# C8051F99x-C8051F98x

## Ultra Low Power, 8-2 kB Flash, Capacitive Sensing MCU

### Ultra Low Power Consumption
- 150 µA/MHz in active mode (24.5 MHz clock)
- 2 µs wakeup time
- 10 nA sleep mode with memory retention
- 50 nA sleep mode with brownout detector
- 300 nA sleep mode with LFO
- 600 nA sleep mode with external crystal

### Supply Voltage 1.8 to 3.6 V
- Built-in LDO regulator allows a high analog supply voltage and low digital core voltage
- 2 built-in supply monitors (brownout detector) for sleep mode and active modes

### 12-Bit or 10-Bit Analog to Digital Converter
- ±1 LSB INL (10-bit mode); ±1.5 LSB INL (12-bit mode) no missing codes
- Programmable throughput up to 300 ksps (10-bit mode) or 75 ksps (12-bit mode)
- Up to 10 external inputs
- On-chip voltage reference; 0.5x gain allows measuring voltages up to twice the reference voltage
- 16-bit auto-averaging accumulator with burst mode provides increased ADC resolution
- Data dependent windowed interrupt generator
- Built-in temperature sensor

### Capacitive Sense Interface (F99x)
- Supports buttons, sliders, wheels, and capacitive proximity sensing
- Fast 40 µs per channel conversion time
- 16-bit resolution, up to 14 input channels
- Auto scan and wake-on-touch
- Auto-accumulate up to 64x samples

### Analog Comparator
- Programmable hysteresis and response time
- Configurable as wake-up or reset source

### 6-Bit Programmable Current Reference
- Up to ±500 µA, can be used as a bias or for generating a custom reference voltage
- PWM enhanced resolution mode

### High-Speed 8051 µC Core
- Pipelined instruction architecture; executes 70% of instructions in 1 or 2 system clocks
- Up to **25 MIPS** throughput with 25 MHz clock
- Expanded interrupt handler

### Memory
- 512 bytes RAM
- 8 kB (F990/1/6/7, F980/1/6/7), 4 kB (F982/3/8/9), or 2 kB (F985) Flash; in-system programmable

### Digital Peripherals
- Up to 17 port I/O; high sink current and programmable drive strength

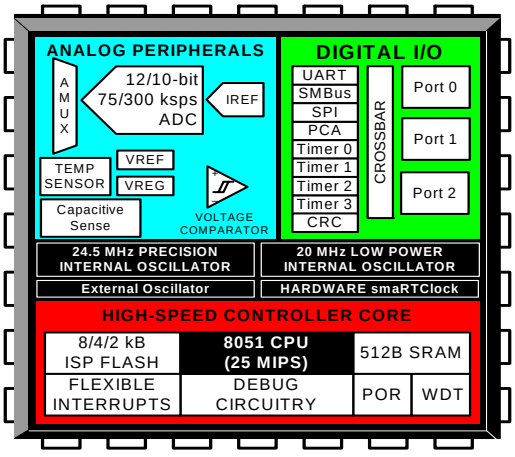
  Hardware SMBus™/I$^2$C™, SPI™, and UART serial ports available concurrently
- Four general purpose 16-bit counter/timers
- Programmable 16-bit counter/timer array with three capture/compare modules and watchdog timer

### Clock Sources
- Internal oscillators: 24.5 MHz, 2% accuracy supports UART operation; 20 MHz low power oscillator requires very little bias current.
- External oscillator: Crystal, RC, C, or CMOS Clock
- SmaRTClock oscillator: 32 kHz Crystal or internal
- Can switch between clock sources on-the-fly; useful in implementing various power saving modes

### 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

### Packages
- 20-pin QFN (3 x 3 mm)
- 24-pin QFN (4 x 4 mm)
- 24-pin QSOP (easy to hand-solder)

### Temperature Range: –40 to +85 °C





Copyright © 2014 by Silicon Laboratories

## Table of Contents

**1. System Overview ..... 17**
1.1. CIP-51™ Microcontroller Core ..... 25
1.1.1. Fully 8051 Compatible ..... 25
1.1.2. Improved Throughput ..... 25
1.1.3. Additional Features ..... 25
1.2. Port Input/Output ..... 26
1.3. Serial Ports ..... 27
1.4. Programmable Counter Array ..... 27
1.5. SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode ..... 28
1.6. Programmable Current Reference (IREF0) ..... 29
1.7. Comparator ..... 29
**2. Ordering Information ..... 31**
**3. Pinout and Package Definitions ..... 32**
**4. Electrical Characteristics ..... 48**
4.1. Absolute Maximum Specifications ..... 48
4.2. Electrical Characteristics ..... 49
**5. SAR ADC with 16-bit Auto-Averaging Accumulator and Autonomous Low Power Burst Mode ..... 66**
5.1. Output Code Formatting ..... 67
5.2. Modes of Operation ..... 68
5.2.1. Starting a Conversion ..... 68
5.2.2. Tracking Modes ..... 69
5.2.3. Burst Mode ..... 70
5.2.4. Settling Time Requirements ..... 71
5.2.5. Gain Setting ..... 72
5.3. 8-Bit Mode ..... 72
5.4. 12-Bit Mode (C8051F980/6 and C8051F990/6 devices only) ..... 72
5.5. Low Power Mode ..... 72
5.6. Programmable Window Detector ..... 80
5.6.1. Window Detector In Single-Ended Mode ..... 82
5.6.2. ADC0 Specifications ..... 82
5.7. ADC0 Analog Multiplexer ..... 83
5.8. Temperature Sensor ..... 85
5.8.1. Calibration ..... 86
5.9. Voltage and Ground Reference Options ..... 88
5.10. External Voltage Reference ..... 89
5.11. Internal Voltage Reference ..... 89
5.12. Analog Ground Reference ..... 89
5.13. Temperature Sensor Enable ..... 89
5.14. Voltage Reference Electrical Specifications ..... 90
**6. Programmable Current Reference (IREF0) ..... 91**
6.1. PWM Enhanced Mode ..... 91

