion Clock | Regular Mode (BURSTEN = '0') | — | — | 3 | MHz |
| Conversion Time in SAR Clocks[1] | | — | 13 | — | clocks |
| Track/Hold Acquisition Time[2] | | 1 | — | — | µs |
| Throughput Rate | | — | — | 200 | ksps |
| **Analog Inputs** | | | | | |
| Input Voltage Range | | 0 | — | $V_{REF}$ | V |
| Input Capacitance | | — | 12 | — | pF |
| **Temperature Sensor** | | | | | |
| Linearity[3,4] | | — | ±0.2 | — | °C |
| Slope[4] | | — | 2.95 | — | mV/°C |
| Slope Error[3] | | — | ±73 | — | µV/°C |
| Offset[4] | (Temp = 0 °C) | — | 900 | — | mV |
| Offset Error[3] | | — | ±17 | — | mV |
| **Power Specifications** | | | | | |
| Power Supply Current ($V_{DD}$ supplied to ADC0) | Operating Mode, 200 ksps | — | 680 | 1000 | µA |
| Burst Mode (Idle) | | — | 100 | — | µA |
| Power Supply Rejection | | — | 1 | — | mV/V |

**Notes:**
1. An additional 2 FCLK cycles are required to start and complete a conversion.
2. Additional tracking time may be required depending on the output impedance connected to the ADC input. See Section "**5.3.6. Settling Time Requirements**" on page **58**.
3. Represents one standard deviation from the mean.
4. Includes ADC offset, gain, and linearity variations.

## Table 5.4. ADC0 Electrical Characteristics ($V_{DD}$ = 2.1 V, $V_{REF}$ = 1.5 V)

$V_{DD}$ = 2.1 V, $V_{REF}$ = 1.5 V (REFSL = 0), –40 to +85 °C unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **DC Accuracy** | | | | | |
| Resolution | | | 12 | | bits |
| Integral Nonlinearity | | — | — | ±1 | LSB |
| Differential Nonlinearity | Guaranteed Monotonic | — | — | ±1 | LSB |
| Offset Error | | — | ±3 | ±10 | LSB |
| Full Scale Error | | — | ±3 | ±10 | LSB |
| **Dynamic Performance (10 kHz sine-wave Single-ended input, 0 to 1 dB below Full Scale, 200 ksps)** | | | | | |
| Signal-to-Noise Plus Distortion | Regular Mode (BURSTEN = '0')<br>Burst Mode (BURSTEN = '0') | 66<br>60 | 68<br>62 | —<br>— | dB |
| Total Harmonic Distortion | Up to the 5th harmonic | — | –75 | — | dB |
| Spurious-Free Dynamic Range | | — | –90 | — | dB |
| **Conversion Rate** | | | | | |
| SAR Conversion Clock | Regular Mode (BURSTEN = '0') | — | — | 3 | MHz |
| Conversion Time in SAR Clocks[1] | | — | 13 | — | clocks |
| Track/Hold Acquisition Time[2] | | 1 | — | — | µs |
| Throughput Rate | | — | — | 200 | ksps |
| **Analog Inputs** | | | | | |
| Input Voltage Range | | 0 | — | $V_{REF}$ | V |
| Input Capacitance | | — | 12 | — | pF |
| **Temperature Sensor** | | | | | |
| Linearity[3,4] | | — | ±0.2 | — | °C |
| Slope[4] | | — | 2.95 | — | mV/°C |
| Slope Error[3] | | — | ±73 | — | µV/°C |
| Offset | (Temp = 0 °C) | — | 900 | — | mV |
| Offset Error[3] | | — | ±17 | — | mV |
| **Power Specifications** | | | | | |
| Power Supply Current ($V_{DD}$ supplied to ADC0) | Operating Mode, 200 ksps | — | 650 | 1000 | µA |
| Burst Mode (Idle) | | — | 100 | — | µA |
| Power Supply Rejection | | — | 1 | — | mV/V |

**Notes:**
1. An additional 2 FCLK cycles are required to start and complete a conversion.
2. Additional tracking time may be required depending on the output impedance connected to the ADC input. See Section "**5.3.6. Settling Time Requirements**" on page **58**.
3. Represents one standard deviation from the mean.
4. Includes ADC offset, gain, and linearity variations.

# 6. 12-Bit Current Mode DACs (IDA0 and IDA1)

The C8051F41x devices include two 12-bit current-mode Digital-to-Analog Converters (IDACs). The maximum current output of the IDACs can be adjusted for four different current settings; 0.25 mA, 0.5 mA, 1 mA, and 2 mA. The IDACs can be individually enabled or disabled using the enable bits in the corresponding IDAC Control Register (IDA0CN or IDA1CN). When both IDACs are enabled, their outputs may be routed to individual pins or merged onto a single pin. An internal bandgap bias generator is used to generate a reference current for the IDACs whenever they are enabled. IDAC updates can be performed on-demand, scheduled on a Timer overflow, or synchronized with an external pin edge. Figure 6.1 shows a block diagram of the IDAC circuitry.

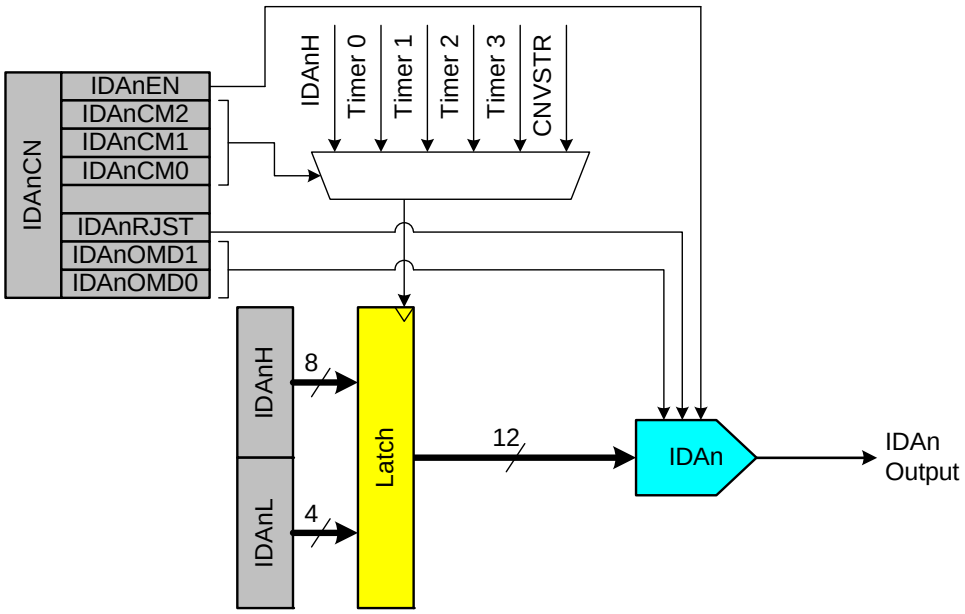



**Figure 6.1. IDAC Functional Block Diagram**

## 6.1. IDAC Output Scheduling

A flexible output update mechanism allows for seamless full-scale changes and supports jitter-free updates for waveform generation. Three update modes are provided, allowing IDAC output updates on a write to the IDAC's data register, on a Timer overflow, or on an external pin edge.

### 6.1.1. Update Output On-Demand

In its default mode (IDAnCN.[6:4] = '111') the IDAC output is updated "on-demand" with a write to the data register high byte (IDAnH). It is important to note that in this mode, writes to the data register low byte (IDAnL) are held and have no effect on the IDAn output until a write to IDAnH takes place. Since data from both the high and low bytes of the data register are immediately latched to IDAn after a write to IDAnH, **the write sequence when writing a full 12-bit word to the IDAC data registers should be IDAnL followed by IDAnH**. When the data word is left justified, the IDAC can be used in 8-bit mode by initializing IDAnL to the desired value (typically 0x00), and writing data only to IDA0H.

### 6.1.2. Update Output Based on Timer Overflow

The IDAC output update can be scheduled on a Timer overflow. This feature is useful in systems where the IDAC is used to generate a waveform of a defined sampling rate, by eliminating the effects of variable interrupt latency and instruction execution on the timing of the IDAC output. When the IDAnCM bits (IDAnCN.[6:4]) are set to ‘000’, ‘001’, ‘010’ or ‘011’, writes to both IDAC data registers (IDAnL and IDAnH) are held until an associated Timer overflow event (Timer 0, Timer 1, Timer 2 or Timer 3, respectively) occurs, at which time the IDAnH:IDAnL contents are copied to the IDAC input latch, allowing the IDAC output to change to the new value. When updates are scheduled based on Timer 2 or 3, updates occur on low-byte overflows if Timer 2 or 3 is in 8-bit mode and high-byte overflows if Timer 2 or 3 is in 16-bit mode.

### 6.1.3. Update Output Based on CNVSTR Edge

The IDAC output can also be configured to update on a rising edge, falling edge, or both edges of the external CNVSTR signal. When the IDAnCM bits (IDAnCN.[6:4]) are set to ‘100’, ‘101’, or ‘110’, writes to the IDAC data registers (IDAnL and IDAnH) are held until an edge occurs on the CNVSTR input pin. The particular setting of the IDAnCM bits determines whether the IDAC output is updated on rising, falling, or both edges of CNVSTR. When a corresponding edge occurs, the IDAnH:IDAnL contents are copied to the IDAC input latch, allowing the IDAC output to change to the new value.

## 6.2. IDAC Output Mapping

The IDAC data word can be Left Justified or Right Justified as shown in Figure 6.2. When Left Justified, the 8 MSBs of the data word (D11-D4) are mapped to bits 7-0 of the IDAnH register and the 4 LSBs of the data word (D3-D0) are mapped to bits 7-4 of the IDAnL register. When Right Justified, the 4 MSBs of the data word (D11-D8) are mapped to bits 3-0 of the IDAnH register and the 8 LSBs of the data word (D7-D0) are mapped to bits 7-0 of the IDAnL register. The IDAC data word justification is selected using the IDAnRJST bit (IDAnCN.2).

The full-scale output current of the IDAC is selected using the IDAnOMD bits (IDAnCN[1:0]). By default, the IDAC is set to a full-scale output current of 2 mA. The IDAnOMD bits can also be configured to provide full-scale output currents of 0.25 mA, 0.5 mA, or 1 mA.

Left Justified Data (IDAnRJST = 0):

| IDAnH | | | | | | | | IDAnL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | | | | |

Right Justified Data (IDAnRJST = 1):

| IDAnH | | | | | | | | IDAnL | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

| IDAn Data Word (D11–D0) | Output Current vs IDAnOMD bit setting | | | |
|---|---|---|---|---|
| | ‘11’ (2 mA) | ‘10’ (1 mA) | ‘01’ (0.5 mA) | ‘00’ (0.25 mA) |
| 0x000 | 0 mA | 0 mA | 0 mA | 0 mA |
| 0x001 | 1/4096 x 2 mA | 1/4096 x 1 mA | 1/4096 x 0.5 mA | 1/4096 x 0.25 mA |
| 0x800 | 2048/4096 x 2 mA | 2048/4096 x 1 mA | 2048/4096 x 0.5 mA | 2048/4096 x 0.25 mA |
| 0xFFF | 4095/4096 x 2 mA | 4095/4096 x 1 mA | 4095/4096 x 0.5 mA | 4095/4096 x 0.25 mA |

**Figure 6.2. IDAC Data Word Mapping**

## SFR Definition 6.1. IDA0CN: IDA0 Control

| R/W | R/W | R/W | R/W | R/W | R | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| IDA0EN | IDA0CM | | | - | IDA0RJST | IDA0OMD | | 01110011 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB9

Bit 7: IDA0EN: IDA0 Enable Bit.
0: IDA0 Disabled.
1: IDA0 Enabled.

Bits 6–4: IDA0CM[2:0]: IDA0 Update Source Select Bits.
000: DAC output updates on Timer 0 overflow.
001: DAC output updates on Timer 1 overflow.
010: DAC output updates on Timer 2 overflow.
011: DAC output updates on Timer 3 overflow.
100: DAC output updates on rising edge of CNVSTR.
101: DAC output updates on falling edge of CNVSTR.
110: DAC output updates on any edge of CNVSTR.
111: DAC output updates on write to IDA0H.

Bit 3: Reserved. Read = 0b, Write = 0b.

Bit 2: IDA0RJST: IDA0 Right Justify Select Bit.
0: IDA0 data in IDA0H:IDA0L is left justified.
1: IDA0 data in IDA0H:IDA0L is right justified.

Bits 1:0: IDA0OMD[1:0]: IDA0 Output Mode Select Bits.
00: 0.25 mA full-scale output current.
01: 0.5 mA full-scale output current.
10: 1.0 mA full-scale output current.
11: 2.0 mA full-scale output current.

## SFR Definition 6.2. IDA0H: IDA0 Data High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x97

Bits 7–0: IDA0 Data Word High-Order Bits.
For IDA0RJST = 0:
Bits 7-0 hold the most significant 8-bits of the 12-bit IDA0 Data Word.
For IDA0RJST = 1:
Bits 3-0 hold the most significant 4-bits of the 12-bit IDA0 Data Word. Bits 7-4 are 0000b.

## SFR Definition 6.3. IDA0L: IDA0 Data Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x96

Bits 7–0: IDA0 Data Word Low-Order Bits.
For IDA0RJST = 0:
Bits 7-4 hold the least significant 4-bits of the 12-bit IDA0 Data Word. Bits 3–0 are 0000b.
For IDA0RJST = 1:
Bits 7–0 hold the least significant 8-bits of the 12-bit IDA0 Data Word.

## SFR Definition 6.4. IDA1CN: IDA1 Control

| R/W | R/W | R/W | R/W | R/W | R | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| IDA1EN | IDA1CM | | | - | IDA1RJST | IDA1OMD | | 01110011 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB5

Bit 7: IDA1EN: IDA0 Enable Bit.
0: IDA1 Disabled.
1: IDA1 Enabled.

Bits 6–4: IDA1CM[2:0]: IDA1 Update Source Select Bits.
000: DAC output updates on Timer 0 overflow.
001: DAC output updates on Timer 1 overflow.
010: DAC output updates on Timer 2 overflow.
011: DAC output updates on Timer 3 overflow.
100: DAC output updates on rising edge of CNVSTR.
101: DAC output updates on falling edge of CNVSTR.
110: DAC output updates on any edge of CNVSTR.
111: DAC output updates on write to IDA1H.

Bit 3: Reserved. Read = 0b, Write = 0b.

Bit 2: IDA1RJST: IDA1 Right Justify Select Bit.
0: IDA1 data in IDA1H:IDA1L is left justified.
1: IDA1 data in IDA1H:IDA1L is right justified.

Bits 1–0: IDA1OMD[1:0]: IDA1 Output Mode Select Bits.
00: 0.25 mA full-scale output current.
01: 0.5 mA full-scale output current.
10: 1.0 mA full-scale output current.
11: 2.0 mA full-scale output current.

### SFR Definition 6.5. IDA1H: IDA0 Data High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF5

Bits 7–0: IDA1 Data Word High-Order Bits.
For IDA0RJST = 0:
Bits 7-0 hold the most significant 8-bits of the 12-bit IDA1 Data Word.
For IDA0RJST = 1:
Bits 3-0 hold the most significant 4-bits of the 12-bit IDA1 Data Word. Bits 7–4 are 0000b.

### SFR Definition 6.6. IDA1L: IDA1 Data Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF4

Bits 7–0: IDA1 Data Word Low-Order Bits.
For IDA0RJST = 0:
Bits 7-4 hold the least significant 4-bits of the 12-bit IDA1 Data Word. Bits 3–0 are 0000b.
For IDA0RJST = 1:
Bits 7–0 hold the least significant 8-bits of the 12-bit IDA1 Data Word.

## 6.3. IDAC External Pin Connections

The IDA0 output is connected to P0.0, and the IDA1 output can be connected to P0.0 or P0.1. The output pin for IDA1 is selected using IDAMRG (REF0CN.7). When the enable bits for both IDACs (IDAnEN) are set to ‘0’, the IDAC outputs behave as a normal GPIO pins. When either IDAC’s enable bit is set to ‘1’, the digital output drivers and weak pullup for the selected IDAC pin are automatically disabled, and the pin is connected to the IDAC output. When using the IDACs, the selected IDAC pin(s) should be skipped in the Crossbar by setting the corresponding PnSKIP bits to a ‘1’. Figure 6.3 shows the pin connections for IDA0 and IDA1.

When both IDACs are enabled and IDAMRG is set to logic 1, the output of both IDACs is merged onto P0.0.

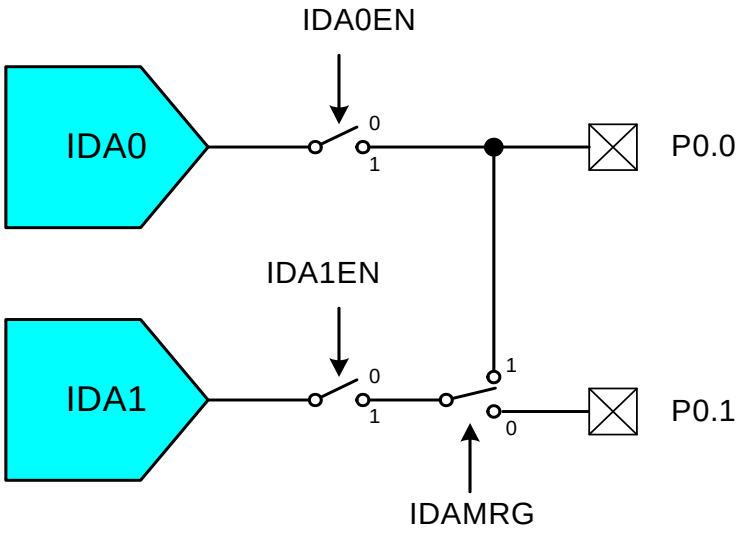



**Figure 6.3. IDAC Pin Connections**

## Table 6.1. IDAC Electrical Characteristics

–40 to +85 °C, $V_{DD}$ = 2.0 V Full-scale output current set to 2 mA unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Static Performance** | | | | | |
| Resolution | | | 12 | | bits |
| Integral Nonlinearity | | — | — | ±10 | LSB |
| Differential Nonlinearity | Guaranteed Monotonic | — | — | ±1 | LSB |
| Output Compliance Range | Guaranteed by Design, Applies to entire VDD range | — | — | $V_{DD}$ – 1.2 | V |
| Offset Error | | — | 0 | — | LSB |
| Gain Error | 2 mA Full Scale Output Current | — | 0.05 | 2 | % |
| Gain-Error Tempco | | — | 320 | — | nA/°C |
| $V_{DD}$ Power Supply Rejection Ratio | | — | 2 | — | µA/V |
| Output Capacitance | | — | 2 | — | pF |
| **Dynamic Performance** | | | | | |
| Startup Time | | — | 10 | — | µs |
| Gain Variation From 2 mA range | 1 mA Full Scale Output Current<br>0.5 mA Full Scale Output Current<br>0.25 mA Full Scale Output Current | — | 0.5<br>0.5<br>0.5 | — | %<br>%<br>% |
| **Power Consumption** | | | | | |
| Power Supply Current | 2 mA Full Scale Output Current<br>1 mA Full Scale Output Current<br>0.5 mA Full Scale Output Current<br>0.25 mA Full Scale Output Current | — | 2.1<br>1.1<br>0.6<br>0.35 | — | mA<br>mA<br>mA<br>mA |

## NOTES:

## 7. Voltage Reference

The Voltage reference MUX on C8051F41x devices is configurable to use an externally connected voltage reference, the internal reference voltage generator, or the $V_{DD}$ power supply voltage (see Figure 7.1). The REFSL bit in the Reference Control register (REF0CN) selects the reference source. For an external source or the internal reference, REFSL should be set to ‘0’. To use $V_{DD}$ as the reference source, REFSL should be set to ‘1’.

The internal voltage reference circuit consists of a temperature stable bandgap voltage reference generator and a gain-of-two output buffer amplifier. The output voltage is selected between 1.5 V and 2.2 V. The internal voltage reference can be driven out on the $V_{REF}$ pin by setting the REFBE bit in register REF0CN to a ‘1’ (see Figure 7.1). The load seen by the $V_{REF}$ pin must draw less than 200 µA to GND. When using the internal voltage reference, bypass capacitors of 0.1 µF and 4.7 µF are recommended from the $V_{REF}$ pin to GND. If the internal reference is not used, the REFBE bit should be cleared to ‘0’.

The BIASE bit enables the internal voltage bias generator, which is used by the ADC, Temperature Sensor, internal oscillators, and IDACs. This bit is forced to logic 1 when any of the aforementioned peripherals are enabled. The bias generator may be enabled manually by writing a ‘1’ to the BIASE bit in register REF0CN; see SFR Definition 7.1 for REF0CN register details.

The electrical specifications for the voltage reference circuit are given in Table 7.1.

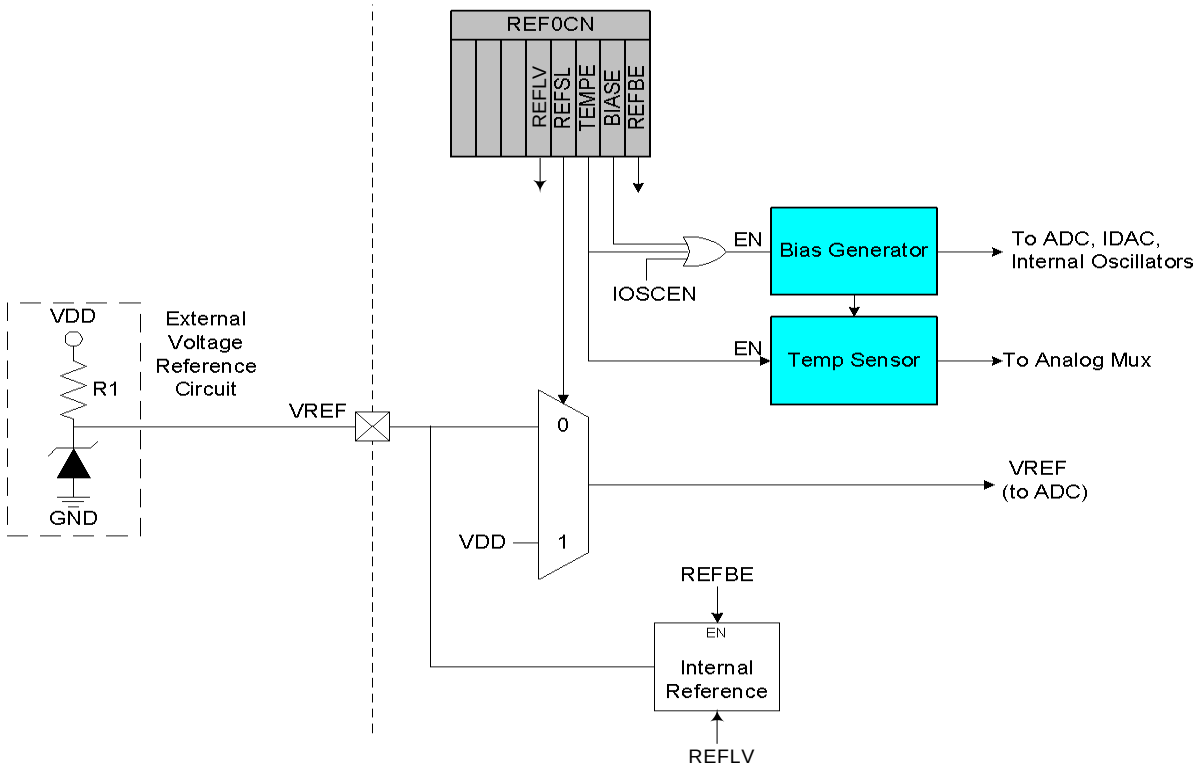



**Figure 7.1. Voltage Reference Functional Block Diagram**

**Important Note About the $V_{REF}$ Pin:** Port pin P1.2 is used as the external $V_{REF}$ input and as an output for the internal $V_{REF}$. When using either an external voltage reference or the internal reference circuitry, P1.2 should be configured as an analog pin, and skipped by the Digital Crossbar. To configure P1.2 as an analog pin, clear Bit 2 in register P1MDIN to ‘0’ and set Bit 2 in register P1 to '1'. To configure the Crossbar to skip P1.2, set Bit 2 in register P1SKIP to ‘1’. Refer to **Section “18. Port Input/Output” on page 147** for complete Port I/O configuration details. 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.

### SFR Definition 7.1. REF0CN: Reference Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| IDAMRG | GF | ZTCEN | REFLV | REFSL | TEMPE | BIASE | REFBE | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0xD1 |

Bit7: IDAMRG: IDAC Output Merge Select.
0: IDA1 Output is P0.1.
1: IDA1 Output is P0.0 (Merged with IDA0 Output).

Bit6: GF. General Purpose Flag.
This bit is a general purpose flag for use under software control.

Bit5: ZTCEN: Zero-TempCo Bias Enable Bit.
0: ZeroTC Bias Generator automatically enabled when needed.
1: ZeroTC Bias Generator forced on.

Bit4: REFLV: Voltage Reference Output Level Select.
This bit selects the output voltage level for the internal voltage reference.
0: Internal voltage reference set to 1.5 V.
1: Internal voltage reference set to 2.2 V.

Bit3: REFSL: Voltage Reference Select.
This bit selects the source for the internal voltage reference.
0: $V_{REF}$ pin used as voltage reference.
1: $V_{DD}$ used as voltage reference.

Bit2: TEMPE: Temperature Sensor Enable Bit.
0: Internal Temperature Sensor off.
1: Internal Temperature Sensor on.

Bit1: BIASE: Internal Analog Bias Generator Enable Bit.
0: Internal Analog Bias Generator automatically enabled when needed.
1: Internal Analog Bias Generator on.

Bit0: REFBE: Internal Reference Buffer Enable Bit.
0: Internal Reference Buffer disabled.
1: Internal Reference Buffer enabled. Internal voltage reference driven on the $V_{REF}$ pin.

## Table 7.1. Voltage Reference Electrical Characteristics

$V_{DD}$ = 2.0 V; –40 to +85 °C unless otherwise specified.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Internal Reference (REFBE = 1)** | | | | | |
| Output Voltage | 25 °C ambient (REFLV = 0)<br>25 °C ambient (REFLV = 1), $V_{DD}$ = 2.5 V | 1.47<br>2.16 | 1.5<br>2.2 | 1.53<br>2.24 | V |
| $V_{REF}$ Short-Circuit Current | | — | 3.0 | — | mA |
| $V_{REF}$ Temperature Coefficient | | — | 35 | — | ppm/°C |
| Load Regulation | Load = 0 to 200 µA to GND | — | 10 | — | ppm/µA |
| $V_{REF}$ Turn-on Time | $V_{DD}$ = 2.5 V, $V_{REF}$ = 1.5 V:<br>4.7 µF tantalum, 0.1 µF ceramic bypass<br>0.1 µF ceramic bypass<br>$V_{DD}$ = 2.5 V, $V_{REF}$ = 2.2 V:<br>4.7 µF tantalum, 0.1 µF ceramic bypass<br>0.1 µF ceramic bypass | <br>—<br>—<br><br>—<br>— | <br>2.5<br>55<br><br>6.8<br>144 | <br>—<br>—<br><br>—<br>— | <br>ms<br>µs<br><br>ms<br>µs |
| Power Supply Rejection | | — | 2 | — | mV/V |
| **External Reference (REFBE = 0)** | | | | | |
| Input Voltage Range | | 0 | — | $V_{DD}$ | V |
| Input Current | Sample Rate = 200 ksps; $V_{REF}$ = 2 V | — | 5 | — | µA |
| **Bias Generators** | | | | | |
| ADC Bias Generator | BIASE = ‘1’ | — | 22 | — | µA |
| Power Consumption (Internal) | | — | 50 | — | µA |

## NOTES:

## 8. Voltage Regulator (REG0)

C8051F41x devices include an on-chip low dropout voltage regulator (REG0). The input to REG0 at the $V_{REGIN}$ pin can be as high as 5.25 V. The output can be selected by software to 2.1 V or 2.5 V. When enabled, the output of REG0 appears on the $V_{DD}$ pin, powers the microcontroller core, and can be used to power external devices. On reset, REG0 is enabled and can be disabled by software.

The input ($V_{REGIN}$) and output ($V_{DD}$) of the voltage regulator should both be protected with a large capacitor (4.7 µF + 0.1 µF) to ground. This capacitor will eliminate power spikes and provide any immediate power required by the microcontroller. A settling time associated with the voltage regulator is shown in Table 8.1.

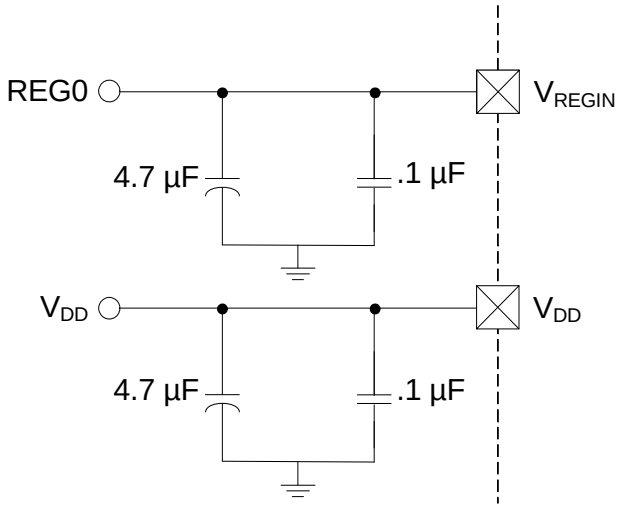



**Figure 8.1. External Capacitors for Voltage Regulator Input/Output**

If the internal voltage regulator is not used, the $V_{REGIN}$ input should be tied to $V_{DD}$, as shown in Figure 8.2.

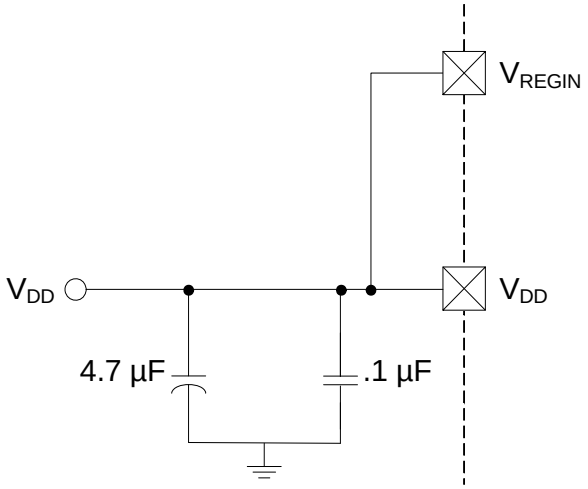



**Figure 8.2. External Capacitors for Voltage Regulator Input/Output**

## SFR Definition 8.1. REG0CN: Regulator Control

| R/W | R/W | R | R/W | R | R | R | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| REGDIS | Reserved | — | REG0MD | — | — | — | DROPOUT | 00010000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xC9

Bit 7: REGDIS: Voltage Regulator Disable Bit.
This bit disables/enables the Voltage Regulator.
0: Voltage Regulator Enabled.
1: Voltage Regulator Disabled.
Bit 6: RESERVED. Read = 0b. Must write 0b.
Bit 5: UNUSED. Read = 0b. Write = don't care.
Bit 4: REG0MD: Voltage Regulator Mode Select Bit.
This bit selects the Voltage Regulator output voltage.
0: Voltage Regulator output is 2.1 V.
1: Voltage Regulator output is 2.5 V (default).
Bits 3–1: UNUSED. Read = 0b. Write = don't care.
Bit 0: DROPOUT: Voltage Regulator Dropout Indicator Bit.
0: Voltage Regulator is not in dropout.
1: Voltage Regulator is in or near dropout.

## Table 8.1. Voltage Regulator Electrical Specifications

$V_{DD}$ = 2.1 or 2.5 V; –40 to +85 °C unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Input Voltage Range ($V_{REGIN}$)* | | (See Note) | — | 5.25 | V |
| Load Current | | — | — | 50 | mA |
| Load Regulation | | — | 7 | 15 | mV/mA |
| Output Voltage ($V_{DD}$) | Output Current = 1 mA<br>REG0MD = '0'<br>REG0MD = '1' | <br>2.0<br>2.35 | <br>2.1<br>2.5 | <br>2.25<br>2.55 | V |
| Bias Current | REG0MD = '0'<br>REG0MD = '1' | —<br>— | 1<br>1 | 1.5<br>1.5 | µA |
| Dropout Indicator Detection Threshold | | — | 65 | — | mV |
| Output Voltage Tempco | | — | 600 | — | µV/ºC |
| VREG Settling Time | 50 mA load with $V_{REGIN}$ = 2.5 V and $V_{DD}$ load capacitor of 4.8 µF | — | 250 | — | µs |

***Note:** Actual Output Voltage ($V_{DD}$) = Nominal Output Voltage ($V_{DD}$) – (Load Regulation x Load Current).

# 9. Comparators

C8051F41x devices include two on-chip programmable voltage comparators: Comparator0 is shown in Figure 9.1; Comparator1 is shown in Figure 9.2. The two comparators operate identically, but only Comparator0 can be used as a reset source.

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 with the device in STOP or SUSPEND mode. When assigned to a Port pin, the Comparator output may be configured as open drain or push-pull (see **Section “18.2. Port I/O Initialization” on page 151**). Comparator0 may also be used as a reset source (see **Section “15.5. Comparator0 Reset” on page 130**).

The Comparator0 inputs are selected in the CPT0MX register (SFR Definition 9.2). The CMX0P3-CMX0P0 bits select the Comparator0 positive input; the CMX0N3-CMX0N0 bits select the Comparator0 negative input. The Comparator1 inputs are selected in the CPT1MX register (SFR Definition 9.4). The CMX1P3-CMX1P0 bits select the Comparator1 positive input; the CMX1N3-CMX1N0 bits select the Comparator1 negative input.

**Important Note About Comparator Inputs:** The Port pins selected as Comparator inputs should be configured as analog inputs in their associated Port configuration register (with a ‘1’ written to the corresponding Port Latch register), and configured to be skipped by the Crossbar (for details on Port configuration, see **Section “18.3. General Purpose Port I/O” on page 154**)

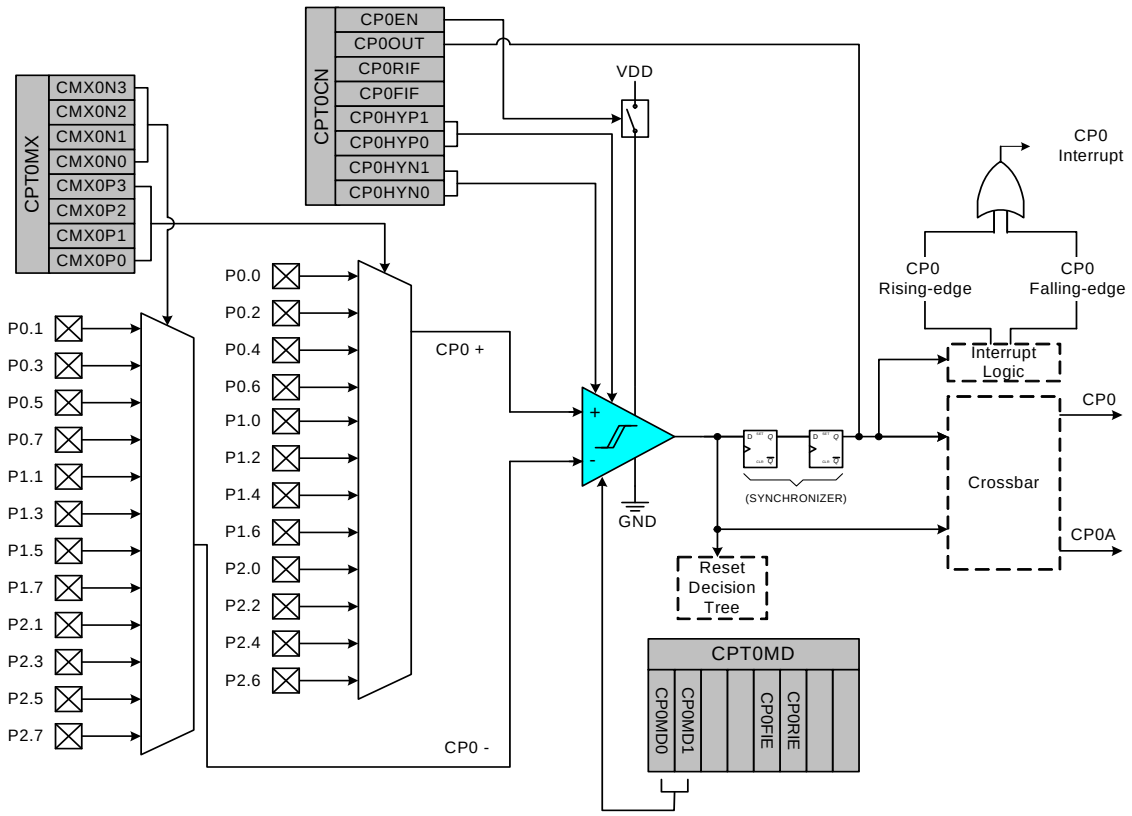



**Figure 9.1. Comparator0 Functional Block Diagram**

The Comparator output can be polled in software, used as an interrupt source, internal oscillator suspend awakening source and/or routed to a Port pin. When routed to a Port pin, the Comparator output is available asynchronous or synchronous to the system clock; the asynchronous output is available even in STOP or SUSPEND mode (with no system clock active). When disabled, the Comparator output (if assigned to a Port I/O pin via the Crossbar) defaults to the logic low state, and its supply current falls to less than 100 nA. See **Section "18.1. Priority Crossbar Decoder" on page 149** for details on configuring Comparator outputs via the digital Crossbar. Comparator inputs can be externally driven from -0.25 V to ($V_{DD}$) + 0.25 V without damage or upset. The complete Comparator electrical specifications are given in Table 9.1.

The Comparator response time may be configured in software via the CPTnMD registers (see SFR Definition 9.3 and SFR Definition 9.5). Selecting a longer response time reduces the Comparator supply current. See Table 9.1 for complete timing and current consumption specifications.

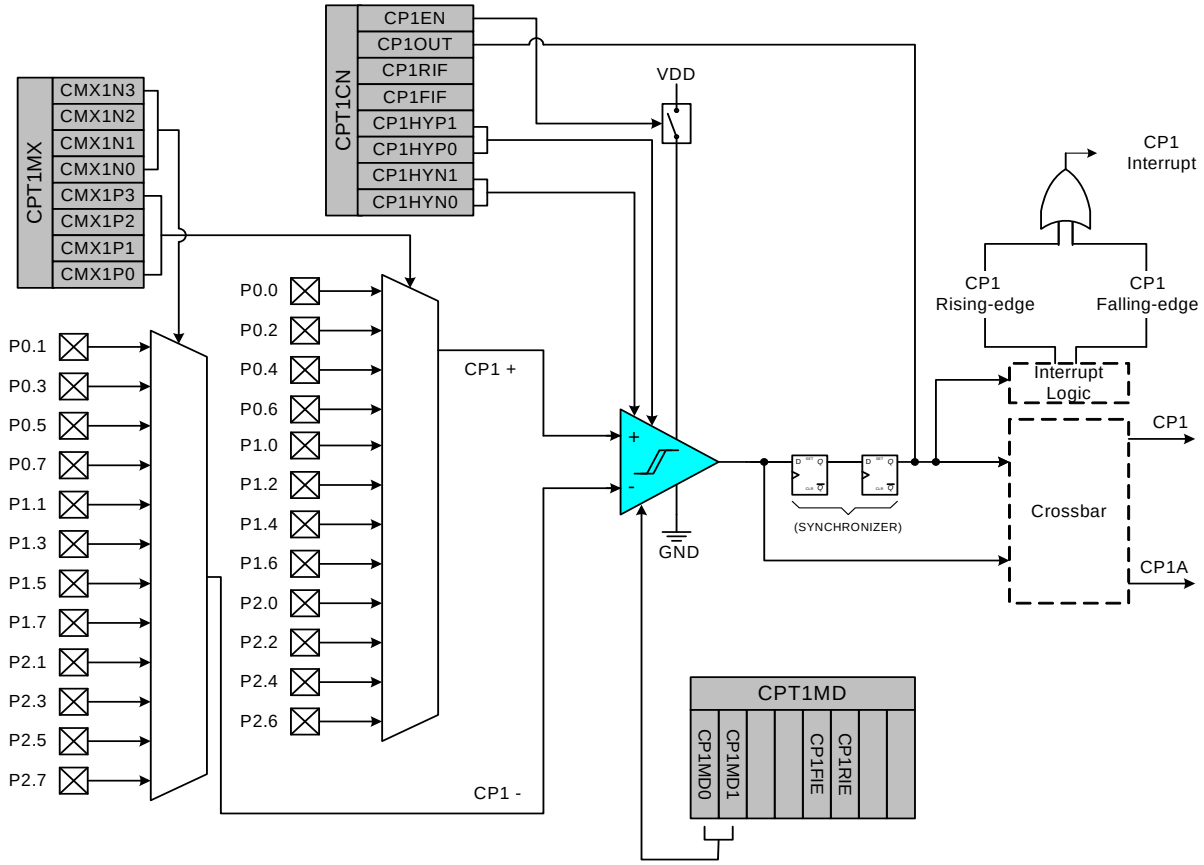



**Figure 9.2. Comparator1 Functional Block Diagram**

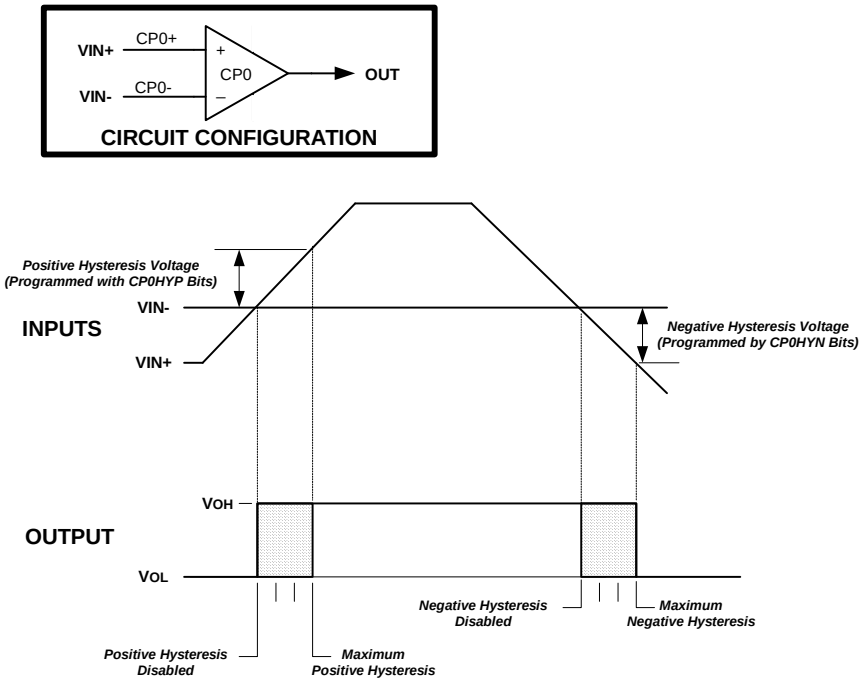



**Figure 9.3. Comparator Hysteresis Plot**

The Comparator hysteresis is software-programmable via its Comparator Control register CPTnCN (for n = 0 or 1). 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.

The Comparator hysteresis is programmed using Bits3-0 in the Comparator Control Register CPTnCN (shown in SFR Definition 9.1 and SFR Definition 9.6). The amount of negative hysteresis voltage is determined by the settings of the CPnHYN bits. As shown in Table 9.1, settings of 20, 10 or 5 mV of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by setting the CPnHYP bits.

Comparator interrupts can be generated on both rising-edge and falling-edge output transitions. (For Interrupt enable and priority control, see **Section "12. Interrupt Handler" on page 110**). The CPnFIF flag is set to logic 1 upon a Comparator falling-edge detect, and the CPnRIF flag is set to logic 1 upon the Comparator rising-edge detect. Once set, these bits remain set until cleared by software. The output state of the Comparator can be obtained at any time by reading the CPnOUT bit. The Comparator is enabled by setting the CPnEN bit to logic 1, and is disabled by clearing this bit to logic 0.

The output state of the Comparator can be obtained at any time by reading the CPnOUT bit. The Comparator is enabled by setting the CPnEN bit to logic 1, and is disabled by clearing this bit to logic 0. When the Comparator is enabled, the internal oscillator is awakened from SUSPEND mode if the Comparator output is logic 0.

Note that false rising edges and falling edges can be detected when the comparator is first powered-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. This Power Up Time is specified in Table 9.1 on page 92.

## SFR Definition 9.1. CPT0CN: Comparator0 Control

| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CP0EN | CP0OUT | CP0RIF | CP0FIF | CP0HYP1 | CP0HYP0 | CP0HYN1 | CP0HYN0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9B |

Bit7: CP0EN: Comparator0 Enable Bit.
0: Comparator0 Disabled.
1: Comparator0 Enabled.

Bit6: CP0OUT: Comparator0 Output State Flag.
0: Voltage on CP0+ < CP0–.
1: Voltage on CP0+ > CP0–.

Bit5: CP0RIF: Comparator0 Rising-Edge Flag.
0: No Comparator0 Rising Edge has occurred since this flag was last cleared.
1: Comparator0 Rising Edge has occurred.

Bit4: CP0FIF: Comparator0 Falling-Edge Flag.
0: No Comparator0 Falling-Edge has occurred since this flag was last cleared.
1: Comparator0 Falling-Edge has occurred.

Bits3–2: CP0HYP1–0: Comparator0 Positive Hysteresis Control Bits.
00: Positive Hysteresis Disabled.
01: Positive Hysteresis = 5 mV.
10: Positive Hysteresis = 10 mV.
11: Positive Hysteresis = 20 mV.

Bits1–0: CP0HYN1–0: Comparator0 Negative Hysteresis Control Bits.
00: Negative Hysteresis Disabled.
01: Negative Hysteresis = 5 mV.
10: Negative Hysteresis = 10 mV.
11: Negative Hysteresis = 20 mV.

## SFR Definition 9.2. CPT0MX: Comparator0 MUX Selection

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CMX0N3 | CMX0N2 | CMX0N1 | CMX0N0 | CMX0P3 | CMX0P2 | CMX0P1 | CMX0P0 | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9F |

Bits7–4: CMX0N3–CMX0N0: Comparator0 Negative Input MUX Select.
These bits select which Port pin is used as the Comparator0 negative input.

| CMX0N3 | CMX0N2 | CMX0N1 | CMX0N0 | Negative Input |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | P0.1 |
| 0 | 0 | 0 | 1 | P0.3 |
| 0 | 0 | 1 | 0 | P0.5 |
| 0 | 0 | 1 | 1 | P0.7 |
| 0 | 1 | 0 | 0 | P1.1 |
| 0 | 1 | 0 | 1 | P1.3 |
| 0 | 1 | 1 | 0 | P1.5 |
| 0 | 1 | 1 | 1 | P1.7 |
| 1 | 0 | 0 | 0 | P2.1 |
| 1 | 0 | 0 | 1 | P2.3* |
| 1 | 0 | 1 | 0 | P2.5* |
| 1 | 0 | 1 | 1 | P2.7 |
| 1 | 1 | x | x | Reserved |
| ***Note:** *Available only on the C8051F410/2.* | | | | |

Bits1–0: CMX0P3–CMX0P0: Comparator0 Positive Input MUX Select.
These bits select which Port pin is used as the Comparator0 positive input.

| CMX0P3 | CMX0P2 | CMX0P1 | CMX0P0 | Positive Input |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | P0.0 |
| 0 | 0 | 0 | 1 | P0.2 |
| 0 | 0 | 1 | 0 | P0.4 |
| 0 | 0 | 1 | 1 | P0.6 |
| 0 | 1 | 0 | 0 | P1.0 |
| 0 | 1 | 0 | 1 | P1.2 |
| 0 | 1 | 1 | 0 | P1.4 |
| 0 | 1 | 1 | 1 | P1.6 |
| 1 | 0 | 0 | 0 | P2.0 |
| 1 | 0 | 0 | 1 | P2.2 |
| 1 | 0 | 1 | 0 | P2.4* |
| 1 | 0 | 1 | 1 | P2.6* |
| 1 | 1 | x | x | Reserved |
| ***Note:** *Available only on the C8051F410/2.* | | | | |

## SFR Definition 9.3. CPT0MD: Comparator0 Mode Selection

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| RESERVED | - | CP0RIE | CP0FIE | - | - | CP0MD1 | CP0MD0 | 00000010 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9D |

Bit7: RESERVED. Read = 0b. Must Write 0b.
Bit6: UNUSED. Read = 0b. Write = don't care.
Bit5: CP0RIE: Comparator Rising-Edge Interrupt Enable.
0: Comparator rising-edge interrupt disabled.
1: Comparator rising-edge interrupt enabled.
Bit4: CP0FIE: Comparator Falling-Edge Interrupt Enable.
0: Comparator falling-edge interrupt disabled.
1: Comparator falling-edge interrupt enabled.
Bits3–2: UNUSED. Read = 00b. Write = don't care.
Bits1–0: CP0MD1–CP0MD0: Comparator0 Mode Select
These bits affect the response time and power consumption for Comparator0.

| Mode | CP0MD1 | CP0MD0 | Effect |
|---|---|---|---|
| 0 | 0 | 0 | Fastest Response Time |
| 1 | 0 | 1 | — |
| 2 | 1 | 0 | — |
| 3 | 1 | 1 | Lowest Power Consumption |

## SFR Definition 9.4. CPT1MX: Comparator1 MUX Selection

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CMX1N3 | CMX1N2 | CMX1N1 | CMX1N0 | CMX1P3 | CMX1P2 | CMX1P1 | CMX1P0 | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9E |

Bits7–4: CMX1N3–CMX1N0: Comparator1 Negative Input MUX Select.
These bits select which Port pin is used as the Comparator1 negative input.

| CMX1N3 | CMX1N2 | CMX1N1 | CMX1N0 | Negative Input |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | P0.1 |
| 0 | 0 | 0 | 1 | P0.3 |
| 0 | 0 | 1 | 0 | P0.5 |
| 0 | 0 | 1 | 1 | P0.7 |
| 0 | 1 | 0 | 0 | P1.1 |
| 0 | 1 | 0 | 1 | P1.3 |
| 0 | 1 | 1 | 0 | P1.5 |
| 0 | 1 | 1 | 1 | P1.7 |
| 1 | 0 | 0 | 0 | P2.1 |
| 1 | 0 | 0 | 1 | P2.3* |
| 1 | 0 | 1 | 0 | P2.5* |
| 1 | 0 | 1 | 1 | P2.7 |
| 1 | 1 | x | x | Reserved |
| ***Note:** *Available only on the C8051F410/2.* | | | | |

Bits3–0: CMX1P3–CMX1P0: Comparator1 Positive Input MUX Select.
These bits select which Port pin is used as the Comparator1 positive input.

| CMX1P3 | CMX1P2 | CMX1P1 | CMX1P0 | Positive Input |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | P0.0 |
| 0 | 0 | 0 | 1 | P0.2 |
| 0 | 0 | 1 | 0 | P0.4 |
| 0 | 0 | 1 | 1 | P0.6 |
| 0 | 1 | 0 | 0 | P1.0 |
| 0 | 1 | 0 | 1 | P1.2 |
| 0 | 1 | 1 | 0 | P1.4 |
| 0 | 1 | 1 | 1 | P1.6 |
| 1 | 0 | 0 | 0 | P2.0 |
| 1 | 0 | 0 | 1 | P2.2 |
| 1 | 0 | 1 | 0 | P2.4* |
| 1 | 0 | 1 | 1 | P2.6* |
| 1 | 1 | x | x | Reserved |
| ***Note:** *Available only on the C8051F410/2.* | | | | |

## SFR Definition 9.5. CPT1MD: Comparator1 Mode Selection

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| RESERVED | - | CP1RIE | CP1FIE | - | - | CP1MD1 | CP1MD0 | 00000010 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9C |

Bit7: RESERVED. Read = 0b. Must Write 0b.
Bit6: UNUSED. Read = 0b. Write = don't care.
Bit5: CP1RIE: Comparator Rising-Edge Interrupt Enable.
0: Comparator rising-edge interrupt disabled.
1: Comparator rising-edge interrupt enabled.
Bit4: CP1FIE: Comparator Falling-Edge Interrupt Enable.
0: Comparator falling-edge interrupt disabled.
1: Comparator falling-edge interrupt enabled.
Bits3–2: UNUSED. Read = 00b. Write = don't care.
Bits1–0: CP1MD1–CP1MD0: Comparator1 Mode Select.
These bits affect the response time and power consumption for Comparator1.

| Mode | CP1MD1 | CP1MD0 | Effect |
|---|---|---|---|
| 0 | 0 | 0 | Fastest Response Time |
| 1 | 0 | 1 | — |
| 2 | 1 | 0 | — |
| 3 | 1 | 1 | Lowest Power Consumption |

## SFR Definition 9.6. CPT1CN: Comparator1 Control

| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CP1EN | CP1OUT | CP1RIF | CP1FIF | CP1HYP1 | CP1HYP0 | CP1HYN1 | CP1HYN0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: 0x9A |

Bit7: CP1EN: Comparator1 Enable Bit.
0: Comparator1 Disabled.
1: Comparator1 Enabled.

Bit6: CP1OUT: Comparator1 Output State Flag.
0: Voltage on CP1+ < CP1-.
1: Voltage on CP1+ > CP1-.

Bit5: CP1RIF: Comparator1 Rising-Edge Flag.
0: No Comparator1 Rising Edge has occurred since this flag was last cleared.
1: Comparator1 Rising Edge has occurred.

Bit4: CP1FIF: Comparator1 Falling-Edge Flag.
0: No Comparator1 Falling-Edge has occurred since this flag was last cleared.
1: Comparator1 Falling-Edge has occurred.

Bits3–2: CP1HYP1–0: Comparator1 Positive Hysteresis Control Bits.
00: Positive Hysteresis Disabled.
01: Positive Hysteresis = 5 mV.
10: Positive Hysteresis = 10 mV.
11: Positive Hysteresis = 20 mV.

Bits1–0: CP1HYN1–0: Comparator1 Negative Hysteresis Control Bits.
00: Negative Hysteresis Disabled.
01: Negative Hysteresis = 5 mV.
10: Negative Hysteresis = 10 mV.
11: Negative Hysteresis = 20 mV.

## Table 9.1. Comparator Electrical Characteristics

$V_{DD}$ = 2.0 V, –40 to +85 °C unless otherwise noted. All specifications apply to both Comparator0 and Comparator1 unless otherwise noted. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Response Time: Mode 0, Vcm[1] = 1.5 V | CP0+ – CP0– = 100 mV | — | 120 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 160 | — | ns |
| Response Time: Mode 1, Vcm[1] = 1.5 V | CP0+ – CP0– = 100 mV | — | 200 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 340 | — | ns |
| Response Time: Mode 2, Vcm[1] = 1.5 V | CP0+ – CP0– = 100 mV | — | 360 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 720 | — | ns |
| Response Time: Mode 3, Vcm[1] = 1.5 V | CP0+ – CP0– = 100 mV | — | 2.2 | — | µs |
| | CP0+ – CP0– = –100 mV | — | 7.2 | — | µs |
| Common-Mode Rejection Ratio[2] | | — | 1.5 | 14 | mV/V |
| Positive Hysteresis 1 | CP0HYP1-0 = 00 | — | 0.5 | 2.0 | mV |
| Positive Hysteresis 2 | CP0HYP1-0 = 01 | 2 | 4.5 | 10 | mV |
| Positive Hysteresis 3 | CP0HYP1-0 = 10 | 5 | 9.0 | 20 | mV |
| Positive Hysteresis 4 | CP0HYP1-0 = 11 | 13 | 18.0 | 40 | mV |
| Negative Hysteresis 1 | CP0HYN1-0 = 00 | — | –0.5 | –2.0 | mV |
| Negative Hysteresis 2 | CP0HYN1-0 = 01 | –2 | –4.5 | –10 | mV |
| Negative Hysteresis 3 | CP0HYN1-0 = 10 | –5 | –9.0 | –20 | mV |
| Negative Hysteresis 4 | CP0HYN1-0 = 11 | –13 | –18.0 | –40 | mV |
| Inverting or Non-Inverting Input Voltage Range | | –0.25 | — | $V_{DD}$ + 0.25 | V |
| Input Capacitance | | — | 4 | — | pF |
| Input Bias Current | | — | 0.5 | — | nA |
| Input Offset Voltage | | –10 | — | 10 | mV |
| **Power Supply** | | | | | |
| Power Supply Rejection[2] | | — | 0.2 | 4 | mV/V |
| Power-up Time | | — | 2.3 | — | µs |
| Supply Current at DC | Mode 0 | — | 13 | 30 | µA |
| | Mode 1 | — | 6.0 | 20 | µA |
| | Mode 2 | — | 3.0 | 10 | µA |
| | Mode 3 | — | 1.0 | 5 | µA |

**Notes:**
1. Vcm is the common-mode voltage on CP0+ and CP0–.
2. Guaranteed by design and/or characterization.

## 10. 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 C8051F41x family has a superset of all the peripherals included with a standard 8051. See Section "**1. System Overview**" on page **19** for more information about the available peripherals. The CIP-51 includes on-chip debug hardware which 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 10.1 for a block diagram). The CIP-51 core includes the following features:

- Fully Compatible with MCS-51 Instruction Set
- 50 MIPS Peak Throughput
- 256 Bytes of Internal RAM
- Extended Interrupt Handler
- Reset Input
- Power Management Modes
- Integrated Debug Logic

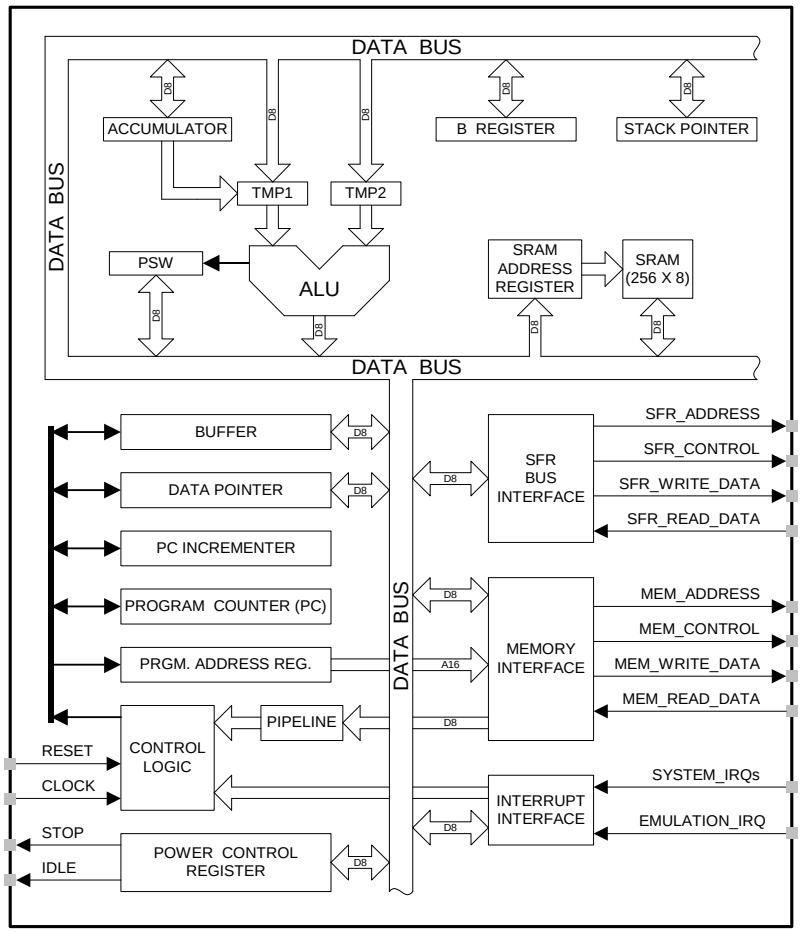



**Figure 10.1. CIP-51 Block Diagram**

**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.

With the CIP-51's system clock running at 50 MHz, it has a peak throughput of 50 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 | 2/4 | 3 | 3/5 | 4 | 5 | 4/6 | 6 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of Instructions | 26 | 50 | 5 | 10 | 7 | 5 | 2 | 1 | 2 | 1 |

**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 (C2) interface. Note that the re-programmable Flash can also be read and written a single byte at a time by the application software using the MOVC and MOVX instructions. This feature allows program memory to be used for non-volatile data storage as well as updating program code under software control.

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.

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

## 10.1. 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.

### 10.1.1. 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 two less clock cycles to complete when the branch is not taken as opposed to when the branch is taken. Table 10.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

### 10.1.2. MOVX Instruction and Program Memory

The MOVX instruction is typically used to access data stored in XDATA memory space. In the CIP-51, the MOVX instruction can also be used to write or erase on-chip program memory space implemented as re-programmable Flash memory. The Flash access feature provides a mechanism for the CIP-51 to update program code and use the program memory space for non-volatile data storage. Refer to **Section "16. Flash Memory" on page 135** for further details.

**Table 10.1. CIP-51 Instruction Set Summary[1]**

| 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 |

**Notes:**
1. Assumes PFEN = 1 for all instruction timing.
2. MOVC instructions take 4 to 7 clock cycles depending on instruction alignment and the FLRT setting (SFR Definition 16.3. FLSCL: Flash Scale).

**Table 10.1. CIP-51 Instruction Set Summary[1] (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| 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 |
| 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 | 4 to 7[2] |
| MOVC A, @A+PC | Move code byte relative PC to A | 1 | 4 to 7[2] |
| 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 |

**Notes:**
1. Assumes PFEN = 1 for all instruction timing.
2. MOVC instructions take 4 to 7 clock cycles depending on instruction alignment and the FLRT setting (SFR Definition 16.3. FLSCL: Flash Scale).

**Table 10.1. CIP-51 Instruction Set Summary[1] (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| 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 |
| 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 | 2/4 |
| JNC rel | Jump if Carry is not set | 2 | 2/4 |
| JB bit, rel | Jump if direct bit is set | 3 | 3/5 |
| JNB bit, rel | Jump if direct bit is not set | 3 | 3/5 |
| JBC bit, rel | Jump if direct bit is set and clear bit | 3 | 3/5 |
| **Program Branching** | | | |
| ACALL addr11 | Absolute subroutine call | 2 | 4 |
| LCALL addr16 | Long subroutine call | 3 | 5 |
| RET | Return from subroutine | 1 | 6 |
| RETI | Return from interrupt | 1 | 6 |
| AJMP addr11 | Absolute jump | 2 | 4 |
| LJMP addr16 | Long jump | 3 | 5 |
| SJMP rel | Short jump (relative address) | 2 | 4 |
| JMP @A+DPTR | Jump indirect relative to DPTR | 1 | 4 |
| JZ rel | Jump if A equals zero | 2 | 2/4 |
| JNZ rel | Jump if A does not equal zero | 2 | 2/4 |
| CJNE A, direct, rel | Compare direct byte to A and jump if not equal | 3 | 3/5 |
| CJNE A, #data, rel | Compare immediate to A and jump if not equal | 3 | 3/5 |
| CJNE Rn, #data, rel | Compare immediate to Register and jump if not equal | 3 | 3/5 |
| CJNE @Ri, #data, rel | Compare immediate to indirect and jump if not equal | 3 | 4/6 |
| DJNZ Rn, rel | Decrement Register and jump if not zero | 2 | 2/4 |
| DJNZ direct, rel | Decrement direct byte and jump if not zero | 3 | 3/5 |
| NOP | No operation | 1 | 1 |

**Notes:**
1. Assumes PFEN = 1 for all instruction timing.
2. MOVC instructions take 4 to 7 clock cycles depending on instruction alignment and the FLRT setting (SFR Definition 16.3. FLSCL: Flash Scale).

**Notes on Registers, Operands and Addressing Modes:**

**Rn** - Register R0-R7 of the currently selected register bank.

**@Ri** - Data RAM location addressed indirectly through R0 or R1.

**rel** - 8-bit, signed (two's complement) offset relative to the first byte of the following instruction. Used by SJMP and all conditional jumps.

**direct** - 8-bit internal data location's address. This could be a direct-access Data RAM location (0x00-0x7F) or an SFR (0x80-0xFF).

**#data** - 8-bit constant

**#data16** - 16-bit constant

**bit** - Direct-accessed bit in Data RAM or SFR

**addr11** - 11-bit destination address used by ACALL and AJMP. The destination must be within the same 2K-byte page of program memory as the first byte of the following instruction.

**addr16** - 16-bit destination address used by LCALL and LJMP. The destination may be anywhere within the 8K-byte program memory space.

There is one unused opcode (0xA5) that performs the same function as NOP.
All mnemonics copyrighted © Intel Corporation 1980.

## 10.2. 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 1. 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 datasheet associated with their corresponding system function.

### SFR Definition 10.1. SP: Stack Pointer

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x81

Bits7–0: SP: Stack Pointer.
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 10.2. DPL: Data Pointer Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x82

Bits7–0: DPL: Data Pointer Low.
The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

## SFR Definition 10.3. DPH: Data Pointer High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x83

Bits7–0: DPH: Data Pointer High.
The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

## SFR Definition 10.4. PSW: Program Status Word

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CY | AC | F0 | RS1 | RS0 | OV | F1 | PARITY | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xD0

Bit7: CY: Carry Flag.
This bit is set when the last arithmetic operation resulted in a carry (addition) or a borrow (subtraction). It is cleared to 0 by all other arithmetic operations.

Bit6: AC: Auxiliary Carry Flag
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 0 by all other arithmetic operations.

Bit5: F0: User Flag 0.
This is a bit-addressable, general purpose flag for use under software control.

Bits4–3: RS1–RS0: Register Bank Select.
These bits select which register bank is used during register accesses.

| RS1 | RS0 | Register Bank | Address |
|---|---|---|---|
| 0 | 0 | 0 | 0x00–0x07 |
| 0 | 1 | 1 | 0x08–0x0F |
| 1 | 0 | 2 | 0x10–0x17 |
| 1 | 1 | 3 | 0x18–0x1F |

Bit2: OV: Overflow Flag.
This bit is set to 1 under the following circumstances:
- An ADD, ADDC, or SUBB instruction causes a sign-change overflow.
- A MUL instruction results in an overflow (result is greater than 255).
- A DIV instruction causes a divide-by-zero condition.

The OV bit is cleared to 0 by the ADD, ADDC, SUBB, MUL, and DIV instructions in all other cases.

Bit1: F1: User Flag 1.
This is a bit-addressable, general purpose flag for use under software control.

Bit0: PARITY: Parity Flag.
This bit is set to 1 if the sum of the eight bits in the accumulator is odd and cleared if the sum is even.

### SFR Definition 10.5. ACC: Accumulator

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| ACC.7 | ACC.6 | ACC.5 | ACC.4 | ACC.3 | ACC.2 | ACC.1 | ACC.0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |
| | | | | | | | SFR Address: 0xE0 | |

Bits7–0: ACC: Accumulator.
This register is the accumulator for arithmetic operations.

### SFR Definition 10.6. B: B Register

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| B.7 | B.6 | B.5 | B.4 | B.3 | B.2 | B.1 | B.0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |
| | | | | | | | SFR Address: 0xF0 | |

Bits7–0: B: B Register.
This register serves as a second accumulator for certain arithmetic operations.

## 10.3. Power Management Modes

The CIP-51 core has two software programmable power management modes: Idle and Stop. Idle mode halts the CPU while leaving the peripherals and internal clocks active. In Stop mode, the CPU is halted, all interrupts and timers (except the Missing Clock Detector) are inactive, and the internal oscillator is stopped (analog peripherals remain in their selected states; the external oscillator is not affected). Since clocks are running in Idle mode, power consumption is dependent upon the system clock frequency and the number of peripherals left in active mode before entering Idle. Stop mode consumes the least power. SFR Definition 10.7 describes the Power Control Register (PCON) used to control the CIP-51's power management modes.

Although the CIP-51 has Idle and Stop modes built in (as with any standard 8051 architecture), power management of the entire MCU is better accomplished by enabling/disabling individual peripherals as needed. Each analog peripheral can be disabled when not in use and placed in low power mode. Digital peripherals, such as timers or serial buses, draw little power when they are not in use. Turning off the oscillators lowers power consumption considerably; however a reset is required to restart the MCU.

The C8051F41x devices feature a low-power SUSPEND mode, which stops the internal oscillator until a wakening event occurs. See Section “**19.1.1. Internal Oscillator Suspend Mode**” on page **166**.

### 10.3.1. 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.

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.

### 10.3.2. 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 internal oscillator, CPU, and all digital peripherals are stopped; the state of 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 period of 100 µs.

### 10.3.3. Suspend Mode

The C8051F41x devices feature a low-power SUSPEND mode, which stops the internal oscillator until a wakening event occurs. See **Section “19.1.1. Internal Oscillator Suspend Mode” on page 166**.

## SFR Definition 10.7. PCON: Power Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | STOP | IDLE | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x87

Bits7–2: Reserved.

Bit1: STOP: STOP Mode Select.
Writing a ‘1’ to this bit will place the CIP-51 into STOP mode. This bit will always read ‘0’.
1: CIP-51 forced into power-down mode. (Turns off internal oscillator).

Bit0: IDLE: IDLE Mode Select.
Writing a ‘1’ to this bit will place the CIP-51 into IDLE mode. This bit will always read ‘0’.
1: CIP-51 forced into IDLE mode. (Shuts off clock to CPU, but clock to Timers, Interrupts, and all peripherals remain active.)

# 11. Memory Organization and SFRs

The memory organization of the C8051F41x 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 map is shown in Figure 11.1.

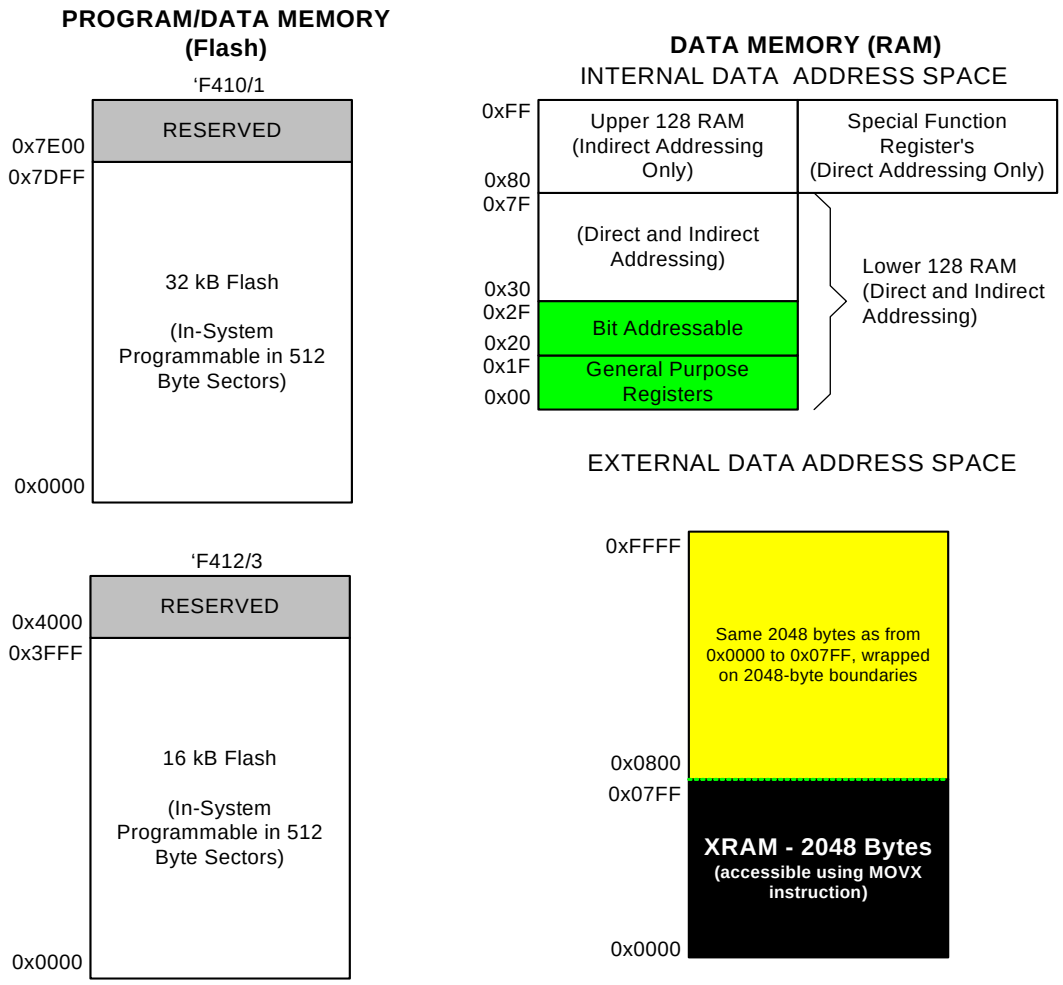



**Figure 11.1. Memory Map**

## 11.1. Program Memory

The CIP-51 core has a 64k-byte program memory space. The C8051F410/1 implement 32k of this program memory space as in-system, re-programmable Flash memory, organized in a contiguous block from addresses 0x0000 to 0x7DFF. Addresses above 0x7DFF are reserved on the 32 kB devices. The C8051F412/3 implement 16 kB of Flash from addresses 0x0000 to 0x3FFF.

Program memory is normally assumed to be read-only. However, the C8051F41x can write to program memory by setting the Program Store Write Enable bit (PSCTL.0) and using the MOVX write instruction. This feature provides a mechanism for updates to program code and use of the program memory space for non-volatile data storage. Refer to **Section "16. Flash Memory" on page 135** for further details.

## 11.2. Data Memory

The C8051F41x includes 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 (SFRs) 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 11.1 illustrates the data memory organization of the C8051F41x.

The C8051F41x family also includes 2048 bytes of on-chip RAM mapped into the external memory (XDATA) space. This RAM can be accessed using the CIP-51 core's MOVX instruction. More information on the XRAM memory can be found in **Section "17. External RAM" on page 145**.

## 11.3. 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 10.4. PSW: Program Status Word). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

## 11.4. 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 bit 7 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.

## 11.5. 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, 0x81) 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.

## 11.6. 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 CIP-51's resources and peripherals. The CIP-51 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 MCU. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 11.1 lists the SFRs implemented in the CIP-51 System Controller.

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, 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 11.2, for a detailed description of each register.

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

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | SPI0CN | PCA0L | PCA0H | PCA0CPL0 | PCA0CPH0 | PCA0CPL4 | PCA0CPH4 | VDM0CN |
| F0 | B | P0MDIN | P1MDIN | P2MDIN | IDA1L | IDA1H | EIP1 | EIP2 |
| E8 | ADC0CN | PCA0CPL1 | PCA0CPH1 | PCA0CPL2 | PCA0CPH2 | PCA0CPL3 | PCA0CPH3 | RSTSRC |
| E0 | ACC | XBR0 | XBR1 | PFE0CN | IT01CF | | EIE1 | EIE2 |
| D8 | PCA0CN | PCA0MD | PCA0CPM0 | PCA0CPM1 | PCA0CPM2 | PCA0CPM3 | PCA0CPM4 | CRC0FLIP |
| 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 | IDA0CN | ADC0TK | ADC0MX | ADC0CF | ADC0L | ADC0H | P1MASK |
| B0 | P0ODEN | OSCXCN | OSCICN | OSCICL | | IDA1CN | FLSCL | FLKEY |
| A8 | IE | CLKSEL | EMI0CN | CLKMUL | RTC0ADR | RTC0DAT | RTC0KEY | ONESHOT |
| A0 | P2 | SPI0CFG | SPI0CKR | SPI0DAT | P0MDOUT | P1MDOUT | P2MDOUT | |
| 98 | SCON0 | SBUF0 | CPT1CN | CPT0CN | CPT1MD | CPT0MD | CPT1MX | CPT0MX |
| 90 | P1 | TMR3CN | TMR3RLL | TMR3RLH | TMR3L | TMR3H | IDA0L | IDA0H |
| 88 | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | CKCON | PSCTL |
| 80 | P0 | SP | DPL | DPH | CRC0CN | CRC0IN | CRC0DAT | PCON |
| | 0(8) | 1(9) | 2(A) | 3(B) | 4(C) | 5(D) | 6(E) | 7(F) |

(bit addressable)

## Table 11.2. Special Function Registers

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

| Register | Address | Description | Page |
|---|---|---|---|
| ACC | 0xE0 | Accumulator | 101 |
| ADC0CF | 0xBC | ADC0 Configuration | 60 |
| ADC0CN | 0xE8 | ADC0 Control | 62 |
| ADC0H | 0xBE | ADC0 | 61 |
| ADC0L | 0xBD | ADC0 | 61 |
| ADC0GTH | 0xC4 | ADC0 Greater-Than Data High Byte | 64 |
| ADC0GTL | 0xC3 | ADC0 Greater-Than Data Low Byte | 64 |
| ADC0LTH | 0xC6 | ADC0 Less-Than Data High Byte | 65 |
| ADC0LTL | 0xC5 | ADC0 Less-Than Data Low Byte | 65 |
| ADC0MX | 0xC6 | ADC0 Channel Select | 59 |
| ADC0TK | 0xBA | ADC0 Tracking Mode Select | 63 |
| B | 0xF0 | B Register | 101 |
| CKCON | 0x8E | Clock Control | 237 |
| CLKMUL | 0xAB | Clock Multiplier | 173 |
| CLKSEL | 0xA9 | Clock Select | 174 |
| CPT0CN | 0x9B | Comparator0 Control | 86 |
| CPT0MD | 0x9D | Comparator0 Mode Selection | 88 |
| CPT0MX | 0x9F | Comparator0 MUX Selection | 87 |
| CPT1CN | 0x9A | Comparator1 Control | 91 |
| CPT1MD | 0x9C | Comparator1 Mode Selection | 90 |
| CPT1MX | 0x9E | Comparator1 MUX Selection | 89 |
| CRC0CN | 0x84 | CRC0 Control | 125 |
| CRC0IN | 0x85 | CRC0 Data Input | 125 |
| CRC0DAT | 0x86 | CRC0 Data Output | 126 |
| CRC0FLIP | 0xDF | CRC0 Bit Flip | 126 |
| DPH | 0x83 | Data Pointer High | 99 |
| DPL | 0x82 | Data Pointer Low | 99 |
| EIE1 | 0xE6 | Extended Interrupt Enable 1 | 114 |
| EIE2 | 0xE7 | Extended Interrupt Enable 2 | 116 |
| EIP1 | 0xF6 | Extended Interrupt Priority 1 | 115 |
| EIP2 | 0xF7 | Extended Interrupt Priority 2 | 116 |
| EMI0CN | 0xAA | External Memory Interface Control | 145 |
| FLKEY | 0xB7 | Flash Lock and Key | 141 |
| FLSCL | 0xB6 | Flash Scale | 142 |
| IDA0H | 0x97 | Current Mode DAC0 High Byte | 71 |
| IDA0L | 0x96 | Current Mode DAC0 Low Byte | 72 |
| IDA0CN | 0xB9 | Current Mode DAC0 Control | 71 |
| IDA1H | 0xF5 | Current Mode DAC1 High Byte | 73 |

## Table 11.2. Special Function Registers (Continued)

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

| Register | Address | Description | Page |
|---|---|---|---|
| IDA1L | 0xF4 | Current Mode DAC1 Low Byte | 73 |
| IDA1CN | 0xB5 | Current Mode DAC1 Control | 72 |
| IE | 0xA8 | Interrupt Enable | 112 |
| IP | 0xB8 | Interrupt Priority | 113 |
| IT01CF | 0xE4 | INT0/INT1 Configuration | 118 |
| ONESHOT | 0xAF | Flash Oneshot Period | 143 |
| OSCICL | 0xB3 | Internal Oscillator Calibration | 167 |
| OSCICN | 0xB2 | Internal Oscillator Control | 167 |
| OSCXCN | 0xB1 | External Oscillator Control | 171 |
| P0 | 0x80 | Port 0 Latch | 155 |
| P0MASK | 0xC7 | Port 0 Mask | 157 |
| P0MAT | 0xD7 | Port 0 Match | 157 |
| P0MDIN | 0xF1 | Port 0 Input Mode Configuration | 155 |
| P0MDOUT | 0xA4 | Port 0 Output Mode Configuration | 156 |
| P0ODEN | 0xB0 | Port 0 Overdrive | 157 |
| P0SKIP | 0xD4 | Port 0 Skip | 156 |
| P1 | 0x90 | Port 1 Latch | 158 |
| P1MASK | 0xBF | Port 1 Mask | 160 |
| P1MAT | 0xCF | Port 1 Match | 160 |
| P1MDIN | 0xF2 | Port 1 Input Mode Configuration | 158 |
| P1MDOUT | 0xA5 | Port 1 Output Mode Configuration | 159 |
| P1SKIP | 0xD5 | Port 1 Skip | 159 |
| P2 | 0xA0 | Port 2 Latch | 161 |
| P2MDIN | 0xF3 | Port 2 Input Mode Configuration | 161 |
| P2MDOUT | 0xA6 | Port 2 Output Mode Configuration | 162 |
| P2SKIP | 0xD6 | Port 2 Skip | 162 |
| PCA0CN | 0xD8 | PCA Control | 261 |
| PCA0CPH0 | 0xFC | PCA Capture 0 High | 264 |
| PCA0CPH1 | 0xEA | PCA Capture 1 High | 264 |
| PCA0CPH2 | 0xEC | PCA Capture 2 High | 264 |
| PCA0CPH3 | 0xEE | PCA Capture 3 High | 264 |
| PCA0CPH4 | 0xFE | PCA Capture 4 High | 264 |
| PCA0CPH5 | 0xD3 | PCA Capture 5 High | 264 |
| PCA0CPL0 | 0xFB | PCA Capture 0 Low | 264 |
| PCA0CPL1 | 0xE9 | PCA Capture 1 Low | 264 |
| PCA0CPL2 | 0xEB | PCA Capture 2 Low | 264 |
| PCA0CPL3 | 0xED | PCA Capture 3 Low | 264 |
| PCA0CPL4 | 0xFD | PCA Capture 4 Low | 264 |

**Table 11.2. Special Function Registers (Continued)**

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

| Register | Address | Description | Page |
|---|---|---|---|
| PCA0CPL5 | 0xD2 | PCA Capture 5 Low | 264 |
| PCA0CPM0 | 0xDA | PCA Module 0 Mode | 263 |
| PCA0CPM1 | 0xDB | PCA Module 1 Mode | 263 |
| PCA0CPM2 | 0xDC | PCA Module 2 Mode | 263 |
| PCA0CPM3 | 0xDD | PCA Module 3 Mode | 263 |
| PCA0CPM4 | 0xDE | PCA Module 4 Mode | 263 |
| PCA0CPM5 | 0xCE | PCA Module 5 Mode | 263 |
| PCA0H | 0xFA | PCA Counter High | 264 |
| PCA0L | 0xF9 | PCA Counter Low | 264 |
| PCA0MD | 0xD9 | PCA Mode | 262 |
| PCON | 0x87 | Power Control | 102 |
| PFE0CN | 0xE3 | Prefetch Engine Control | 119 |
| PSCTL | 0x8F | Program Store R/W Control | 141 |
| PSW | 0xD0 | Program Status Word | 100 |
| REF0CN | 0xD1 | Voltage Reference Control | 78 |
| REG0CN | 0xC9 | Voltage Regulator Control | 82 |
| RTC0ADR | 0xAC | smaRTClock Address | 181 |
| RTC0DAT | 0xAD | smaRTClock Data | 182 |
| RTC0KEY | 0xAE | smaRTClock Lock and Key | 180 |
| RSTSRC | 0xEF | Reset Source Configuration/Status | 133 |
| SBUF0 | 0x99 | UART0 Data Buffer | 213 |
| SCON0 | 0x98 | UART0 Control | 212 |
| SMB0CF | 0xC1 | SMBus Configuration | 197 |
| SMB0CN | 0xC0 | SMBus Control | 199 |
| SMB0DAT | 0xC2 | SMBus Data | 201 |
| SP | 0x81 | Stack Pointer | 98 |
| SPI0CFG | 0xA1 | SPI Configuration | 223 |
| SPI0CKR | 0xA2 | SPI Clock Rate Control | 225 |
| SPI0CN | 0xF8 | SPI Control | 224 |
| SPI0DAT | 0xA3 | SPI Data | 226 |
| TCON | 0x88 | Timer/Counter Control | 235 |
| TH0 | 0x8C | Timer/Counter 0 High | 238 |
| TH1 | 0x8D | Timer/Counter 1 High | 238 |
| TL0 | 0x8A | Timer/Counter 0 Low | 238 |
| TL1 | 0x8B | Timer/Counter 1 Low | 238 |
| TMOD | 0x89 | Timer/Counter Mode | 236 |
| TMR2CN | 0xC8 | Timer/Counter 2 Control | 242 |
| TMR2H | 0xCD | Timer/Counter 2 High | 243 |

**Table 11.2. Special Function Registers (Continued)**

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

| Register | Address | Description | Page |
|---|---|---|---|
| TMR2L | 0xCC | Timer/Counter 2 Low | 243 |
| TMR2RLH | 0xCB | Timer/Counter 2 Reload High | 243 |
| TMR2RLL | 0xCA | Timer/Counter 2 Reload Low | 243 |
| TMR3CN | 0x91 | Timer/Counter 3Control | 247 |
| TMR3H | 0x95 | Timer/Counter 3 High | 248 |
| TMR3L | 0x94 | Timer/Counter 3 Low | 248 |
| TMR3RLH | 0x93 | Timer/Counter 3 Reload High | 248 |
| TMR3RLL | 0x92 | Timer/Counter 3 Reload Low | 248 |
| VDM0CN | 0xFF | $V_{DD}$ Monitor Control | 130 |
| XBR0 | 0xE1 | Port I/O Crossbar Control 0 | 153 |
| XBR1 | 0xE2 | Port I/O Crossbar Control 1 | 154 |

# 12. Interrupt Handler

The C8051F41x family includes an extended interrupt system supporting a total of 18 interrupt sources with 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. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

If interrupts are enabled for the source, an 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.)

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.

Some interrupt-pending flags are automatically cleared by the 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. MCU Interrupt Sources and Vectors

The MCUs support 18 interrupt sources. 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. MCU interrupt sources, associated vector addresses, priority order, and control bits are summarized in Table 12.1 on page 111. 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).

## 12.2. 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. Each interrupt has an associated interrupt priority bit in an SFR (IP or EIP1) 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, given in Table 12.1.

## 12.3. 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 (/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 (/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 | 0x0043 | 8 | ALRM (RTC0CN.2)<br>OSCFAIL (RTC0CN.5) | N | N | ERTC0 (EIE1.1) | PRTC0 (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) |
| Voltage Regulator Dropout | 0x007B | 15 | N/A | N/A | N/A | EREG0 (EIE2.0) | PREG0 (EIP2.0) |
| Port Match | 0x0083 | 16 | N/A | N/A | N/A | EMAT (EIE2.1) | PMAT (EIP2.1) |

## 12.4. Interrupt Register Descriptions

The SFRs used to enable the interrupt sources and set their priority level are described below. 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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| EA | ESPI0 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xA8

Bit 7: EA: Global Interrupt Enable.
This bit globally enables/disables all interrupts. It overrides the individual interrupt mask settings.
0: Disable all interrupt sources.
1: Enable each interrupt according to its individual mask setting.

Bit 6: ESPI0: Enable Serial Peripheral Interface (SPI0) Interrupt.
This bit sets the masking of the SPI0 interrupts.
0: Disable all SPI0 interrupts.
1: Enable interrupt requests generated by SPI0.

Bit 5: ET2: Enable Timer 2 Interrupt.
This bit sets the masking of the Timer 2 interrupt.
0: Disable Timer 2 interrupt.
1: Enable interrupt requests generated by the TF2L or TF2H flags.

Bit 4: ES0: Enable UART0 Interrupt.
This bit sets the masking of the UART0 interrupt.
0: Disable UART0 interrupt.
1: Enable UART0 interrupt.

Bit 3: ET1: Enable Timer 1 Interrupt.
This bit sets the masking of the Timer 1 interrupt.
0: Disable all Timer 1 interrupt.
1: Enable interrupt requests generated by the TF1 flag.

Bit 2: EX1: Enable External Interrupt 1.
This bit sets the masking of External Interrupt 1.
0: Disable external interrupt 1.
1: Enable interrupt requests generated by the /INT1 input.

Bit 1: ET0: Enable Timer 0 Interrupt.
This bit sets the masking of the Timer 0 interrupt.
0: Disable all Timer 0 interrupt.
1: Enable interrupt requests generated by the TF0 flag.

Bit 0: EX0: Enable External Interrupt 0.
This bit sets the masking of External Interrupt 0.
0: Disable external interrupt 0.
1: Enable interrupt requests generated by the /INT0 input.

## SFR Definition 12.2. IP: Interrupt Priority

| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | PSPI0 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 | 10000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xB8

Bit 7: UNUSED. Read = 1, Write = don't care.

Bit 6: PSPI0: Serial Peripheral Interface (SPI0) Interrupt Priority Control.
This bit sets the priority of the SPI0 interrupt.
0: SPI0 interrupt set to low priority level.
1: SPI0 interrupt set to high priority level.

Bit 5: PT2: Timer 2 Interrupt Priority Control.
This bit sets the priority of the Timer 2 interrupt.
0: Timer 2 interrupt set to low priority level.
1: Timer 2 interrupt set to high priority level.