6.2. IREF0 Specifications ..... 92
**7. Comparator ..... 93**
7.1. Comparator Inputs ..... 93
7.2. Comparator Outputs ..... 94
7.3. Comparator Response Time ..... 94
7.4. Comparator Hysteresis ..... 94
7.5. Comparator Register Descriptions ..... 95
7.6. Comparator0 Analog Multiplexer ..... 98
**8. Capacitive Sense (CS0) ..... 100**
8.1. Configuring Port Pins as Capacitive Sense Inputs ..... 101
8.2. Initializing the Capacitive Sensing Peripheral ..... 101
8.3. Capacitive Sense Start-Of-Conversion Sources ..... 101
8.4. CS0 Multiple Channel Enable ..... 102
8.5. CS0 Gain Adjustment ..... 102
8.6. Wake from Suspend ..... 102
8.7. Using CS0 in Applications that Utilize Sleep Mode ..... 102
8.8. Automatic Scanning (Method 1—CS0SMEN = 0) ..... 103
8.9. Automatic Scanning (Method 2—CS0SMEN = 1) ..... 104
8.10. CS0 Comparator ..... 104
8.11. CS0 Conversion Accumulator ..... 105
8.12. CS0 Pin Monitor ..... 106
8.13. Adjusting CS0 For Special Situations ..... 106
8.14. Capacitive Sense Multiplexer ..... 117
**9. CIP-51 Microcontroller ..... 119**
9.1. Performance ..... 119
9.2. Programming and Debugging Support ..... 120
9.3. Instruction Set ..... 120
9.3.1. Instruction and CPU Timing ..... 120
9.4. CIP-51 Register Descriptions ..... 125
**10. Memory Organization ..... 128**
10.1. Program Memory ..... 129
10.1.1. MOVX Instruction and Program Memory ..... 129
10.2. Data Memory ..... 129
10.2.1. Internal RAM ..... 129
10.2.2. External RAM ..... 130
**11. On-Chip XRAM ..... 131**
11.1. Accessing XRAM ..... 131
11.1.1. 16-Bit MOVX Example ..... 131
11.1.2. 8-Bit MOVX Example ..... 131
**12. Special Function Registers ..... 132**
12.1. SFR Paging ..... 133
**13. Interrupt Handler ..... 138**
13.1. Enabling Interrupt Sources ..... 138
13.2. MCU Interrupt Sources and Vectors ..... 138
13.3. Interrupt Priorities ..... 139

13.4.Interrupt Latency ..... 139
13.5.Interrupt Register Descriptions ..... 141
13.6.External Interrupts INT0 and INT1 ..... 148
**14. Flash Memory ..... 150**
14.1.Programming the Flash Memory ..... 150
14.1.1.Flash Lock and Key Functions ..... 150
14.1.2.Flash Erase Procedure ..... 151
14.1.3.Flash Write Procedure ..... 151
14.2.Non-volatile Data Storage ..... 151
14.3.Security Options ..... 152
14.4.Determining the Device Part Number at Run Time ..... 154
14.5.Flash Write and Erase Guidelines ..... 156
14.5.1.$V_{DD}$ Maintenance and the $V_{DD}$ Monitor ..... 156
14.5.2.PSWE Maintenance ..... 157
14.5.3.System Clock ..... 157
14.6.Minimizing Flash Read Current ..... 158
**15. Power Management ..... 162**
15.1.Normal Mode ..... 163
15.2.Idle Mode ..... 164
15.3.Stop Mode ..... 164
15.4.Suspend Mode ..... 165
15.5.Sleep Mode ..... 165
15.6.Configuring Wakeup Sources ..... 166
15.7.Determining the Event that Caused the Last Wakeup ..... 167
15.8.Power Management Specifications ..... 171
**16. Cyclic Redundancy Check Unit (CRC0) ..... 172**
16.1.CRC Algorithm ..... 172
16.2.Preparing for a CRC Calculation ..... 174
16.3.Performing a CRC Calculation ..... 174
16.4.Accessing the CRC0 Result ..... 174
16.5.CRC0 Bit Reverse Feature ..... 179
**17. Voltage Regulator (VREG0) ..... 180**
17.1.Voltage Regulator Electrical Specifications ..... 180
**18. Reset Sources ..... 181**
18.1.Power-On Reset ..... 182
18.2.Power-Fail Reset ..... 183
18.3.External Reset ..... 184
18.4.Missing Clock Detector Reset ..... 185
18.5.Comparator0 Reset ..... 185
18.6.PCA Watchdog Timer Reset ..... 185
18.7.Flash Error Reset ..... 185
18.8.SmaRTClock (Real Time Clock) Reset ..... 186
18.9.Software Reset ..... 186
**19. Clocking Sources ..... 188**
19.1.Programmable Precision Internal Oscillator ..... 189

19.2.Low Power Internal Oscillator ..... 189
19.3.External Oscillator Drive Circuit ..... 189
19.3.1.External Crystal Mode ..... 189
19.3.2.External RC Mode ..... 191
19.3.3.External Capacitor Mode ..... 192
19.3.4.External CMOS Clock Mode ..... 192
19.4.Special Function Registers for Selecting and Configuring the System Clock 193
**20. SmaRTClock (Real Time Clock) ..... 197**
20.1.SmaRTClock Interface ..... 198
20.1.1.SmaRTClock Lock and Key Functions ..... 198
20.1.2.Using RTC0ADR and RTC0DAT to Access SmaRTClock Internal Registers ..... 199
20.1.3.RTC0ADR Short Strobe Feature ..... 199
20.1.4.SmaRTClock Interface Autoread Feature ..... 199
20.1.5.RTC0ADR Autoincrement Feature ..... 200
20.2.SmaRTClock Clocking Sources ..... 203
20.2.1.Using the SmaRTClock Oscillator with a Crystal or External CMOS Clock ..... 203
20.2.2.Using the SmaRTClock Oscillator in Self-Oscillate Mode ..... 204
20.2.3.Using the Low Frequency Oscillator (LFO) ..... 204
20.2.4.Programmable Load Capacitance ..... 205
20.2.5.Automatic Gain Control (Crystal Mode Only) and SmaRTClock Bias Doubling ..... 206
20.2.6.Missing SmaRTClock Detector ..... 208
20.2.7.SmaRTClock Oscillator Crystal Valid Detector ..... 208
20.3.SmaRTClock Timer and Alarm Function ..... 208
20.3.1.Setting and Reading the SmaRTClock Timer Value ..... 208
20.3.2.Setting a SmaRTClock Alarm ..... 209
20.3.3.Software Considerations for using the SmaRTClock Timer and Alarm . 210
**21. Port Input/Output ..... 215**
21.1.Port I/O Modes of Operation ..... 216
21.1.1.Port Pins Configured for Analog I/O ..... 216
21.1.2.Port Pins Configured For Digital I/O ..... 216
21.1.3.Interfacing Port I/O to 5 V Logic ..... 217
21.1.4.Increasing Port I/O Drive Strength ..... 217
21.2.Assigning Port I/O Pins to Analog and Digital Functions ..... 217
21.2.1.Assigning Port I/O Pins to Analog Functions ..... 217
21.2.2.Assigning Port I/O Pins to Digital Functions ..... 218
21.2.3.Assigning Port I/O Pins to External Digital Event Capture Functions .... 218
21.3.Priority Crossbar Decoder ..... 219
21.4.Port Match ..... 225
21.5.Special Function Registers for Accessing and Configuring Port I/O ..... 227
**22. SMBus ..... 235**
22.1.Supporting Documents ..... 236
22.2.SMBus Configuration ..... 236