Bit 4: PS0: UART0 Interrupt Priority Control.
This bit sets the priority of the UART0 interrupt.
0: UART0 interrupt set to low priority level.
1: UART0 interrupt set to high priority level.

Bit 3: PT1: Timer 1 Interrupt Priority Control.
This bit sets the priority of the Timer 1 interrupt.
0: Timer 1 interrupt set to low priority level.
1: Timer 1 interrupt set to high priority level.

Bit 2: PX1: External Interrupt 1 Priority Control.
This bit sets the priority of the External Interrupt 1 interrupt.
0: External Interrupt 1 set to low priority level.
1: External Interrupt 1 set to high priority level.

Bit 1: PT0: Timer 0 Interrupt Priority Control.
This bit sets the priority of the Timer 0 interrupt.
0: Timer 0 interrupt set to low priority level.
1: Timer 0 interrupt set to high priority level.

Bit 0: PX0: External Interrupt 0 Priority Control.
This bit sets the priority of the External Interrupt 0 interrupt.
0: External Interrupt 0 set to low priority level.
1: External Interrupt 0 set to high priority level.

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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| ET3 | ECP1 | ECP0 | EPCA0 | EADC0 | EWADC0 | ERTC0 | ESMB0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE6

Bit 7: ET3: Enable Timer 3 Interrupt.
This bit sets the masking of the Timer 3 interrupt.
0: Disable Timer 3 interrupts.
1: Enable interrupt requests generated by the TF3L or TF3H flags.

Bit 6: ECP1: Enable Comparator1 (CP1) Interrupt.
This bit sets the masking of the CP1 interrupt.
0: Disable CP1 interrupts.
1: Enable interrupt requests generated by the CP1RIF or CP1FIF flags.

Bit 5: ECP0: Enable Comparator0 (CP0) Interrupt.
This bit sets the masking of the CP0 interrupt.
0: Disable CP0 interrupts.
1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags.

Bit 4: EPCA0: Enable Programmable Counter Array (PCA0) Interrupt.
This bit sets the masking of the PCA0 interrupts.
0: Disable all PCA0 interrupts.
1: Enable interrupt requests generated by PCA0.

Bit 3: EADC0: Enable ADC0 Conversion Complete Interrupt.
This bit sets the masking of the ADC0 Conversion Complete interrupt.
0: Disable ADC0 Conversion Complete interrupt.
1: Enable interrupt requests generated by the AD0INT flag.

Bit 2: EWADC0: Enable ADC0 Window Comparison Interrupt.
This bit sets the masking of the ADC0 Window Comparison interrupt.
0: Disable ADC0 Window Comparison interrupt.
1: Enable interrupt requests generated by the AD0WINT flag.

Bit 1: ERTC0: Enable smaRTClock Interrupt.
This bit sets the masking of the smaRTClock interrupt.
0: Disable smaRTClock interrupts.
1: Enable interrupt requests generated by the ALRM and OSCFAIL flag.

Bit 0: ESMB0: Enable SMBus (SMB0) Interrupt.
This bit sets the masking of the SMB0 interrupt.
0: Disable all SMB0 interrupts.
1: Enable interrupt requests generated by SMB0.

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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| PT3 | PCP1 | PCP0 | PPCA0 | PADC0 | PWADC0 | PRTC0 | PSMB0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF6

Bit 7: PT3: Timer 3 Interrupt Priority Control.
This bit sets the priority of the Timer 3 interrupt.
0: Timer 3 interrupts set to low priority level.
1: Timer 3 interrupts set to high priority level.

Bit 6: PCP1: Comparator1 (CP1) Interrupt Priority Control.
This bit sets the priority of the CP1 interrupt.
0: CP1 interrupt set to low priority level.
1: CP1 interrupt set to high priority level.

Bit 5: PCP0: Comparator0 (CP0) Interrupt Priority Control.
This bit sets the priority of the CP0 interrupt.
0: CP0 interrupt set to low priority level.
1: CP0 interrupt set to high priority level.

Bit 4: PPCA0: Programmable Counter Array (PCA0) Interrupt Priority Control.
This bit sets the priority of the PCA0 interrupt.
0: PCA0 interrupt set to low priority level.
1: PCA0 interrupt set to high priority level.

Bit 3: PADC0: ADC0 Conversion Complete Interrupt Priority Control.
This bit sets the priority of the ADC0 Conversion Complete interrupt.
0: ADC0 Conversion Complete interrupt set to low priority level.
1: ADC0 Conversion Complete interrupt set to high priority level.

Bit 2: PWADC0: ADC0 Window Comparison Interrupt Priority Control.
This bit sets the priority of the ADC0 Window Comparison interrupt.
0: ADC0 Window Comparison interrupt set to low priority level.
1: ADC0 Window Comparison interrupt set to high priority level.

Bit 1: PRTC0: smaRTClock Interrupt Priority Control.
This bit sets the priority of the smaRTClock interrupt.
0: smaRTClock interrupt set to low priority level.
1: smaRTClock interrupt set to high priority level.

Bit 0: PSMB0: SMBus (SMB0) Interrupt Priority Control.
This bit sets the priority of the SMB0 interrupt.
0: SMB0 interrupt set to low priority level.
1: SMB0 interrupt set to high priority level.

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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | EMAT | EREG0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE7

Bits 7–2: UNUSED. Read = 000000b. Write = don't care.
Bit 1: EMAT: Enable Port Match Interrupt.
This bit sets the masking of the Port Match interrupt.
0: Disable the Port Match interrupt.
1: Enable the Port Match interrupt.
Bit 0: EREG0: Enable Voltage Regulator Interrupt.
This bit sets the masking of the Voltage Regulator Dropout interrupt.
0: Disable the Voltage Regulator Dropout interrupt.
1: Enable the Voltage Regulator Dropout interrupt.

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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | PMAT | PREG0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF7

Bits 7–2: UNUSED. Read = 000000b. Write = don't care.
Bit 1: EMAT: Port Match Interrupt Priority Control.
This bit sets the priority of the Port Match interrupt.
0: Port Match interrupt set to low priority level.
1: Port Match interrupt set to high priority level.
Bit 0: PREG0: Voltage Regulator Interrupt Priority Control.
This bit sets the priority of the Voltage Regulator interrupt.
0: Voltage Regulator interrupt set to low priority level.
1: Voltage Regulator interrupt set to high priority level.

## 12.5. External Interrupts

The /INT0 and /INT1 external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL (/INT0 Polarity) and IN1PL (/INT1 Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (**Section "24.1. Timer 0 and Timer 1" on page 231**) select level or edge sensitive. The table below lists the possible configurations.

| IT0 | IN0PL | /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 | /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 |

/INT0 and /INT1 are assigned to Port pins as defined in the IT01CF register (see SFR Definition 12.7). Note that /INT0 and /INT0 Port pin assignments are independent of any Crossbar assignments. /INT0 and /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 /INT0 and/or /INT1, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see **Section "18.1. Priority Crossbar Decoder" on page 149** for complete details on configuring the Crossbar).

IE0 (TCON.1) and IE1 (TCON.3) serve as the interrupt-pending flags for the /INT0 and /INT1 external interrupts, respectively. If an /INT0 or /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: INT0/INT1 Configuration

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| IN1PL | IN1SL2 | IN1SL1 | IN1SL0 | IN0PL | IN0SL2 | IN0SL1 | IN0SL0 | 00000001 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE4

Note: Refer to SFR Definition 24.1. “TCON: Timer Control” on page 235 for INT0/1 edge- or level-sensitive interrupt selection.

Bit 7: IN1PL: /INT1 Polarity
0: /INT1 input is active low.
1: /INT1 input is active high.

Bits 6–4: IN1SL2–0: /INT1 Port Pin Selection Bits
These bits select which Port pin is assigned to /INT1. Note that this pin assignment is independent of the Crossbar; /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 (accomplished by setting to ‘1’ the corresponding bit in register P0SKIP).

| IN1SL2–0 | /INT1 Port Pin |
|---|---|
| 000 | P0.0 |
| 001 | P0.1 |
| 010 | P0.2 |
| 011 | P0.3 |
| 100 | P0.4 |
| 101 | P0.5 |
| 110 | P0.6 |
| 111 | P0.7 |

Bit 3: IN0PL: /INT0 Polarity
0: /INT0 interrupt is active low.
1: /INT0 interrupt is active high.

Bits 2–0: INT0SL2–0: /INT0 Port Pin Selection Bits
These bits select which Port pin is assigned to /INT0. Note that this pin assignment is independent of the Crossbar. /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 (accomplished by setting to ‘1’ the corresponding bit in register P0SKIP).

| IN0SL2–0 | /INT0 Port Pin |
|---|---|
| 000 | P0.0 |
| 001 | P0.1 |
| 010 | P0.2 |
| 011 | P0.3 |
| 100 | P0.4 |
| 101 | P0.5 |
| 110 | P0.6 |
| 111 | P0.7 |

## 13. Prefetch Engine

The C8051F41x family of devices incorporate a 2-byte prefetch engine. Due to Flash access time specifications, the prefetch engine is necessary for full-speed (50 MHz) code execution. Instructions are read from Flash memory two bytes at a time by the prefetch engine, and given to the CIP-51 processor core to execute. When running linear code (code without any jumps or branches), the prefetch engine allows instructions to be executed at full speed. When a code branch occurs, the processor may be stalled for up to two clock cycles while the next set of code bytes is retrieved from Flash memory. The FLRT bit (FLSCL.4) determines how many clock cycles are used to read each set of two code bytes from Flash. When operating from a system clock of 25 MHz or less, the FLRT bit should be set to ‘0’ so that the prefetch engine takes only one clock cycle for each read. When operating with a system clock of greater than 25 MHz (up to 50 MHz), the FLRT bit should be set to ‘1’, so that each prefetch code read lasts for two clock cycles.

### SFR Definition 13.1. PFE0CN: Prefetch Engine Control

| R | R | R/W | R | R | R | R | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | PFEN | | | | | FLBWE | 00100000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE3

Bits 7–6: Unused. Read = 00b; Write = Don’t Care
Bit 5: PFEN: Prefetch Enable.
This bit enables the prefetch engine.
0: Prefetch engine is disabled.
1: Prefetch engine is enabled.
Bits 4–1: Unused. Read = 0000b; Write = Don’t Care
Bit 0: FLBWE: Flash Block Write Enable.
This bit allows block writes to Flash memory from software.
0: Each byte of a software Flash write is written individually.
1: Flash bytes are written in groups of two.

Note: The prefetch engine should be disabled when changes to FLRT are made. See **Section “16. Flash Memory” on page 135**.

## NOTES:

# 14. Cyclic Redundancy Check Unit (CRC0)

C8051F41x 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 14.1. CRC0 also has a bit reverse register for quick data manipulation.

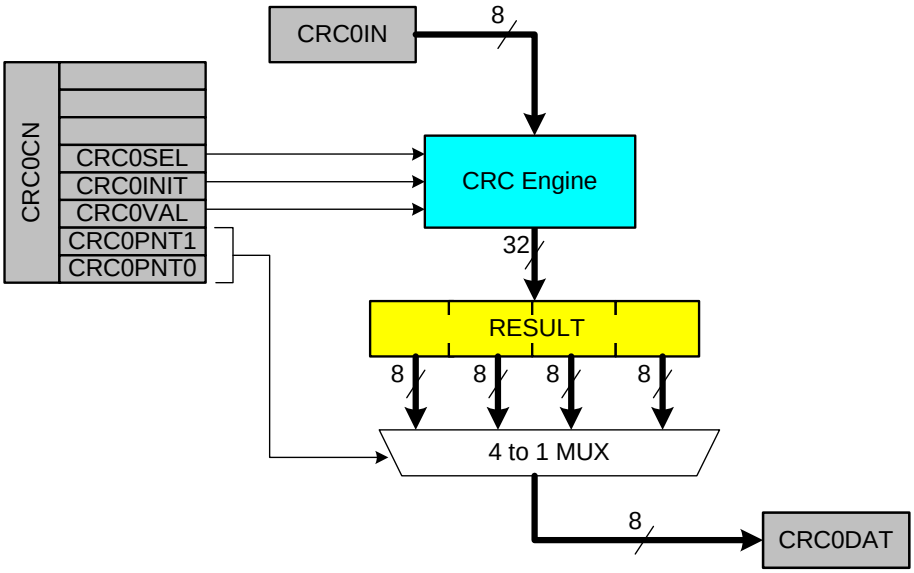



**Figure 14.1. CRC0 Block Diagram**

## 14.1. 16-bit CRC Algorithm

The C8051F41x 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:

Step 1. XOR the most-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 (0x0000 or 0xFFFF).

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

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

Step 3. Repeat at Step 2a for the number of input bits (8).

For example, the 16-bit 'F41x 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 'F41x CRC algorithm (an initial value of 0xFFFF is used):

**Table 14.1. Example 16-bit CRC Outputs**

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

## 14.2. 32-bit CRC Algorithm

The C8051F41x 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 'F41x 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 'F41x CRC algorithm (an initial value of 0xFFFFFFFF is used):

**Table 14.2. Example 32-bit CRC Outputs**

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

## 14.3. 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.

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

## 14.4. 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.

## 14.5. 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 CRC0 result register until it is set, overwritten, or additional data is written to CRC0IN.

## 14.6. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 14.2. 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.

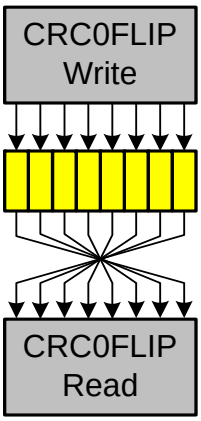



**Figure 14.2. Bit Reverse Register**

## SFR Definition 14.1. CRC0CN: CRC0 Control

| R | R | R | R/W | W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | CRC0SEL | CRC0INIT | CRC0VAL | CRC0PNT | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x84

Bits 7–5: UNUSED. Read = 0b. Write = don't care.
Bit 4: CRC0SEL: CRC0 Polynomial Select Bit.
0: CRC0 uses the 32-bit polynomial 0x04C11DB7 for calculating the CRC result.
1: CRC0 uses the 16-bit polynomial 0x1021 for calculating the CRC result.
Bit 3: CRC0INIT: CRC0 Result Initialization Bit.
Writing a '1' to this bit initializes the entire CRC result based on CRC0VAL.
Bit 2: CRC0VAL: CRC0 Set Value Select Bit
This bit selects the set value of the CRC result.
0: CRC result is set to 0x00000000 on write of '1' to CRC0INIT.
1: CRC result is set to 0xFFFFFFFF on write of '1' to CRC0INIT.
Bits 1–0: CRC0PNT. CRC0 Result Pointer.
These bits specify which byte of the CRC result will be read/written on the next access to CRC0DAT. The value of these bits will auto-increment upon each read or write.
For CRC0SEL = 0:
00: CRC0DAT accesses bits 7–0 of the 32-bit CRC result.
01: CRC0DAT accesses bits 15–8 of the 32-bit CRC result.
10: CRC0DAT accesses bits 23–16 of the 32-bit CRC result.
11: CRC0DAT accesses bits 31–24 of the 32-bit CRC result.
For CRC0SEL = 1:
00: CRC0DAT accesses bits 7–0 of the 16-bit CRC result.
01: CRC0DAT accesses bits 15–8 of the 16-bit CRC result.
10: CRC0DAT accesses bits 7–0 of the 16-bit CRC result.
11: CRC0DAT accesses bits 15–8 of the 16-bit CRC result.

## SFR Definition 14.2. CRC0IN: CRC0 Data Input

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x85

Bits 7–0: CRC0IN: CRC Data Input
Each write to CRCIN results in the written data being computed into the existing CRC result.

## SFR Definition 14.3. CRC0DAT: CRC0 Data Output

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x86

Bits 7–0: CRC0DAT: Indirect CRC Result Data Bits.
Each operation performed on CRC0DAT targets the CRC result bits pointed to by CRC0PNT.

## SFR Definition 14.4. CRC0FLIP: CRC0 Bit Flip

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xDF

Bits 7–0: CRC0FLIP: CRC Bit Flip.
Any byte written to CRC0FLIP is read back in a bit-reversed order, i.e. the written LSB becomes the MSB. For example:

If 0xC0 is written to CRC0FLIP, the data read back will be 0x03.
If 0x05 is written to CRC0FLIP, the data read back will be 0xA0.

## 15. 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 detailed descriptions. The contents of internal data memory are unaffected during a reset; any previously stored data is preserved. However, 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 the internal oscillator. Refer to **Section "19. Oscillators" on page 165** 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 "25.3. Watchdog Timer Mode" on page 257** details the use of the Watchdog Timer). Program execution begins at location 0x0000.

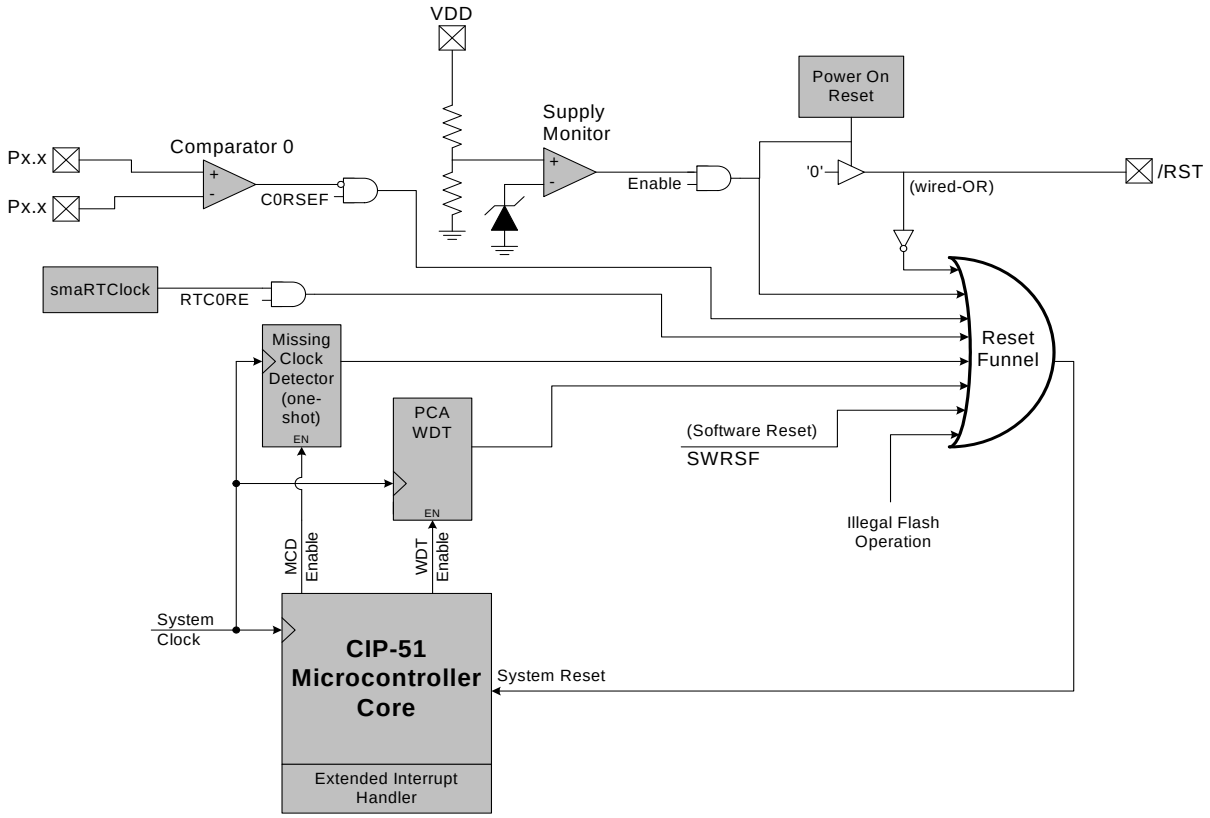



**Figure 15.1. Reset Sources**

## 15.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until $V_{DD}$ settles above $V_{RST}$. An additional delay occurs before the device is released from reset; the delay decreases as the $V_{DD}$ ramp time increases ($V_{DD}$ ramp time is defined as how fast $V_{DD}$ ramps from 0 V to $V_{RST}$). Figure 15.2 plots the power-on and $V_{DD}$ monitor reset timing. For valid ramp times (less than 1 ms), the power-on reset delay ($T_{PORDelay}$) is typically less than 0.3 ms.

**Note:** The maximum $V_{DD}$ ramp time is 1 ms; slower ramp times may cause the device to be released from reset before $V_{DD}$ reaches the $V_{RST}$ 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. The $V_{DD}$ monitor is enabled following a power-on reset.

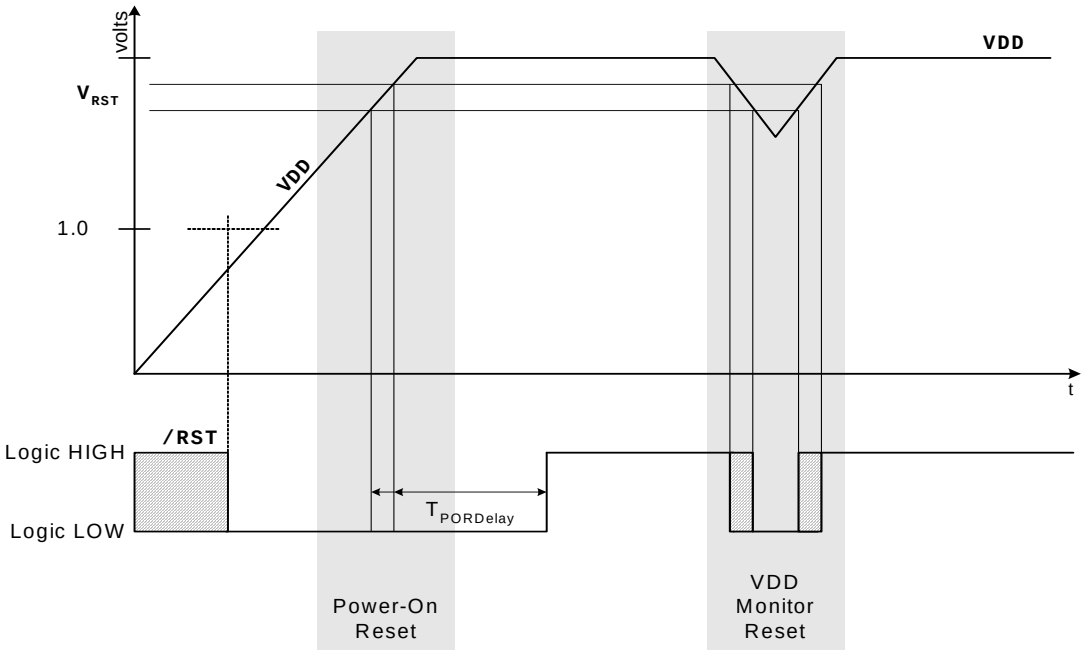



**Figure 15.2. Power-On and $V_{DD}$ Monitor Reset Timing**

## 15.2. Power-Fail Reset / $V_{DD}$ Monitor

When the VDD Monitor is selected as a reset source and a power-down transition or power irregularity causes $V_{DD}$ to drop below $V_{RST}$, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 15.2). When $V_{DD}$ returns to a level above $V_{RST}$, the CIP-51 will be released from the reset state. Note that even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if $V_{DD}$ dropped below the level required for data retention. If the PORSF flag reads ‘1’, the data may no longer be valid. The $V_{DD}$ monitor is enabled and is selected as a reset source after power-on resets; however its defined state (enabled/disabled) is not altered by any other reset source. For example, if the $V_{DD}$ monitor is disabled by software, and a software reset is performed, the $V_{DD}$ monitor will still be disabled after the reset. **To protect the integrity of Flash contents, the $V_{DD}$ monitor must be enabled to the higher setting (VDMLVL = '1') and selected as a reset source if software contains routines which erase or write Flash memory. If the $V_{DD}$ monitor is not enabled, any erase or write performed on Flash memory will cause a Flash Error device reset.**

**The $V_{DD}$ monitor must be enabled before it is selected as a reset source.** Selecting the $V_{DD}$ monitor as a reset source before it is enabled and stabilized may cause a system reset. The procedure for re-enabling the $V_{DD}$ monitor and configuring the $V_{DD}$ monitor as a reset source is shown below:

Step 1. Enable the $V_{DD}$ monitor (VDMEN bit in VDM0CN = ‘1’).
Step 2. Wait for the $V_{DD}$ monitor to stabilize (approximately 5 µs).
**Note: This delay should be omitted if software contains routines which erase or write Flash memory.**
Step 3. Select the $V_{DD}$ monitor as a reset source (PORSF bit in RSTSRC = ‘1’).

See Figure 15.2 for $V_{DD}$ monitor timing; note that the reset delay is not incurred after a $V_{DD}$ monitor reset. See Table 15.1 for complete electrical characteristics of the $V_{DD}$ monitor.

**Note: 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.**

### SFR Definition 15.1. VDM0CN: $V_{DD}$ Monitor Control

| R/W | R | R/W | R | R | R | R | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| VDMEN | VDDSTAT | VDMLVL | Reserved | Reserved | Reserved | Reserved | Reserved | 1v000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xFF

Bit7: VDMEN: $V_{DD}$ Monitor Enable.
This bit turns the $V_{DD}$ monitor circuit on/off. The $V_{DD}$ Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 15.2). The $V_{DD}$ Monitor can be allowed to stabilize before it is selected as a reset source. **Selecting the $V_{DD}$ monitor as a reset source before it has stabilized may generate a system reset.** See Table 15.1 for the minimum $V_{DD}$ Monitor turn-on time.
0: $V_{DD}$ Monitor Disabled.
1: $V_{DD}$ Monitor Enabled (default).

Bit6: VDDSTAT: $V_{DD}$ Status.
This bit indicates the current power supply status ($V_{DD}$ Monitor output).
0: $V_{DD}$ is at or below the $V_{DD}$ Monitor Threshold.
1: $V_{DD}$ is above the $V_{DD}$ Monitor Threshold.

Bit5: VDMLVL: $V_{DD}$ Level Select.
0: $V_{DD}$ Monitor Threshold is set to $V_{RST\text{-}LOW}$ (default).
1: $V_{DD}$ Monitor Threshold is set to $V_{RST\text{-}HIGH}$. This setting is recommended for any system that includes code that writes to and/or erases Flash.

Bits4–0: Reserved. Read = Variable. Write = don't care.

## 15.3. External Reset

The external $\overline{RST}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{RST}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{RST}$ pin may be necessary to avoid erroneous noise-induced resets. See Table 15.1 for complete $\overline{RST}$ pin specifications. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

## 15.4. 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 state of the $\overline{RST}$ pin is unaffected by this reset.

## 15.5. 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 state of the $\overline{RST}$ pin is unaffected by this reset.

## 15.6. 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 “25.3. Watchdog Timer Mode” on page 257**; 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 state of the $\overline{RST}$ pin is unaffected by this reset.

## 15.7. 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 “16.3. Security Options” on page 137**).
- 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 $\overline{RST}$ pin is unaffected by this reset.

## 15.8. 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 state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 15.9. 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 15.2. RSTSRC: Reset Source

| R/W | R | R/W | R/W | R | R/W | R/W | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| RTC0RE | FERROR | C0RSEF | SWRSF | WDTRSF | MCDRSF | PORSF | PINRSF | Variable |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xEF

**Note:** For bits that act as both reset source enables (on a write) and reset indicator flags (on a read), read-modify-write instructions read and modify the source enable only. [This applies to bits: RTC0RE, C0RSEF, SWRSF, MCDRSF, PORSF].

Bit7: RTC0RE: smaRTClock (Real Time Clock) Reset Enable and Flag.
0: **Read:** Source of last reset was not a smaRTClock alarm or oscillator fail event.
**Write:** smaRTClock is not a reset source.
1: **Read:** Source of last reset was a smaRTClock alarm or oscillator fail event.
**Write:** smaRTClock is a reset source.

Bit6: FERROR: Flash Error Indicator.
0: Source of last reset was not a Flash read/write/erase error.
1: Source of last reset was a Flash read/write/erase error.

Bit5: C0RSEF: Comparator0 Reset Enable and Flag.
0: **Read:** Source of last reset was not Comparator0.
**Write:** Comparator0 is not a reset source.
1: **Read:** Source of last reset was Comparator0.
**Write:** Comparator0 is a reset source (active-low).

Bit4: SWRSF: Software Reset Force and Flag.
0: **Read:** Source of last reset was not a write to the SWRSF bit.
**Write:** No Effect.
1: **Read:** Source of last was a write to the SWRSF bit.
**Write:** Forces a system reset.

Bit3: WDTRSF: Watchdog Timer Reset Flag.
0: Source of last reset was not a WDT timeout.
1: Source of last reset was a WDT timeout.

Bit2: MCDRSF: Missing Clock Detector Flag.
0: **Read:** Source of last reset was not a Missing Clock Detector timeout.
**Write:** Missing Clock Detector disabled.
1: **Read:** Source of last reset was a Missing Clock Detector timeout.
**Write:** Missing Clock Detector enabled; triggers a reset if a missing clock condition is detected.

Bit1: PORSF: Power-On Reset Force and Flag.
This bit is set anytime a power-on reset occurs. Writing this bit enables/disables the $V_{DD}$ monitor as a reset source. **Note: writing '1' to this bit before the $V_{DD}$ monitor is enabled and stabilized may cause a system reset.** See register VDM0CN (SFR Definition 15.1)
0: **Read:** Last reset was not a power-on or $V_{DD}$ monitor reset.
**Write:** $V_{DD}$ monitor is not a reset source.
1: **Read:** Last reset was a power-on or $V_{DD}$ monitor reset; all other reset flags indeterminate.
**Write:** $V_{DD}$ monitor is a reset source.

Bit0: PINRSF: HW Pin Reset Flag.
0: Source of last reset was not $\overline{RST}$ pin.
1: Source of last reset was $\overline{RST}$ pin.

## Table 15.1. Reset Electrical Characteristics

–40 to +85 °C unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| $\overline{RST}$ Output Low Voltage | **$V_{IO}$ = 2.0 V:**<br>$I_{OL}$ = 70 µA<br>$I_{OL}$ = 8.5 mA<br>**$V_{IO}$ = 4.0 V:**<br>$I_{OL}$ = 70 µA<br>$I_{OL}$ = 8.5 mA | <br>—<br>—<br><br>—<br>— | <br>—<br>—<br><br>—<br>— | <br>50<br>800<br><br>40<br>400 | mV |
| $\overline{RST}$ Input High Voltage | | 0.7 x $V_{IO}$ | — | — | V |
| $\overline{RST}$ Input Low Voltage | | — | — | 0.3 x $V_{IO}$ | V |
| $\overline{RST}$ Input Pullup Impedance | $V_{IO}$ = 2.0 V<br>$V_{IO}$ = 5.0 V | —<br>— | 150<br>70 | —<br>— | kΩ |
| $V_{DD}$ Monitor Threshold ($V_{RST-LOW}$) | | 1.9 | 1.95 | 2.0 | V |
| $V_{DD}$ Monitor Threshold ($V_{RST-HIGH}$) | | 2.25 | 2.3 | 2.35 | V |
| Missing Clock Detector Timeout | Time from last system clock rising edge to reset initiation | 50 | 350 | 650 | µs |
| Reset Time Delay | Delay between release of any reset source and code execution at location 0x0000 | — | — | 180 | µs |
| Minimum $\overline{RST}$ Low Time to Generate a System Reset | | 20 | — | — | µs |
| $V_{DD}$ Monitor Supply Current | | — | 0.7 | 70 | µA |
| $V_{DD}$ Ramp Time | $V_{DD}$ = 0 V to $V_{DD}$ = $V_{RST}$ V | — | — | 1 | ms |

# 16. 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 16.2 for complete Flash memory electrical characteristics.

## 16.1. 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 "26. C2 Interface" on page 265**. For detailed guidelines on writing or erasing Flash from firmware, please see **Section "16.4. Flash Write and Erase Guidelines" on page 139**.

**To ensure the integrity of the Flash contents, the on-chip VDD Monitor must be enabled to the higher setting (VDMLVL = '1') 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 disabled will cause a Flash Error device reset.**

### 16.1.1. 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 16.2.

### 16.1.2. Flash Erase Procedure

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 writing to Flash memory using MOVX, Flash write 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.

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 Flash memory is organized in 512-byte pages. The erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire 512-byte page, perform the following steps:

Step 1. Disable interrupts (recommended).
Step 2. Write the first key code to FLKEY: 0xA5.
Step 3. Write the second key code to FLKEY: 0xF1.
Step 4. Set the PSEE bit (register PSCTL).
Step 5. Set the PSWE bit (register PSCTL).
Step 6. Using the MOVX instruction, write a data byte to any location within the 512-byte page to be erased.
Step 7. Clear the PSWE and PSEE bits.
Step 8. Re-enable interrupts.

### 16.1.3. Flash Write Procedure

Bytes in Flash memory can be written one byte at a time, or in groups of two. The FLBWE bit in register PFE0CN (SFR Definition 13.1) controls whether a single byte or a block of two bytes is written to Flash during a write operation. When FLBWE is cleared to ‘0’, the Flash will be written one byte at a time. When FLBWE is set to ‘1’, the Flash will be written in two-byte blocks. Block writes are performed in the same amount of time as single-byte writes, which can save time when storing large amounts of data to Flash memory.

During a single-byte write to Flash, bytes are written individually, and a Flash write will be performed after each MOVX write instruction. The recommended procedure for writing Flash in single bytes is:

Step 1. Disable interrupts.
Step 2. Clear the FLBWE bit (register PFE0CN) to select single-byte write mode.
Step 3. Write '0000' to FLSCL.3–0.
Step 4. Write the first key code to FLKEY: 0xA5.
Step 5. Write the second key code to FLKEY: 0xF1.
Step 6. Set the PSWE bit (register PSCTL).
Step 7. Clear the PSEE bit (register PSCTL).
Step 8. Using the MOVX instruction, write a single data byte to the desired location within the 512-byte sector.
Step 9. Clear the PSWE bit.
Step 10. Re-enable interrupts.

Steps 3–9 must be repeated for each byte to be written.

For block Flash writes, the Flash write procedure is only performed after the last byte of each block is written with the MOVX write instruction. A Flash write block is two bytes long, from even addresses to odd addresses. Writes must be performed sequentially (i.e. addresses ending in 0b and 1b must be written in order). The Flash write will be performed following the MOVX write that targets the address ending in 1b. If a byte in the block does not need to be updated in Flash, it should be written to 0xFF. The recommended procedure for writing Flash in blocks is:

Step 1. Disable interrupts.
Step 2. Set the FLBWE bit (register PFE0CN) to select block write mode.
Step 3. Write '0000' to FLSCL.3–0.
Step 4. Write the first key code to FLKEY: 0xA5.
Step 5. Write the second key code to FLKEY: 0xF1.
Step 6. Set the PSWE bit (register PSCTL).
Step 7. Clear the PSEE bit (register PSCTL).
Step 8. Using the MOVX instruction, write the first data byte to the even block location (ending in 0b).
Step 9. Clear the PSWE bit (register PSCTL).
Step 10. Write the first key code to FLKEY: 0xA5.
Step 11. Write the second key code to FLKEY: 0xF1.
Step 12. Set the PSWE bit (register PSCTL).
Step 13. Clear the PSEE bit (register PSCTL).
Step 14. Using the MOVX instruction, write the second data byte to the odd block location (ending in 1b).
Step 15. Clear the PSWE bit (register PSCTL).
Step 16. Re-enable interrupts.

Steps 3-15 must be repeated for each block to be written.

## 16.2. 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.

## 16.3. 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* 512-byte Flash pages, starting at page 0 (addresses 0x0000 to 0x01FF), where *n* is the 1's 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 example below for an C8051F410.

| | |
|---|---|
| Security Lock Byte: | 11111101b |
| 1's Complement: | 00000010b |
| Flash pages locked: | 3 (First two Flash pages + Lock Byte Page) |
| Addresses locked: | 0x0000 to 0x03FF (first two Flash pages) and<br>0x7C00 to 0x7DFF (Lock Byte Page) |

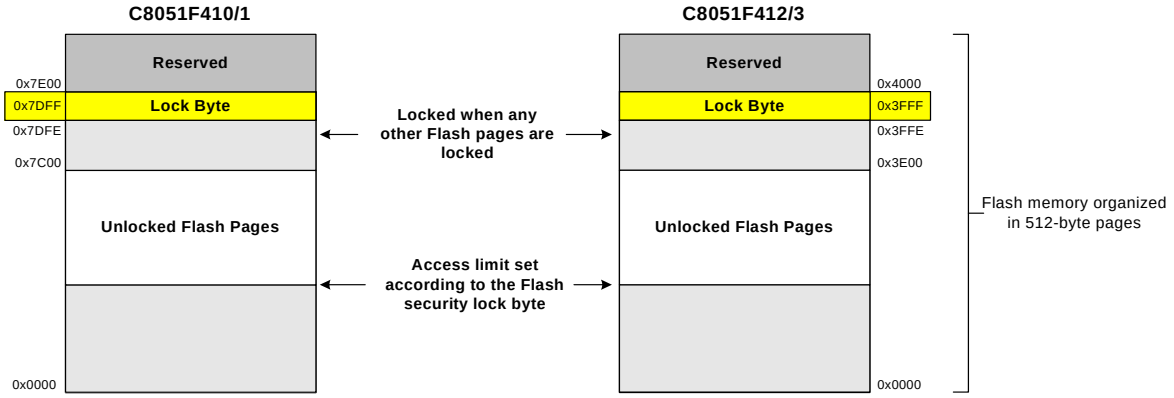



**Figure 16.1. Flash Program Memory Map**

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 16.1 summarizes the Flash security features of the 'F41x devices.

**Table 16.1. 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 C2DE | FEDR | FEDR |
| Lock additional pages (change '1's to '0's in the Lock Byte) | Not Permitted | FEDR | FEDR |
| Unlock individual pages (change '0's to '1's 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.

## 16.4. 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 C8051F41x 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.

### 16.4.1. 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 VDD 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 /RST pin of the device that holds the device in reset until VDD reaches $V_{RST}$ and re-asserts /RST if VDD drops below $V_{RST}$.
3. Keep the on-chip VDD Monitor enabled and enable the VDD 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 VDD Monitor and enabling the VDD Monitor as a reset source. Code examples showing this can be found in AN201, "Writing to Flash from Firmware", available from the Silicon Laboratories web site.

   **Note:** On C8051F41x devices, both the VDD Monitor and the VDD Monitor reset source must be enabled to write or erase Flash without generating a Flash Error Device Reset.

4. As an added precaution, explicitly enable the VDD Monitor and enable the VDD Monitor as a reset source inside the functions that write and erase Flash memory. The VDD 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.

### 16.4.2. 16.4.2 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 web site.
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.

### 16.4.3. 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.

## SFR Definition 16.1. PSCTL: Program Store R/W Control

| R | R | R | R | R | R | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | PSEE | PSWE | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8F

Bits7–2: UNUSED: Read = 000000b, Write = don't care.

Bit1: PSEE: Program Store Erase Enable
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.
0: Flash program memory erasure disabled.
1: Flash program memory erasure enabled.

Bit0: PSWE: Program Store Write Enable
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.
0: Writes to Flash program memory disabled.
1: Writes to Flash program memory enabled; the MOVX write instruction targets Flash memory.

## SFR Definition 16.2. FLKEY: Flash Lock and Key

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB7

Bits7–0: FLKEY: Flash Lock and Key Register
Write:
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 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.
Read:
When read, bits 1-0 indicate the current Flash lock state.
00: Flash is write/erase locked.
01: The first key code has been written (0xA5).
10: Flash is unlocked (writes/erases allowed).
11: Flash writes/erases disabled until the next reset.

## 16.5. Flash Read Timing

On reset, the C8051F41x Flash read timing is configured for operation with system clocks up to 25 MHz. If the system clock will not be increased above 25 MHz, then the Flash timing registers may be left at their reset value.

For every Flash read or fetch, the system provides an internal Flash read strobe to the Flash memory. The Flash read strobe lasts for one or two system clock cycles, based on FLRT (FLSCL.4). **If the system clock is greater than 25 MHz, the FLRT bit must be set to logic 1**, otherwise data read or fetched from Flash may not represent the actual contents of Flash.

When the Flash read strobe is asserted, Flash memory is active. When it is de-asserted, Flash memory is in a low power state. The Flash read strobe does not need to be asserted for longer than 80 ns in order for Flash reads and fetches to be reliable. For system clocks greater than 12.5 MHz (but less than 25 MHz), the Flash read strobe width is limited by the system clock period. For system clocks less than 12.5 MHz, the Flash read strobe is limited by a programmable one shot with a default period of 80 ns (1/12.5 MHz). This is a power saving feature that is very beneficial for very slow system clocks (e.g. 32.768 kHz where the system clock period is greater than 30,000 ns).

For additional power savings, the one shot can be programmed to values less than 80 ns. The one shot can be trimmed according the equation in the ONESHOT register description in Figure 16.4. The one shot period must not be programmed less than the minimum read cycle time specified in Table 16.2.

The recommended procedure for updating FLRT or the ONESHOT period is:

Step 1. Select SYSCLK to 25 MHz or less.
Step 2. Disable the prefetch engine (PFEN = ‘0’ in PFE0CN register).
Step 3. Clear FLRT to ‘0’ (FLSCL register).
Step 4. Set the ONESHOT period bits.
Step 5. Set FLRT to ‘1’ if SYSCLK > 25 MHz.
Step 6. Enable the prefetch engine (PFEN = ‘1’ in PFE0CN register).

### SFR Definition 16.3. FLSCL: Flash Scale

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| Reserved | Reserved | Reserved | FLRT | Reserved | Reserved | Reserved | Reserved | 00000011 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB6

Bits7–5: RESERVED. Read = 000b. Must Write 000b.
Bit 4: FLRT: Flash Read Time Control.
This bit should be programmed to the smallest allowed value, according to the system clock speed.
0: SYSCLK $\leq$ 25 MHz (Flash read strobe is one system clock).
1: SYSCLK > 25 MHz (Flash read strobe is two system clocks).
Bits3–0: RESERVED. Must Write 0000b.

## SFR Definition 16.4. ONESHOT: Flash Oneshot Period

| R | R | R | R | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | - | PERIOD | | | | 00001111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAF

Bits7–4: UNUSED. Read = 0000b. Write = don't care.

Bits3–0: PERIOD: Oneshot Period Control Bits.
These bits limit the internal Flash read strobe width as follows. When the Flash read strobe is de-asserted, the Flash memory enters a low-power state for the remainder of the system clock cycle. These bits have no effect when the system clocks is greater than 12.5 MHz and FLRT = 0.

$$FLASH_{RDMAX} = 5ns + (PERIOD \times 5ns)$$

## Table 16.2. Flash Electrical Characteristics

$V_{DD}$ = 2.0 to 2.75 V; –40 to +85 ºC unless otherwise specified. Typical values are given at 25 ºC.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Flash Size | C8051F410/1<br>C8051F412/3 | 32768*<br>16384 | — | — | bytes |
| Endurance | $V_{DD}$ is 2.2 V or greater | 20 k | 90 k | — | Erase/Write |
| Erase Cycle Time | FLSCL.3–0 written to '0000' | 16 | 20 | 24 | ms |
| Write Cycle Time | FLSCL.3–0 written to '0000' | 38 | 46 | 57 | µs |
| Read Cycle Time | | 40 | — | — | ns |
| $V_{DD}$ | Write/Erase Operations | 2.25 | — | — | V |

***Note:** 512 bytes at addresses 0x7E00 to 0x7FFF are reserved.

## NOTES:

## 17. External RAM

The C8051F41x devices include 2048 bytes of 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. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN as shown in SFR Definition 17.1). Note: the MOVX instruction is also used for writes to the Flash memory. See **Section "16. Flash Memory" on page 135** for details. The MOVX instruction accesses XRAM by default.

For a 16-bit MOVX operation (@DPTR), the upper 5-bits of the 16-bit external data memory address word are "don't cares." As a result, the RAM is mapped modulo style over the entire 64 k external data memory address range. For example, the XRAM byte at address 0x0000 is shadowed at addresses 0x0800, 0x1000, 0x1800, 0x2000, etc. This is a useful feature when performing a linear memory fill, as the address pointer doesn't have to be reset when reaching the RAM block boundary.

### SFR Definition 17.1. EMI0CN: External Memory Interface Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | PGSEL | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAA

Bits 7–3: UNUSED. Read = 00000b. Write = don't care.

Bits 2–0: PGSEL: XRAM Page Select.
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, the PGSEL determines which page of XRAM is accessed.

For Example: If EMI0CN = 0x01, addresses 0x0100 through 0x01FF will be accessed.

## NOTES:

## 18. Port Input/Output

Digital and analog resources are available through up to 24 I/O pins. Port pins are organized as three byte-wide Ports. Each of the Port pins can be defined as general-purpose I/O (GPIO) or analog input/output; Port pins P0.0 - P2.7 can be assigned to one of the internal digital resources as shown in Figure 18.3. The designer has complete control over which functions are assigned, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. Note that the state of a Port I/O pin can always be read in the corresponding Port latch, regardless of the Crossbar settings.

The Crossbar assigns the selected internal digital resources to the I/O pins based on the peripheral priority order of the Priority Decoder (Figure 18.3 and Figure 18.4). The registers XBR0 and XBR1, defined in SFR Definition 18.1 and SFR Definition 18.2, are used to select internal digital functions.

Port I/Os on P0 are 5 V tolerant over the operating range of $V_{IO}$. Port I/Os on P1 and P2 should not be driven above $V_{IO}$ or they will sink current. Figure 18.2 shows the Port cell circuit. The Port I/O cells are configured as either push-pull or open-drain in the Port Output Mode registers (PnMDOUT, where n = 0,1,2). Complete Electrical Specifications for Port I/O are given in Table 18.1 on page 163.

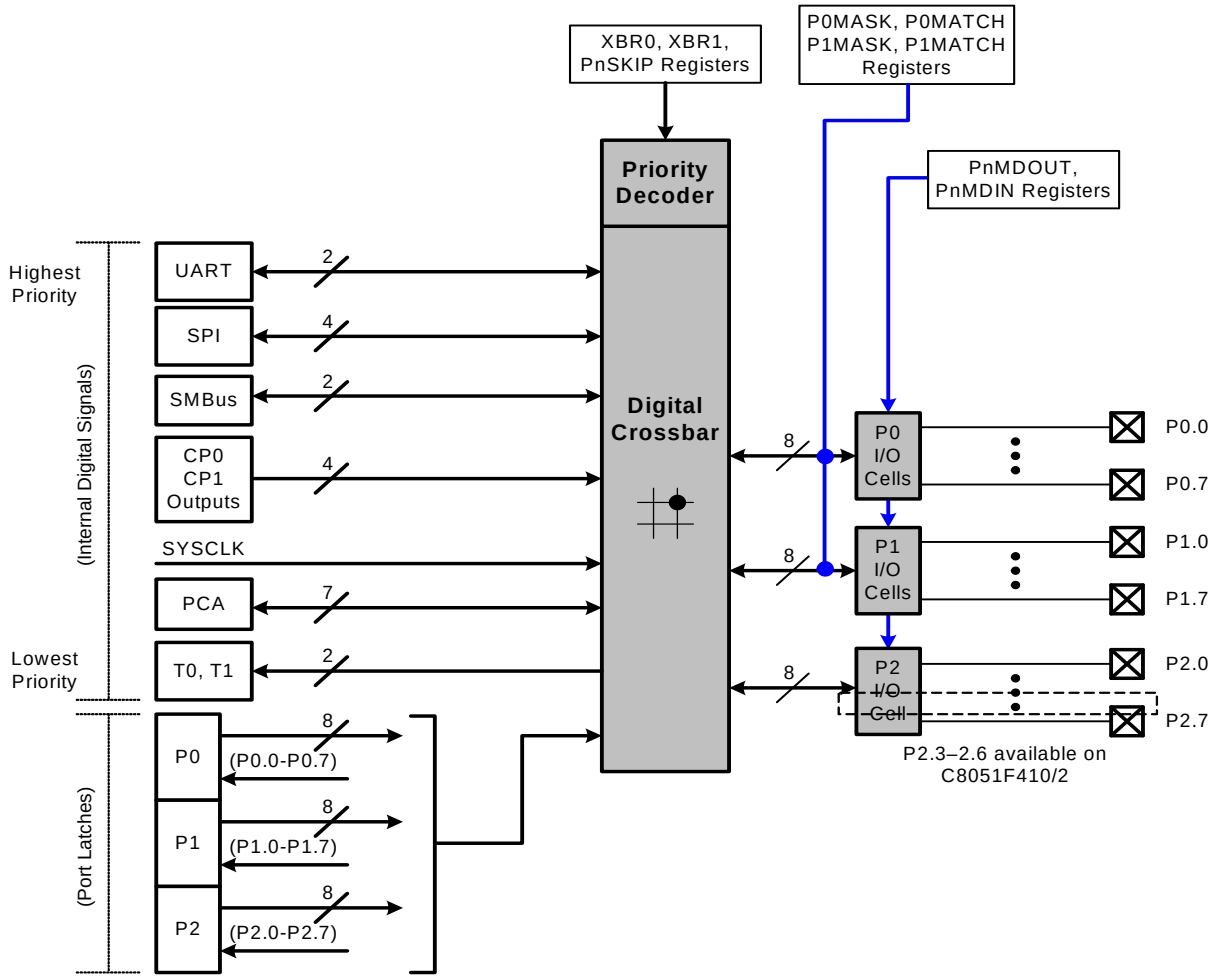



**Figure 18.1. Port I/O Functional Block Diagram**

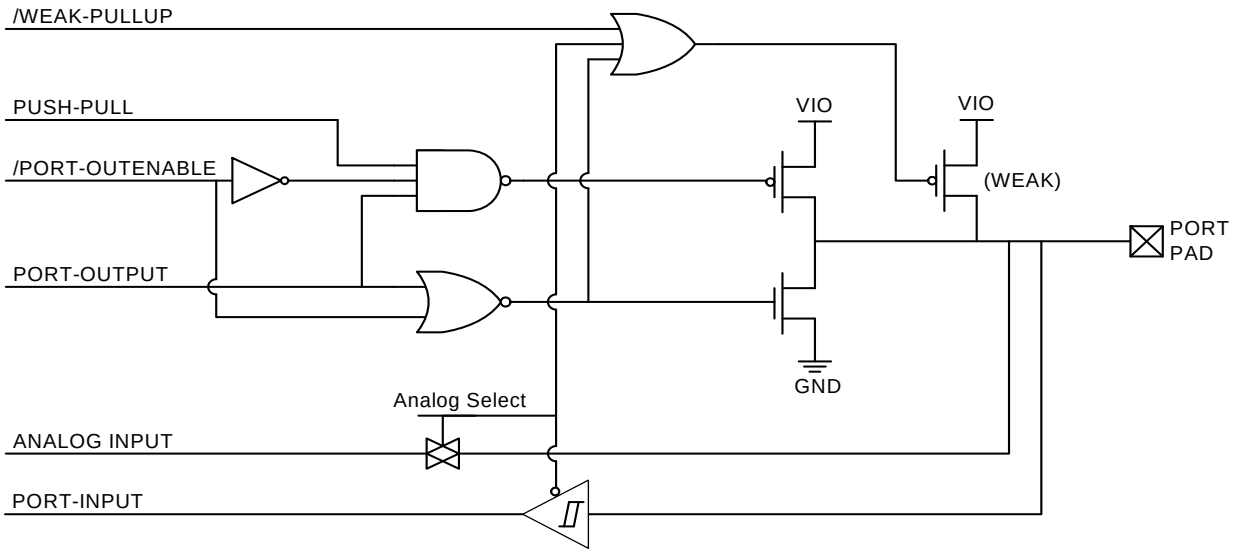



**Figure 18.2. Port I/O Cell Block Diagram**

## 18.1. Priority Crossbar Decoder

The Priority Crossbar Decoder (Figure 18.3) assigns a priority to each I/O function, starting at the top with UART0. When a digital resource is selected, the least-significant unassigned Port pin is assigned to that resource (excluding UART0, which will be assigned to pins P0.4 and P0.5). If a Port pin is assigned, the Crossbar skips that pin when assigning the next selected resource. Additionally, the Crossbar will skip Port pins whose associated bits in the PnSKIP registers are set. The PnSKIP registers allow software to skip Port pins that are to be used for analog input, dedicated functions, or GPIO.

**Important Note on Crossbar Configuration:** If a Port pin is claimed by a peripheral without use of the Crossbar, its corresponding PnSKIP bit should be set. This applies to P1.0 and/or P1.1 for the external oscillator, P1.2 for $V_{REF}$, P0.6 for the external CNVSTR signal, P0.0 for IDA0, P0.1 for IDA1, and any selected ADC or comparator inputs. The Crossbar skips selected pins as if they were already assigned, and moves to the next unassigned pin. Figure 18.3 shows the Crossbar Decoder priority with no Port pins skipped (P0SKIP, P1SKIP, P2SKIP = 0x00); Figure 18.4 shows the Crossbar Decoder priority with the XTAL1 (P1.0) and XTAL2 (P1.1) pins skipped (P1SKIP = 0x03).

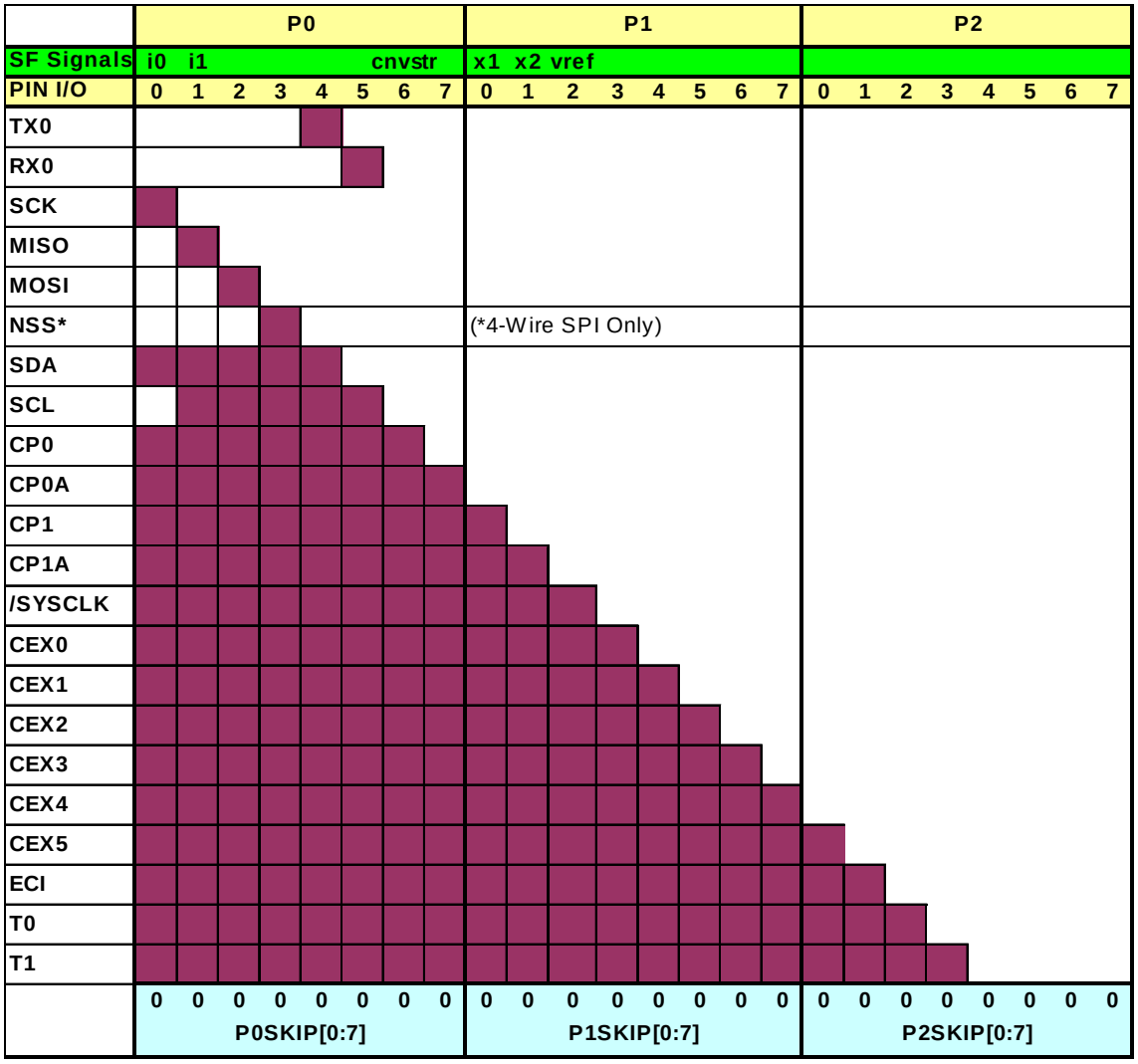
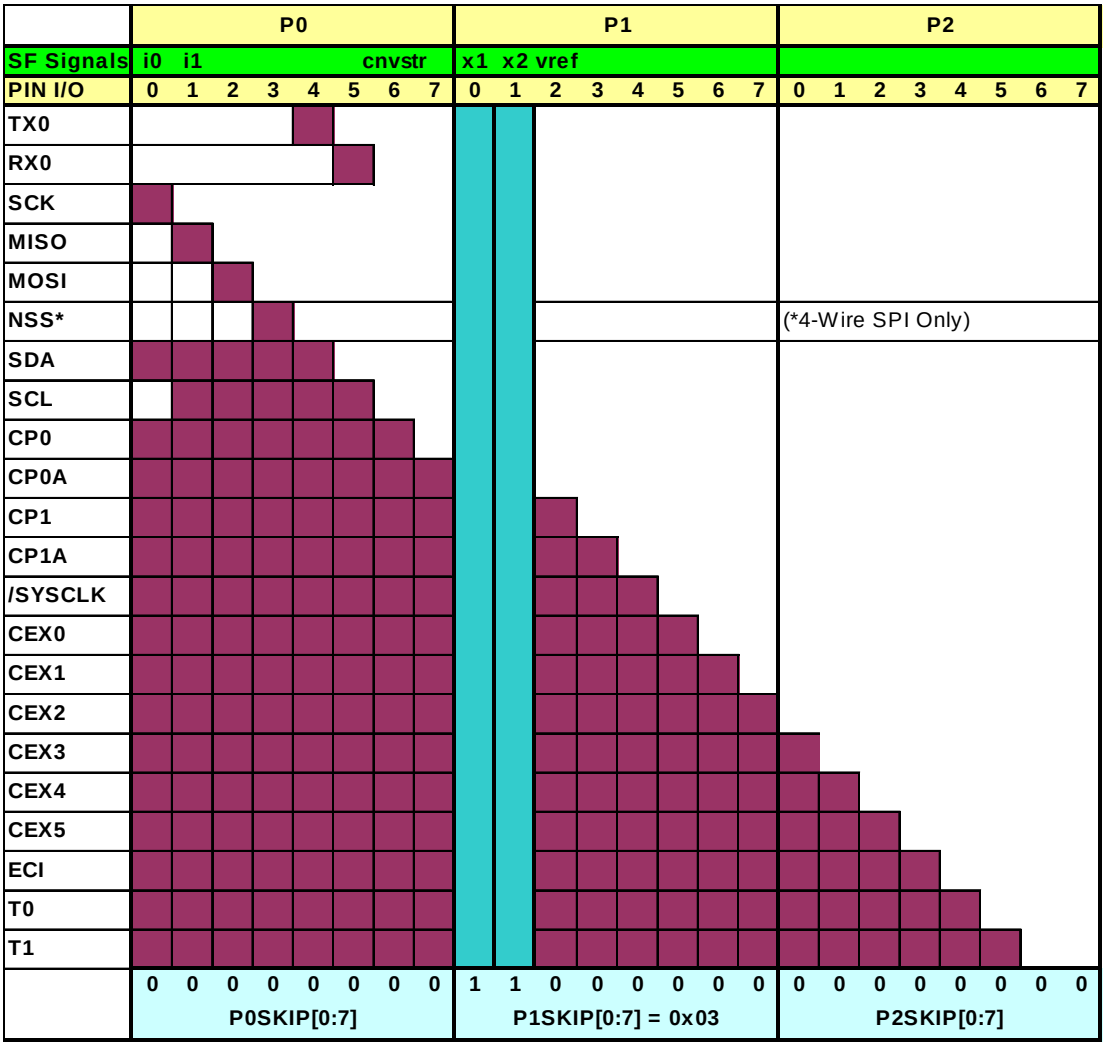



**Figure 18.3. Crossbar Priority Decoder with No Pins Skipped**




Port pin potentially assignable to peripheral

**SF Signals** Special Function Signals are not assigned by the crossbar. When these signals are enabled, the CrossBar must be manually configured to skip their corresponding port pins.

**Figure 18.4. Crossbar Priority Decoder with Crystal Pins Skipped**

Registers XBR0 and XBR1 are used to assign the digital I/O resources to the physical I/O Port pins. Note that when the SMBus is selected, the Crossbar assigns both pins associated with the SMBus (SDA and SCL); when the UART is selected, the Crossbar assigns both pins associated with the UART (TX and RX). UART0 pin assignments are fixed for bootloading purposes: UART TX0 is always assigned to P0.4; UART RX0 is always assigned to P0.5. Standard Port I/Os appear contiguously starting at P0.0 after prioritized functions and skipped pins are assigned.

**Important Note:** The SPI can be operated in either 3-wire or 4-wire modes, depending on the state of the NSSMD1-NSSMD0 bits in register SPI0CN. According to the SPI mode, the NSS signal may or may not be routed to a Port pin.

## 18.2. Port I/O Initialization

Port I/O initialization consists of the following steps:

Step 1. Select the input mode (analog or digital) for all Port pins, using the Port Input Mode register (PnMDIN). **If the pin is in analog mode, a '1' must also be written to the corresponding Port Latch.**
Step 2. Select the output mode (open-drain or push-pull) for all Port pins, using the Port Output Mode register (PnMDOUT).
Step 3. Select any pins to be skipped by the I/O Crossbar using the Port Skip registers (PnSKIP).
Step 4. Assign Port pins to desired peripherals using the XBRn registers.
Step 5. Enable the Crossbar (XBARE = ‘1’).

All Port pins must be configured as either analog or digital inputs. Any pins to be used as Comparator or ADC inputs should be configured as an analog inputs. When a pin is configured as an analog input, its weak pullup, digital driver, and digital receiver are disabled. This process saves power and reduces noise on the analog input. Pins configured as digital inputs may still be used by analog peripherals; however, this practice is not recommended.

Additionally, all analog input pins should be configured to be skipped by the Crossbar (accomplished by setting the associated bits in PnSKIP). Port input mode is set in the PnMDIN register, where a ‘1’ indicates a digital input, and a ‘0’ indicates an analog input. All port pins in analog mode must have a '1' set in the corresponding Port Latch register. All pins default to digital inputs on reset. See SFR Definition 18.4 for the PnMDIN register details.

**Important Note:** Port 0 pins are 5 V tolerant across the operating range of $V_{IO}$. Figure 18.5 shows the input current range of P0 pins when overdriven above $V_{IO}$ (when $V_{IO}$ is 3.3 V nominal). There are two overdrive modes for Port 0: Normal and High-Impedance. When the corresponding bit in P0ODEN is logic 0, Normal Overdrive Mode is selected and the port pin requires 150 µA peak overdrive current when its voltage reaches approximately $V_{IO}$ + 0.7 V. When the corresponding bit in P0ODEN is logic 1, High-Impedance Overdrive Mode is selected and the port pin does not require any additional overdrive current. Pins configured to High-Impedance Overdrive Mode consume slightly more power from $V_{IO}$ than pins configured to Normal Overdrive Mode. Note that Port 1 and Port 2 pins cannot be overdriven above $V_{IO}$ and have the same behavior as P0 in Normal Mode.

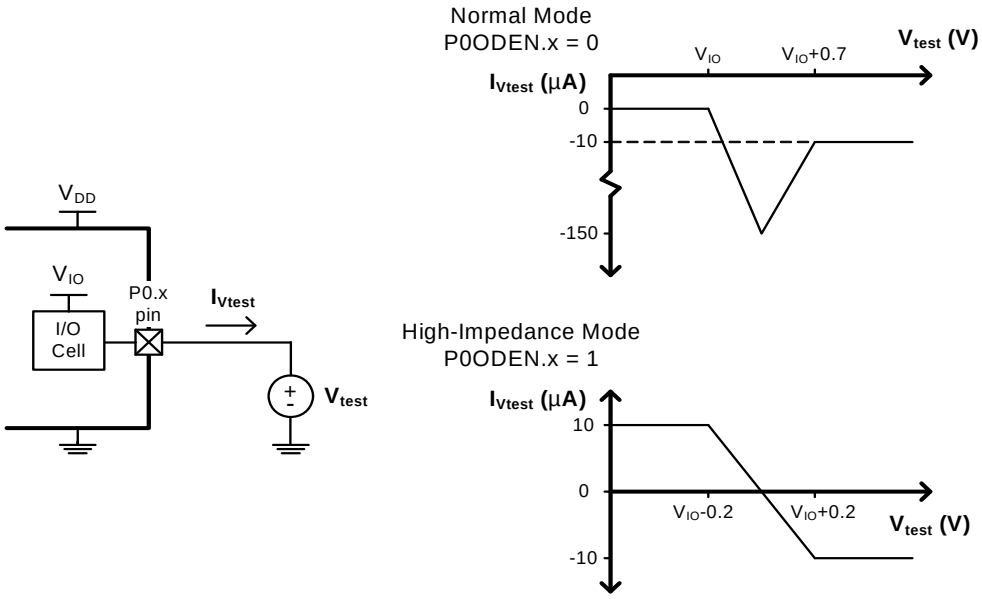



**Figure 18.5. Port 0 Input Overdrive Current Range**

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. When the WEAKPUD bit in XBR1 is ‘0’, a weak pullup is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pullup is turned off on an output that is driving a ‘0’ and for pins configured for analog input mode to avoid unnecessary power dissipation.

Registers XBR0 and XBR1 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR1 to ‘1’ enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard Port I/O (in input mode), regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table.

The Crossbar must be enabled to use Port pins as standard Port I/O in output mode. **Port output drivers are disabled while the Crossbar is disabled.**

## SFR Definition 18.1. XBR0: Port I/O Crossbar Register 0

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CP1AE | CP1E | CP0AE | CP0E | SYSCKE | SMB0E | SPI0E | URT0E | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE1

Bit7: CP1AE: Comparator1 Asynchronous Output Enable
0: Asynchronous CP1 unavailable at Port pin.
1: Asynchronous CP1 routed to Port pin.

Bit6: CP1E: Comparator1 Output Enable
0: CP1 unavailable at Port pin.
1: CP1 routed to Port pin.

Bit5: CP0AE: Comparator0 Asynchronous Output Enable
0: Asynchronous CP0 unavailable at Port pin.
1: Asynchronous CP0 routed to Port pin.

Bit4: CP0E: Comparator0 Output Enable
0: CP0 unavailable at Port pin.
1: CP0 routed to Port pin.

Bit3: SYSCKE: /SYSCLK Output Enable
0: /SYSCLK unavailable at Port pin.
1: /SYSCLK output routed to Port pin.

Bit2: SMB0E: SMBus I/O Enable
0: SMBus I/O unavailable at Port pins.
1: SMBus I/O routed to Port pins.

Bit1: SPI0E: SPI I/O Enable
0: SPI I/O unavailable at Port pins.
1: SPI I/O routed to Port pins. Note that the SPI can be assigned either 3 or 4 GPIO pins.

Bit0: URT0E: UART I/O Output Enable
0: UART I/O unavailable at Port pin.
1: UART TX0, RX0 routed to Port pins P0.4 and P0.5.

**SFR Definition 18.2. XBR1: Port I/O Crossbar Register 1**

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| WEAKPUD | XBARE | T1E | T0E | ECIE | PCA0ME | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xE2

Bit7: WEAKPUD: Port I/O Weak Pullup Disable.
0: Weak Pullups enabled (except for Ports whose I/O are configured as analog input).
1: Weak Pullups disabled.

Bit6: XBARE: Crossbar Enable.
0: Crossbar disabled.
1: Crossbar enabled.

Bit5: T1E: T1 Enable
0: T1 unavailable at Port pin.
1: T1 routed to Port pin.

Bit4: T0E: T0 Enable
0: T0 unavailable at Port pin.
1: T0 routed to Port pin.

Bit3: ECIE: PCA0 External Counter Input Enable
0: ECI unavailable at Port pin.
1: ECI routed to Port pin.

Bits2–0: PCA0ME: PCA Module I/O Enable Bits.
000: All PCA I/O unavailable at Port pins.
001: CEX0 routed to Port pin.
010: CEX0, CEX1 routed to Port pins.
011: CEX0, CEX1, CEX2 routed to Port pins.
100: CEX0, CEX1, CEX2, CEX3 routed to Port pins.
101: CEX0, CEX1, CEX2, CEX3, CEX4 routed to Port pins.
110: CEX0, CEX1, CEX2, CEX3, CEX4, CEX5 routed to Port pins.
111: Reserved.

## 18.3. General Purpose Port I/O

Port pins that remain unassigned by the Crossbar and are not used by analog peripherals can be used for general purpose I/O. Ports P0-P2 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.

In addition to performing general purpose I/O, P0 and P1 can generate a port match event if the logic levels of the Port’s input pins match a software controlled value. A port match event is generated if (P0 & P0MASK) does not equal (P0MATCH & P0MASK) or if (P1 & P1MASK) does not equal

(P1MATCH & P1MASK). This allows Software to be notified if a certain change or pattern occurs on P0 or P1 input pins regardless of the XBRn settings. A port match event can cause an interrupt if EMAT (EIE2.1) is set to '1' or cause the internal oscillator to awaken from SUSPEND mode. See Section "**19.1.1. Internal Oscillator Suspend Mode**" on page **166** for more information.

### SFR Definition 18.3. P0: Port0

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| P0.7 | P0.6 | P0.5 | P0.4 | P0.3 | P0.2 | P0.1 | P0.0 | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0x80

Bits7–0: P0.[7:0]
Write - Output appears on I/O pins per Crossbar Registers.
0: Logic Low Output.
1: Logic High Output (high impedance if corresponding P0MDOUT.n bit = 0).
Read - Always reads '0' if selected as analog input in register P0MDIN. Directly reads Port pin when configured as digital input.
0: P0.n pin is logic low.
1: P0.n pin is logic high.

### SFR Definition 18.4. P0MDIN: Port0 Input Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF1

Bits7–0: Analog Input Configuration Bits for P0.7–P0.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P0.n pin is configured as an analog input. **In order for the P0.n pin to be in analog input mode, there MUST be a '1' in the Port Latch register corresponding to that pin.**
1: Corresponding P0.n pin is not configured as an analog input.

## SFR Definition 18.5. P0MDOUT: Port0 Output Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA4

Bits7–0: Output Configuration Bits for P0.7–P0.0 (respectively): ignored if corresponding bit in register P0MDIN is logic 0.
0: Corresponding P0.n Output is open-drain.
1: Corresponding P0.n Output is push-pull.

(Note: When SDA and SCL appear on any of the Port I/O, each are open-drain regardless of the value of P0MDOUT).

## SFR Definition 18.6. P0SKIP: Port0 Skip

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD4

Bits7–0: P0SKIP[7:0]: Port0 Crossbar Skip Enable Bits.
These bits select Port pins to be skipped by the Crossbar Decoder. Port pins used as analog inputs (for ADC or Comparator) or used as special functions ($V_{REF}$ input, external oscillator circuit, CNVSTR input) should be skipped by the Crossbar.
0: Corresponding P0.n pin is not skipped by the Crossbar.
1: Corresponding P0.n pin is skipped by the Crossbar.

## SFR Definition 18.7. P0MAT: Port0 Match

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD7

Bits7–0: P0MAT[7:0]: Port0 Match Value.
These bits control the value that unmasked P0 Port pins are compared against. A Port Match event is generated if (P0 & P0MASK) does not equal (P0MAT & P0MASK).

## SFR Definition 18.8. P0MASK: Port0 Mask

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xC7

Bits7–0: P0MASK[7:0]: Port0 Mask Value.
These bits select which Port pins will be compared to the value stored in P0MAT.
0: Corresponding P0.n pin is ignored and cannot cause a Port Match event.
1: Corresponding P0.n pin is compared to the corresponding bit in P0MAT.

## SFR Definition 18.9. P0ODEN: Port0 Overdrive Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB0

Bits7–0: High Impedance Overdrive Mode Enable Bits for P0.7–P0.0 (respectively).
Port pins configured to High-Impedance Overdrive Mode do not require additional overdrive current, although selecting this mode results in a slight increase in supply current. Port pins configured to Normal Overdrive Mode require approximately 150 µA of input overdrive current when the voltage at the pin reaches $V_{IO}$+0.7 V.
0: Corresponding P0.n pin is configured to Normal Overdrive Mode.
1: Corresponding P0.n pin is configured to High-Impedance Overdrive Mode.

## SFR Definition 18.10. P1: Port1

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| P1.7 | P1.6 | P1.5 | P1.4 | P1.3 | P1.2 | P1.1 | P1.0 | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0x90

Bits7–0: P1.[7:0]
Write - Output appears on I/O pins per Crossbar Registers.
0: Logic Low Output.
1: Logic High Output (high impedance if corresponding P1MDOUT.n bit = 0).
Read - Always reads ‘0’ if selected as analog input in register P1MDIN. Directly reads Port pin when configured as digital input.
0: P1.n pin is logic low.
1: P1.n pin is logic high.

## SFR Definition 18.11. P1MDIN: Port1 Input Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF2

Bits7–0: Analog Input Configuration Bits for P1.7–P1.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P1.n pin is configured as an analog input. **In order for the P1.n pin to be in analog input mode, there MUST be a '1' in the Port Latch register corresponding to that pin.**
1: Corresponding P1.n pin is not configured as an analog input.

## SFR Definition 18.12. P1MDOUT: Port1 Output Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA5

Bits7–0: Output Configuration Bits for P1.7–P1.0 (respectively): ignored if corresponding bit in register P1MDIN is logic 0.
0: Corresponding P1.n Output is open-drain.
1: Corresponding P1.n Output is push-pull.

## SFR Definition 18.13. P1SKIP: Port1 Skip

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD5

Bits7–0: P1SKIP[7:0]: Port1 Crossbar Skip Enable Bits.
These bits select Port pins to be skipped by the Crossbar Decoder. Port pins used as analog inputs (for ADC or Comparator) or used as special functions ($V_{REF}$ input, external oscillator circuit, CNVSTR input) should be skipped by the Crossbar.
0: Corresponding P1.n pin is not skipped by the Crossbar.
1: Corresponding P1.n pin is skipped by the Crossbar.

## SFR Definition 18.14. P1MAT: Port1 Match

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xCF

Bits7–0: P1MAT[7:0]: Port1 Match Value.
These bits control the value that unmasked P0 Port pins are compared against. A Port Match event is generated if (P1 & P1MASK) does not equal (P1MAT & P1MASK).

## SFR Definition 18.15. P1MASK: Port1 Mask

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xBF

Bits7–0: P1MASK[7:0]: Port1 Mask Value.
These bits select which Port pins will be compared to the value stored in P1MAT.
0: Corresponding P1.n pin is ignored and cannot cause a Port Match event.
1: Corresponding P1.n pin is compared to the corresponding bit in P1MAT.

## SFR Definition 18.16. P2: Port2

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| P2.7 | P2.6 | P2.5 | P2.4 | P2.3 | P2.2 | P2.1 | P2.0 | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xA0

Bits7–0: P2.[7:0]
Write - Output appears on I/O pins per Crossbar Registers.
0: Logic Low Output.
1: Logic High Output (high impedance if corresponding P2MDOUT.n bit = 0).
Read - Always reads ‘0’ if selected as analog input in register P2MDIN. Directly reads Port pin when configured as digital input.
0: P2.n pin is logic low.
1: P2.n pin is logic high.

## SFR Definition 18.17. P2MDIN: Port2 Input Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF3

Bits7–0: Analog Input Configuration Bits for P2.7–P2.0 (respectively).
Port pins configured as analog inputs have their weak pullup, digital driver, and digital receiver disabled.
0: Corresponding P2.n pin is configured as an analog input. **In order for the P2.n pin to be in analog input mode, there MUST be a '1' in the Port Latch register corresponding to that pin.**
1: Corresponding P2.n pin is not configured as an analog input.

## SFR Definition 18.18. P2MDOUT: Port2 Output Mode

| R | R | R | R | R | R | R | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA6

Bits7–0: Output Configuration Bits for P2.7–P2.0 (respectively): ignored if corresponding bit in register P2MDIN is logic 0.
0: Corresponding P2.n Output is open-drain.
1: Corresponding P2.n Output is push-pull.

## SFR Definition 18.19. P2SKIP: Port2 Skip

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD6

Bits7–0: P2SKIP[7:0]: Port2 Crossbar Skip Enable Bits.
These bits select Port pins to be skipped by the Crossbar Decoder. Port pins used as analog inputs (for ADC or Comparator) or used as special functions ($V_{REF}$ input, external oscillator circuit, CNVSTR input) should be skipped by the Crossbar.
0: Corresponding P2.n pin is not skipped by the Crossbar.
1: Corresponding P2.n pin is skipped by the Crossbar.

## Table 18.1. Port I/O DC Electrical Characteristics

$V_{IO}$ = 2.0 to 5.25 V, –40 to +85 °C unless otherwise specified. Typical values are given at 25 ºC.

| Parameters | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Output High Voltage | $I_{OH}$ = –3 mA, Port I/O push-pull | $V_{IO}$ – 0.5 | — | — | V |
| | $I_{OH}$ = –70 µA, Port I/O push-pull | $V_{IO}$ – 50 mV | — | — | |
| Output Low Voltage | **$V_{IO}$ = 2.0 V:** | | | | mV |
| | $I_{OL}$ = 70 µA | — | — | 50 | |
| | $I_{OL}$ = 8.5 mA | — | — | 800 | |
| | **$V_{IO}$ = 4.0 V:** | | | | |
| | $I_{OL}$ = 70 µA | — | — | 40 | |
| | $I_{OL}$ = 8.5 mA | — | — | 400 | |
| Input High Voltage | | $V_{IO}$ x 0.7 | — | — | V |
| Input Low Voltage | | — | — | $V_{IO}$ x 0.3 | V |
| Input Leakage Current | Weak Pullup Off | — | < 0.1 | ±1 | µA |
| Weak Pullup Impedance | | — | 120 | — | kΩ |

## NOTES:

# 19. Oscillators

C8051F41x devices include a programmable internal oscillator, an external oscillator drive circuit, and a Clock Multiplier. The internal oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 19.1. The system clock (SYSCLK) can be derived from the internal oscillator, external oscillator circuit, or smaRTClock oscillator. The clock multiplier can produce three possible base outputs which can be scaled by a programmable factor of 1, 2/3, 2/4 (or 1/2), 2/5, 2/6 (or 1/3), or 2/7: Internal Oscillator x 2, External Oscillator x 2, or External Oscillator x 4. Oscillator electrical specifications are given in Table 19.1 on page 175.

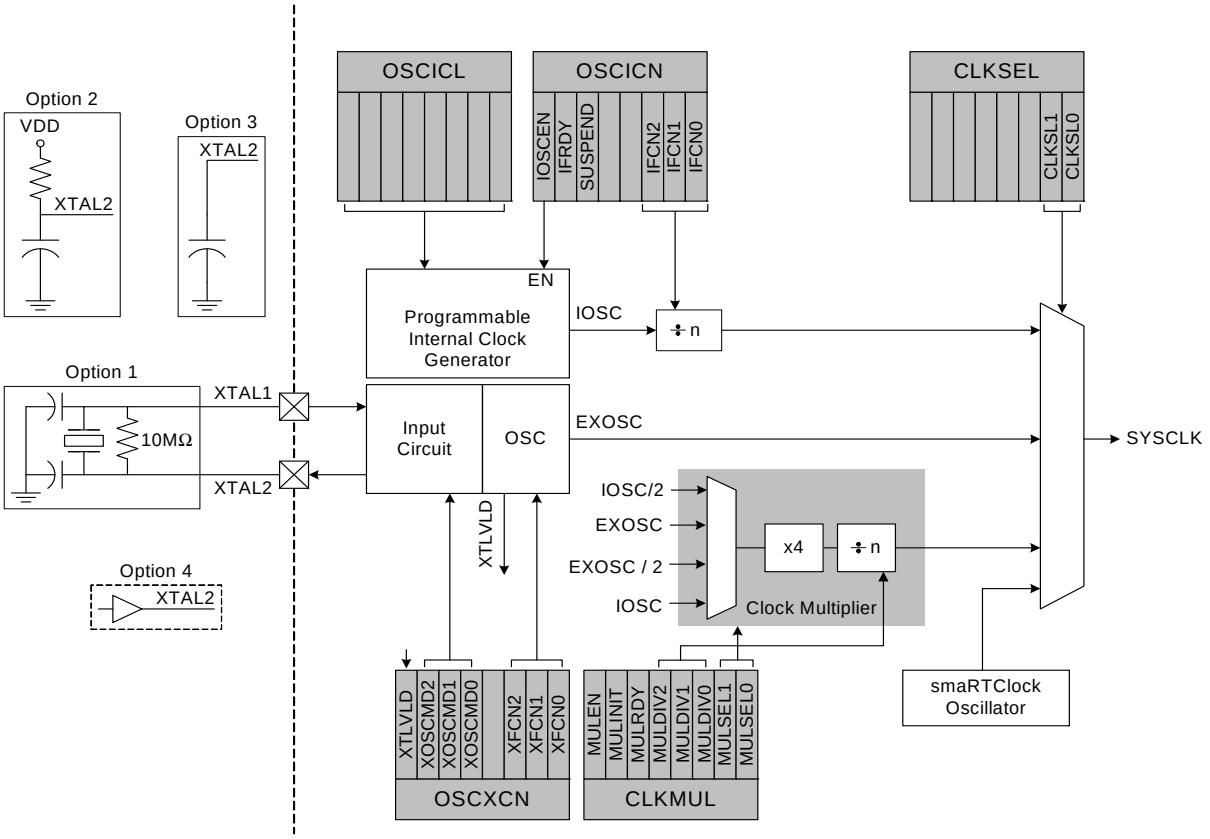



**Figure 19.1. Oscillator Diagram**

## 19.1. Programmable Internal Oscillator

All C8051F41x devices include a programmable internal oscillator that defaults as the system clock after a system reset. The internal oscillator period can be programmed via the OSCICL register, shown in SFR Definition 19.2. On C8051F41x devices, OSCICL is factory calibrated to obtain a 24.5 MHz frequency.

Electrical specifications for the precision internal oscillator are given in Table 19.1 on page 175. Note that the system clock may be derived from the programmed internal oscillator divided by 1, 2, 4, 8, 16, 32, 64, or 128 as defined by the IFCN bits in register OSCICN. The divide value defaults to 128 following a reset.

### 19.1.1. Internal Oscillator Suspend Mode

When software writes a logic 1 to SUSPEND (OSCICN.5), the internal oscillator is suspended. If the system clock is derived from the internal oscillator, the input clock to the peripheral or CIP-51 will be stopped until one of the following events occur:

- Port 0 Match Event.
- Port 1 Match Event.
- Comparator 0 enabled and output is logic 0.
- Comparator 1 enabled and output is logic 0.
- smaRTClock Oscillator Fail Event.
- smaRTClock Alarm Event.

When one of the internal oscillator awakening events occur, the internal oscillator, CIP-51, and affected peripherals resume normal operation, regardless of whether the event also causes an interrupt. The CPU resumes execution at the instruction following the write to SUSPEND.

## SFR Definition 19.1. OSCICN: Internal Oscillator Control

| R/W | R | R/W | R | R | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| IOSCEN | IFRDY | SUSPEND | - | - | IFCN2 | IFCN1 | IFCN0 | 11000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB2

Bit7: IOSCEN: Internal Oscillator Enable Bit.
0: Internal Oscillator Disabled.
1: Internal Oscillator Enabled.

Bit6: IFRDY: Internal Oscillator Frequency Ready Flag.
0: Internal Oscillator is not running at programmed frequency.
1: Internal Oscillator is running at programmed frequency.

Bit5: SUSPEND: Internal Oscillator Suspend Enable Bit.
Setting this bit to logic 1 places the internal oscillator in SUSPEND mode. The internal oscillator resumes operation when one of the SUSPEND mode awakening events occur.

Bits4–3: UNUSED. Read = 00b, Write = don't care.

Bits2–0: IFCN2–0: Internal Oscillator Frequency Control Bits.
000: SYSCLK derived from Internal Oscillator divided by 128 (default).
001: SYSCLK derived from Internal Oscillator divided by 64.
010: SYSCLK derived from Internal Oscillator divided by 32.
011: SYSCLK derived from Internal Oscillator divided by 16.
100: SYSCLK derived from Internal Oscillator divided by 8.
101: SYSCLK derived from Internal Oscillator divided by 4.
110: SYSCLK derived from Internal Oscillator divided by 2.
111: SYSCLK derived from Internal Oscillator divided by 1.

## SFR Definition 19.2. OSCICL: Internal Oscillator Calibration

| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | OSCICL | | | | | | | Varies |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB3

Bit7: UNUSED. Read = 0. Write = don’t care.

Bits 6–0: OSCICL: Internal Oscillator Calibration Register.
This register determines the internal oscillator period. On C8051F41x devices, the reset value is factory calibrated to generate an internal oscillator frequency of 24.5 MHz.

## 19.2. External Oscillator Drive Circuit

The external oscillator circuit may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. For a crystal or ceramic resonator configuration, the crystal/resonator must be wired across the XTAL1 and XTAL2 pins as shown in Option 1 of Figure 19.1. A 10 MΩ resistor also must be wired across the XTAL1 and XTAL2 pins for the crystal/resonator configuration. In RC, capacitor, or CMOS clock configuration, the clock source should be wired to the XTAL2 pin as shown in Option 2, 3, or 4 of Figure 19.1. The type of external oscillator must be selected in the OSCXCN register, and the frequency control bits (XFCN) must be selected appropriately (see SFR Definition 19.3. OSCXCN: External Oscillator Control).

**Important Note on External Oscillator Usage:** Port pins must be configured when using the external oscillator circuit. When the external oscillator drive circuit is enabled in crystal/resonator mode, Port pins P1.0 and P1.1 are used as XTAL1 and XTAL2 respectively. When the external oscillator drive circuit is enabled in capacitor, RC, or CMOS clock mode, Port pin P1.1 is used as XTAL2. The Port I/O Crossbar should be configured to skip the Port pins used by the oscillator circuit; see **Section "18.1. Priority Crossbar Decoder" on page 149** for Crossbar configuration. Additionally, when using the external oscillator circuit in crystal/resonator, capacitor, or RC mode, the associated Port pins should be configured as **analog inputs** (with '1's in the corresponding Port Latch). In CMOS clock mode, the associated pin should be configured as a **digital input**. See **Section "18.2. Port I/O Initialization" on page 151** for details on Port input mode selection.

The frequency of the external oscillator can be measured with respect to the smaRTClock Oscillator using Timer 2 or Timer 3. Section "**24.2.3. External/smaRTClock Capture Mode**" on page **241** shows how this can be accomplished.

### 19.2.1. Clocking Timers Directly Through the External Oscillator

The external oscillator source divided by eight is a clock option for the timers (**Section "24. Timers" on page 231**) and the Programmable Counter Array (PCA) (**Section "25. Programmable Counter Array (PCA0)" on page 249**). When the external oscillator is used to clock these peripherals, but is not used as the system clock, the external oscillator frequency must be less than or equal to the system clock frequency. In this configuration, the clock supplied to the peripheral (external oscillator / 8) is synchronized with the system clock; the jitter associated with this synchronization is limited to ±0.5 system clock cycles.

### 19.2.2. External Crystal Example

If a crystal or ceramic resonator is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 19.1, Option 1. The External Oscillator Frequency Control value (XFCN) should be chosen from the Crystal column of the table in SFR Definition 19.3. For example, a 12 MHz crystal requires an XFCN setting of 111b.

When the crystal oscillator is first enabled, the oscillator amplitude detection circuit requires a settling time to achieve proper bias. Introducing a delay of 1 ms between enabling the oscillator and checking the XTLVLD bit will prevent a premature switch to the external oscillator as the system clock. Switching to the external oscillator before the crystal oscillator has stabilized can result in unpredictable behavior. The recommended procedure is:

Step 1. Force the XTAL1 and XTAL2 pins low by writing 0's to the port latch.
Step 2. Configure XTAL1 and XTAL2 as analog inputs.
Step 3. Release the crystal pins by writing ‘1's to the port latch.
Step 4. Enable the external oscillator.
Step 5. Wait at least 1 ms.
Step 6. Poll for XTLVLD => '1'.
Step 7. Switch the system clock to the external oscillator.

**Note:** Tuning-fork crystals may require additional settling time before XTLVLD returns a valid result.

The capacitors shown in the external crystal configuration provide the load capacitance required by the crystal for correct oscillation. These capacitors are "in series" as seen by the crystal and "in parallel" with the stray capacitance of the XTAL1 and XTAL2 pins.

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

For example, a tuning-fork crystal of 32.768 kHz with a recommended load capacitance of 12.5 pF should use the configuration shown in Figure 19.1, Option 1. The total value of the capacitors and the stray capacitance of the XTAL pins should equal 25 pF. With a stray capacitance of 3 pF per pin, the 22 pF capacitors yield an equivalent capacitance of 12.5 pF across the crystal, as shown in Figure 19.2.