22.3.SMBus Operation ..... 237
22.3.1.Transmitter vs. Receiver ..... 237
22.3.2.Arbitration ..... 238
22.3.3.Clock Low Extension ..... 238
22.3.4.SCL Low Timeout ..... 238
22.3.5.SCL High (SMBus Free) Timeout ..... 238
22.4.Using the SMBus ..... 239
22.4.1.SMBus Configuration Register ..... 240
22.4.2.SMB0CN Control Register ..... 243
22.4.3.Hardware Slave Address Recognition ..... 246
22.4.4.Data Register ..... 248
22.5.SMBus Transfer Modes ..... 249
22.5.1.Write Sequence (Master) ..... 249
22.5.2.Read Sequence (Master) ..... 250
22.5.3.Write Sequence (Slave) ..... 251
22.5.4.Read Sequence (Slave) ..... 252
22.6.SMBus Status Decoding ..... 252
**23. UART0 ..... 257**
23.1.Enhanced Baud Rate Generation ..... 258
23.2.Operational Modes ..... 259
23.2.1.8-Bit UART ..... 259
23.2.2.9-Bit UART ..... 260
23.3.Multiprocessor Communications ..... 260
**24. Enhanced Serial Peripheral Interface (SPI0) ..... 265**
24.1.Signal Descriptions ..... 266
24.1.1.Master Out, Slave In (MOSI) ..... 266
24.1.2.Master In, Slave Out (MISO) ..... 266
24.1.3.Serial Clock (SCK) ..... 266
24.1.4.Slave Select (NSS) ..... 266
24.2.SPI0 Master Mode Operation ..... 266
24.3.SPI0 Slave Mode Operation ..... 268
24.4.SPI0 Interrupt Sources ..... 269
24.5.Serial Clock Phase and Polarity ..... 269
24.6.SPI Special Function Registers ..... 271
**25. Timers ..... 278**
25.1.Timer 0 and Timer 1 ..... 280
25.1.1.Mode 0: 13-bit Counter/Timer ..... 280
25.1.2.Mode 1: 16-bit Counter/Timer ..... 281
25.1.3.Mode 2: 8-bit Counter/Timer with Auto-Reload ..... 282
25.1.4.Mode 3: Two 8-bit Counter/Timers (Timer 0 Only) ..... 283
25.2.Timer 2 ..... 288
25.2.1.16-bit Timer with Auto-Reload ..... 288
25.2.2.8-bit Timers with Auto-Reload ..... 289
25.2.3.Comparator 0/SmaRTClock Capture Mode ..... 290
25.3.Timer 3 ..... 294

25.3.1.16-bit Timer with Auto-Reload ..... 294
25.3.2.8-Bit Timers with Auto-Reload ..... 295
25.3.3.SmaRTClock/External Oscillator Capture Mode ..... 296
**26. Programmable Counter Array ..... 300**
26.1.PCA Counter/Timer ..... 301
26.2.PCA0 Interrupt Sources ..... 302
26.3.Capture/Compare Modules ..... 303
26.3.1.Edge-triggered Capture Mode ..... 304
26.3.2.Software Timer (Compare) Mode ..... 305
26.3.3.High-Speed Output Mode ..... 306
26.3.4.Frequency Output Mode ..... 306
26.3.5. 8-bit, 9-bit, 10-bit and 11-bit Pulse Width Modulator Modes ..... 307
26.3.6. 16-Bit Pulse Width Modulator Mode ..... 310
26.4.Watchdog Timer Mode ..... 311
26.4.1.Watchdog Timer Operation ..... 311
26.4.2.Watchdog Timer Usage ..... 312
26.5.Register Descriptions for PCA0 ..... 313
**27. C2 Interface ..... 319**
27.1.C2 Interface Registers ..... 319
27.2.C2 Pin Sharing ..... 322
**Document Change List ..... 323**
**Contact Information ..... 325**

## List of Figures

Figure 1.1. C8051F980 Block Diagram ..... 18
Figure 1.2. C8051F981 Block Diagram ..... 18
Figure 1.3. C8051F982 Block Diagram ..... 19
Figure 1.4. C8051F983 Block Diagram ..... 19
Figure 1.5. C8051F985 Block Diagram ..... 20
Figure 1.6. C8051F986 Block Diagram ..... 20
Figure 1.7. C8051F987 Block Diagram ..... 21
Figure 1.8. C8051F988 Block Diagram ..... 21
Figure 1.9. C8051F989 Block Diagram ..... 22
Figure 1.10. C8051F990 Block Diagram ..... 22
Figure 1.11. C8051F991 Block Diagram ..... 23
Figure 1.12. C8051F996 Block Diagram ..... 23
Figure 1.13. C8051F997 Block Diagram ..... 24
Figure 1.14. Port I/O Functional Block Diagram ..... 26
Figure 1.15. PCA Block Diagram ..... 27
Figure 1.16. ADC0 Functional Block Diagram ..... 28
Figure 1.17. ADC0 Multiplexer Block Diagram ..... 29
Figure 1.18. Comparator 0 Functional Block Diagram ..... 30
Figure 3.1. QFN-20 Pinout Diagram (Top View) ..... 35
Figure 3.2. QFN-24 Pinout Diagram (Top View) ..... 36
Figure 3.3. QSOP-24 Pinout Diagram (Top View) ..... 37
Figure 3.4. QFN-20 Package Marking Diagram ..... 38
Figure 3.5. QFN-24 Package Marking Diagram ..... 38
Figure 3.6. QSOP-24 Package Marking Diagram ..... 39
Figure 3.7. QFN-20 Package Drawing ..... 40
Figure 3.8. Typical QFN-20 Landing Diagram ..... 41
Figure 3.9. QFN-24 Package Drawing ..... 43
Figure 3.10. Typical QFN-24 Landing Diagram ..... 44
Figure 3.11. QSOP-24 Package Diagram ..... 46
Figure 3.12. QSOP-24 Landing Diagram ..... 47
Figure 4.1. Active Mode Current (External CMOS Clock) ..... 52
Figure 4.2. Idle Mode Current (External CMOS Clock) ..... 53
Figure 4.3. Typical VOH Curves, 1.8–3.6 V ..... 55
Figure 4.4. Typical VOL Curves, 1.8–3.6 V ..... 56
Figure 5.1. ADC0 Functional Block Diagram ..... 66
Figure 5.2. 10-Bit ADC Track and Conversion Example Timing (BURSTEN = 0) ..... 69
Figure 5.3. Burst Mode Tracking Example with Repeat Count Set to 4 ..... 70
Figure 5.4. ADC0 Equivalent Input Circuits ..... 71
Figure 5.5. ADC Window Compare Example: Right-Justified Single-Ended Data ..... 82
Figure 5.6. ADC Window Compare Example: Left-Justified Single-Ended Data ..... 82
Figure 5.7. ADC0 Multiplexer Block Diagram ..... 83
Figure 5.8. Temperature Sensor Transfer Function ..... 85
Figure 5.9. Temperature Sensor Error with 1-Point Calibration ($V_{REF}$ = 1.65 V) ..... 86