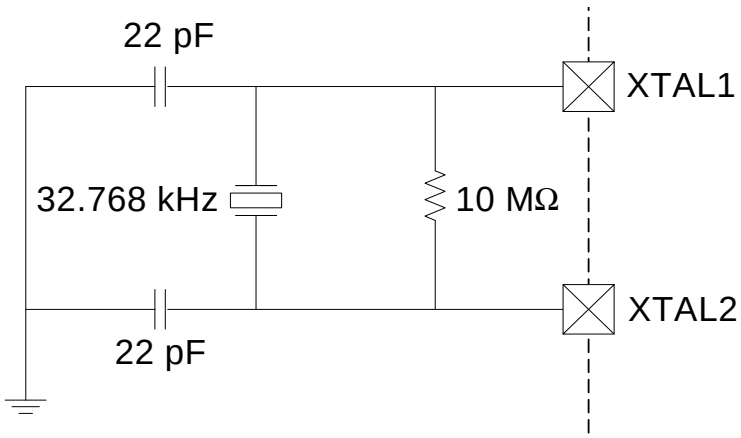



**Figure 19.2. 32.768 kHz External Crystal Example**

**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.

### 19.2.3. External RC Example

If an RC network is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 19.1, Option 2. 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. 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. If the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

$f = 1.23( 10^3 ) / RC = 1.23 ( 10^3 ) / [ 246 \times 50 ] = 0.1 \text{ MHz} = 100 \text{ kHz}$

Referring to the table in SFR Definition 19.3, the required XFCN setting is 010b. Programming XFCN to a higher setting in RC mode will improve frequency accuracy at a slightly increased external oscillator supply current.

### 19.2.4. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 19.1, Option 3. 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. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, select the frequency of oscillation and calculate the capacitance to be used from the equations below. Assume $V_{DD}$ = 2.0 V and f = 75 kHz:

$f = KF / (C \times V_{DD})$
$0.075 \text{ MHz} = KF / (C \times 2.0)$

Since the frequency of roughly 75 kHz is desired, select the K Factor from the table in SFR Definition 19.3 as KF = 7.7:

$0.075 \text{ MHz} = 7.7 / (C \times 2.0)$

$C \times 2.0 = 7.7 / 0.075 \text{ MHz}$

$C = 102.6 / 2.0 \text{ pF} = 51.3 \text{ pF}$

Therefore, the XFCN value to use in this example is 010b.

## SFR Definition 19.3. OSCXCN: External Oscillator Control

| R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| XTLVLD | XOSCMD2 | XOSCMD1 | XOSCMD0 | Reserved | XFCN2 | XFCN1 | XFCN0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xB1

Bit7: XTLVLD: Crystal Oscillator Valid Flag. (Read only when XOSCMD = 11x.)
0: Crystal Oscillator is unused or not yet stable.
1: Crystal Oscillator is running and stable.

Bits6–4: XOSCMD2–0: External Oscillator Mode Bits.
00x: External Oscillator circuit off.
010: External CMOS Clock Mode.
011: External CMOS Clock Mode with divide by 2 stage.
100: RC Oscillator Mode.
101: Capacitor Oscillator Mode.
110: Crystal Oscillator Mode.
111: Crystal Oscillator Mode with divide by 2 stage.

Bit3: RESERVED. Read = 0b; Must write 0b.

Bits2–0: XFCN2–0: External Oscillator Frequency Control Bits.
000-111: See table below:

| XFCN | Crystal (XOSCMD = 11x) | RC (XOSCMD = 10x) | C (XOSCMD = 10x) |
|---|---|---|---|
| 000 | f ≤ 20 kHz | f ≤ 25 kHz | K Factor = 0.87 |
| 001 | 20 kHz < f ≤ 58 kHz | 25 kHz < f ≤ 50 kHz | K Factor = 2.6 |
| 010 | 58 kHz < f ≤ 155 kHz | 50 kHz < f ≤ 100 kHz | K Factor = 7.7 |
| 011 | 155 kHz < f ≤ 415 kHz | 100 kHz < f ≤ 200 kHz | K Factor = 22 |
| 100 | 415 kHz < f ≤ 1.1 MHz | 200 kHz < f ≤ 400 kHz | K Factor = 65 |
| 101 | 1.1 MHz < f ≤ 3.1 MHz | 400 kHz < f ≤ 800 kHz | K Factor = 180 |
| 110 | 3.1 MHz < f ≤ 8.2 MHz | 800 kHz < f ≤ 1.6 MHz | K Factor = 664 |
| 111 | 8.2 MHz < f ≤ 25 MHz | 1.6 MHz < f ≤ 3.2 MHz | K Factor = 1590 |

**Crystal Mode** (Circuit from Figure 19.1, Option 1; XOSCMD = 11x)
Choose XFCN value to match crystal or resonator frequency.

**RC Mode** (Circuit from Figure 19.1, Option 2; XOSCMD = 10x)
Choose XFCN value to match frequency range:
**$f = 1.23(10^3) / (R \times C)$**, where
f = frequency of clock in MHz
C = capacitor value in pF
R = Pullup resistor value in kΩ

**C Mode** (Circuit from Figure 19.1, Option 3; XOSCMD = 10x)
Choose K Factor (KF) for the oscillation frequency desired:
**$f = KF / (C \times V_{DD})$**, where
f = frequency of clock in MHz
C = capacitor value the XTAL2 pin in pF
$V_{DD}$ = Power Supply on MCU in volts

## 19.3. Clock Multiplier

The Clock Multiplier generates an output clock which is 4 times the input clock frequency scaled by a programmable factor of 1, 2/3, 2/4 (or 1/2), 2/5, 2/6 (or 1/3), or 2/7. The Clock Multiplier's input can be selected from the external oscillator, or the internal or external oscillators divided by 2. This produces three possible base outputs which can be scaled by a programmable factor: Internal Oscillator x 2, External Oscillator x 2, or External Oscillator x 4. See **Section 19.4** for details on system clock selection.

The Clock Multiplier is configured via the CLKMUL register (SFR Definition 19.4). The procedure for configuring and enabling the Clock Multiplier is as follows:

1. Reset the Multiplier by writing 0x00 to register CLKMUL.
2. Select the Multiplier input source via the MULSEL bits.
3. Select the Multiplier output scaling factor via the MULDIV bits
4. Enable the Multiplier with the MULEN bit (CLKMUL | = 0x80).
5. Delay for >5 µs.
6. Initialize the Multiplier with the MULINIT bit (CLKMUL | = 0xC0).
7. Poll for MULRDY => '1'.

**Important Note: When using an external oscillator as the input to the Clock Multiplier, the external source must be enabled and stable before the Multiplier is initialized. See Section 19.4 for details on selecting an external oscillator source.**

The Clock Multiplier allows faster operation of the CIP-51 core and is intended to generate an output frequency between 25 and 50 MHz. The clock multiplier can also be used with slow input clocks. However, if the clock is below the minimum Clock Multiplier input frequency ($FCM_{min}$) specified in Table 19.1, the generated clock will consist of four fast pulses followed by a long delay until the next input clock rising edge. The average frequency of the output is equal to 4x the input, but the instantaneous frequency may be faster. See Figure 19.3 for more information.

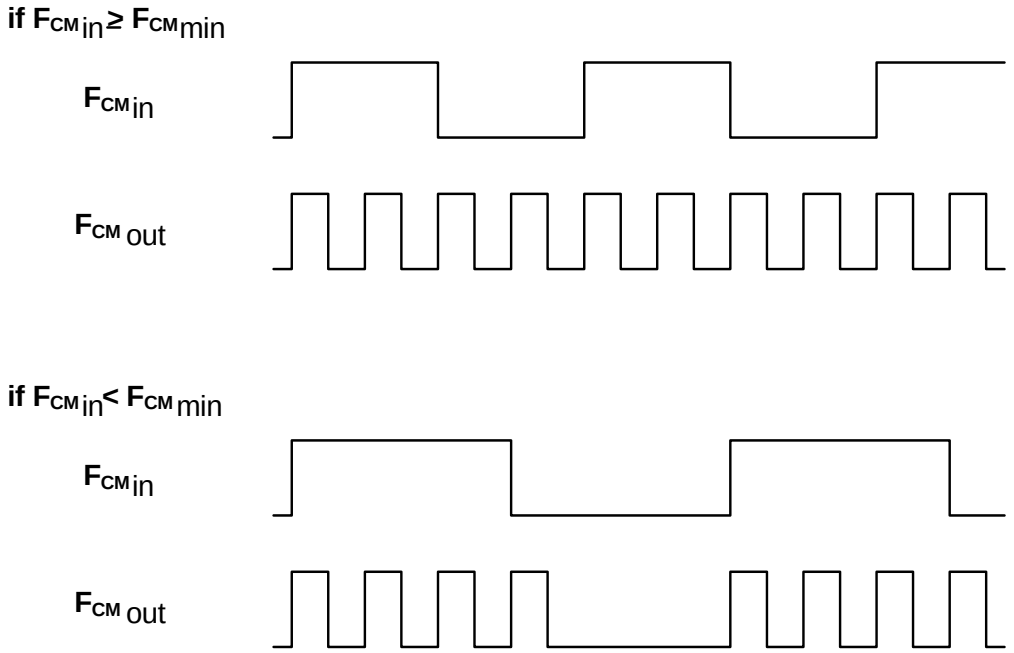



**Figure 19.3. Example Clock Multiplier Output**

## SFR Definition 19.4. CLKMUL: Clock Multiplier Control

| R/W | R/W | R | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| MULEN | MULINIT | MULRDY | MULDIV | | | MULSEL | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAB

**Note:** The maximum SYSCLK is 50 MHz, so the Clock Multiplier output should be scaled accordingly.

Bit7: MULEN: Clock Multiplier Enable
0: Clock Multiplier disabled.
1: Clock Multiplier enabled.

Bit6: MULINIT: Clock Multiplier Initialize
This bit should be a ‘0’ when the Clock Multiplier is enabled. Once enabled, writing a ‘1’ to this bit will initialize the Clock Multiplier. The MULRDY bit reads ‘1’ when the Clock Multiplier is stabilized.

Bit5: MULRDY: Clock Multiplier Ready
This read-only bit indicates the status of the Clock Multiplier.
0: Clock Multiplier not ready.
1: Clock Multiplier ready (locked).

Bits4–2: MULDIV: Clock Multiplier Output Scaling Factor
These bits scale the Clock Multiplier output.
000: Clock Multiplier Output scaled by a factor of 1.
001: Clock Multiplier Output scaled by a factor of 1.
010: Clock Multiplier Output scaled by a factor of 1.
011: Clock Multiplier Output scaled by a factor of 2/3*.
100: Clock Multiplier Output scaled by a factor of 2/4 (or 1/2).
101: Clock Multiplier Output scaled by a factor of 2/5*.
110: Clock Multiplier Output scaled by a factor of 2/6 (or 1/3).
111: Clock Multiplier Output scaled by a factor of 2/7*.
***Note:** The Clock Multiplier Output duty cycle is not 50% for these settings.

Bits1–0: MULSEL: Clock Multiplier Input Select
These bits select the clock supplied to the Clock Multiplier.

| MULSEL | Selected Input Clock | Clock Multiplier Output for MULDIV = 000b |
|---|---|---|
| 00 | Internal Oscillator / 2 | Internal Oscillator x 2 |
| 01 | External Oscillator | External Oscillator x 4 |
| 10 | External Oscillator / 2 | External Oscillator x 2 |
| 11 | Internal Oscillator | Internal Oscillator x 4 |

## 19.4. System Clock Selection

The internal oscillator requires little start-up time and may be selected as the system clock immediately following the OSCICN write that enables the internal oscillator. External crystals and ceramic resonators typically require a start-up time before they are settled and ready for use. The Crystal Valid Flag (XTLVLD in register OSCXCN) is set to '1' by hardware when the external oscillator is settled. **To avoid reading a false XTLVLD, in crystal mode software should delay at least 1 ms between enabling the external oscillator and checking XTLVLD.** RC and C modes typically require no startup time.

The CLKSL[1:0] bits in register CLKSEL select which oscillator source is used as the system clock. CLKSL[1:0] must be set to 01b for the system clock to run from the external oscillator; however the external oscillator may still clock certain peripherals (timers, PCA) when another oscillator is selected as the system clock. The system clock may be switched on-the-fly between the internal oscillator, external oscillator, smaRTClock oscillator, and Clock Multiplier, as long as the selected clock source is enabled and has settled.

### SFR Definition 19.5. CLKSEL: Clock Select

| R | R | R/W | R/W | R | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| - | - | CLKDIV | | - | Reserved | CLKSL | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA9

Bits7–6: Unused. Read = 00b; Write = don't care.
Bits5–4: CLKDIV1–0: Output /SYSCLK Divide Value
These bits can be used to pre-divide the /SYSCLK output before it is sent to a port pin through the Crossbar.
00: Output will be SYSCLK.
01: Output will be SYSCLK/2.
10: Output will be SYSCLK/4.
11: Output will be SYSCLK/8.
Bit3: Unused. Read = 0b; Write = don't care.
Bit2: Reserved. Read = 0b; Must write 0b.
Bits1–0: CLKSL1–0: System Clock Select
These bits select the system clock source.

| CLKSL | Selected Clock |
|---|---|
| 00 | Internal Oscillator (as determined by the IFCN bits in register OSCICN) |
| 01 | External Oscillator |
| 10 | Clock Multiplier |
| 11 | smaRTClock Oscillator |

## Table 19.1. Oscillator Electrical Characteristics

–40 to +85 °C unless otherwise specified.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Internal Oscillator Frequency | Reset Frequency | 24 | 24.5 | 25 | MHz |
| Internal Oscillator Supply Current (from $V_{DD}$) | OSCICN.7 = 1 | — | 400 | — | µA |
| Minimum Clock Multiplier Input Frequency ($FCM_{min}$) | T = 25 °C | — | 1.6 | — | MHz |

## NOTES:

## 20. smaRTClock (Real Time Clock)

C8051F41x devices include a low power smaRTClock Peripheral (Real Time Clock). The smaRTClock has a dedicated 32 kHz oscillator that can be configured for use with or without a crystal, a 47-bit smaRTClock timer with alarm, a backup supply regulator and 64 bytes of battery-backed SRAM. When the backup supply voltage ($V_{RTC\text{-}BACKUP}$) is powered, the smaRTClock peripheral remains fully functional if the core supply voltage ($V_{DD}$) is lost.

The smaRTClock allows a maximum of 137 year 47-bit independent time-keeping when used with a 32.768 kHz Watch Crystal and backup supply voltage of at least 1V. The switchover logic powers smaRTClock from the backup supply when the voltage at $V_{RTC\text{-}BACKUP}$ is greater than $V_{DD}$. The smaRTClock Alarm and Missing Clock Detector can interrupt the CIP-51, wake the internal oscillator from SUSPEND mode, or generate a device reset if the smaRTClock timer reaches a pre-set value or the oscillator stops.

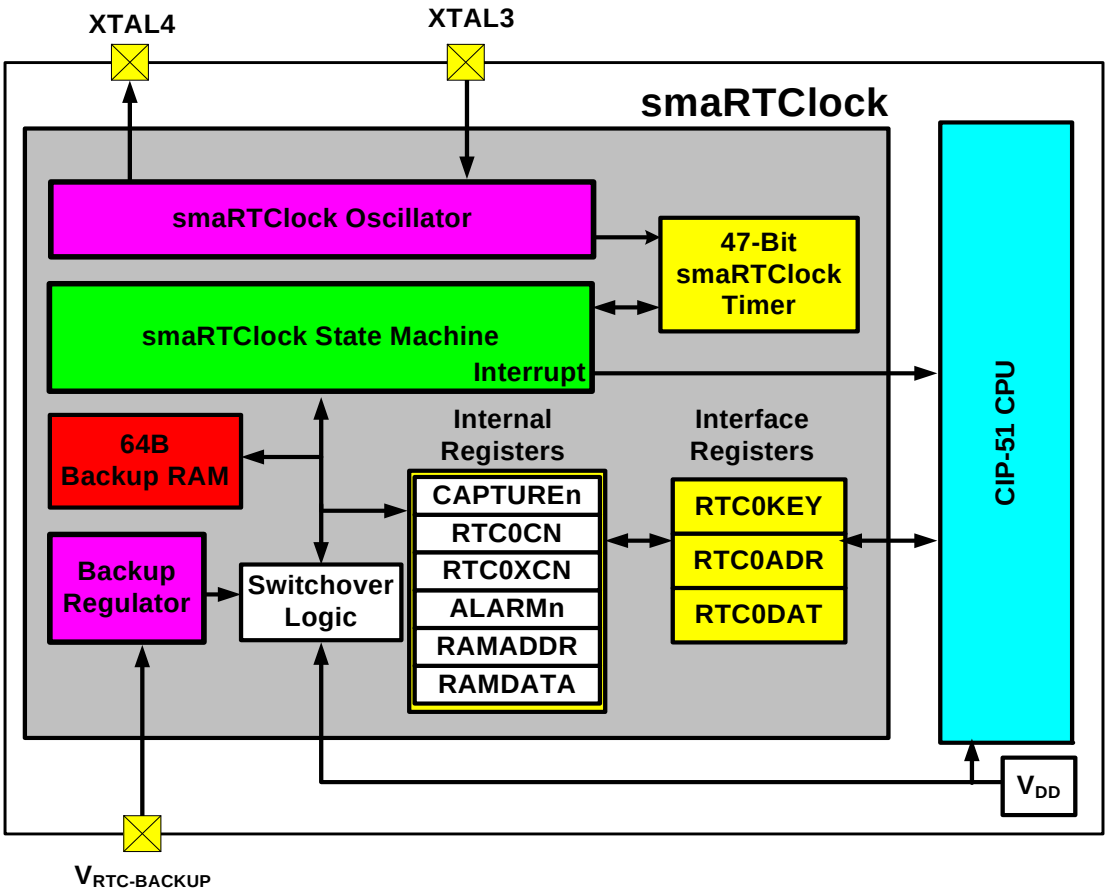



**Figure 20.1. smaRTClock Block Diagram**

## 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.

### 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 invalid read or write is attempted, further writes and reads to RTC0ADR and RTC0DAT will be disabled until the next system reset. Once the smaRTClock interface is unlocked, software may perform accesses of the smaRTClock registers until an invalid access, the interface is locked, or a system reset.

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. Prior to each read or write, BUSY (RTC0ADR.7) should be checked to make sure the smaRTClock Interface is not busy performing another 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.

Step 1. Poll BUSY (RTC0ADR.7) until it returns a '0'.
Step 2. Write 0x06 to RTC0ADR. This selects the internal RTC0CN register at smaRTClock Address 0x06.
Step 3. Write 0x00 to RTC0DAT. This operation writes 0x00 to the internal RTC0CN register.

An 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.

Step 1. Poll BUSY (RTC0ADR.7) until it returns a '0'.
Step 2. Write 0x06 to RTC0ADR. This selects the internal RTC0CN register at smaRTClock Address 0x06.
Step 3. Write '1' to BUSY. This initiates the transfer of data from RTC0CN to RTC0DAT.
Step 4. Poll BUSY (RTC0ADR.7) until it returns a '0'.
Step 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. 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 begin-

ning of each series of consecutive reads. Software must check if the smaRTClock Interface is busy prior to reading RTC0DAT. Autoread is enabled by setting AUTORD (RTC0ADR.6) to logic 1.

### 20.1.4. RTC0ADR Autoincrement Feature

For ease of reading and writing the 48-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.

**Table 20.1. smaRTClock Internal Registers**

| smaRTClock Address | smaRTClock Register | Register Name | Description |
|---|---|---|---|
| 0x00 - 0x05 | CAPTUREn | smaRTClock Capture Registers | Six Registers used for setting the 47-bit smaRTClock timer or reading its current value. The LSB of CAPTURE0 is not used. |
| 0x06 | RTC0CN | smaRTClock Control Register | Controls the operation of the smaRTClock State Machine. |
| 0x07 | RTC0XCN | smaRTClock Oscillator Control Register | Controls the operation of the smaRTClock Oscillator. |
| 0x08–0x0D | ALARMn | smaRTClock Alarm Registers | Six registers used to set or read the 47-bit smaRTClock alarm value. The LSB of ALARM0 is not used. |
| 0x0E | RAMADDR | smaRTClock Backup RAM Indirect Address Register | Used as an index to the 64 byte smaRTClock backup RAM. |
| 0x0F | RAMDATA | smaRTClock Backup RAM Indirect Data Register | Used to read or write the byte pointed to by RAMADDR. |

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

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAE

Bits 7–0: RTC0STATE. smaRTClock State Bits

**Read:**

0x00: smaRTClock Interface is locked.

0x01: smaRTClock Interface is locked. First key code (0xA5) has been written, waiting for second key code.

0x02: smaRTClock Interface is unlocked. First and second key codes (0xA5, 0xF1) have been written.

0x03: smaRTClock Interface is disabled until the next system reset.

**Write:**

When RTC0STATE = 0x00 (locked), writing 0xA5 followed by 0xF1 unlocks the smaRTClock Interface.

When RTC0STATE = 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.

When RTC0STATE = 0x02 (unlocked), any write to RTC0KEY will lock the smaRTClock Interface.

When RTC0STATE = 0x03 (disabled), writes to RTC0KEY have no effect.

## SFR Definition 20.2. RTC0ADR: smaRTClock Address

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| BUSY | AUTORD | VREGEN | SHORT | RTC0ADDR | | | | Variable |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAC

Bit 7: BUSY: smaRTClock Interface Busy bit.
Writing a ‘1’ to this bit initiates a smaRTClock indirect read operation. This bit is automatically cleared by hardware when the operation is complete.
0: smaRTClock Interface is not busy.
1: smaRTClock Interface is busy performing a read or write operation.

Bit 6: AUTORD: smaRTClock Interface Auto Read Enable.
0: BUSY must be written manually for each smaRTClock indirect read operation.
1: The next smaRTClock indirect read operation is initiated when RTC0DAT is read by software.

Bit 5: VREGEN: Backup Supply Voltage Regulator Enable.
This bit is automatically set to 1b when $V_{RTC\text{-}BACKUP} > V_{DD}$.
0: Backup Supply Voltage Regulator Disabled (smaRTClock powered from $V_{DD}$).
1: Force Backup Supply Voltage Regulator Enabled (smaRTClock powered from $V_{RTC\text{-}BACKUP}$).

Bit 4: SHORT: Short Read/Write Timing Enable.
0: smaRTClock reads and writes are 4 system clocks wide.
1: smaRTClock reads and writes are 1 system clock wide.
Note: Increasing the speed of the smaRTClock reads and writes may also slightly increase power consumption.

Bits 3–0: RTC0ADDR: smaRTClock Address Bits
These bits select the smaRTClock internal register that is targeted by reads/writes to RTC0DAT.

| RTC0ADDR | smaRTClock Internal Register |
|---|---|
| 0000 | CAPTURE0 |
| 0001 | CAPTURE1 |
| 0010 | CAPTURE2 |
| 0011 | CAPTURE3 |
| 0100 | CAPTURE4 |
| 0101 | CAPTURE5 |
| 0110 | RTC0CN |
| 0111 | RTC0XCN |
| 1000 | ALARM0 |
| 1001 | ALARM1 |
| 1010 | ALARM2 |
| 1011 | ALARM3 |
| 1100 | ALARM4 |
| 1101 | ALARM5 |
| 1110 | RAMADDR |
| 1111 | RAMDATA |

**Note: The RTC0ADDR bits increment after each indirect read/write operation that targets a CAPTUREn or ALARMn internal register.**

**SFR Definition 20.3. RTC0DAT: smaRTClock Data**

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Variable |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xAD

**Note:** Software should avoid read modify write instructions when writing values to RTC0DAT.

Bits 7–0: RTC0DAT. smaRTClock Data Bits
Holds data transferred to/from the internal smaRTClock register selected by RTC0ADR.

## 20.2. smaRTClock Clocking Sources

The smaRTClock peripheral is clocked from its own timebase, independent of SYSCLK. The RTCCLK timebase is derived from the smaRTClock oscillator circuit. This oscillator has two modes of operation: Crystal Mode, and Self-Oscillate Mode. The oscillation frequency is 32.768 kHz in Crystal Mode and can be configured to roughly 20 kHz or 40 kHz in Self-Oscillate Mode. The frequency of the smaRTClock oscillator can be measured with respect to another oscillator using Timer 2 or Timer 3. Section "**24.2.3. External/smaRTClock Capture Mode**" on page **241** shows how this can be accomplished.

**Note:** The smaRTClock clock can be selected as system clock and routed to a port pin. See SFR Definition 19.5. "CLKSEL: Clock Select" on page 174 and Section "**18. Port Input/Output**" on page **147**.

### 20.2.1. Using the smaRTClock Oscillator in Crystal Mode

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:

Step 1. Set smaRTClock to Crystal Mode (XMODE = 1).
Step 2. *Optional.* Enable Automatic Gain Control (AGCEN = 1).
Step 3. *Optional.* Enable smaRTClock Bias Doubling (BIASX2 = 1).
Step 4. Enable power to the smaRTClock oscillator circuit (RTC0EN = 1).
Step 5. Poll the smaRTClock Clock Valid Bit (CLKVLD) until the crystal oscillator stabilizes.
Step 6. *Optional*. Clear BIASX2 to '0' after the oscillator stabilizes to conserve power.

### 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 following steps show how to configure smaRTClock for use in Self-Oscillate Mode:

Step 1. Set smaRTClock to Self-Oscillate Mode (XMODE = 0).
Step 2. Set the desired oscillation frequency:
For oscillation at about 20 kHz, set BIASX2 = 0.
For oscillation at about 40 kHz, set BIASX2 = 1.
Step 3. The oscillator starts oscillating instantaneously.

### 20.2.3. Automatic Gain Control (Crystal Mode Only)

Automatic Gain Control is enabled by setting AGCEN (RTC0XCN.7) to a logic 1. When enabled, the smaRTClock oscillator trims the oscillation amplitude to save power. This mode is useful for preserving battery life in systems where oscillator performance is not critical and external conditions are stable.

**Note:** Setting the AGCEN to a logic 1 in self-oscillator mode can lead to drastic changes in the smaRTClock oscillator frequency.

### 20.2.4. smaRTClock Bias Doubling

The smaRTClock Bias Doubling is enabled by setting BIASX2 (RTC0XCN.5) to 1b. When enabled, the bias current to smaRTClock is doubled allowing for more robust oscillator performance. When the smaRTClock oscillator is in Self-Oscillate mode, the oscillation frequency is increased from 20 to 40 kHz. When operating in Crystal Mode, the oscillator is less likely to be affected by external conditions when BIASX2 = ‘1’. Enabling Bias Doubling increases the power consumption of smaRTClock; therefore, it is not recommended for use in power-critical systems.

### 20.2.5. smaRTClock Missing Clock Detector

The smaRTClock Missing Clock Detector is a one-shot circuit enabled by setting MCLKEN (RTC0CN.6) to a logic 1. When the smaRTClock Missing Clock Detector is enabled, OSCFAIL (RTC0CN.5) is set by hardware if RTCCLK remains high or low for more than 50 µs. A smaRTClock Missing Clock detector timeout triggers three events:

1. Awakening the internal oscillator from Suspend Mode.
2. smaRTClock Interrupt (If the smaRTClock Interrupt is enabled).
3. MCU reset (If smaRTClock is enabled as a reset source).

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

## Internal Register Definition 20.4. RTC0CN: smaRTClock Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| RTC0EN | MCLKEN | OSCFAIL | RTC0TR | RTC0AEN | ALRM | RTC0SET | RTC0CAP | Variable |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | smaRTClock Address: |
| Note: This register is not an SFR. It can only be accessed indirectly through RTC0ADR and RTC0DAT. | | | | | | | | 0x06 |

Bit 7: RTC0EN: smaRTClock Enable Bit.
0: smaRTClock bias and crystal oscillator disabled. smaRTClock is powered from $V_{DD}$ only.
1: smaRTClock bias and crystal oscillator enabled. smaRTClock can switch to the backup battery if $V_{DD}$ fails.

Bit 6: MCLKEN: smaRTClock Missing Clock Detector Enable Bit.
When enabled, the smaRTClock missing clock detector sets the OSCFAIL bit if the smaRT-Clock clock frequency falls below approximately 20 kHz.
0: smaRTClock missing clock detector disabled.
1: smaRTClock missing clock detector enabled.

Bit 5: OSCFAIL: smaRTClock Clock Fail Flag.
Set by hardware when a missing clock detector timeout occurs. When the smaRTClock Interrupt is enabled, setting this bit causes the CPU to vector to the smaRTClock interrupt service routine. This bit is not automatically cleared by hardware.

Bit 4: RTC0TR: smaRTClock Timer Run Control.
0: smaRTClock timer holds its current value.
1: smaRTClock timer increments every smaRTClock clock period.

Bit 3: RTC0AEN: smaRTClock Alarm Enable.
0: smaRTClock alarm events disabled.
1: smaRTClock alarm events enabled.

Bit 2: ALRM: smaRTClock Alarm Event Flag.
Set by hardware when the smaRTClock timer value is **greater than or equal to** the value of the ALARMn registers. When the smaRTClock Interrupt is enabled, setting this bit causes the CPU to vector to the smaRTClock interrupt service routine. This bit is not automatically cleared by hardware.

Bit 1: RTC0SET: smaRTClock Set Bit.
Writing a ‘1’ to this bit causes the 47-bit value in CAPTUREn registers to be transferred to the smaRTClock timer. This bit is automatically cleared by hardware once the transfer is complete.

Bit 0: RTC0CAP: smaRTClock Capture Bit.
Writing a ‘1’ to this bit causes the 47-bit smaRTClock timer value to be transferred to the CAPTUREn registers. This bit is automatically cleared by hardware once the transfer is complete.

### Internal Register Definition 20.5. RTC0XCN: smaRTClock Oscillator Control

| R/W | R/W | R/W | R | R | R | R | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| AGCEN | XMODE | BIASX2 | CLKVLD | - | - | - | VBATEN | Variable |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | smaRTClock Address: |

Note: This register is not an SFR. It can only be accessed indirectly through RTC0ADR and RTC0DAT. 0x07

Bit 7: AGCEN: Crystal Oscillator Automatic Gain Control Enable Bit (Crystal Mode only).
0: Automatic Gain Control disabled.
1: Automatic Gain Control enabled.

Bit 6: XMODE: smaRTClock Mode Select Bit.
This bit selects whether smaRTClock will be used with or without a crystal.
0: smaRTClock is configured to Self-Oscillate Mode.
1: smaRTClock is configured to Crystal Mode.

Bit 5: BIASX2: smaRTClock Bias Double Enable Bit.
0: smaRTClock Bias Current Doubling is disabled.
1: smaRTClock Bias Current Doubling is enabled.

Bit 4: CLKVLD: smaRTClock Clock Valid Bit.
Set by hardware when the smaRTClock crystal oscillator is nearly stable. This bit always reads 1b when smaRTClock is used in Self-Oscillate Mode (XMODE = 0). This bit should be checked at least 1 ms after enabling the smaRTClock oscillator circuit and should not be used for an oscillator fail detect (use OSCFAIL in RTC0CN instead).

Bits 3–1: UNUSED. Read = 000b. Write = don't care.

Bit 0: VBATEN: smaRTClock $V_{BAT}$ Indicator.
Note: This bit always reads 1b when smaRTClock is disabled (RTC0EN = 0).
For smaRTClock enabled (RTC0EN = 1):
0: smaRTClock is powered from $V_{DD}$.
1: smaRTClock is powered from the $V_{RTC\text{-}BACKUP}$ supply.

## 20.3. smaRTClock Timer and Alarm Function

The smaRTClock timer is a 47-bit counter that, when running (RTC0TR = 1), is incremented every RTC-CLK cycle. The timer has an alarm function that can be set to generate an interrupt, reset the MCU, or release the internal oscillator from Suspend Mode at a specific time.

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

The 47-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:

Step 1. Write the desired 47-bit set value to the CAPTUREn registers (the LSB of CAPTURE0 is not used).
Step 2. Write '1' to RTC0SET. This will transfer the contents of the CAPTUREn registers to the timer.
Step 3. Operation is complete when RTC0SET is cleared to '0' by hardware.

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

Step 1. Write ‘1’ to RTC0CAP. This will transfer the contents of the timer to the CAPTUREn registers (the LSB of the smaRTClock timer will be found in CAPTURE0.1).
Step 2. Poll RTC0CAP until it is cleared to ‘0’ by hardware.
Step 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 47-bit value of smaRTClock Timer to the value of the ALARMn registers. An alarm event is triggered if the smaRTClock timer is **greater than or equal to** the ALARMn registers. If the smaRTClock Interrupt is enabled, the CIP-51 will vector to the smaRTClock Interrupt Service Routine when an alarm event occurs. If smaRTClock is enabled as a reset source, the MCU will be reset when an alarm event occurs. Also, the internal oscillator will awaken from suspend mode on a smaRTClock alarm event.

The following steps can be used to set up a smaRTClock Alarm:

Step 1. Disable smaRTClock Alarm Events (RTC0AEN = 0).
Step 2. Set the ALARMn registers to the desired value.
Step 3. Enable smaRTClock Alarm Events (RTC0AEN = 1).

**Note:** When an alarm event occurs and smaRTClock interrupts are enabled, software should clear the ALRM bit and set the ALARM5-0 registers to the maximum possible value to avoid continuous alarm interrupts.

### Internal Register Definition 20.6. CAPTUREn: smaRTClock Timer Capture

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

smaRTClock Addresses: CAPTURE0: 0x00; CAPTURE1: 0x01; CAPTURE2: 0x02; CAPTURE3: 0x03; CAPTURE4: 0x04; CAPTURE5: 0x05
Note: These registers are not SFRs. They can only be accessed indirectly through RTC0ADR and RTC0DAT.

Bits 7–0: CAPTUREn: smaRTClock Set/Capture Value.
These 6 registers (CAPTURE5–CAPTURE0) are used to read or set the 47-bit smaRTClock timer. Data is transferred to or from the smaRTClock timer when the RTC0SET or RTC0CAP bits are set.

**Note:** The LSB of CAPTURE0 is not used. The LSB of the 47-bit smaRTClock timer will appear in CAPTURE0.1.

### Internal Register Definition 20.7. ALARMn: smaRTClock Alarm

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 11111111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

smaRTClock Addresses: ALARM0: 0x08; ALARM1: 0x09; ALARM2: 0x0A; ALARM3: 0x0B; ALARM4: 0x0C; ALARM5: 0x0D
Note: These registers are not SFRs. They can only be accessed indirectly through RTC0ADR and RTC0DAT.

Bits 7–0: ALARMn: smaRTClock Alarm Target.
These 6 registers (ALARM5–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 LSB of ALARM0 is not used. The LSB of the 47-bit smaRTClock timer will be compared against ALARM0.1.

## 20.4. Backup Regulator and RAM

The smaRTClock includes a backup supply regulator that keeps the smaRTClock peripheral fully functional when $V_{DD}$ is turned off. The backup supply regulator regulates the $V_{RTC\text{-}BACKUP}$ supply voltage, which can range from 1 V to 5.25 V. Switchover logic automatically powers smaRTClock from the backup supply when the voltage at $V_{RTC\text{-}BACKUP}$ is greater than $V_{DD}$.

The smaRTClock also includes 64 bytes of backup RAM. This memory can be read and written indirectly using the RAMADDR and RAMDATA internal registers.

### Internal Register Definition 20.8. RAMADDR: smaRTClock Backup RAM Address

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | smaRTClock Address: |

Note: This register is not an SFR. It can only be accessed indirectly through RTC0ADR and RTC0DAT. 0x0E

Bit 7: RAMADDR: smaRTClock Battery Backup RAM Address Bits
These bits select the smaRTClock Backup RAM byte that is targeted by RAMDATA. This address auto-increments after each read or write of RAMDATA.

## Internal Register Definition 20.9. RAMDATA: smaRTClock Backup RAM Data

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | smaRTClock Address: |
| Note: This register is not an SFR. It can only be accessed indirectly through RTC0ADR and RTC0DAT. | | | | | | | | 0x0F |

Bit 7: RAMDATA: smaRTClock Battery Backup RAM Data Bits.
These bits provide read and write access to the smaRTClock Backup RAM byte that is selected by RAMADDR.

Reads and writes of RAMDATA load the value at address RAMADDR into RTC0DAT. The following example writes 0xA5 to address 0x20 in the RAM and reads the value back to a temporary variable:

```
// in 'C':
unsigned char temp = 0x00;

// Unlock the smaRTClock interface
RTC0KEY = 0xA5;
RTC0KEY = 0xF1;

// Enable the smaRTClock
RTC0ADR = 0x06; // address the RTC0CN register
RTC0DAT = 0x80; // enable the smaRTClock
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit

// Write to the smaRTClock RAM
RTC0ADR = 0x0E;// address the RAMADDR register
RTC0DAT = 0x20;// write the address of 0x20 to RAMADDR
while ((RTC0ADR & 0x80) == 0x80);// poll on the BUSY bit

RTC0ADR = 0x0F;// address the RAMDATA register
RTC0DAT = 0xA5;// write 0xA5 to RAM address 0x20
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit

// Read from the smaRTClock RAM
RTC0ADR = 0x0E;// address the RAMADDR register
RTC0DAT = 0x20;// write the address of 0x20 to RAMADDR
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit

RTC0ADR = 0x0F;    // address the RAMDATA register
RTC0ADR |= 0x80; // initiate a read of the RAMDATA register
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit
temp = RTC0DAT; // read the value of RAM address 0x20


; in assembly:
   ; Unlock the smaRTClock interface
   mov RTC0KEY, #0A5h
   mov RTC0KEY, #0F1h
```

```
   ; Enable the smaRTClock
   mov RTC0ADR, #06h ; address the RTC0CN register
   mov RTC0DAT, #080h ; enable the smaRTClock
L0: mov A, RTC0ADR   ; poll on the BUSY bit
   jb ACC.7, L0

   ; Write to the smaRTClock RAM
   mov RTC0ADR, #0Eh; address the RAMADDR register
   mov RTC0DAT, #20h; write the address of 0x20 to RAMADDR
L1: mov A, RTC0ADR     ; poll on the BUSY bit
   jb ACC.7, L1

   mov RTC0ADR, #0Fh; address the RAMDATA register
   mov RTC0DAT, #0A5h; write 0xA5 to RAM address 0x20
L2: mov A, RTC0ADR     ; poll on the BUSY bit
   jb ACC.7, L2


   ; Read from the smaRTClock RAM
   mov RTC0ADR, #0Eh; address the RAMADDR register
   mov RTC0DAT, #20h; write the address of 0x20 to RAMADDR
L3: mov A, RTC0ADR   ; poll on the BUSY bit
   jb ACC.7, L3

   mov RTC0ADR, #0Fh   ; address the RAMDATA register
   orl RTC0ADR, #80h   ; initiate a read of the RAMDATA register
L4: mov A, RTC0ADR   ; poll on the BUSY bit
   jb ACC.7, L4
   movR0, #80h
   mov@R0, RTC0DAT    ; read the value of RAM address 0x20 into
             ; the 128-byte internal RAM
```

To reduce the number of instructions necessary to read and write sections of the 64-byte RAM, the RAMADDR register automatically increments after each write or read. The following C example initializes the entire 64-byte RAM to 0xA5 and copies this value from the RAM to an array using the auto-increment feature:

```
// in 'C':
unsigned char RAM_data[64] = 0x00;
unsigned char addr;

// Unlock smaRTClock, enable smaRTClock

// Write to the entire smaRTClock RAM
RTC0ADR = 0x0E;// address the RAMADDR register
RTC0DAT = 0x00;// write the address of 0x00 to RAMADDR
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit
RTC0ADR = 0x0F;// address the RAMDATA register
for (addr = 0; addr < 64; addr++)
{
   RTC0DAT = 0xA5; // write 0xA5 to every RAM address
   while ((RTC0ADR & 0x80) == 0x80);// poll on the BUSY bit
}

// Read from the entire smaRTClock RAM
RTC0ADR = 0x0E;// address the RAMADDR register
```

```
RTC0DAT = 0x00;// write the address of 0x00 to RAMADDR
while ((RTC0ADR & 0x80) == 0x80);   // poll on the BUSY bit
RTC0ADR = 0x0F; // address the RAMDATA register
for (addr = 0; addr < 64; addr++)
{
   RTC0ADR |= 0x80;   // initiate a read of the RAMDATA register
   while ((RTC0ADR & 0x80) == 0x80); // poll on the BUSY bit
   RAM_data[addr] = RTC0DAT; // copy the data from the entire RAM
}
```

## 21. 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 2, and compatible with the I2C 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. Three SFRs are associated with the SMBus: SMB0CF configures the SMBus; SMB0CN controls the status of the SMBus; and SMB0DAT is the data register, used for both transmitting and receiving SMBus data and slave addresses.

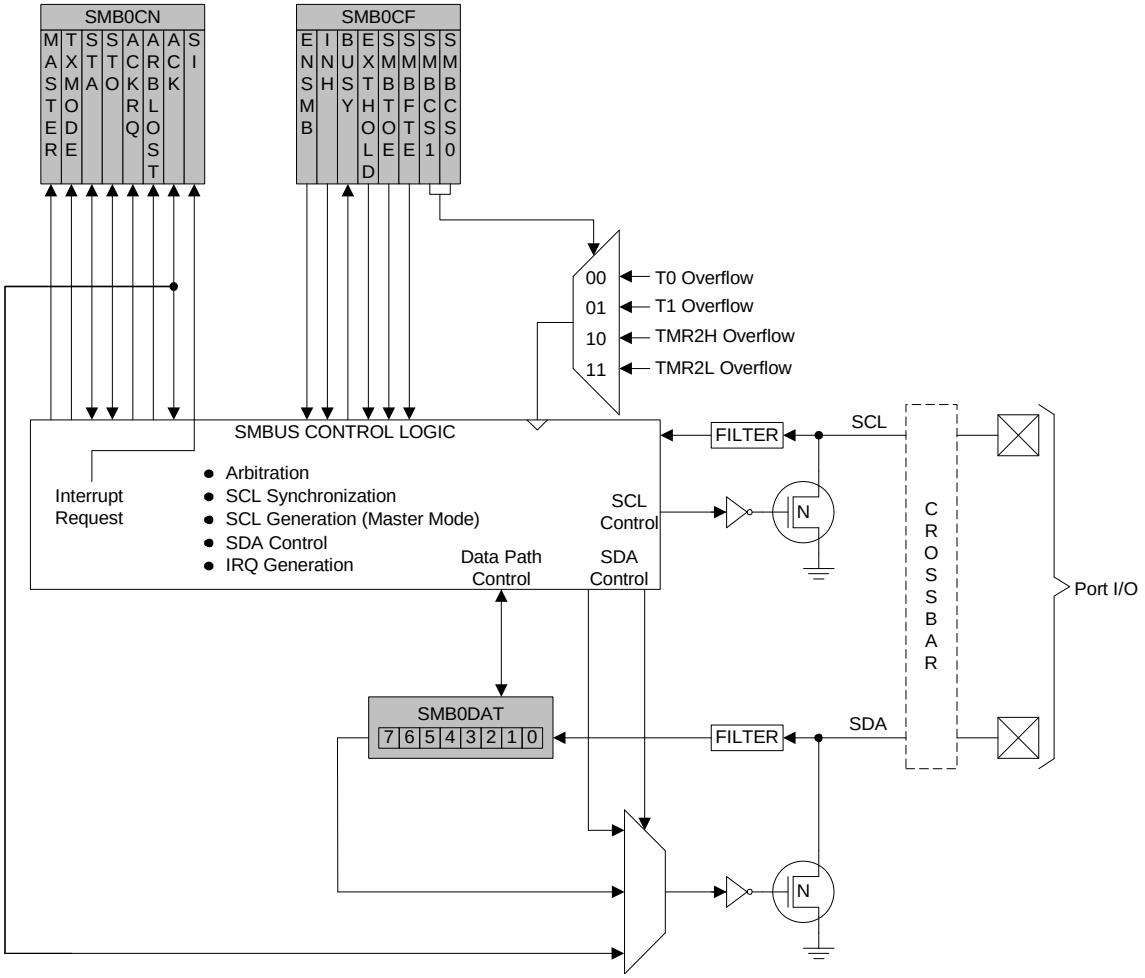



**Figure 21.1. SMBus Block Diagram**

## 21.1. Supporting Documents

It is assumed the reader is familiar with or has access to the following supporting documents:

1. The I2C Manual (AN10216-01), Philips Semiconductor.
2. System Management Bus Specification -- Version 2, SBS Implementers Forum.

## 21.2. SMBus Configuration

Figure 21.2 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.

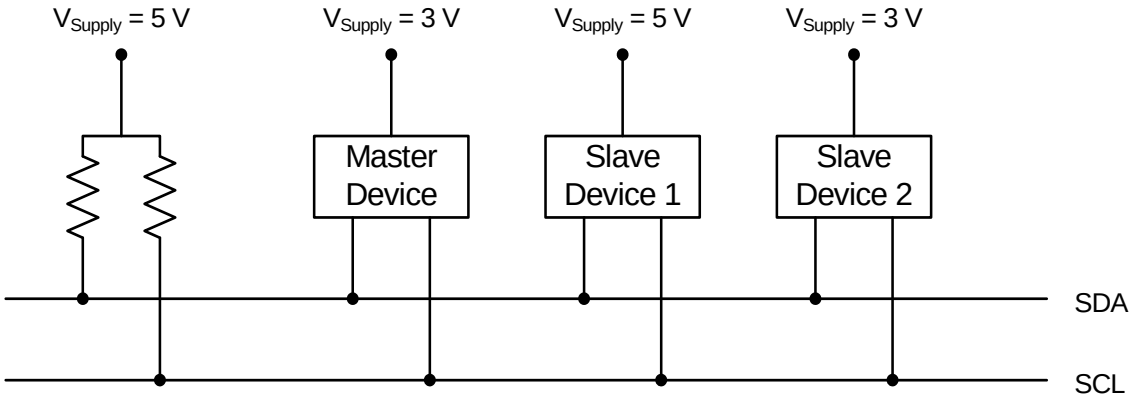



**Figure 21.2. Typical SMBus Configuration**

**Note:** It is recommended that the SDA and SCL pins be configured for high impedance overdrive mode. See **Section "18. Port Input/Output" on page 147** for more information.

## 21.3. 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. Each byte that is received (by a master or slave) must be acknowledged (ACK) with a low SDA during a high SCL (see Figure 21.3). 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 21.3 illustrates a typical SMBus transaction.

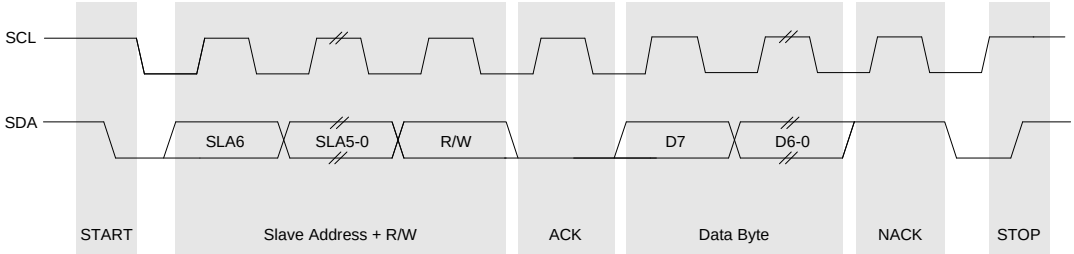



**Figure 21.3. SMBus Transaction**

### 21.3.1. 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 "21.3.4. SCL High (SMBus Free) Timeout" on page 194**). 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.

### 21.3.2. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I2C, 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.

### 21.3.3. 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.

### 21.3.4. 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. 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. **Enabling the Bus Free Timeout is recommended.**

## 21.4. 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

SMBus interrupts are generated for each data byte or slave address that is transferred. When transmitting, this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data, this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. See **Section “21.5. SMBus Transfer Modes” on page 201** 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 “21.4.2. SMB0CN Control Register” on page 198**; Table 21.4 provides a quick SMB0CN decoding reference.

SMBus configuration options include:

- Timeout detection (SCL Low Timeout and/or Bus Free Timeout)
- SDA setup and hold time extensions
- Slave event enable/disable
- Clock source selection

These options are selected in the SMB0CF register, as described in **Section "21.4.1. SMBus Configuration Register" on page 195**.

### 21.4.1. 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 21.1. 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 Bus 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 21.1. 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 "24. Timers" on page 231**.

$$T_{HighMin} = T_{LowMin} = \frac{1}{f_{ClockSourceOverflow}}$$

**Equation 21.1. 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 21.1. 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 21.2.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

**Equation 21.2. Typical SMBus Bit Rate**

Figure 21.4 shows the typical SCL generation described by Equation 21.2. 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 21.1.

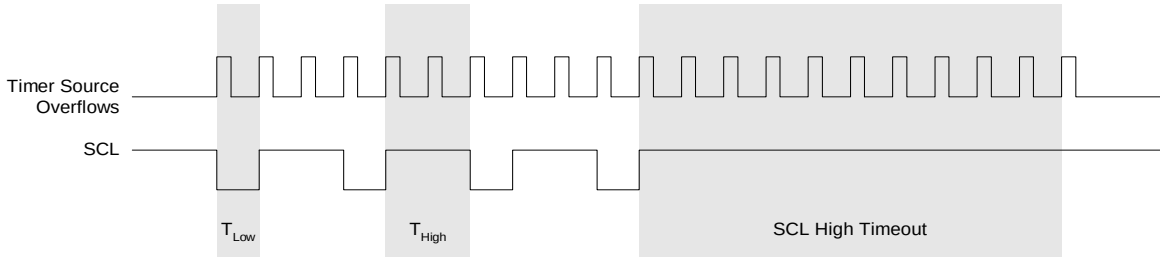



**Figure 21.4. Typical SMBus SCL Generation**

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 21.2 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.

Note: For SCL operation above 100 kHz, EXTHOLD should be cleared to ‘0’.

**Table 21.2. 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. 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 “21.3.3. SCL Low Timeout” on page 194**). 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 21.4). When a Free Timeout is detected, the interface will respond as if a STOP was detected (an interrupt will be generated, and STO will be set). **Enabling the Bus Free Timeout is recommended.**

## SFR Definition 21.1. SMB0CF: SMBus Clock/Configuration

| R/W | R/W | R | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| ENSMB | INH | BUSY | EXTHOLD | SMBTOE | SMBFTE | SMBCS1 | SMBCS0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xC1

Bit7: ENSMB: SMBus Enable.
This bit enables/disables the SMBus interface. When enabled, the interface constantly monitors the SDA and SCL pins.
0: SMBus interface disabled.
1: SMBus interface enabled.

Bit6: INH: SMBus Slave Inhibit.
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.
0: SMBus Slave Mode enabled.
1: SMBus Slave Mode inhibited.

Bit5: BUSY: SMBus Busy Indicator.
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.

Bit4: EXTHOLD: SMBus Setup and Hold Time Extension Enable.
This bit controls the SDA setup and hold times according to Table 21.2.
0: SDA Extended Setup and Hold Times disabled.
1: SDA Extended Setup and Hold Times enabled.

Bit3: SMBTOE: SMBus SCL Timeout Detection Enable.
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. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication.

Bit2: SMBFTE: SMBus Free Timeout Detection Enable.
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.

Bits1–0: SMBCS1–SMBCS0: SMBus Clock Source Selection.
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 21.1.

| 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 |

### 21.4.2. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information (see SFR Definition 21.2). 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 and TXMODE indicate the master/slave state and transmit/receive modes, respectively.

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.

As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received on 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.

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 21.3 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.

Table 21.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 21.4 for SMBus status decoding using the SMB0CN register.

## SFR Definition 21.2. SMB0CN: SMBus Control

| R | R | R/W | R/W | R | R | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| MASTER | TXMODE | STA | STO | ACKRQ | ARBLOST | ACK | SI | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xC0

Bit7: MASTER: SMBus Master/Slave Indicator.
This read-only bit indicates when the SMBus is operating as a master.
0: SMBus operating in Slave Mode.
1: SMBus operating in Master Mode.

Bit6: TXMODE: SMBus Transmit Mode Indicator.
This read-only bit indicates when the SMBus is operating as a transmitter.
0: SMBus in Receiver Mode.
1: SMBus in Transmitter Mode.

Bit5: STA: SMBus Start Flag.
Write:
0: No Start generated.
1: When operating as a master, a START condition is transmitted if the bus is free (If the bus is not free, the START is transmitted after a STOP is received or a timeout is detected). If STA is set by software as an active Master, a repeated START will be generated after the next ACK cycle.
Read:
0: No Start or repeated Start detected.
1: Start or repeated Start detected.

Bit4: STO: SMBus Stop Flag. **If set by hardware, this bit must be cleared by software.**
Write:
0: No STOP condition is transmitted.
1: Setting STO to logic 1 causes a STOP condition to be transmitted after the next ACK cycle. When the STOP condition is generated, hardware clears STO to logic 0. If both STA and STO are set, a STOP condition is transmitted followed by a START condition.
Read:
0: No Stop condition detected.
1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).

Bit3: ACKRQ: SMBus Acknowledge Request
This read-only bit is set to logic 1 when the SMBus has received a byte and needs the ACK bit to be written with the correct ACK response value.

Bit2: ARBLOST: SMBus Arbitration Lost Indicator.
This read-only bit is set to logic 1 when the SMBus loses arbitration while operating as a transmitter. A lost arbitration while a slave indicates a bus error condition.

Bit1: ACK: SMBus Acknowledge Flag.
This bit defines the out-going ACK level and records incoming ACK levels. It should be written each time a byte is received (when ACKRQ=1), or read after each byte is transmitted.
0: A "not acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).
1: An "acknowledge" has been received (if in Transmitter Mode) OR will be transmitted (if in Receiver Mode).

Bit0: SI: SMBus Interrupt Flag.
This bit is set by hardware under the conditions listed in Table 21.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled.

**Table 21.3. 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.<br>• If STO is set by hardware, it must be cleared by software. |
| ACKRQ | • A byte has been received and an ACK response value is needed. | • 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. |

### 21.4.3. 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 21.3. SMB0DAT: SMBus Data**

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xC2

Bits7–0: SMB0DAT: SMBus Data.
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.

## 21.5. 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; however, note that the interrupt is generated before the ACK cycle when operating as a receiver, and after the ACK cycle when operating as a transmitter.

### 21.5.1. Master Transmitter Mode

Serial data is transmitted on SDA while the serial clock is output on SCL. 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 21.5 shows a typical Master Transmitter sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode.

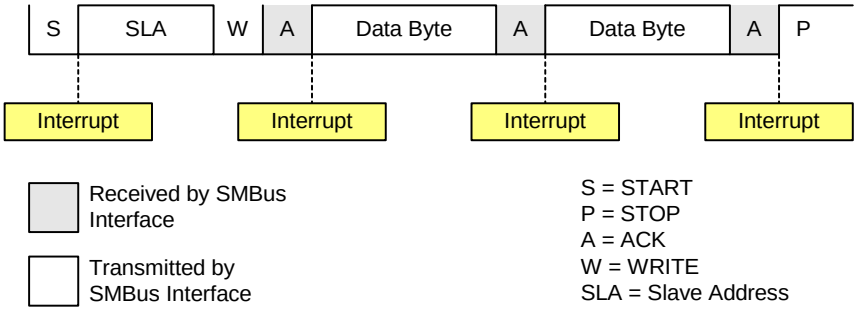



**Figure 21.5. Typical Master Transmitter Sequence**

### 21.5.2. Master Receiver Mode

Serial data is received on SDA while the serial clock is output on SCL. 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. After each byte is received, ACKRQ is set to ‘1’ and an interrupt is generated. Software must write the ACK bit (SMB0CN.1) to define the outgoing acknowledge value (Note: 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 after the last byte is received, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. Note that the interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 21.6 shows a typical Master Receiver sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur **before** the ACK cycle in this mode.

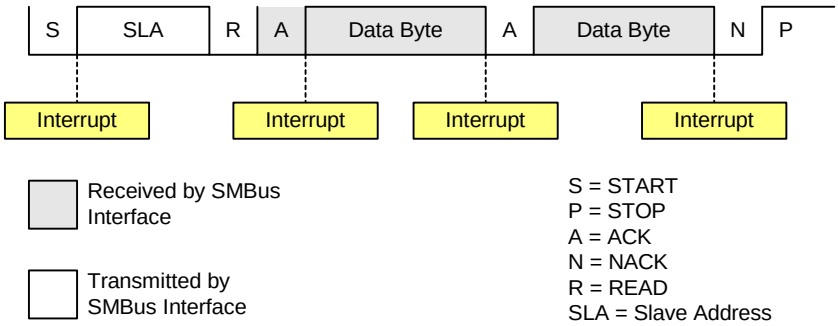



**Figure 21.6. Typical Master Receiver Sequence**

### 21.5.3. Slave Receiver Mode

Serial data is received on SDA and the clock is received on SCL. 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. Upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. Software responds to the received slave address with an ACK, or ignores the received slave address with a NACK. If the received slave address is ignored, 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. Software must write the ACK bit after each received byte to ACK or NACK the received byte. 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 21.7 shows a typical Slave Receiver sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur **before** the ACK cycle in this mode.

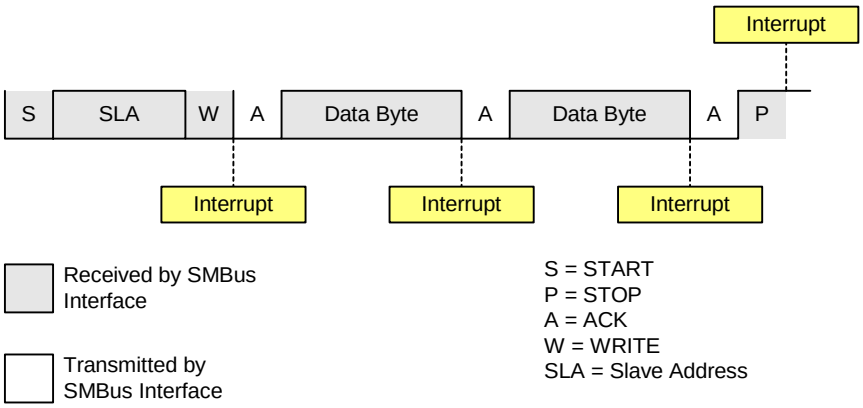



**Figure 21.7. Typical Slave Receiver Sequence**

### 21.5.4. Slave Transmitter Mode

Serial data is transmitted on SDA and the clock is received on SCL. 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. Upon entering Slave Transmitter Mode, an interrupt is generated and the ACKRQ bit is set. Software responds to the received slave address with an ACK, or ignores the received slave address with a NACK. If the received slave address is ignored, slave interrupts will be inhibited until a START is detected. 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 21.8 shows a typical Slave Transmitter sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode.

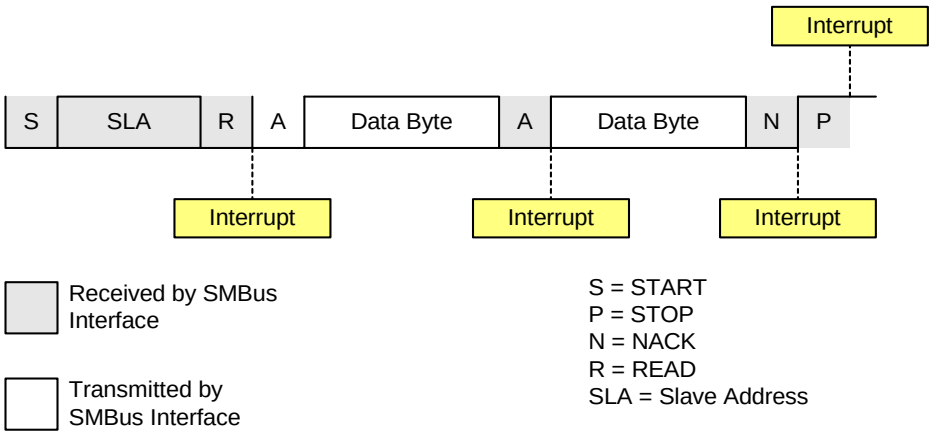



**Figure 21.8. Typical Slave Transmitter Sequence**

## 21.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. In the table below, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. Note that 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 but do not conform to the SMBus specification.

**Table 21.4. SMBus Status Decoding**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values Written | | |
|---|---|---|---|---|---|---|---|---|---|
| | 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 | 0 | 0 | 0 | A master data or address byte was transmitted; NACK received. | Set STA to restart transfer. | 1 | 0 | X |
| | | | | | | 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 |
| | | | | | | End transfer with STOP. | 0 | 1 | X |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X |
| | | | | | | Send repeated START. | 1 | 0 | X |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). | 0 | 0 | X |
| Master Receiver | 1000 | 1 | 0 | X | A master data byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 |
| | | | | | | 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 |
| | | | | | | Send ACK followed by repeated START. | 1 | 0 | 1 |
| | | | | | | Send NACK to indicate last byte, and send repeated START. | 1 | 0 | 0 |
| | | | | | | Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 1 |
| | | | | | | Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 0 |

**Table 21.4. SMBus Status Decoding (Continued)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values Written | | |
|---|---|---|---|---|---|---|---|---|---|
| | 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 |
| | | 0 | 0 | 1 | A slave byte was transmitted; ACK received. | Load SMB0DAT with next data byte to transmit. | 0 | 0 | X |
| | | 0 | 1 | X | A Slave byte was transmitted; error detected. | No action required (expecting Master to end transfer). | 0 | 0 | X |
| | 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 was received; ACK requested. | Acknowledge received address. | 0 | 0 | 1 |
| | | | | | | Do not acknowledge received address. | 0 | 0 | 0 |
| | | 1 | 1 | X | Lost arbitration as master; slave address received; ACK requested. | Acknowledge received address. | 0 | 0 | 1 |
| | | | | | | Do not acknowledge received address. | 0 | 0 | 0 |
| | | | | | | Reschedule failed transfer; do not acknowledge received address. | 1 | 0 | 0 |
| | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X |
| | 0001 | 1 | 1 | X | Lost arbitration while attempting a STOP. | No action required (transfer complete/aborted). | 0 | 0 | 0 |
| | | 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 due to a detected STOP. | Abort transfer. | 0 | 0 | X |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X |
| | 0000 | 1 | 0 | X | A slave byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 |
| | | | | | | Do not acknowledge received byte. | 0 | 0 | 0 |
| | | 1 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | 0 |
| | | | | | | Reschedule failed transfer. | 1 | 0 | 0 |

## 22. 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 "22.1. Enhanced Baud Rate Generation" on page 208**). 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).

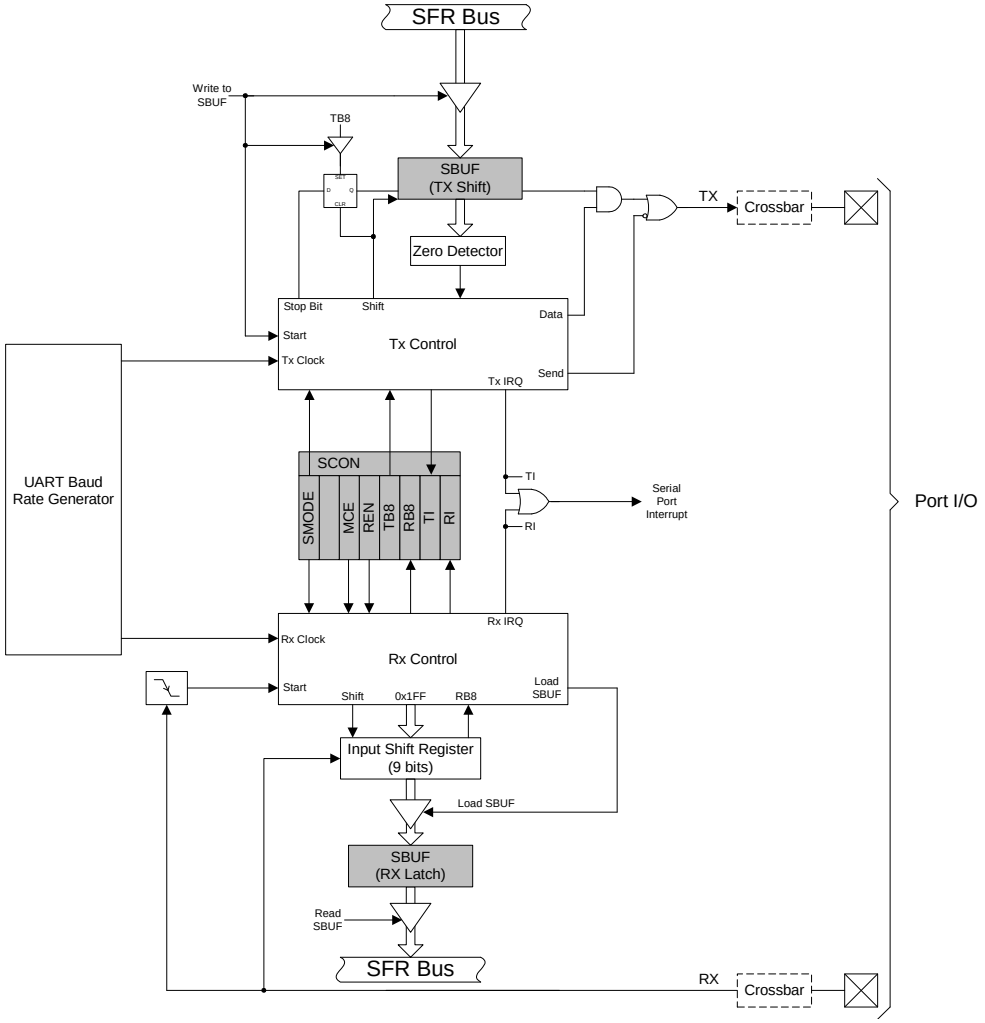



**Figure 22.1. UART0 Block Diagram**

## 22.1. 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 22.2), 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.

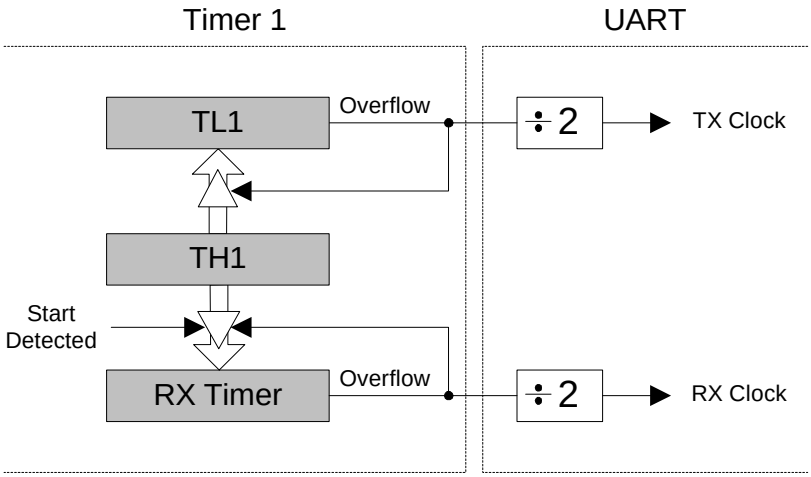



**Figure 22.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see **Section "24.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload" on page 233**). 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. The UART0 baud rate is determined by Equation 22.1-A and Equation 22.1-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 22.1. 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 (8-bit auto-reload mode reload value). Timer 1 clock frequency is selected as described in **Section "24. Timers" on page 231**. A quick reference for typical baud rates and system clock frequencies is given in Table 22.1 through Table 22.6. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

## 22.2. 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.

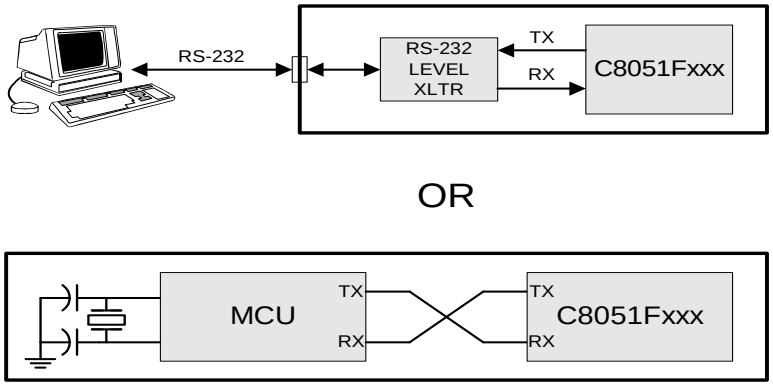



**Figure 22.3. UART Interconnect Diagram**

### 22.2.1. 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.

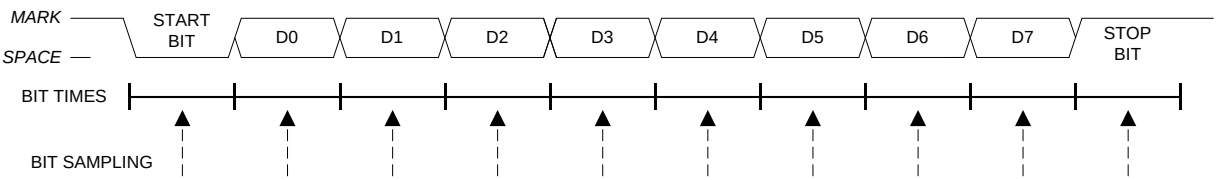



**Figure 22.4. 8-Bit UART Timing Diagram**

### 22.2.2. 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’.

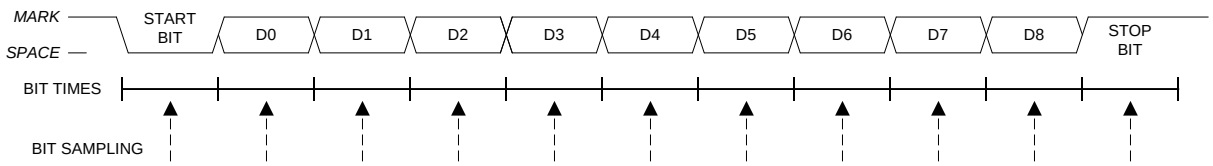



**Figure 22.5. 9-Bit UART Timing Diagram**

## 22.3. 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).

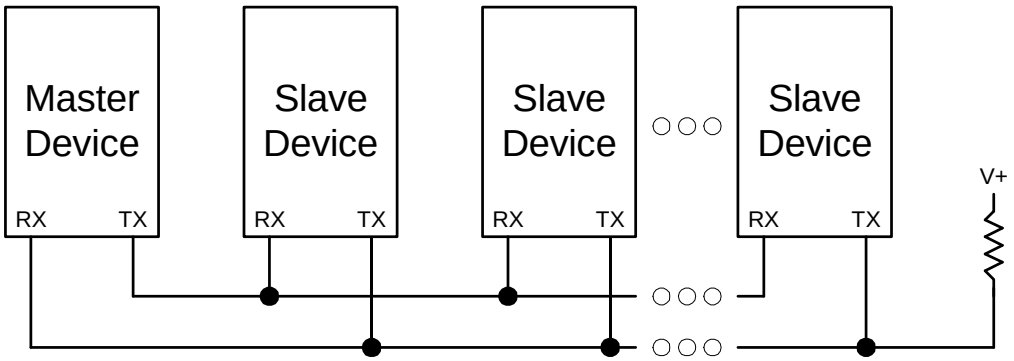



**Figure 22.6. UART Multi-Processor Mode Interconnect Diagram**

## SFR Definition 22.1. SCON0: Serial Port 0 Control

| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| S0MODE | - | MCE0 | REN0 | TB80 | RB80 | TI0 | RI0 | 01000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0x98

Bit7: S0MODE: Serial Port 0 Operation Mode.
This bit selects the UART0 Operation Mode.
0: 8-bit UART with Variable Baud Rate.
1: 9-bit UART with Variable Baud Rate.

Bit6: UNUSED. Read = 1b. Write = don't care.

Bit5: MCE0: Multiprocessor Communication Enable.
The function of this bit is dependent on the Serial Port 0 Operation Mode.
S0MODE = 0: Checks for valid stop bit.
0: Logic level of stop bit is ignored.
1: RI0 will only be activated if stop bit is logic level 1.
S0MODE = 1: Multiprocessor Communications Enable.
0: Logic level of ninth bit is ignored.
1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1.

Bit4: REN0: Receive Enable.
This bit enables/disables the UART receiver.
0: UART0 reception disabled.
1: UART0 reception enabled.

Bit3: TB80: Ninth Transmission Bit.
The logic level of this bit will be assigned to the ninth transmission bit in 9-bit UART Mode. It is not used in 8-bit UART Mode. Set or cleared by software as required.

Bit2: RB80: Ninth Receive Bit.
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.

Bit1: TI0: Transmit Interrupt Flag.
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.

Bit0: RI0: Receive Interrupt Flag.
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 22.2. SBUF0: Serial (UART0) Port Data Buffer

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x99

Bits7–0: SBUF0[7:0]: Serial Data Buffer Bits 7-0 (MSB-LSB)
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 22.1. Timer Settings for Standard Baud Rates Using the Internal Oscillator**

| | Frequency: 24.5 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from Internal Osc. | 230400 | -0.32% | 106 | SYSCLK | XX | 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 |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

**Table 22.2. Timer Settings for Standard Baud Rates Using an External 25.0 MHz Oscillator**

| | Frequency: 25.0 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | -0.47% | 108 | SYSCLK | XX | 1 | 0xCA |
| | 115200 | 0.45% | 218 | SYSCLK | XX | 1 | 0x93 |
| | 57600 | -0.01% | 434 | SYSCLK | XX | 1 | 0x27 |
| | 28800 | 0.45% | 872 | SYSCLK / 4 | 01 | 0 | 0x93 |
| | 14400 | -0.01% | 1736 | SYSCLK / 4 | 01 | 0 | 0x27 |
| | 9600 | 0.15% | 2608 | EXTCLK / 8 | 11 | 0 | 0x5D |
| | 2400 | 0.45% | 10464 | SYSCLK / 48 | 10 | 0 | 0x93 |
| | 1200 | -0.01% | 20832 | SYSCLK / 48 | 10 | 0 | 0x27 |
| SYSCLK from Internal Osc. | 57600 | -0.47% | 432 | EXTCLK / 8 | 11 | 0 | 0xE5 |
| | 28800 | -0.47% | 864 | EXTCLK / 8 | 11 | 0 | 0xCA |
| | 14400 | 0.45% | 1744 | EXTCLK / 8 | 11 | 0 | 0x93 |
| | 9600 | 0.15% | 2608 | EXTCLK / 8 | 11 | 0 | 0x5D |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

**Table 22.3. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator**

| | Frequency: 22.1184 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 96 | SYSCLK | XX | 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 |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

**Table 22.4. Timer Settings for Standard Baud Rates Using an External 18.432 MHz Oscillator**

| | Frequency: 18.432 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 80 | SYSCLK | XX | 1 | 0xD8 |
| | 115200 | 0.00% | 160 | SYSCLK | XX | 1 | 0xB0 |
| | 57600 | 0.00% | 320 | SYSCLK | XX | 1 | 0x60 |
| | 28800 | 0.00% | 640 | SYSCLK / 4 | 01 | 0 | 0xB0 |
| | 14400 | 0.00% | 1280 | SYSCLK / 4 | 01 | 0 | 0x60 |
| | 9600 | 0.00% | 1920 | SYSCLK / 12 | 00 | 0 | 0xB0 |
| | 2400 | 0.00% | 7680 | SYSCLK / 48 | 10 | 0 | 0xB0 |
| | 1200 | 0.00% | 15360 | SYSCLK / 48 | 10 | 0 | 0x60 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 80 | EXTCLK / 8 | 11 | 0 | 0xFB |
| | 115200 | 0.00% | 160 | EXTCLK / 8 | 11 | 0 | 0xF6 |
| | 57600 | 0.00% | 320 | EXTCLK / 8 | 11 | 0 | 0xEC |
| | 28800 | 0.00% | 640 | EXTCLK / 8 | 11 | 0 | 0xD8 |
| | 14400 | 0.00% | 1280 | EXTCLK / 8 | 11 | 0 | 0xB0 |
| | 9600 | 0.00% | 1920 | EXTCLK / 8 | 11 | 0 | 0x88 |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

**Table 22.5. Timer Settings for Standard Baud Rates Using an External 11.0592 MHz Oscillator**

| | Frequency: 11.0592 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 48 | SYSCLK | XX | 1 | 0xE8 |
| | 115200 | 0.00% | 96 | SYSCLK | XX | 1 | 0xD0 |
| | 57600 | 0.00% | 192 | SYSCLK | XX | 1 | 0xA0 |
| | 28800 | 0.00% | 384 | SYSCLK | XX | 1 | 0x40 |
| | 14400 | 0.00% | 768 | SYSCLK / 12 | 00 | 0 | 0xE0 |
| | 9600 | 0.00% | 1152 | SYSCLK / 12 | 00 | 0 | 0xD0 |
| | 2400 | 0.00% | 4608 | SYSCLK / 12 | 00 | 0 | 0x40 |
| | 1200 | 0.00% | 9216 | SYSCLK / 48 | 10 | 0 | 0xA0 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 48 | EXTCLK / 8 | 11 | 0 | 0xFD |
| | 115200 | 0.00% | 96 | EXTCLK / 8 | 11 | 0 | 0xFA |
| | 57600 | 0.00% | 192 | EXTCLK / 8 | 11 | 0 | 0xF4 |
| | 28800 | 0.00% | 384 | EXTCLK / 8 | 11 | 0 | 0xE8 |
| | 14400 | 0.00% | 768 | EXTCLK / 8 | 11 | 0 | 0xD0 |
| | 9600 | 0.00% | 1152 | EXTCLK / 8 | 11 | 0 | 0xB8 |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

**Table 22.6. Timer Settings for Standard Baud Rates Using an External 3.6864 MHz Oscillator**

| | Frequency: 3.6864 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscilla-tor Divide Factor | Timer Clock Source | SCA1-SCA0 (pre-scale select)* | T1M* | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 16 | SYSCLK | XX | 1 | 0xF8 |
| | 115200 | 0.00% | 32 | SYSCLK | XX | 1 | 0xF0 |
| | 57600 | 0.00% | 64 | SYSCLK | XX | 1 | 0xE0 |
| | 28800 | 0.00% | 128 | SYSCLK | XX | 1 | 0xC0 |
| | 14400 | 0.00% | 256 | SYSCLK | XX | 1 | 0x80 |
| | 9600 | 0.00% | 384 | SYSCLK | XX | 1 | 0x40 |
| | 2400 | 0.00% | 1536 | SYSCLK / 12 | 00 | 0 | 0xC0 |
| | 1200 | 0.00% | 3072 | SYSCLK / 12 | 00 | 0 | 0x80 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 16 | EXTCLK / 8 | 11 | 0 | 0xFF |
| | 115200 | 0.00% | 32 | EXTCLK / 8 | 11 | 0 | 0xFE |
| | 57600 | 0.00% | 64 | EXTCLK / 8 | 11 | 0 | 0xFC |
| | 28800 | 0.00% | 128 | EXTCLK / 8 | 11 | 0 | 0xF8 |
| | 14400 | 0.00% | 256 | EXTCLK / 8 | 11 | 0 | 0xF0 |
| | 9600 | 0.00% | 384 | EXTCLK / 8 | 11 | 0 | 0xE8 |

X = Don't care

***Note:** SCA1-SCA0 and T1M bit definitions can be found in **Section 24.1**.

## 23. Enhanced Serial Peripheral Interface (SPI0)

The 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.

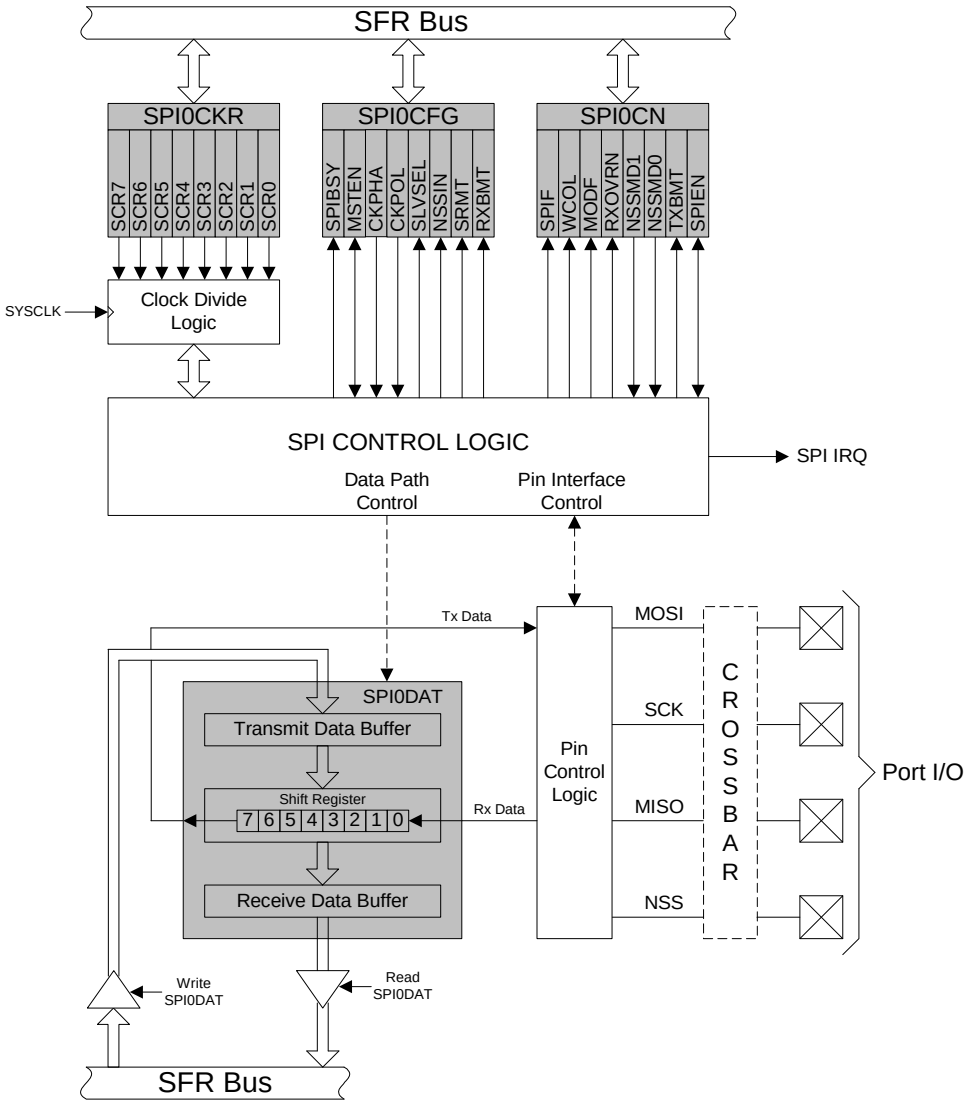



**Figure 23.1. SPI Block Diagram**

## 23.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

### 23.1.1. 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.

### 23.1.2. 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.

### 23.1.3. 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.

### 23.1.4. 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 23.2, Figure 23.3, and Figure 23.4 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 "**18. Port Input/Output**" on page **147** for general purpose port I/O and crossbar information.

## 23.2. 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 data 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 (SPI0CFG.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 23.2 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 23.3 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 23.4 shows a connection diagram for a master device in 4-wire master mode and two slave devices.

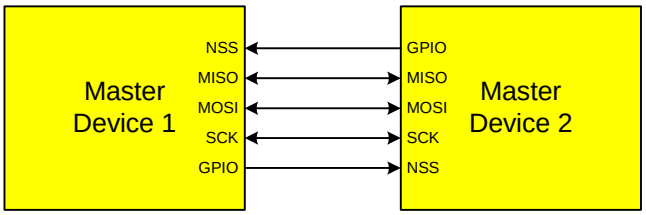



**Figure 23.2. Multiple-Master Mode Connection Diagram**

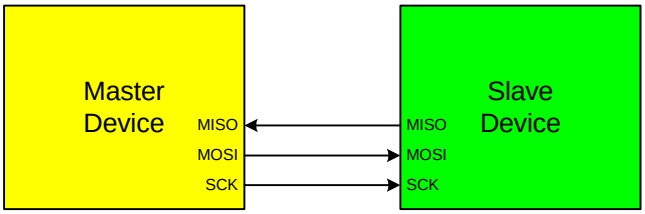



**Figure 23.3. 3-Wire Single Master and Slave Mode Connection Diagram**

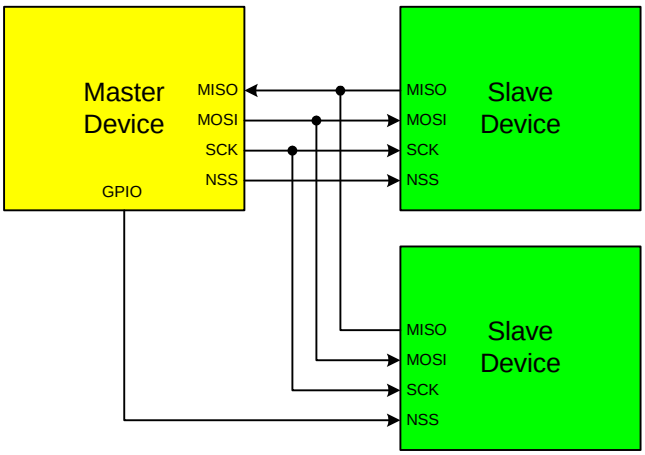



**Figure 23.4. 4-Wire Single Master and Slave Mode Connection Diagram**

## 23.3. 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 into 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.

The shift register contents are locked after the slave detects the first edge of SCK. Writes to SPI0DAT that occur after the first SCK edge will be held in the TX latch until the end of the current 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 23.4 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 not a 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 23.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

## 23.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

***Note that all of the following interrupt bits must be cleared by software.***

1. 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.
2. 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.
3. The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master in multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits are set to logic 0 to disable SPI0 and allow another master device to access the bus.
4. The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed while 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.

## 23.5. Serial Clock Timing

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 a rising edge or a falling edge. 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 are shown in Figure 23.5.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 23.3 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.

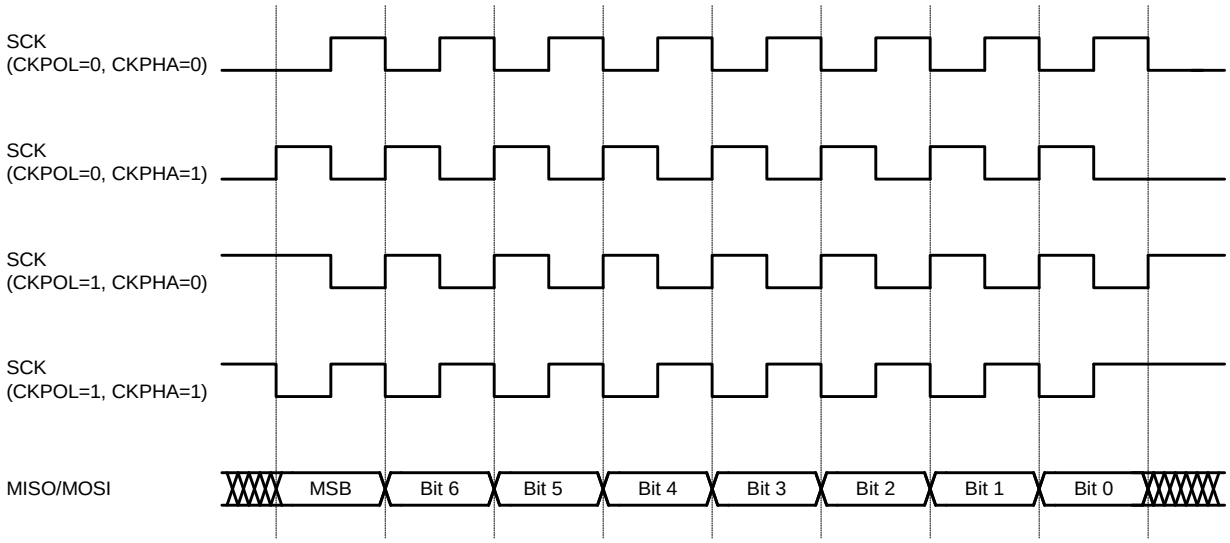



**Figure 23.5. Data/Clock Timing Relationship**

## 23.6. 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 23.1. SPI0CFG: SPI0 Configuration

| R | R/W | R/W | R/W | R | R | R | R | Reset Value |
|---|---|---|---|---|---|---|---|---|
| SPIBSY | MSTEN | CKPHA | CKPOL | SLVSEL | NSSIN | SRMT | RXBMT | 00000111 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA1

Bit 7: SPIBSY: SPI Busy (read only).
This bit is set to logic 1 when a SPI transfer is in progress (Master or Slave Mode).

Bit 6: MSTEN: Master Mode Enable.
0: Disable master mode. Operate in slave mode.
1: Enable master mode. Operate as a master.

Bit 5: CKPHA: SPI0 Clock Phase.
This bit controls the SPI0 clock phase.
0: Data centered on first edge of SCK period.*
1: Data centered on second edge of SCK period.*

Bit 4: CKPOL: SPI0 Clock Polarity.
This bit controls the SPI0 clock polarity.
0: SCK line low in idle state.
1: SCK line high in idle state.

Bit 3: SLVSEL: Slave Selected Flag (read only).
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.

Bit 2: NSSIN: NSS Instantaneous Pin Input (read only).
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.