Figure 5.10. Voltage Reference Functional Block Diagram ..... 88
Figure 7.1. Comparator 0 Functional Block Diagram ..... 93
Figure 7.2. Comparator Hysteresis Plot ..... 95
Figure 7.3. CP0 Multiplexer Block Diagram ..... 98
Figure 8.1. CS0 Block Diagram ..... 100
Figure 8.2. Auto-Scan Example ..... 103
Figure 8.3. CS0 Multiplexer Block Diagram ..... 117
Figure 9.1. CIP-51 Block Diagram ..... 119
Figure 10.1. C8051F99x-C8051F98x Memory Map ..... 128
Figure 10.2. Flash Program Memory Map ..... 129
Figure 14.1. Flash Program Memory Map (8 kB and smaller devices) ..... 152
Figure 15.1. C8051F99x-C8051F98x Power Distribution ..... 163
Figure 16.1. CRC0 Block Diagram ..... 172
Figure 16.2. Bit Reverse Register ..... 179
Figure 18.1. Reset Sources ..... 181
Figure 18.2. Power-Fail Reset Timing Diagram ..... 182
Figure 19.1. Clocking Sources Block Diagram ..... 188
Figure 19.2. 25 MHz External Crystal Example ..... 190
Figure 20.1. SmaRTClock Block Diagram ..... 197
Figure 20.2. Interpreting Oscillation Robustness (Duty Cycle) Test Results ..... 206
Figure 21.1. Port I/O Functional Block Diagram ..... 215
Figure 21.2. Port I/O Cell Block Diagram ..... 216
Figure 21.3. Peripheral Availability on Port I/O Pins ..... 219
Figure 21.4. Crossbar Priority Decoder in Example Configuration (No Pins Skipped) ..... 220
Figure 21.5. Crossbar Priority Decoder in Example Configuration (4 Pins Skipped) ..... 220
Figure 22.1. SMBus Block Diagram ..... 235
Figure 22.2. Typical SMBus Configuration ..... 236
Figure 22.3. SMBus Transaction ..... 237
Figure 22.4. Typical SMBus SCL Generation ..... 240
Figure 22.5. Typical Master Write Sequence ..... 249
Figure 22.6. Typical Master Read Sequence ..... 250
Figure 22.7. Typical Slave Write Sequence ..... 251
Figure 22.8. Typical Slave Read Sequence ..... 252
Figure 23.1. UART0 Block Diagram ..... 257
Figure 23.2. UART0 Baud Rate Logic ..... 258
Figure 23.3. UART Interconnect Diagram ..... 259
Figure 23.4. 8-Bit UART Timing Diagram ..... 259
Figure 23.5. 9-Bit UART Timing Diagram ..... 260
Figure 23.6. UART Multi-Processor Mode Interconnect Diagram ..... 261
Figure 24.1. SPI Block Diagram ..... 265
Figure 24.2. Multiple-Master Mode Connection Diagram ..... 267
Figure 24.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram ..... 267

Figure 24.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram ..... 268
Figure 24.5. Master Mode Data/Clock Timing ..... 270
Figure 24.6. Slave Mode Data/Clock Timing (CKPHA = 0) ..... 270
Figure 24.7. Slave Mode Data/Clock Timing (CKPHA = 1) ..... 271
Figure 24.8. SPI Master Timing (CKPHA = 0) ..... 275
Figure 24.9. SPI Master Timing (CKPHA = 1) ..... 275
Figure 24.10. SPI Slave Timing (CKPHA = 0) ..... 276
Figure 24.11. SPI Slave Timing (CKPHA = 1) ..... 276
Figure 25.1. T0 Mode 0 Block Diagram ..... 281
Figure 25.2. T0 Mode 2 Block Diagram ..... 282
Figure 25.3. T0 Mode 3 Block Diagram ..... 283
Figure 25.4. Timer 2 16-Bit Mode Block Diagram ..... 288
Figure 25.5. Timer 2 8-Bit Mode Block Diagram ..... 289
Figure 25.6. Timer 2 Capture Mode Block Diagram ..... 290
Figure 25.7. Timer 3 16-Bit Mode Block Diagram ..... 294
Figure 25.8. Timer 3 8-Bit Mode Block Diagram ..... 295
Figure 25.9. Timer 3 Capture Mode Block Diagram ..... 296
Figure 26.1. PCA Block Diagram ..... 300
Figure 26.2. PCA Counter/Timer Block Diagram ..... 301
Figure 26.3. PCA Interrupt Block Diagram ..... 302
Figure 26.4. PCA Capture Mode Diagram ..... 304
Figure 26.5. PCA Software Timer Mode Diagram ..... 305
Figure 26.6. PCA High-Speed Output Mode Diagram ..... 306
Figure 26.7. PCA Frequency Output Mode ..... 307
Figure 26.8. PCA 8-Bit PWM Mode Diagram ..... 308
Figure 26.9. PCA 9, 10 and 11-Bit PWM Mode Diagram ..... 309
Figure 26.10. PCA 16-Bit PWM Mode ..... 310
Figure 26.11. PCA Module 2 with Watchdog Timer Enabled ..... 311
Figure 27.1. Typical C2 Pin Sharing ..... 322

## List of Tables

Table 2.1. Product Selection Guide . . . . . 31
Table 3.1. Pin Definitions for the C8051F99x-C8051F98x . . . . . 32
Table 3.2. QFN-20 Package Dimensions . . . . . 40
Table 3.3. PCB Land Pattern . . . . . 42
Table 3.4. QFN-24 Package Dimensions . . . . . 43
Table 3.5. PCB Land Pattern . . . . . 45
Table 3.6. QSOP-24 Package Dimensions . . . . . 46
Table 3.7. PCB Land Pattern . . . . . 47
Table 4.1. Absolute Maximum Ratings . . . . . 48
Table 4.2. Global Electrical Characteristics . . . . . 49
Table 4.3. Port I/O DC Electrical Characteristics . . . . . 54
Table 4.4. Reset Electrical Characteristics . . . . . 57
Table 4.5. Power Management Electrical Specifications . . . . . 58
Table 4.6. Flash Electrical Characteristics . . . . . 58
Table 4.7. Internal Precision Oscillator Electrical Characteristics . . . . . 58
Table 4.8. Internal Low-Power Oscillator Electrical Characteristics . . . . . 58
Table 4.9. SmaRTClock Characteristics . . . . . 59
Table 4.10. ADC0 Electrical Characteristics . . . . . 59
Table 4.11. Temperature Sensor Electrical Characteristics . . . . . 60
Table 4.12. Voltage Reference Electrical Characteristics . . . . . 61
Table 4.13. IREF0 Electrical Characteristics . . . . . 62
Table 4.14. Comparator Electrical Characteristics . . . . . 63
Table 4.15. VREG0 Electrical Characteristics . . . . . 64
Table 4.16. Capacitive Sense Electrical Characteristics . . . . . 65
Table 5.1. Representative Conversion Times and Energy Consumption for the SAR ADC with 1.65 V High-Speed VREF . . . . . 73
Table 8.1. Operation with Auto-scan and Accumulate . . . . . 105
Table 9.1. CIP-51 Instruction Set Summary . . . . . 121
Table 12.1. Special Function Register (SFR) Memory Map (Page 0x0) . . . . . 132
Table 12.2. Special Function Register (SFR) Memory Map (Page 0xF) . . . . . 133
Table 12.3. Special Function Registers . . . . . 134
Table 13.1. Interrupt Summary . . . . . 140
Table 14.1. Flash Security Summary . . . . . 153
Table 15.1. Power Modes . . . . . 162
Table 16.1. Example 16-bit CRC Outputs . . . . . 173
Table 19.1. Recommended XFCN Settings for Crystal Mode . . . . . 190
Table 19.2. Recommended XFCN Settings for RC and C modes . . . . . 191
Table 20.1. SmaRTClock Internal Registers . . . . . 198
Table 20.2. SmaRTClock Load Capacitance Settings . . . . . 205
Table 20.3. SmaRTClock Bias Settings . . . . . 207
Table 21.1. Port I/O Assignment for Analog Functions . . . . . 217
Table 21.2. Port I/O Assignment for Digital Functions . . . . . 218
Table 21.3. Port I/O Assignment for External Digital Event Capture Functions . . 218