Bit 1: SRMT: Shift Register Empty (Valid in Slave Mode, read only).
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.
NOTE: SRMT = 1 when in Master Mode.

Bit 0: RXBMT: Receive Buffer Empty (Valid in Slave Mode, read only).
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.
NOTE: RXBMT = 1 when in Master Mode.

***Note:** See Table 23.1 for timing parameters.

## SFR Definition 23.2. SPI0CN: SPI0 Control

| R/W | R/W | R/W | R/W | R/W | R/W | R | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| SPIF | WCOL | MODF | RXOVRN | NSSMD1 | NSSMD0 | TXBMT | SPIEN | 00000110 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xF8

Bit 7: SPIF: SPI0 Interrupt Flag.
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 SPI0 interrupt service routine. This bit is not automatically cleared by hardware. It must be cleared by software.

Bit 6: WCOL: Write Collision Flag.
This bit is set to logic 1 by hardware if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. It must be cleared by software.

Bit 5: MODF: Mode Fault Flag.
This bit is set to logic 1 by hardware (and generates a SPI0 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.

Bit 4: RXOVRN: Receive Overrun Flag (Slave Mode only).
This bit is set to logic 1 by hardware (and generates a SPI0 interrupt) 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. This bit is not automatically cleared by hardware. It must be cleared by software.

Bits 3–2: NSSMD1–NSSMD0: Slave Select Mode.
Selects between the following NSS operation modes:
(See **Section "23.2. SPI0 Master Mode Operation" on page 219** and **Section "23.3. SPI0 Slave Mode Operation" on page 220**).
00: 3-Wire Slave or 3-wire Master Mode. NSS signal is not routed to a port pin.
01: 4-Wire Slave or Multi-Master Mode (Default). NSS is always an input to the device.
1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.

Bit 1: TXBMT: Transmit Buffer Empty.
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.

Bit 0: SPIEN: SPI0 Enable.
This bit enables/disables the SPI.
0: SPI disabled.
1: SPI enabled.

## SFR Definition 23.3. SPI0CKR: SPI0 Clock Rate

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| SCR7 | SCR6 | SCR5 | SCR4 | SCR3 | SCR2 | SCR1 | SCR0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xA2

Bits 7–0: SCR7–SCR0: SPI0 Clock Rate.
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.

$$f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR + 1)}$$

for 0 <= SPI0CKR <= 255

Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04,

$$f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$$

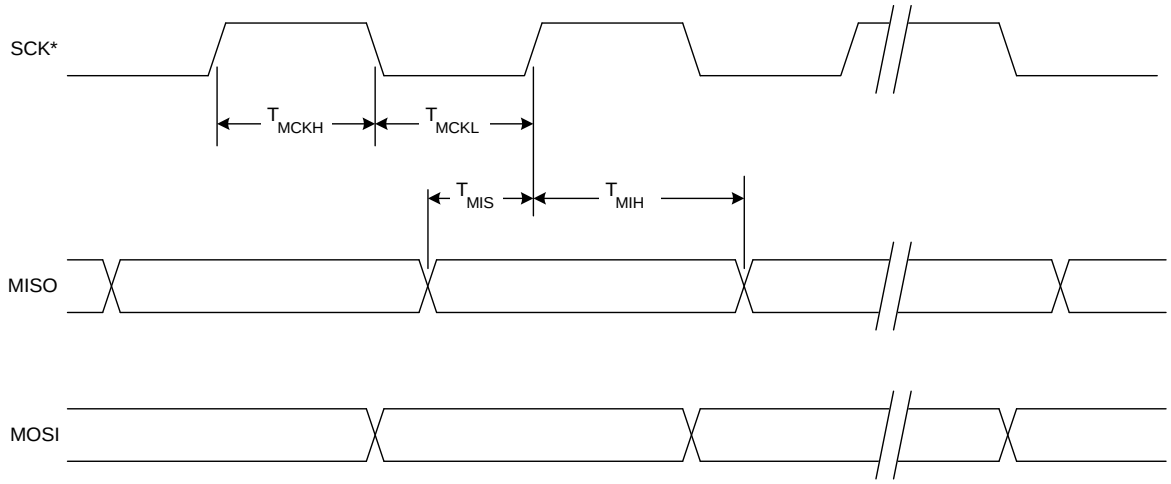
$$f_{SCK} = 200kHz$$

## SFR Definition 23.4. SPI0DAT: SPI0 Data

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

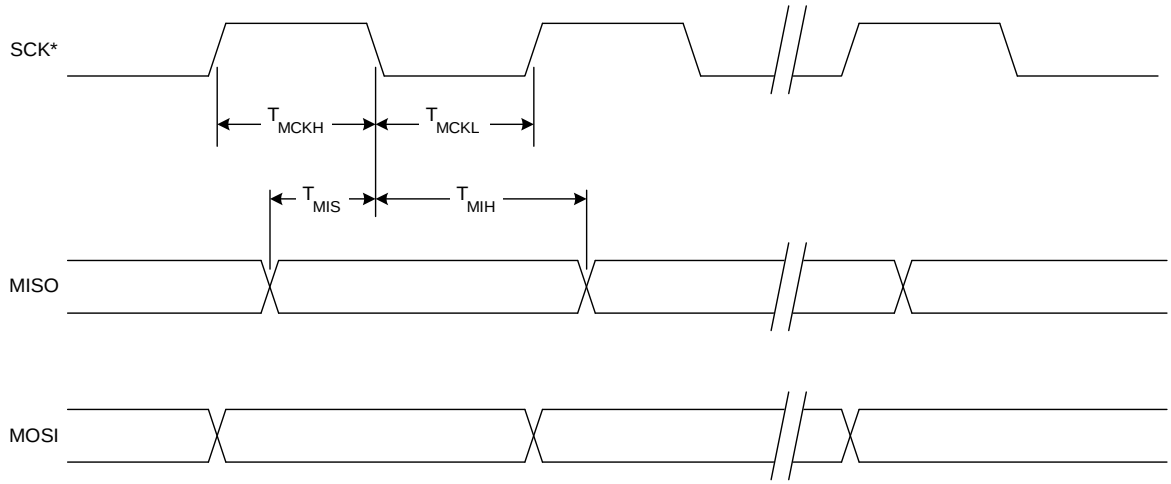
SFR Address: 0xA3

Bits 7–0: SPI0DAT: SPI0 Transmit and Receive Data.
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.



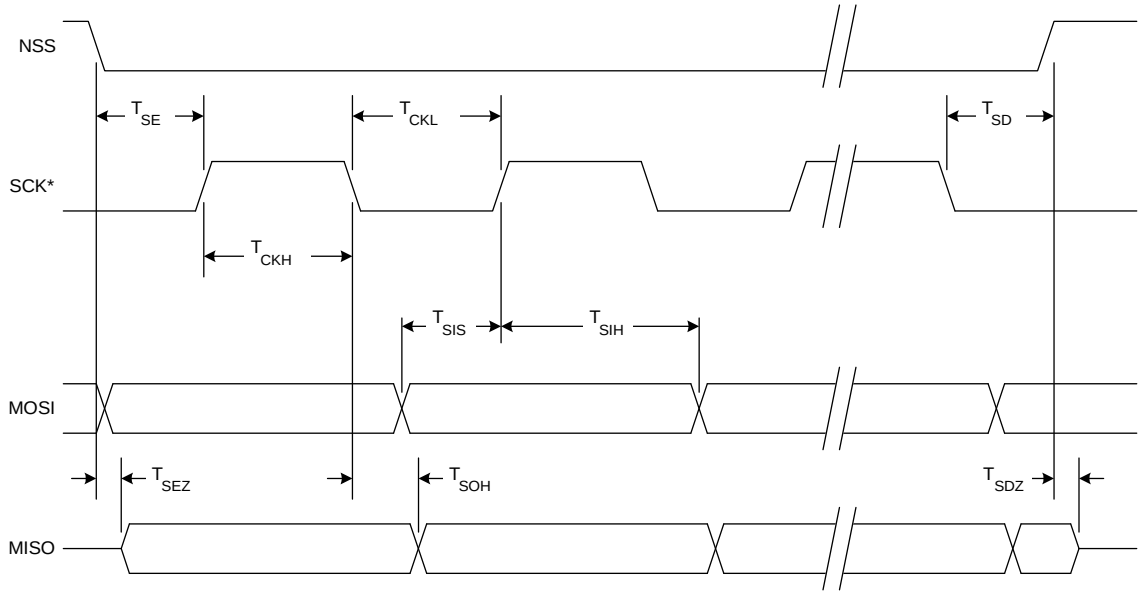

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.6. SPI Master Timing (CKPHA = 0)**



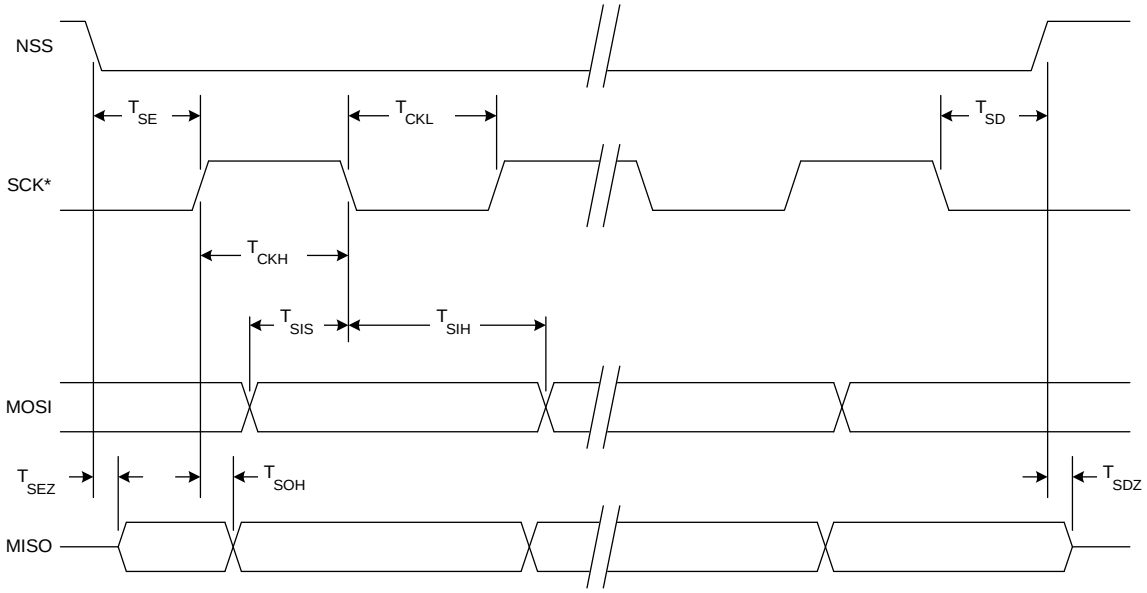

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.7. SPI Master Timing (CKPHA = 1)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.8. SPI Slave Timing (CKPHA = 0)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.9. SPI Slave Timing (CKPHA = 1)**

## Table 23.1. SPI Slave Timing Parameters

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing* (See Figure 23.6 and Figure 23.7)** | | | | |
| $T_{MCKH}$ | SCK High Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MCKL}$ | SCK Low Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MIS}$ | MISO Valid to SCK Sample Edge | 20 | — | ns |
| $T_{MIH}$ | SCK Sample Edge to MISO Change | 0 | — | ns |
| **Slave Mode Timing* (See Figure 23.8 and Figure 23.9)** | | | | |
| $T_{SE}$ | 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 |
| ***Note:** $T_{SYSCLK}$ is equal to one period of the device system clock (SYSCLK) in ns. | | | | |

## NOTES:

# 24. 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 other device peripherals 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. Timer 2 and Timer 3 also have a smaRTClock Capture Mode that can be used to measure the smaRTClock clock with respect to another oscillator.

<table>
<tr><th>Timer 0 and Timer 1 Modes:</th><th>Timer 2 Modes:</th><th>Timer 3 Modes:</th></tr>
<tr><td>13-bit counter/timer</td><td rowspan="2">16-bit timer with auto-reload</td><td rowspan="2">16-bit timer with auto-reload</td></tr>
<tr><td>16-bit counter/timer</td></tr>
<tr><td>8-bit counter/timer with auto-reload</td><td rowspan="2">Two 8-bit timers with auto-reload</td><td rowspan="2">Two 8-bit timers with auto-reload</td></tr>
<tr><td>Two 8-bit counter/timers<br>(Timer 0 only)</td></tr>
</table>

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 24.3 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, or the external oscillator clock source divided by 8.

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's frequency can be counted. The input signal need not be periodic, but it must be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

## 24.1. 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 “12.4. Interrupt Register Descriptions” on page 112**); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (**Section 12.4**). 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.

### 24.1.1. 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 "18.1. Priority Crossbar Decoder" on page 149** 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 24.3).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or the input signal /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7. "IT01CF: INT0/INT1 Configuration" on page 118). Setting GATE0 to '1' allows the timer to be controlled by the external input signal /INT0 (see **Section "12.4. Interrupt Register Descriptions" on page 112**), facilitating pulse width measurements.

| TR0 | GATE0 | /INT0 | Counter/Timer |
|---|---|---|---|
| 0 | X | X | Disabled |
| 1 | 0 | X | Enabled |
| 1 | 1 | 0 | Disabled |
| 1 | 1 | 1 | Enabled |
| 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 /INT1 is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 12.7. "IT01CF: INT0/INT1 Configuration" on page 118).

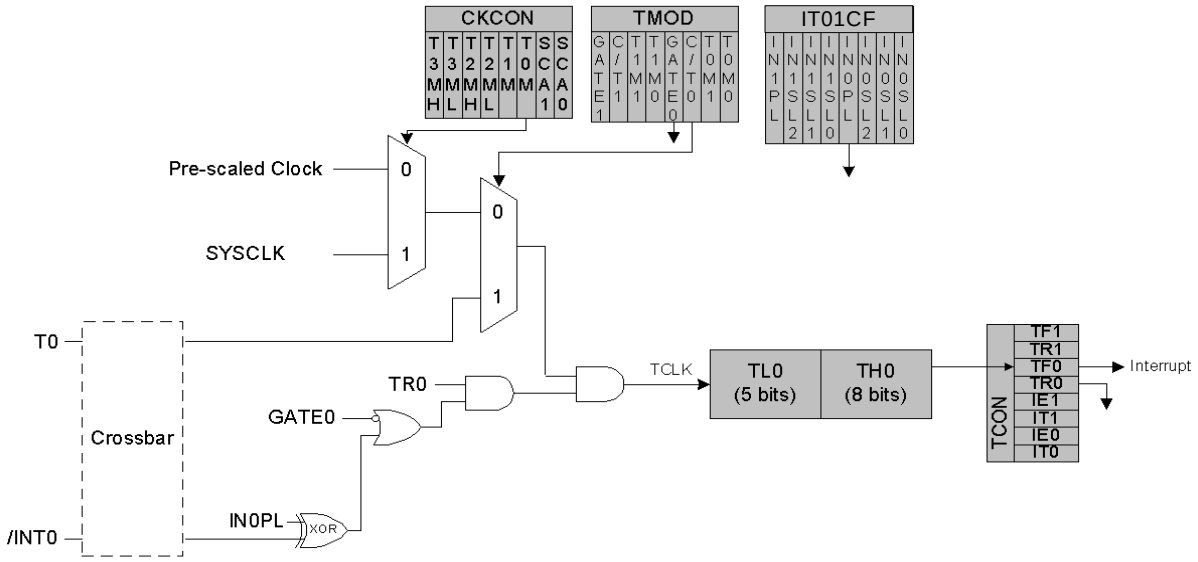



**Figure 24.1. T0 Mode 0 Block Diagram**

### 24.1.2. 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.

### 24.1.3. 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 /INT0 is active as defined by bit IN0PL in register IT01CF (see **Section "12.5. External Interrupts" on page 117** for details on the external input signals /INT0 and /INT1).

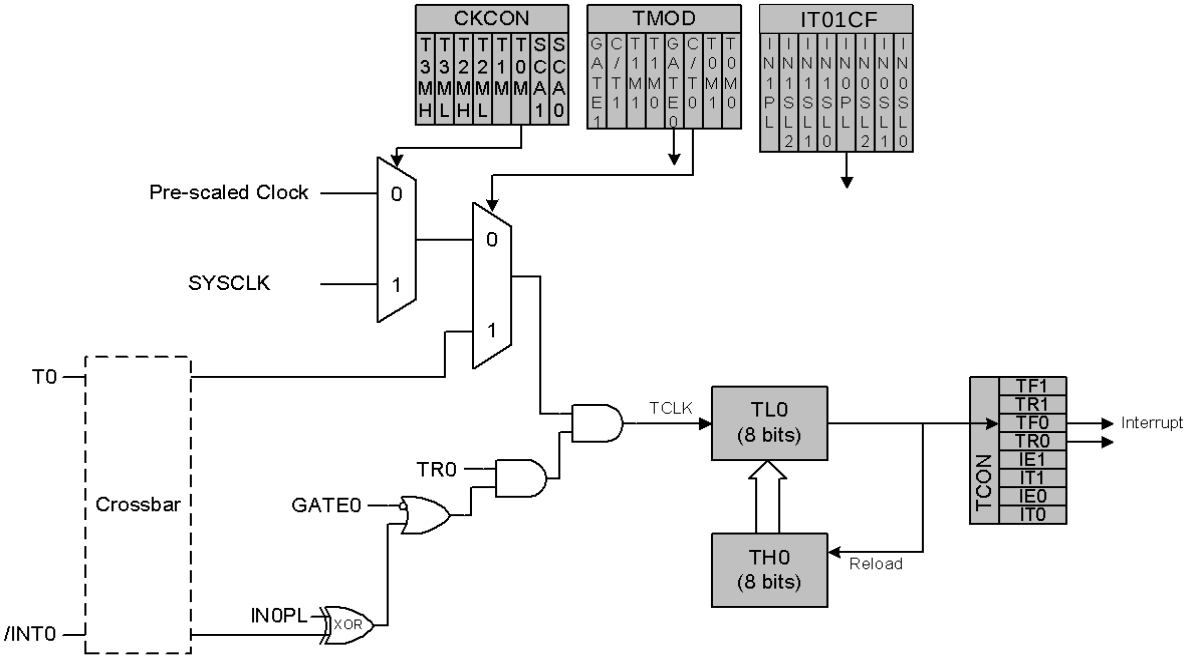



**Figure 24.2. T0 Mode 2 Block Diagram**

### 24.1.4. 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 UART. 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.

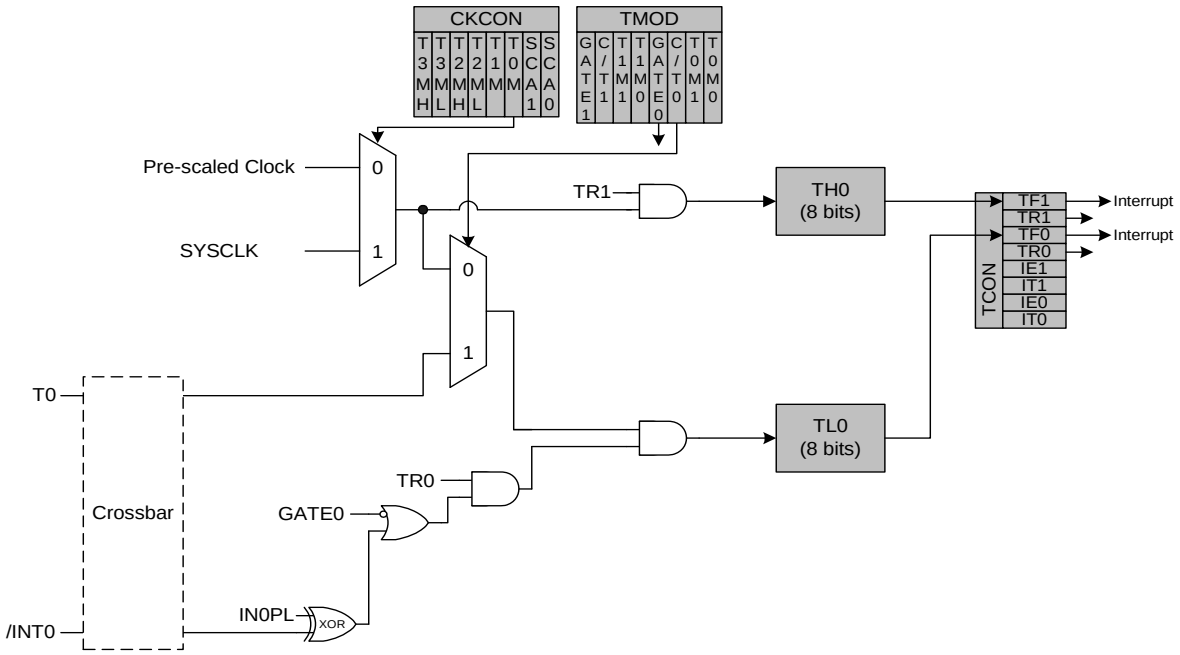



**Figure 24.3. T0 Mode 3 Block Diagram**

## SFR Definition 24.1. TCON: Timer Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0x88

Bit7: TF1: Timer 1 Overflow Flag.
Set 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.
0: No Timer 1 overflow detected.
1: Timer 1 has overflowed.

Bit6: TR1: Timer 1 Run Control.
0: Timer 1 disabled.
1: Timer 1 enabled.

Bit5: TF0: Timer 0 Overflow Flag.
Set 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.
0: No Timer 0 overflow detected.
1: Timer 0 has overflowed.

Bit4: TR0: Timer 0 Run Control.
0: Timer 0 disabled.
1: Timer 0 enabled.

Bit3: IE1: External Interrupt 1.
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 if IT1 = 1. When IT1 = 0, this flag is set to ‘1’ when /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 12.7. “IT01CF: INT0/INT1 Configuration” on page 118).

Bit2: IT1: Interrupt 1 Type Select.
This bit selects whether the configured /INT1 interrupt will be edge or level sensitive. /INT1 is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 12.7. “IT01CF: INT0/INT1 Configuration” on page 118).
0: /INT1 is level triggered.
1: /INT1 is edge triggered.

Bit1: IE0: External Interrupt 0.
This flag is set by hardware when an edge/level of type defined by IT0 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine if IT0 = 1. When IT0 = 0, this flag is set to ‘1’ when /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7. “IT01CF: INT0/INT1 Configuration” on page 118).

Bit0: IT0: Interrupt 0 Type Select.
This bit selects whether the configured /INT0 interrupt will be edge or level sensitive. /INT0 is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 12.7. “IT01CF: INT0/INT1 Configuration” on page 118).
0: /INT0 is level triggered.
1: /INT0 is edge triggered.

## SFR Definition 24.2. TMOD: Timer Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| GATE1 | C/T1 | T1M1 | T1M0 | GATE0 | C/T0 | T0M1 | T0M0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x89

Bit7: GATE1: Timer 1 Gate Control.
0: Timer 1 enabled when TR1 = 1 irrespective of /INT1 logic level.
1: Timer 1 enabled only when TR1 = 1 AND /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 12.7. "IT01CF: INT0/INT1 Configuration" on page 118).

Bit6: C/T1: Counter/Timer 1 Select.
0: Timer Function: Timer 1 incremented by clock defined by T1M bit (CKCON.4).
1: Counter Function: Timer 1 incremented by high-to-low transitions on external input pin (T1).

Bits5–4: T1M1–T1M0: Timer 1 Mode Select.
These bits select the Timer 1 operation mode.

| T1M1 | T1M0 | Mode |
|---|---|---|
| 0 | 0 | Mode 0: 13-bit counter/timer |
| 0 | 1 | Mode 1: 16-bit counter/timer |
| 1 | 0 | Mode 2: 8-bit counter/timer with auto-reload |
| 1 | 1 | Mode 3: Timer 1 inactive |

Bit3: GATE0: Timer 0 Gate Control.
0: Timer 0 enabled when TR0 = 1 irrespective of /INT0 logic level.
1: Timer 0 enabled only when TR0 = 1 AND /INT0 is active as defined by bit IN0PL in register IT01CF (see SFR Definition 12.7. "IT01CF: INT0/INT1 Configuration" on page 118).

Bit2: C/T0: Counter/Timer Select.
0: Timer Function: Timer 0 incremented by clock defined by T0M bit (CKCON.3).
1: Counter Function: Timer 0 incremented by high-to-low transitions on external input pin (T0).

Bits1–0: T0M1–T0M0: Timer 0 Mode Select.
These bits select the Timer 0 operation mode.

| T0M1 | T0M0 | Mode |
|---|---|---|
| 0 | 0 | Mode 0: 13-bit counter/timer |
| 0 | 1 | Mode 1: 16-bit counter/timer |
| 1 | 0 | Mode 2: 8-bit counter/timer with auto-reload |
| 1 | 1 | Mode 3: Two 8-bit counter/timers |

## SFR Definition 24.3. CKCON: Clock Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| T3MH | T3ML | T2MH | T2ML | T1M | T0M | SCA1 | SCA0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8E

Bit7: T3MH: Timer 3 High Byte Clock Select.
This bit selects the clock supplied to the Timer 3 high byte if Timer 3 is configured in split 8-bit timer mode. T3MH is ignored if Timer 3 is in any other mode.
0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN.
1: Timer 3 high byte uses the system clock.

Bit6: T3ML: Timer 3 Low Byte Clock Select.
This bit selects the clock supplied to Timer 3. If Timer 3 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.
0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN.
1: Timer 3 low byte uses the system clock.

Bit5: T2MH: Timer 2 High Byte Clock Select.
This bit selects the clock supplied to the Timer 2 high byte if Timer 2 is configured in split 8-bit timer mode. T2MH is ignored if Timer 2 is in any other mode.
0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN.
1: Timer 2 high byte uses the system clock.

Bit4: T2ML: Timer 2 Low Byte Clock Select.
This bit 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.
0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN.
1: Timer 2 low byte uses the system clock.

Bit3: T1M: Timer 1 Clock Select.
This select the clock source supplied to Timer 1. T1M is ignored when C/T1 is set to logic 1.
0: Timer 1 uses the clock defined by the prescale bits, SCA1-SCA0.
1: Timer 1 uses the system clock.

Bit2: T0M: Timer 0 Clock Select.
This bit selects the clock source supplied to Timer 0. T0M is ignored when C/T0 is set to logic 1.
0: Counter/Timer 0 uses the clock defined by the prescale bits, SCA1-SCA0.
1: Counter/Timer 0 uses the system clock.

Bits1–0: SCA1–SCA0: Timer 0/1 Prescale Bits.
These bits control the division of the clock supplied to Timer 0 and Timer 1 if configured to use prescaled clock inputs.

| SCA1 | SCA0 | Prescaled Clock |
|---|---|---|
| 0 | 0 | System clock divided by 12 |
| 0 | 1 | System clock divided by 4 |
| 1 | 0 | System clock divided by 48 |
| 1 | 1 | External clock divided by 8 |

Note: External clock divided by 8 is synchronized with the system clock.

## SFR Definition 24.4. TL0: Timer 0 Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8A

Bits 7–0: TL0: Timer 0 Low Byte.
The TL0 register is the low byte of the 16-bit Timer 0.

## SFR Definition 24.5. TL1: Timer 1 Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8B

Bits 7–0: TL1: Timer 1 Low Byte.
The TL1 register is the low byte of the 16-bit Timer 1.

## SFR Definition 24.6. TH0: Timer 0 High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8C

Bits 7–0: TH0: Timer 0 High Byte.
The TH0 register is the high byte of the 16-bit Timer 0.

## SFR Definition 24.7. TH1: Timer 1 High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x8D

Bits 7–0: TH1: Timer 1 High Byte.
The TH1 register is the high byte of the 16-bit Timer 1.

## 24.2. 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 clock frequency or the External Oscillator clock frequency.

Timer 2 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. The external oscillator source divided by 8 is synchronized with the system clock.

### 24.2.1. 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, or the external oscillator clock source divided by 8. 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 24.4, 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.

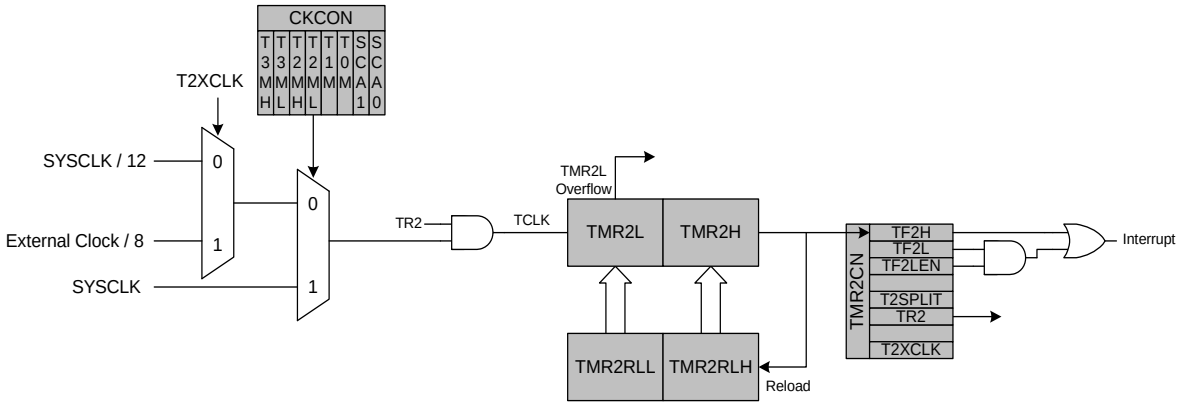



**Figure 24.4. Timer 2 16-Bit Mode Block Diagram**

### 24.2.2. 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 24.5. 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, or the external oscillator clock source divided by 8. 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 bit (T2XCLK in TMR2CN), as follows:

| T2MH | T2XCLK | TMR2H Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 1 | X | SYSCLK |

| T2ML | T2XCLK | TMR2L Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 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.

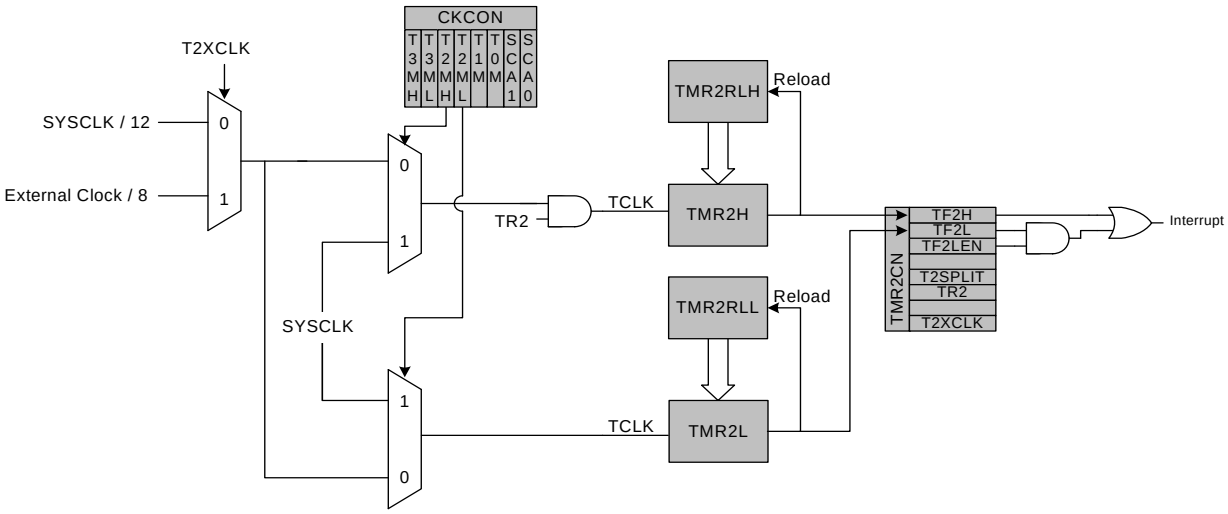



**Figure 24.5. Timer 2 8-Bit Mode Block Diagram**

### 24.2.3. External/smaRTClock Capture Mode

Capture Mode allows either the external oscillator or the smaRTClock clock to be measured against the system clock. The external oscillator and smaRTClock clock can also be compared against each other. Timer 2 can be clocked from the system clock, the system clock divided by 12, the external oscillator divided by 8, or the smaRTClock clock divided by 8, depending on the T2ML (CKCON.4), T2XCLK, and T2RCLK settings. The timer will capture either every 8 external clock cycles or every 8 smaRTClock clock cycles, depending on the T2RCLK setting. When a capture event is generated, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set. By recording the difference between two successive timer capture values, the external oscillator or smaRTClock clock 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. Timer 2 should be in 16-bit auto-reload mode when using Capture Mode.

For example, if T2ML = 1b, T2RCLK = 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:

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 external oscillator frequency when an RC network or capacitor is used to generate the signal.

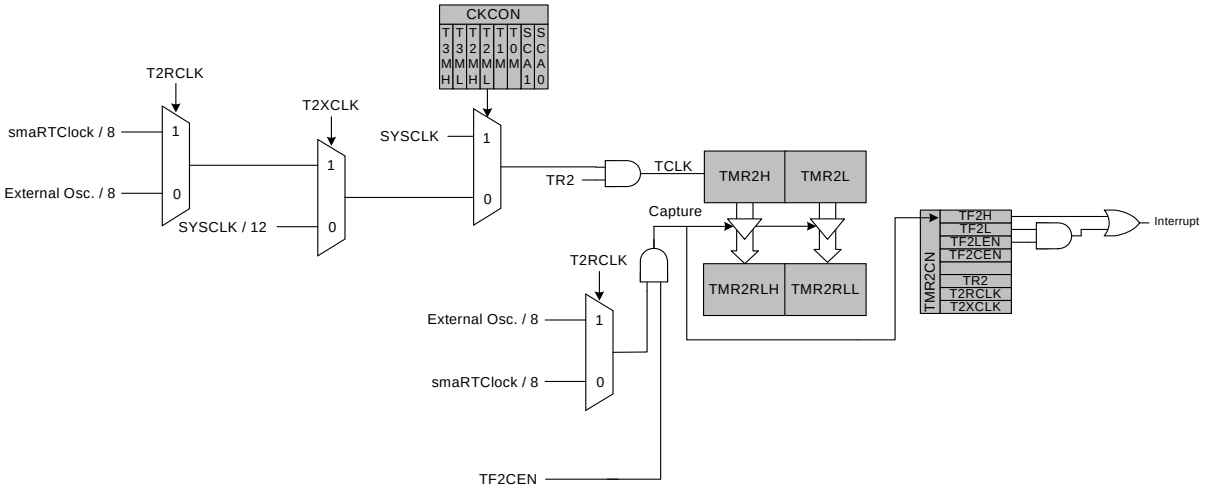



**Figure 24.6. Timer 2 Capture Mode Block Diagram**

## SFR Definition 24.8. TMR2CN: Timer 2 Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| TF2H | TF2L | TF2LEN | TF2CEN | T2SPLIT | TR2 | T2RCLK | T2XCLK | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xC8

Bit7: TF2H: Timer 2 High Byte Overflow Flag.
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. TF2H is not automatically cleared by hardware and must be cleared by software.

Bit6: TF2L: Timer 2 Low Byte Overflow Flag.
Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. When this bit is set, an interrupt will be generated if TF2LEN is set and Timer 2 interrupts are enabled. TF2L will set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware.

Bit5: TF2LEN: Timer 2 Low Byte Interrupt Enable.
This bit enables/disables Timer 2 Low Byte interrupts. If TF2LEN is set and Timer 2 interrupts are enabled, an interrupt will be generated when the low byte of Timer 2 overflows. This bit should be cleared when operating Timer 2 in 16-bit mode.
0: Timer 2 Low Byte interrupts disabled.
1: Timer 2 Low Byte interrupts enabled.

Bit4: TF2CEN. Timer 2 Capture Enable.
0: Timer 2 capture mode disabled.
1: Timer 2 capture mode enabled.

Bit3: T2SPLIT: Timer 2 Split Mode Enable.
When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload.
0: Timer 2 operates in 16-bit auto-reload mode.
1: Timer 2 operates as two 8-bit auto-reload timers.

Bit2: TR2: Timer 2 Run Control.
This bit enables/disables Timer 2. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in this mode.
0: Timer 2 disabled.
1: Timer 2 enabled.

Bit1: T2RCLK: Timer 2 Capture Mode.
This bit controls the Timer 2 capture source when TF2CEN=1. If T2XCLK = 1 and T2ML (CKCON.4) = 0, this bit also controls the clock source for Timer 2.
0: Capture every smaRTClock clock/8. If T2XCLK = 1 and T2ML (CKCON.4) = 0, count at external oscillator/8.
1: Capture every external oscillator/8. If T2XCLK = 1 and T2ML (CKCON.4) = 0, count at smaRTClock clock/8.

Bit0: T2XCLK: Timer 2 External Clock Select.
This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the 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.
0: Timer 2 external clock selection is the system clock divided by 12.
1: Timer 2 external clock uses the clock defined by the T2RCLK bit.

## SFR Definition 24.9. TMR2RLL: Timer 2 Reload Register Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xCA

Bits 7–0: TMR2RLL: Timer 2 Reload Register Low Byte.
TMR2RLL holds the low byte of the reload value for Timer 2.

## SFR Definition 24.10. TMR2RLH: Timer 2 Reload Register High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xCB

Bits 7–0: TMR2RLH: Timer 2 Reload Register High Byte.
The TMR2RLH holds the high byte of the reload value for Timer 2.

## SFR Definition 24.11. TMR2L: Timer 2 Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xCC

Bits 7–0: TMR2L: Timer 2 Low Byte.
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 24.12. TMR2H Timer 2 High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xCD

Bits 7–0: TMR2H: Timer 2 High Byte.
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.

## 24.3. 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 smaRTClock clock frequency or the External Oscillator clock frequency.

Timer 3 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. Note that the external oscillator source divided by 8 is synchronized with the system clock.

### 24.3.1. 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, or the external oscillator clock source divided by 8. 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 TM3RLL) is loaded into the Timer 3 register as shown in Figure 24.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled, 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.

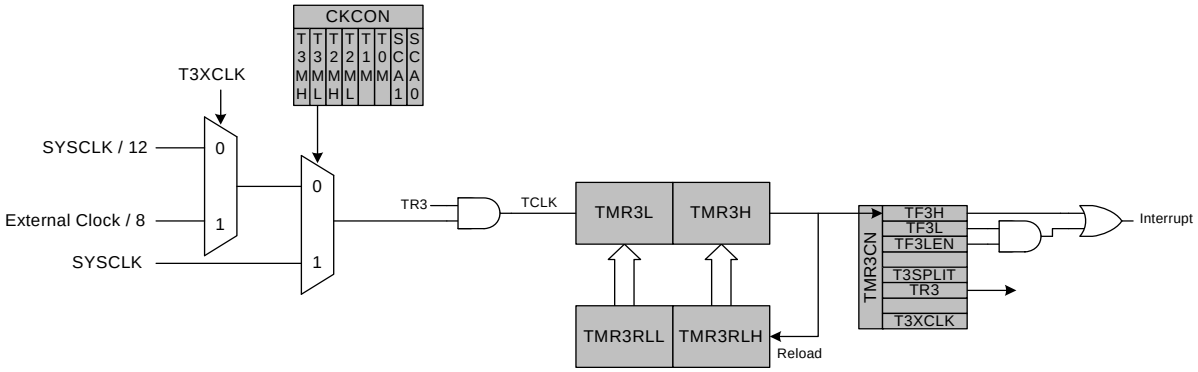



**Figure 24.7. Timer 3 16-Bit Mode Block Diagram**

### 24.3.2. 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 24.8. 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, or the external oscillator clock source divided by 8. 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 bit (T3XCLK in TMR3CN), as follows:

| T3MH | T3XCLK | TMR3H Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 1 | X | SYSCLK |

| T3ML | T3XCLK | TMR3L Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 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.

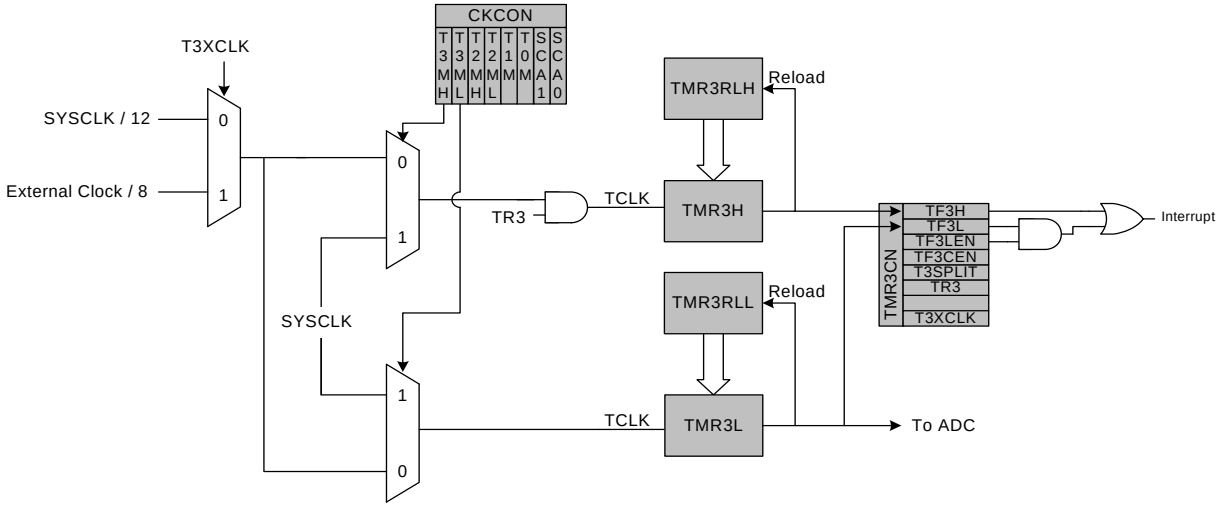



**Figure 24.8. Timer 3 8-Bit Mode Block Diagram**

### 24.3.3. External/smaRTClock Capture Mode

Capture Mode allows either the external oscillator or the smaRTClock clock to be measured against the system clock. The external oscillator and smaRTClock clock can also be compared against each other. Timer 3 can be clocked from the system clock, the system clock divided by 12, the external oscillator divided by 8, or the smaRTClock clock divided by 8, depending on the T3ML (CKCON.6), T3XCLK, and T3RCLK settings. The timer will capture either every 8 external clock cycles or every 8 smaRTClock clock cycles, depending on the T3RCLK setting. When a capture event is generated, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set. By recording the difference between two successive timer capture values, the external oscillator or smaRTClock clock 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. Timer 3 should be in 16-bit auto-reload mode when using Capture Mode.

For example, if T3ML = 1b, T3RCLK = 0b, and TF3CEN = 1b, Timer 3 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:

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 external oscillator frequency when an RC network or capacitor is used to generate the signal.

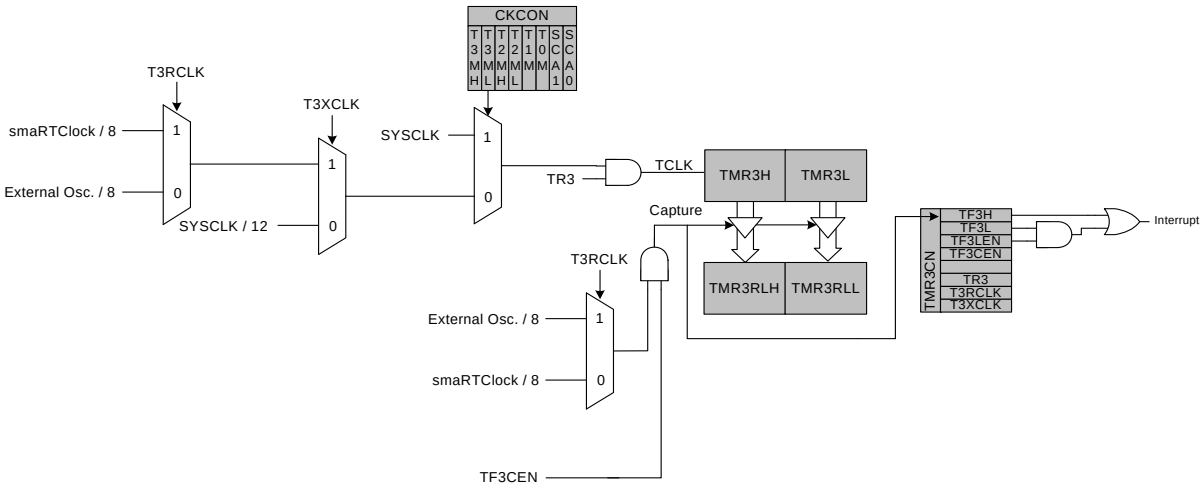



**Figure 24.9. Timer 3 Capture Mode Block Diagram**

## SFR Definition 24.13. TMR3CN: Timer 3 Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| TF3H | TF3L | TF3LEN | TF3CEN | T3SPLIT | TR3 | T3RCLK | T3XCLK | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x91

Bit7: TF3H: Timer 3 High Byte Overflow Flag.
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. TF3H is not automatically cleared by hardware and must be cleared by software.

Bit6: TF3L: Timer 3 Low Byte Overflow Flag.
Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. When this bit is set, an interrupt will be generated if TF3LEN is set and Timer 3 interrupts are enabled. TF3L will set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware.

Bit5: TF3LEN: Timer 3 Low Byte Interrupt Enable.
This bit enables/disables Timer 3 Low Byte interrupts. If TF3LEN is set and Timer 3 interrupts are enabled, an interrupt will be generated when the low byte of Timer 3 overflows. This bit should be cleared when operating Timer 3 in 16-bit mode.
0: Timer 3 Low Byte interrupts disabled.
1: Timer 3 Low Byte interrupts enabled.

Bit4: TF3CEN: Timer 3 Capture Enable.
0: Timer 3 capture mode disabled.
1: Timer 3 capture mode enabled.

Bit3: T3SPLIT: Timer 3 Split Mode Enable.
When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload.
0: Timer 3 operates in 16-bit auto-reload mode.
1: Timer 3 operates as two 8-bit auto-reload timers.

Bit2: TR3: Timer 3 Run Control.
This bit enables/disables Timer 3. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in this mode.
0: Timer 3 disabled.
1: Timer 3 enabled.

Bit1: T3RCLK: Timer 3 Capture Mode.
This bit controls the Timer 3 capture source when TF3CEN=1. If T3XCLK = 1 and T3ML (CKCON.6) = 0, this bit also controls the clock source for Timer 3.
0: Capture every smaRTClock clock/8. If T3XCLK = 1 and T3ML (CKCON.6) = 0, count at external oscillator/8.
1: Capture every external oscillator/8. If T3XCLK = 1 and T3ML (CKCON.6) = 0, count at smaRTClock clock/8.

Bit0: T3XCLK: Timer 3 External Clock Select.
This bit selects the external clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the 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.
0: Timer 3 external clock selection is the system clock divided by 12.
1: Timer 3 external clock uses the clock defined by the T3RCLK bit.

## SFR Definition 24.14. TMR3RLL: Timer 3 Reload Register Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x92

Bits 7-0: TMR3RLL: Timer 3 Reload Register Low Byte.
TMR3RLL holds the low byte of the reload value for Timer 3.

## SFR Definition 24.15. TMR3RLH: Timer 3 Reload Register High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x93

Bits 7-0: TMR3RLH: Timer 3 Reload Register High Byte.
The TMR3RLH holds the high byte of the reload value for Timer 3.

## SFR Definition 24.16. TMR3L: Timer 3 Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x94

Bits 7-0: TMR3L: Timer 3 Low Byte.
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 24.17. TMR3H Timer 3 High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0x95

Bits 7-0: TMR3H: Timer 3 High Byte.
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.

## 25. Programmable Counter Array (PCA0)

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 (See **Section "18.1. Priority Crossbar Decoder" on page 149** for details on configuring the Crossbar). The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, smaRTClock Clock divided by 8, Timer 0 overflow, 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-Bit PWM, or 16-Bit PWM (each mode is described in **Section "25.2. Capture/Compare Modules" on page 251**). The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 25.1

**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 25.3** for details.

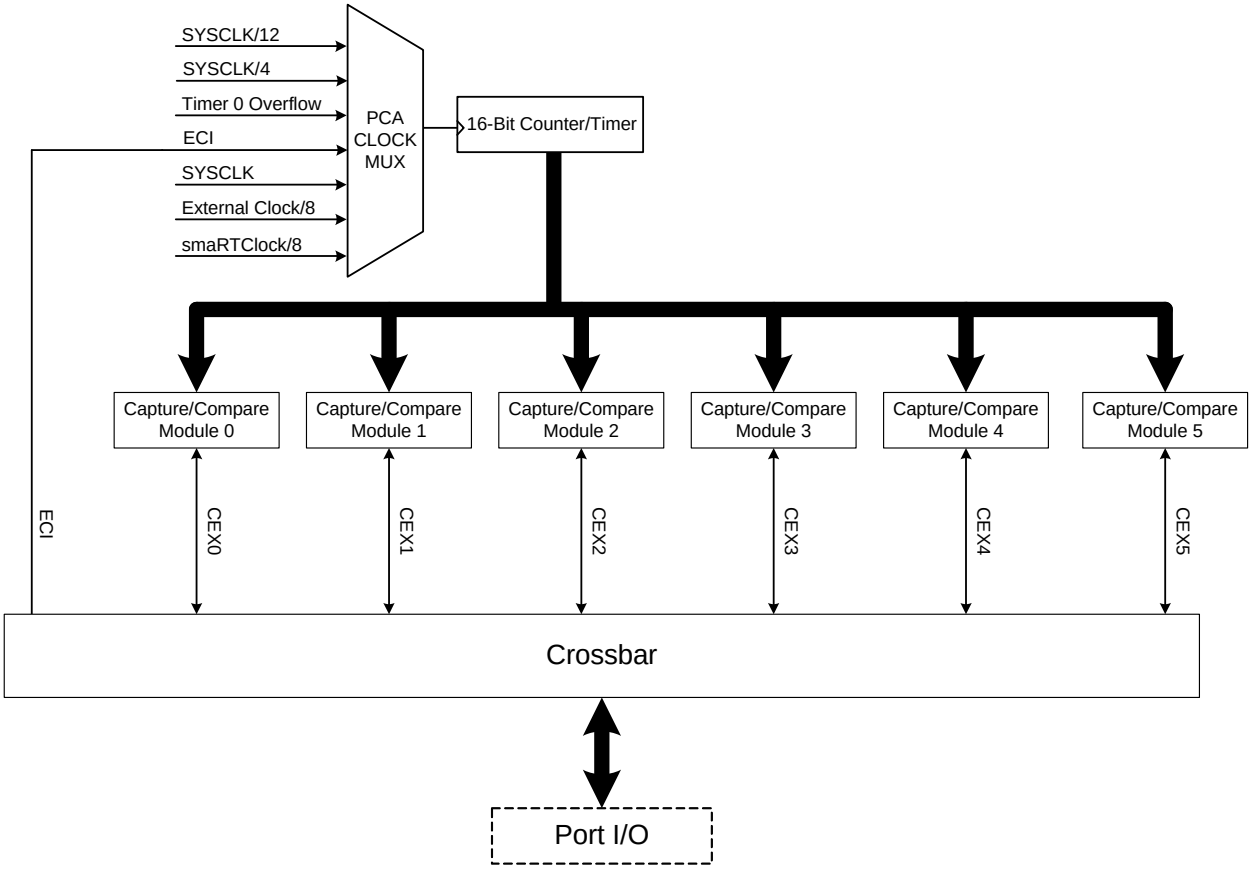



**Figure 25.1. PCA Block Diagram**

## 25.1. 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 25.1.

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 (Note: PCA0 interrupts must be globally enabled before CF interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit (IE.7) and the EPCA0 bit in EIE1 to logic 1). Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 25.1. 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 | smaRTClock clock divided by 8* |
| ***Note:** External clock divided by 8 and smaRTClock clock divided by 8 are synchronized with the system clock. | | | |

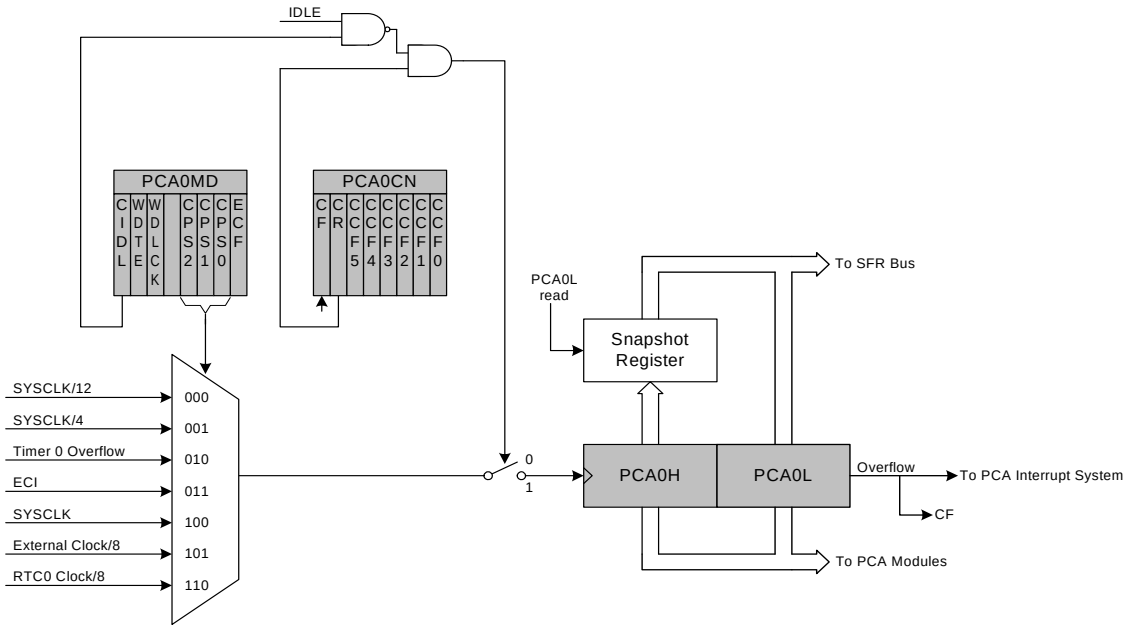



**Figure 25.2. PCA Counter/Timer Block Diagram**

## 25.2. 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-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 25.2 summarizes the bit settings in the PCA0CPMn registers used to select the PCA capture/compare module's operating modes. Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt. Note: PCA0 interrupts must be globally enabled before individual CCFn interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1. See Figure 25.3 for details on the PCA interrupt configuration.

**Table 25.2. PCA0CPM Register Settings for PCA Capture/Compare Modules**

| PWM16 | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF | Operation Mode |
|---|---|---|---|---|---|---|---|---|
| X | X | 1 | 0 | 0 | 0 | 0 | X | Capture triggered by positive edge on CEXn |
| X | X | 0 | 1 | 0 | 0 | 0 | X | Capture triggered by negative edge on CEXn |
| X | X | 1 | 1 | 0 | 0 | 0 | X | Capture triggered by transition on CEXn |
| X | 1 | 0 | 0 | 1 | 0 | 0 | X | Software Timer |
| X | 1 | 0 | 0 | 1 | 1 | 0 | X | High Speed Output |
| X | 1 | 0 | 0 | X | 1 | 1 | X | Frequency Output |
| 0 | 1 | 0 | 0 | X | 0 | 1 | X | 8-Bit Pulse Width Modulator |
| 1 | 1 | 0 | 0 | X | 0 | 1 | X | 16-Bit Pulse Width Modulator |
| X = Don't Care | | | | | | | | |

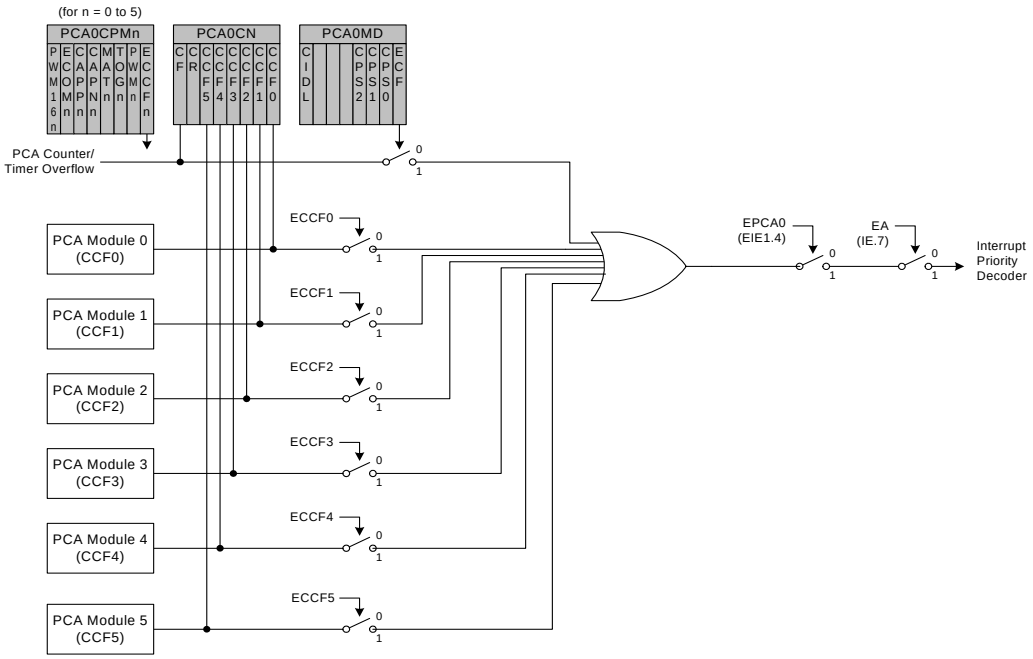



**Figure 25.3. PCA Interrupt Block Diagram**

### 25.2.1. 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 and an interrupt request is generated if CCF interrupts are 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.

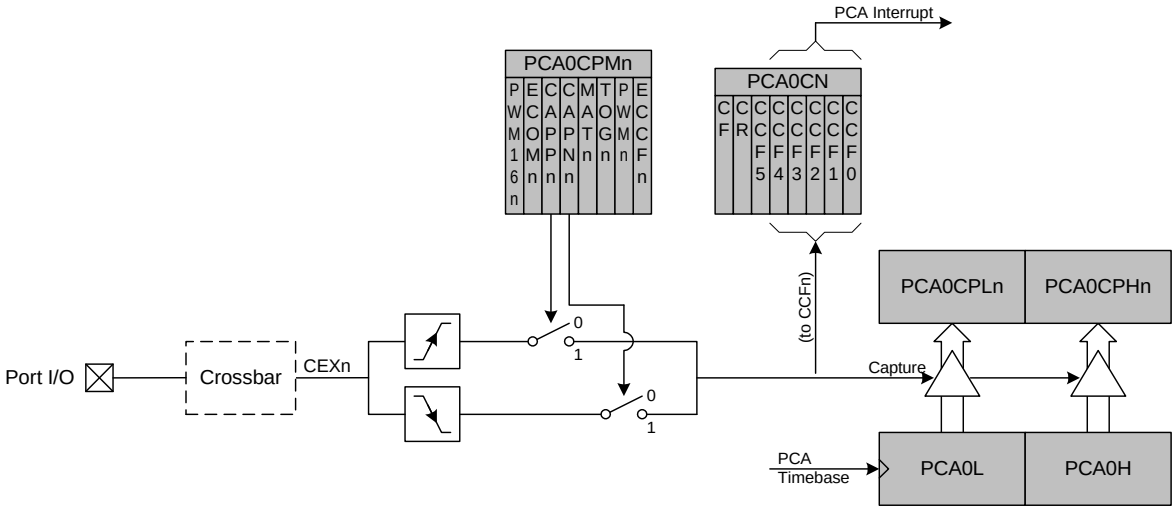



**Figure 25.4. PCA Capture Mode Diagram**

**Note:** The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

### 25.2.2. 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 and an interrupt request is generated if CCF interrupts are 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’.

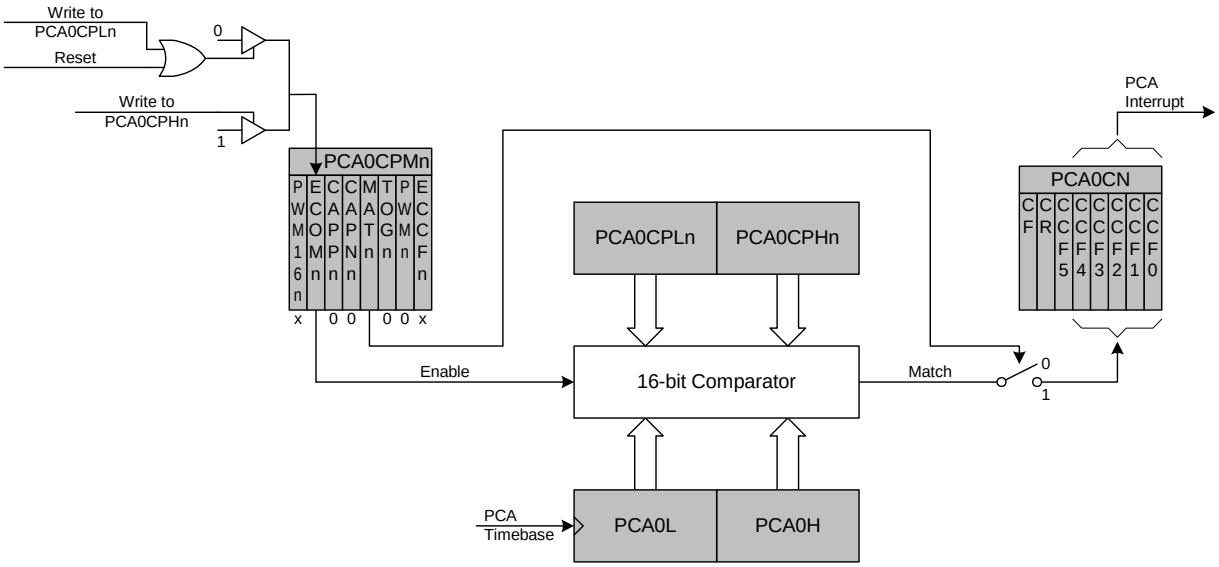



**Figure 25.5. PCA Software Timer Mode Diagram**

### 25.2.3. 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) Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output 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'.

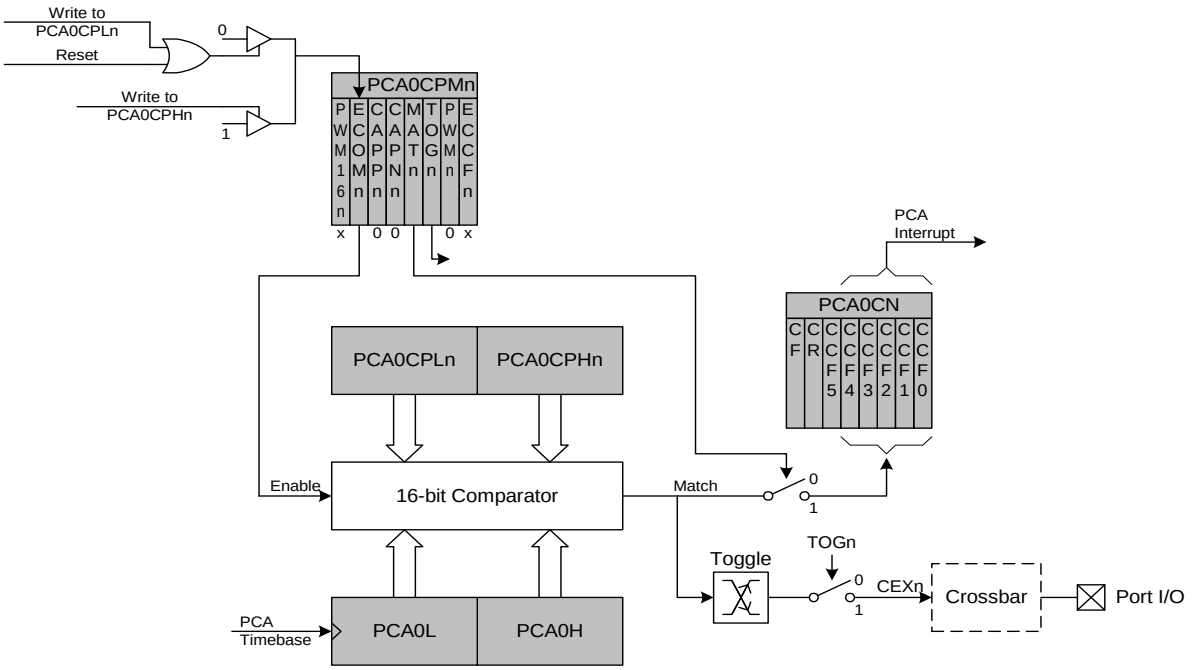



**Figure 25.6. PCA High-Speed Output Mode Diagram**

**Note:** The initial state of the Toggle output is logic 1 and is initialized to this state when the module enters High Speed Output Mode.

### 25.2.4. 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 25.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

**Note:** A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

**Equation 25.1. 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.

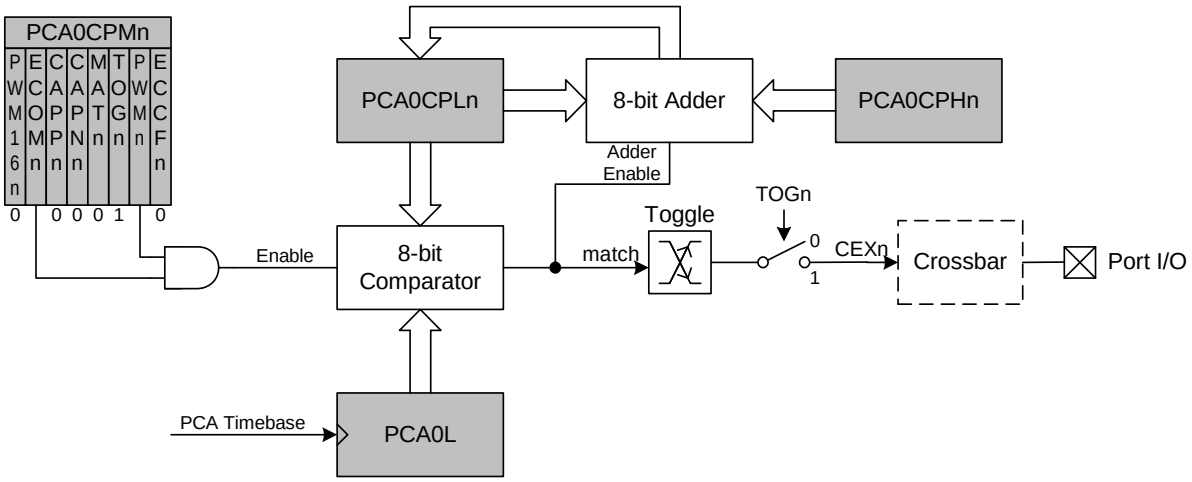



**Figure 25.7. PCA Frequency Output Mode**

### 25.2.5. 8-Bit Pulse Width Modulator Mode

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. The duty cycle of the PWM output signal is varied using the module's PCA0CPHn 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 25.8). 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 enables 8-Bit Pulse Width Modulator mode. The duty cycle for 8-Bit PWM Mode is given by Equation 25.2.

**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'.

$$DutyCycle = \frac{(256 - PCA0CPHn)}{256}$$

**Equation 25.2. 8-Bit PWM Duty Cycle**

Using Equation 25.2, 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'.

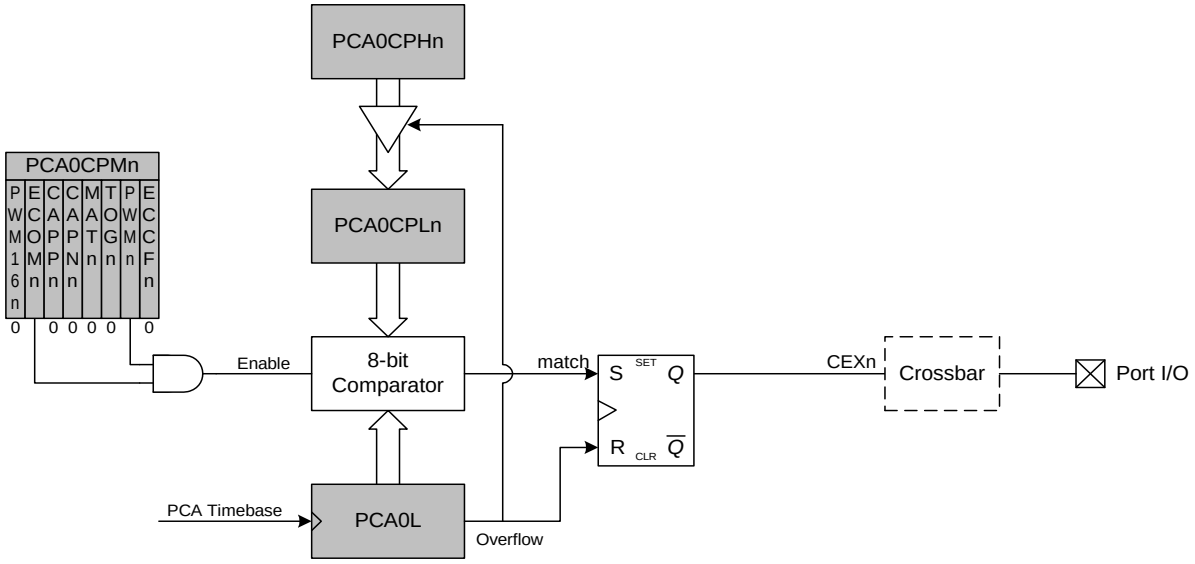



**Figure 25.8. PCA 8-Bit PWM Mode Diagram**

### 25.2.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 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 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. The duty cycle for 16-Bit PWM Mode is given by Equation 25.3.

**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’.

$$DutyCycle = \frac{(65536 - PCA0CPn)}{65536}$$

**Equation 25.3. 16-Bit PWM Duty Cycle**

Using Equation 25.3, 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’.

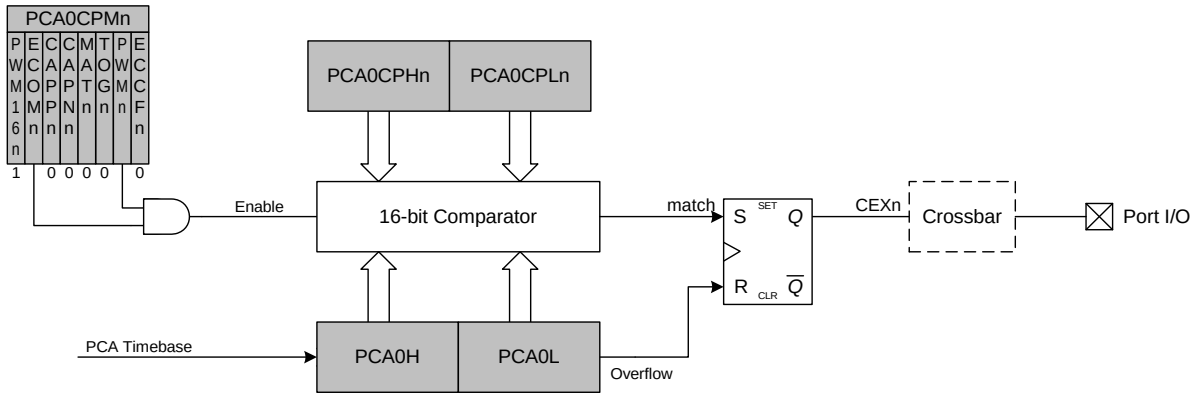



**Figure 25.9. PCA 16-Bit PWM Mode**

## 25.3. 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.**

### 25.3.1. 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 (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 25.10).

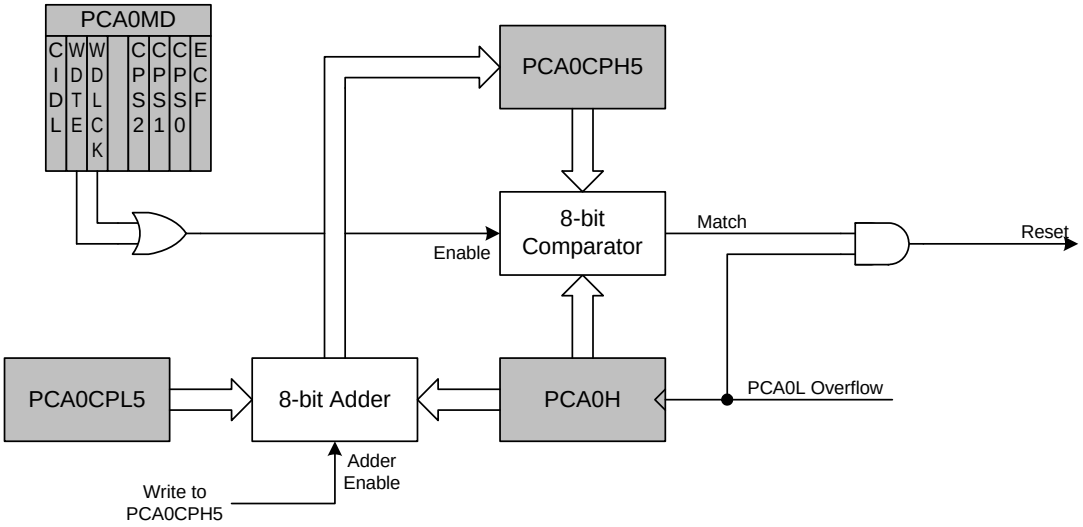



**Figure 25.10. PCA Module 5 with Watchdog Timer Enabled**

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 25.4, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL5) + (256 - PCA0L)$$

**Equation 25.4. 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.

### 25.3.2. 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’.

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 25.4, this results in a WDT timeout interval of 3072 system clock cycles. Table 25.3 lists some example timeout intervals for typical system clocks.

**Table 25.3. 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 |
| 18,432,000 | 255 | 42.7 |
| 18,432,000 | 128 | 21.5 |
| 18,432,000 | 32 | 5.5 |
| 11,059,200 | 255 | 71.1 |
| 11,059,200 | 128 | 35.8 |
| 11,059,200 | 32 | 9.2 |
| 3,062,500 | 255 | 257 |
| 3,062,500 | 128 | 129.5 |
| 3,062,500 | 32 | 33.1 |
| 191,406[2] | 255 | 4109 |
| 191,406[2] | 128 | 2070 |
| 191,406[2] | 32 | 530 |
| 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 oscillator reset frequency.

## 25.4. Register Descriptions for PCA

Following are detailed descriptions of the special function registers related to the operation of the PCA.

### SFR Definition 25.1. PCA0CN: PCA Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CF | CR | CCF5 | CCF4 | CCF3 | CCF2 | CCF1 | CCF0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xD8

Bit7: CF: PCA Counter/Timer Overflow Flag.
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.

Bit6: CR: PCA Counter/Timer Run Control.
This bit enables/disables the PCA Counter/Timer.
0: PCA Counter/Timer disabled.
1: PCA Counter/Timer enabled.

Bit0: CCF5: PCA Module 5 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF5 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.

Bit4: CCF4: PCA Module 4 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF4 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.

Bit3: CCF3: PCA Module 3 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF3 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.

Bit2: CCF2: PCA Module 2 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF2 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.

Bit1: CCF1: PCA Module 1 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF1 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.

Bit0: CCF0: PCA Module 0 Capture/Compare Flag.
This bit is set by hardware when a match or capture occurs. When the CCF0 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 25.2. PCA0MD: PCA Mode

| R/W | R/W | R/W | R | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CIDL | WDTE | WDLCK | - | CPS2 | CPS1 | CPS0 | ECF | 01000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD9

Bit7: CIDL: PCA Counter/Timer Idle Control.
Specifies PCA behavior when CPU is in Idle Mode.
0: PCA continues to function normally while the system controller is in Idle Mode.
1: PCA operation is suspended while the system controller is in Idle Mode.

Bit6: WDTE: Watchdog Timer Enable
If this bit is set, PCA Module 5 is used as the watchdog timer.
0: Watchdog Timer disabled.
1: PCA Module 5 enabled as Watchdog Timer.

Bit5: WDLCK: Watchdog Timer Lock
This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.
0: Watchdog Timer Enable unlocked.
1: Watchdog Timer Enable locked.

Bit4: UNUSED. Read = 0b, Write = don't care.

Bits3–1: CPS2–CPS0: PCA Counter/Timer Pulse Select.
These bits select the timebase source for the PCA counter.

| 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 clock divided by 8* |
| 1 | 1 | 0 | smaRTClock clock divided by 8* |
| 1 | 1 | 1 | Reserved |

***Note:** External clock divided by 8 and smaRTClock clock divided by 8 are synchronized with the system clock.

Bit0: ECF: PCA Counter/Timer Overflow Interrupt Enable.
This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.
0: Disable the CF interrupt.
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 PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled.**

## SFR Definition 25.3. PCA0CPMn: PCA Capture/Compare Mode

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| PWM16n | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: PCA0CPM0: 0xDA, PCA0CPM1: 0xDB, PCA0CPM2: 0xDC, PCA0CPM3: 0xDD, PCA0CPM4: 0xDE, PCA0CPM5: 0xCE

Bit7: PWM16n: 16-bit Pulse Width Modulation Enable.
This bit selects 16-bit mode when Pulse Width Modulation mode is enabled (PWMn = 1).
0: 8-bit PWM selected.
1: 16-bit PWM selected.

Bit6: ECOMn: Comparator Function Enable.
This bit enables/disables the comparator function for PCA module n.
0: Disabled.
1: Enabled.

Bit5: CAPPn: Capture Positive Function Enable.
This bit enables/disables the positive edge capture for PCA module n.
0: Disabled.
1: Enabled.

Bit4: CAPNn: Capture Negative Function Enable.
This bit enables/disables the negative edge capture for PCA module n.
0: Disabled.
1: Enabled.

Bit3: MATn: Match Function Enable.
This bit enables/disables the match function for PCA module n. 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.
0: Disabled.
1: Enabled.

Bit2: TOGn: Toggle Function Enable.
This bit enables/disables the toggle function for PCA module n. 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.
0: Disabled.
1: Enabled.

Bit1: PWMn: Pulse Width Modulation Mode Enable.
This bit enables/disables the PWM function for PCA module n. When enabled, a pulse width modulated signal is output on the CEXn pin. 8-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: Disabled.
1: Enabled.

Bit0: ECCFn: Capture/Compare Flag Interrupt Enable.
This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt.
0: Disable CCFn interrupts.
1: Enable a Capture/Compare Flag interrupt request when CCFn is set.

## SFR Definition 25.4. PCA0L: PCA Counter/Timer Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xF9

Bits 7–0: PCA0L: PCA Counter/Timer Low Byte.
The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

## SFR Definition 25.5. PCA0H: PCA Counter/Timer High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | SFR Address: |

SFR Address: 0xFA

Bits 7–0: PCA0H: PCA Counter/Timer High Byte.
The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer.

## SFR Definition 25.6. PCA0CPLn: PCA Capture Module Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: PCA0CPL0: 0xFB, PCA0CPL1: 0xE9, PCA0CPL2: 0xEB, PCA0CPL3: 0xED, PCA0CPL4: 0xFD, PCA0CPL5: 0xD2

Bits7–0: PCA0CPLn: PCA Capture Module Low Byte.
The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n.

## SFR Definition 25.7. PCA0CPHn: PCA Capture Module High Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: PCA0CPH0: 0xFC, PCA0CPH1: 0xEA, PCA0CPH2: 0xEC, PCA0CPH3: 0xEE, PCA0CPH4: 0xFE, PCA0CPH5: 0xD3

Bits7–0: PCA0CPHn: PCA Capture Module High Byte.
The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n.

# 26. C2 Interface

C8051F41x devices include an on-chip Silicon Laboratories 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.

## 26.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming functions through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

### C2 Register Definition 26.1. C2ADD: C2 Address

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |

Bits7-0: 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 (DEVICEID) |
| 0x01 | Selects the Revision ID register for Data Read instructions (REVID) |
| 0x02 | Selects the C2 Flash Programming Control register for Data Read/Write instructions (FPCTL) |
| 0xB4 | Selects the C2 Flash Programming Data register for Data Read/Write instructions (FPDAT) |

### C2 Register Definition 26.2. DEVICEID: C2 Device ID

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00001011 |

This read-only register returns the 8-bit device ID: 0x0C (C8051F41x).

## C2 Register Definition 26.3. REVID: C2 Revision ID

| | | | | | | | | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

This read-only register returns the 8-bit revision ID: 0x00 (Revision A).

## C2 Register Definition 26.4. FPCTL: C2 Flash Programming Control

| | | | | | | | | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

Bits7-0 FPCTL: Flash Programming Control Register.
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 26.5. FPDAT: C2 Flash Programming Data

| | | | | | | | | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

Bits7-0: FPDAT: C2 Flash Programming Data Register.
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 |

## 26.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and Flash programming functions 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 (/RST) and C2D (P2.0) 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 26.1.

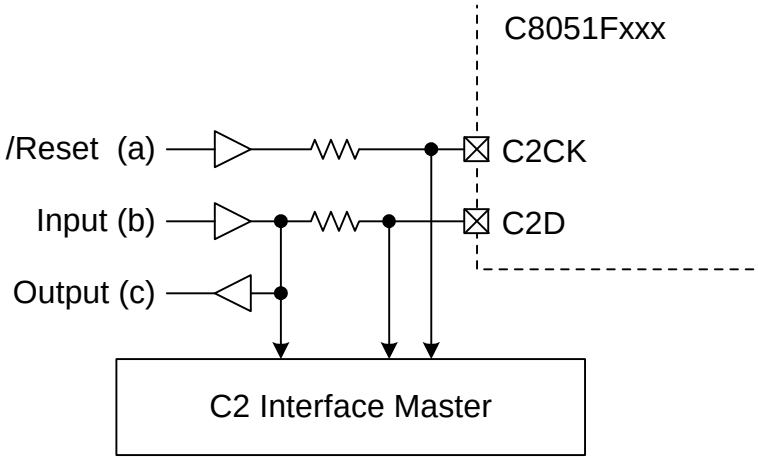



**Figure 26.1. Typical C2 Pin Sharing**

The configuration in Figure 26.1 assumes the following:

1. The user input (b) cannot change state while the target device is halted.
2. The /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 0.7 to Revision 0.8

- Updated specification tables with most recently available characterization data.
- Corrected references to configuring pins for Analog Mode - Port Latch must contain a '1'.
- SFR Definition 5.6: Address correction to 0xBA.
- Added Figure 8.2 showing power connection diagram without using on-chip regulator.
- Section 9 : Removed references to "High Speed Analog Mode".
- Table 11.2 : Corrected SFR Name P2MDIN on location 0xF3.
- Section 14 : Corrected operational description of CRC engine.
- Section 18, Important Note on page 151 : Added "and have the same behavior as P0 in Normal Mode." to last sentence.
- Section 19.2.2 : Inserted Step 3 "Release the crystal pins by writing ‘1's to the port latch."
- Section 19.3 : Added Figure 19.3 and text to describe behavior of clock multiplier with slower input frequencies.
- Section 21: Corrected SMBus maximum rate to 1/20th system clock.
- Table 21.4 : Made corrections to SMBus state descriptions.
- Figure 24.6 : Corrected T2RCLK Mux selection options.
- Figure 24.9 : Corrected T3RCLK Mux selection options.
- C2 Register Definition 26.2 : Corrected DEVICEID value to 0x0C.

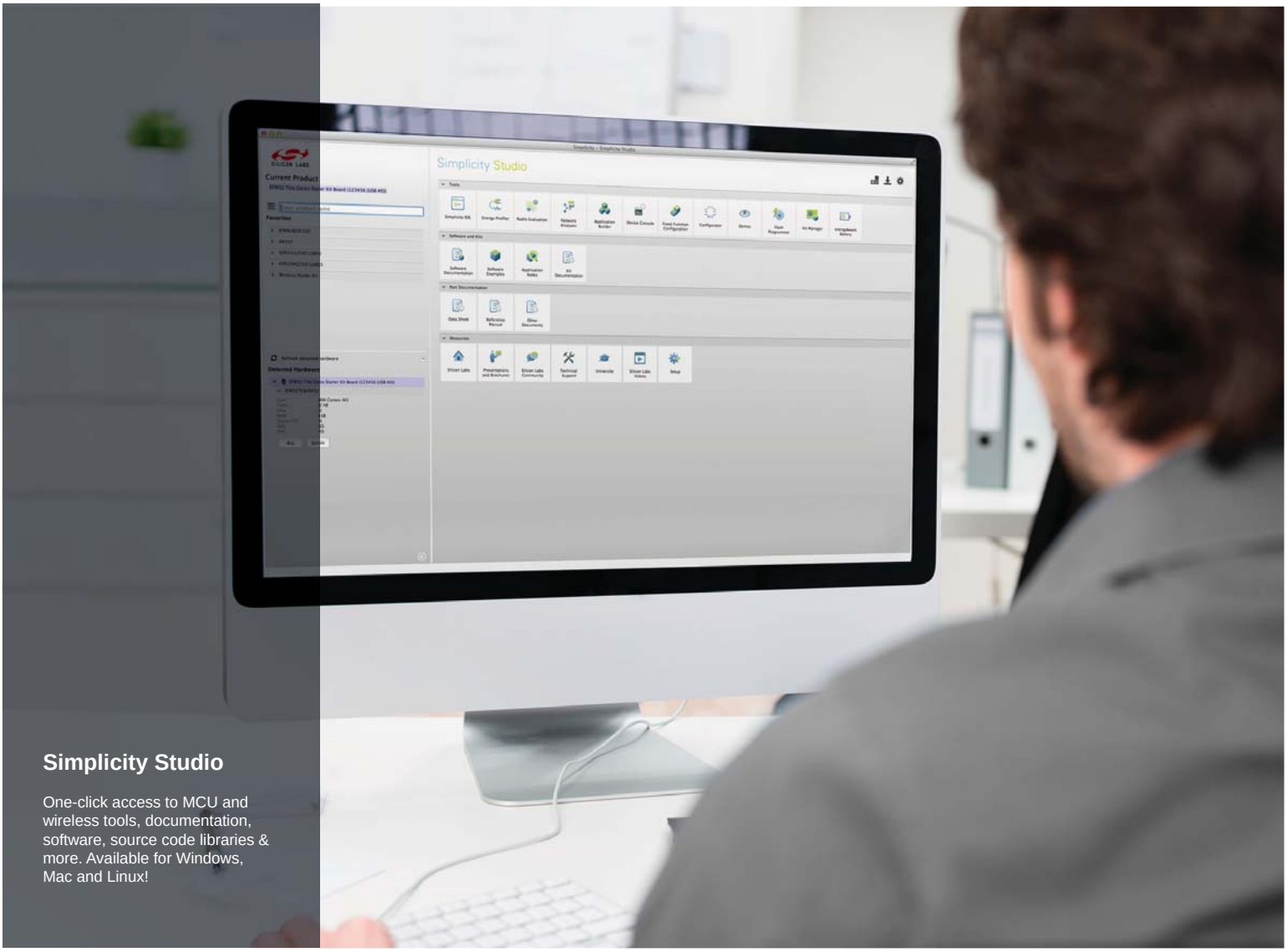
## Revision 0.8 to Revision 1.0

- Updated specification tables with full characterization data.
- Updated Flash write and erase procedures to include a write to FLSCL.3-0.
- Changed /RST pin comments in Table 4.1, “Pin Definitions for the C8051F41x,” on page 41 for the recommended pull-up resistor.
- Changed the reset value of the SFR Definition 16.3. FLSCL: Flash Scale.
- Removed the "Optional GND Connection" from Figure 4.5. ’Typical QFN-28 Landing Diagram’ on page 48.
- Added a note regarding the maximum SYSCLK frequency to SFR Definition 19.4. CLKMUL: Clock Multiplier Control.

## Revision 1.0 to Revision 1.1

- Updated Figure 4.3. ’LQFP-32 Package Diagram’ on page 46, Figure 4.5. ’QFN-28 Package Drawing’ on page 48, and Figure 4.6. ’QFN-28 Recommended PCB Land Pattern’ on page 49.
- Added note that VIO must be ≥ VDD in Table 3.1, “Global DC Electrical Characteristics,” on page 36.
- Added information about ADC0 output register auto-clearing in SFR Definition 5.2.
- Corrected ADC0 Tracking time equation in SFR Definition 5.6.
- Clarified Voltage Regulator Electrical Specifications in Table 8.1 on page 82.
- Added information about 16-bit and 32-bit CRC algorithms in Section 14.

## NOTES:




## Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

**IoT Portfolio**
*www.silabs.com/IoT*

**SW/HW**
*www.silabs.com/simplicity*

**Quality**
*www.silabs.com/quality*

**Support and Community**
*community.silabs.com*

**Disclaimer**
Silicon Laboratories intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Laboratories products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Laboratories reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Laboratories shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Silicon Laboratories. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Laboratories products are generally not intended for military applications. Silicon Laboratories products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

**Trademark Information**
Silicon Laboratories Inc., Silicon Laboratories, Silicon Labs, SiLabs and the Silicon Labs logo, CMEMS®, EFM, EFM32, EFR, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZMac®, EZRadio®, EZRadioPRO®, DSPLL®, ISOmodem ®, Precision32®, ProSLIC®, SiPHY®, USBXpress® and others are trademarks or registered trademarks of Silicon Laboratories Inc. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

**Silicon Laboratories Inc.**
**400 West Cesar Chavez**
**Austin, TX 78701**
**USA**

**http://www.silabs.com**

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Silicon Laboratories:

C8051F411-GM C8051F412-GQ C8051F413-GM C8051F411-GMR C8051F412-GQR C8051F413-GMR