Table 22.1. SMBus Clock Source Selection . . . . . 240
Table 22.2. Minimum SDA Setup and Hold Times . . . . . 241
Table 22.3. Sources for Hardware Changes to SMB0CN . . . . . 245
Table 22.4. Hardware Address Recognition Examples (EHACK = 1) . . . . . 246
Table 22.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) . . . . . 253
Table 22.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) . . . . . 255
Table 23.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator . . . . . 264
Table 23.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator . . . . . 264
Table 24.1. SPI Slave Timing Parameters . . . . . 277
Table 25.1. Timer 0 Running Modes . . . . . 280
Table 26.1. PCA Timebase Input Options . . . . . 301
Table 26.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules . . . . . 303
Table 26.3. Watchdog Timer Timeout Intervals1 . . . . . 312

## List of Registers

SFR Definition 5.1. ADC0CN: ADC0 Control . . . . . 74
SFR Definition 5.2. ADC0CF: ADC0 Configuration . . . . . 75
SFR Definition 5.3. ADC0AC: ADC0 Accumulator Configuration . . . . . 76
SFR Definition 5.4. ADC0PWR: ADC0 Burst Mode Power-Up Time . . . . . 77
SFR Definition 5.5. ADC0TK: ADC0 Burst Mode Track Time . . . . . 78
SFR Definition 5.6. ADC0H: ADC0 Data Word High Byte . . . . . 79
SFR Definition 5.7. ADC0L: ADC0 Data Word Low Byte . . . . . 79
SFR Definition 5.8. ADC0GTH: ADC0 Greater-Than High Byte . . . . . 80
SFR Definition 5.9. ADC0GTL: ADC0 Greater-Than Low Byte . . . . . 80
SFR Definition 5.10. ADC0LTH: ADC0 Less-Than High Byte . . . . . 81
SFR Definition 5.11. ADC0LTL: ADC0 Less-Than Low Byte . . . . . 81
SFR Definition 5.12. ADC0MX: ADC0 Input Channel Select . . . . . 84
SFR Definition 5.13. TOFFH: ADC0 Data Word High Byte . . . . . 87
SFR Definition 5.14. TOFFL: ADC0 Data Word Low Byte . . . . . 87
SFR Definition 5.15. REF0CN: Voltage Reference Control . . . . . 90
SFR Definition 6.1. IREF0CN: Current Reference Control . . . . . 91
SFR Definition 6.2. IREF0CF: Current Reference Configuration . . . . . 92
SFR Definition 7.1. CPT0CN: Comparator 0 Control . . . . . 96
SFR Definition 7.2. CPT0MD: Comparator 0 Mode Selection . . . . . 97
SFR Definition 7.3. CPT0MX: Comparator0 Input Channel Select . . . . . 99
SFR Definition 8.1. CS0CN: Capacitive Sense Control . . . . . 107
SFR Definition 8.2. CS0CF: Capacitive Sense Configuration . . . . . 108
SFR Definition 8.3. CS0DH: Capacitive Sense Data High Byte . . . . . 109
SFR Definition 8.4. CS0DL: Capacitive Sense Data Low Byte . . . . . 109
SFR Definition 8.5. CS0SCAN0: Capacitive Sense Channel Scan Mask 0 . . . . . 110
SFR Definition 8.6. CS0SCAN1: Capacitive Sense Channel Scan Mask 1 . . . . . 110
SFR Definition 8.7. CS0SS: Capacitive Sense Auto-Scan Start Channel . . . . . 111
SFR Definition 8.8. CS0SE: Capacitive Sense Auto-Scan End Channel . . . . . 111
SFR Definition 8.9. CS0THH: Capacitive Sense Comparator Threshold High Byte . . 112
SFR Definition 8.10. CS0THL: Capacitive Sense Comparator Threshold Low Byte . 112
SFR Definition 8.11. CS0MD1: Capacitive Sense Mode 1 . . . . . 113
SFR Definition 8.12. CS0MD2: Capacitive Sense Mode 2 . . . . . 114
SFR Definition 8.13. CS0MD3: Capacitive Sense Mode 3 . . . . . 115
SFR Definition 8.14. CS0PM: Capacitive Sense Pin Monitor . . . . . 116
SFR Definition 8.15. CS0MX: Capacitive Sense Mux Channel Select . . . . . 118
SFR Definition 9.1. DPL: Data Pointer Low Byte . . . . . 125
SFR Definition 9.2. DPH: Data Pointer High Byte . . . . . 125
SFR Definition 9.3. SP: Stack Pointer . . . . . 126
SFR Definition 9.4. ACC: Accumulator . . . . . 126
SFR Definition 9.5. B: B Register . . . . . 126
SFR Definition 9.6. PSW: Program Status Word . . . . . 127
SFR Definition 12.1. SFR Page: SFR Page . . . . . 134
SFR Definition 13.1. IE: Interrupt Enable . . . . . 142

SFR Definition 13.2. IP: Interrupt Priority . . . . . 143
SFR Definition 13.3. EIE1: Extended Interrupt Enable 1 . . . . . 144
SFR Definition 13.4. EIP1: Extended Interrupt Priority 1 . . . . . 145
SFR Definition 13.5. EIE2: Extended Interrupt Enable 2 . . . . . 146
SFR Definition 13.6. EIP2: Extended Interrupt Priority 2 . . . . . 147
SFR Definition 13.7. IT01CF: INT0/INT1 Configuration . . . . . 149
SFR Definition 14.1. DEVICEID: Device Identification . . . . . 154
SFR Definition 14.2. REVID: Revision Identification . . . . . 155
SFR Definition 14.3. PSCTL: Program Store R/W Control . . . . . 159
SFR Definition 14.4. FLKEY: Flash Lock and Key . . . . . 160
SFR Definition 14.5. FLSCL: Flash Scale . . . . . 161
SFR Definition 14.6. FLWR: Flash Write Only . . . . . 161
SFR Definition 15.1. PMU0CF: Power Management Unit Configuration[1,2,3] . . . . . 168
SFR Definition 15.2. PMU0FL: Power Management Unit Flag[1,2] . . . . . 169
SFR Definition 15.3. PMU0MD: Power Management Unit Mode . . . . . 170
SFR Definition 15.4. PCON: Power Management Control Register . . . . . 171
SFR Definition 16.1. CRC0CN: CRC0 Control . . . . . 175
SFR Definition 16.2. CRC0IN: CRC0 Data Input . . . . . 176
SFR Definition 16.3. CRC0DAT: CRC0 Data Output . . . . . 176
SFR Definition 16.4. CRC0AUTO: CRC0 Automatic Control . . . . . 177
SFR Definition 16.5. CRC0CNT: CRC0 Automatic Flash Sector Count . . . . . 178
SFR Definition 16.6. CRC0FLIP: CRC0 Bit Flip . . . . . 179
SFR Definition 17.1. REG0CN: Voltage Regulator Control . . . . . 180
SFR Definition 18.1. VDM0CN: VDD Supply Monitor Control . . . . . 184
SFR Definition 18.2. RSTSRC: Reset Source . . . . . 187
SFR Definition 19.1. CLKSEL: Clock Select . . . . . 193
SFR Definition 19.2. OSCICN: Internal Oscillator Control . . . . . 194
SFR Definition 19.3. OSCICL: Internal Oscillator Calibration . . . . . 195
SFR Definition 19.4. OSCXCN: External Oscillator Control . . . . . 196
SFR Definition 20.1. RTC0KEY: SmaRTClock Lock and Key . . . . . 201
SFR Definition 20.2. RTC0ADR: SmaRTClock Address . . . . . 202
SFR Definition 20.3. RTC0DAT: SmaRTClock Data . . . . . 202
Internal Register Definition 20.4. RTC0CN: SmaRTClock Control . . . . . 211
Internal Register Definition 20.5. RTC0XCN: SmaRTClock Oscillator Control . . . . . 212
Internal Register Definition 20.6. RTC0XCF: SmaRTClock Oscillator Configuration . 213
Internal Register Definition 20.7. CAPTUREn: SmaRTClock Timer Capture . . . . . 214
Internal Register Definition 20.8. ALARMn: SmaRTClock Alarm Programmed Value 214
SFR Definition 21.1. XBR0: Port I/O Crossbar Register 0 . . . . . 222
SFR Definition 21.2. XBR1: Port I/O Crossbar Register 1 . . . . . 223
SFR Definition 21.3. XBR2: Port I/O Crossbar Register 2 . . . . . 224
SFR Definition 21.4. P0MASK: Port0 Mask Register . . . . . 225
SFR Definition 21.5. P0MAT: Port0 Match Register . . . . . 225
SFR Definition 21.6. P1MASK: Port1 Mask Register . . . . . 226
SFR Definition 21.7. P1MAT: Port1 Match Register . . . . . 226
SFR Definition 21.8. P0: Port0 . . . . . 228

SFR Definition 21.9. P0SKIP: Port0 Skip . . . . . 228
SFR Definition 21.10. P0MDIN: Port0 Input Mode . . . . . 229
SFR Definition 21.11. P0MDOUT: Port0 Output Mode . . . . . 229
SFR Definition 21.12. P0DRV: Port0 Drive Strength . . . . . 230
SFR Definition 21.13. P1: Port1 . . . . . 231
SFR Definition 21.14. P1SKIP: Port1 Skip . . . . . 231
SFR Definition 21.15. P1MDIN: Port1 Input Mode . . . . . 232
SFR Definition 21.16. P1MDOUT: Port1 Output Mode . . . . . 232
SFR Definition 21.17. P1DRV: Port1 Drive Strength . . . . . 233
SFR Definition 21.18. P2: Port2 . . . . . 233
SFR Definition 21.19. P2MDOUT: Port2 Output Mode . . . . . 234
SFR Definition 21.20. P2DRV: Port2 Drive Strength . . . . . 234
SFR Definition 22.1. SMB0CF: SMBus Clock/Configuration . . . . . 242
SFR Definition 22.2. SMB0CN: SMBus Control . . . . . 244
SFR Definition 22.3. SMB0ADR: SMBus Slave Address . . . . . 247
SFR Definition 22.4. SMB0ADM: SMBus Slave Address Mask . . . . . 247
SFR Definition 22.5. SMB0DAT: SMBus Data . . . . . 248
SFR Definition 23.1. SCON0: Serial Port 0 Control . . . . . 262
SFR Definition 23.2. SBUF0: Serial (UART0) Port Data Buffer . . . . . 263
SFR Definition 24.1. SPI0CFG: SPI0 Configuration . . . . . 272
SFR Definition 24.2. SPI0CN: SPI0 Control . . . . . 273
SFR Definition 24.3. SPI0CKR: SPI0 Clock Rate . . . . . 274
SFR Definition 24.4. SPI0DAT: SPI0 Data . . . . . 274
SFR Definition 25.1. CKCON: Clock Control . . . . . 279
SFR Definition 25.2. TCON: Timer Control . . . . . 284
SFR Definition 25.3. TMOD: Timer Mode . . . . . 285
SFR Definition 25.4. TL0: Timer 0 Low Byte . . . . . 286
SFR Definition 25.5. TL1: Timer 1 Low Byte . . . . . 286
SFR Definition 25.6. TH0: Timer 0 High Byte . . . . . 287
SFR Definition 25.7. TH1: Timer 1 High Byte . . . . . 287
SFR Definition 25.8. TMR2CN: Timer 2 Control . . . . . 291
SFR Definition 25.9. TMR2RLL: Timer 2 Reload Register Low Byte . . . . . 292
SFR Definition 25.10. TMR2RLH: Timer 2 Reload Register High Byte . . . . . 292
SFR Definition 25.11. TMR2L: Timer 2 Low Byte . . . . . 293
SFR Definition 25.12. TMR2H Timer 2 High Byte . . . . . 293
SFR Definition 25.13. TMR3CN: Timer 3 Control . . . . . 297
SFR Definition 25.14. TMR3RLL: Timer 3 Reload Register Low Byte . . . . . 298
SFR Definition 25.15. TMR3RLH: Timer 3 Reload Register High Byte . . . . . 298
SFR Definition 25.16. TMR3L: Timer 3 Low Byte . . . . . 299
SFR Definition 25.17. TMR3H Timer 3 High Byte . . . . . 299
SFR Definition 26.1. PCA0CN: PCA Control . . . . . 313
SFR Definition 26.2. PCA0MD: PCA Mode . . . . . 314
SFR Definition 26.3. PCA0PWM: PCA PWM Configuration . . . . . 315
SFR Definition 26.4. PCA0CPMn: PCA Capture/Compare Mode . . . . . 316
SFR Definition 26.5. PCA0L: PCA Counter/Timer Low Byte . . . . . 317

SFR Definition 26.6. PCA0H: PCA Counter/Timer High Byte . . . . . 317
SFR Definition 26.7. PCA0CPLn: PCA Capture Module Low Byte . . . . . 318
SFR Definition 26.8. PCA0CPHn: PCA Capture Module High Byte . . . . . 318
C2 Register Definition 27.1. C2ADD: C2 Address . . . . . 319
C2 Register Definition 27.2. DEVICEID: C2 Device ID . . . . . 320
C2 Register Definition 27.3. REVID: C2 Revision ID . . . . . 320
C2 Register Definition 27.4. FPCTL: C2 Flash Programming Control . . . . . 321
C2 Register Definition 27.5. FPDAT: C2 Flash Programming Data . . . . . 321

# 1. System Overview

C8051F99x-C8051F98x devices are fully integrated mixed-signal system-on-a-chip MCUs. Highlighted features are listed below. Refer to Table 2.1 for specific product feature selection and part ordering numbers.

- Ultra low power consumption in active and sleep modes.
- High-speed pipelined 8051-compatible microcontroller core (up to 25 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- 10-bit 300 ksps or 12-bit 75 ksps single-ended ADC with analog multiplexer
- 6-bit programmable current reference (resolution can be increased with PWM)
- Precision programmable 24.5 MHz internal oscillator with spread spectrum technology.
- 8 kB, 4 kB, or 2 kB of on-chip Flash memory
- 512 bytes of on-chip RAM
- SMBus/$I^2C$, Enhanced UART, and Enhanced SPI serial interfaces implemented in hardware
- Four general-purpose 16-bit timers
- Programmable counter/timer array (PCA) with three capture/compare modules and watchdog timer function
- On-chip power-on reset, $V_{DD}$ monitor, and temperature sensor
- One on-chip voltage comparator
- Up to 14 Capacitive Touch Inputs
- Up to 17 Port I/O

With on-chip power-on reset, $V_{DD}$ monitor, watchdog timer, and clock oscillator, the C8051F99x-C8051F98x devices are truly stand-alone system-on-a-chip solutions. The Flash memory can be reprogrammed even in-circuit, providing non-volatile data storage, and also allowing field upgrades of the 8051 firmware. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

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

Each device is specified for 1.8 to 3.6 V operation over the industrial temperature range (–40 to +85 °C). The Port I/O and $\overline{RST}$ pins are powered from the supply voltage. The C8051F99x-C8051F98x devices are available in 20-pin or 24-pin QFN or 24-pin QSOP packages. All package options are lead-free and RoHS compliant. See Table 2.1 for ordering information. Block diagrams are included in Figure 1.1 through Figure 1.9.

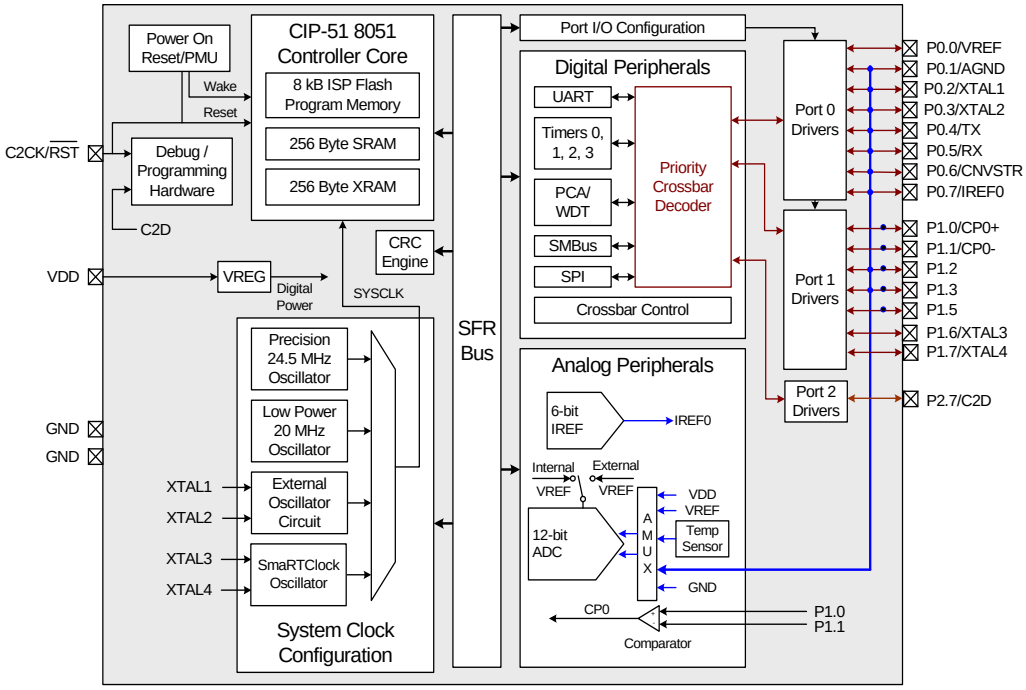



**Figure 1.1. C8051F980 Block Diagram**

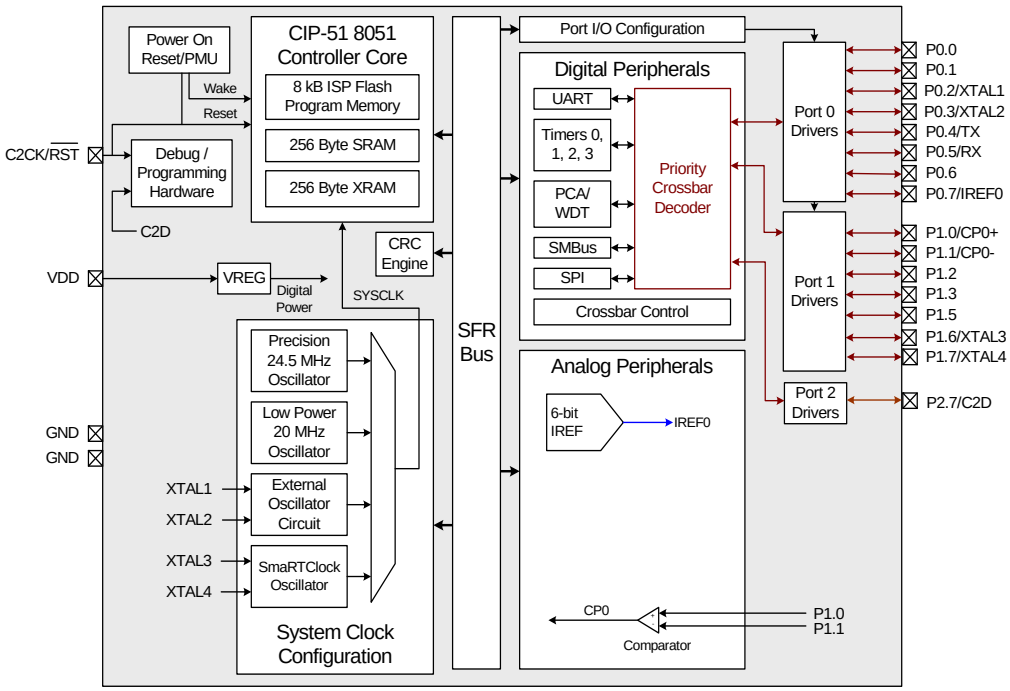



**Figure 1.2. C8051F981 Block Diagram**

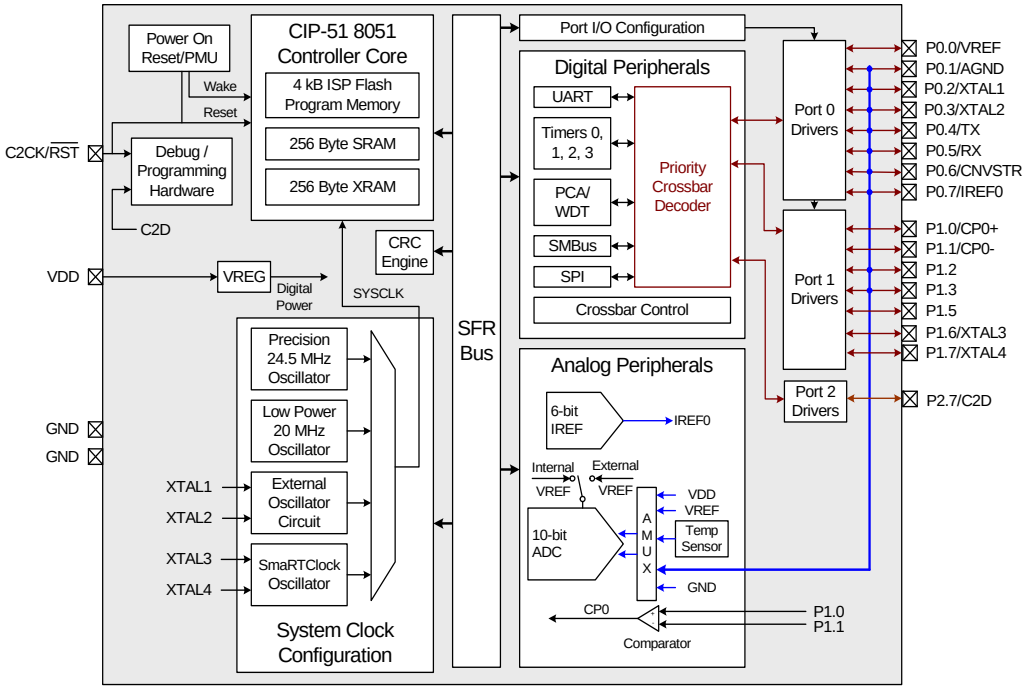



**Figure 1.3. C8051F982 Block Diagram**

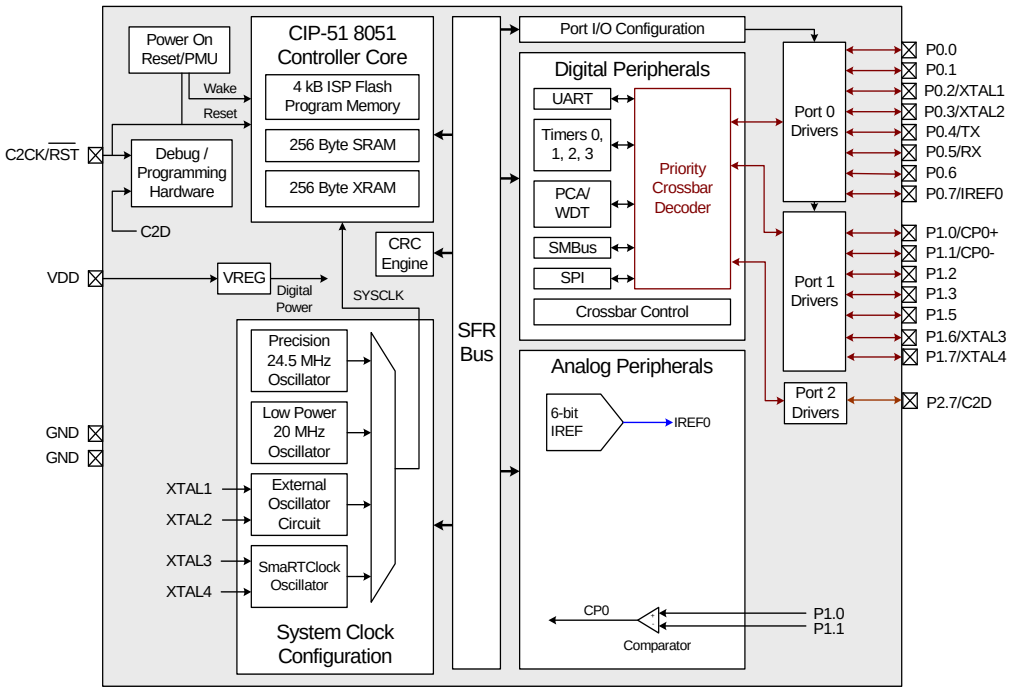



**Figure 1.4. C8051F983 Block Diagram**

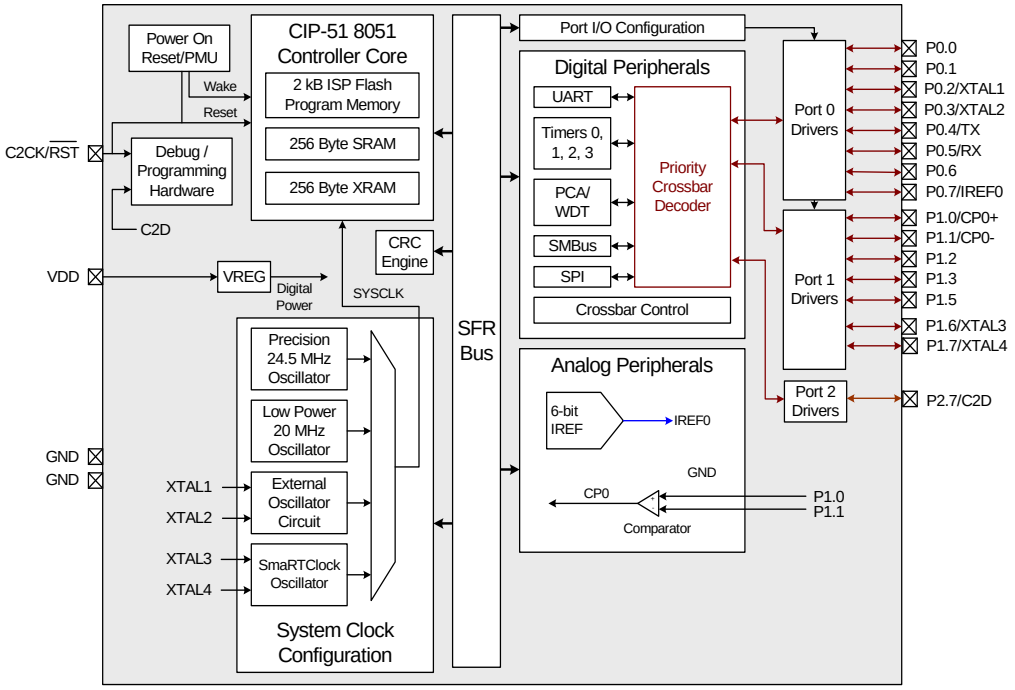



**Figure 1.5. C8051F985 Block Diagram**

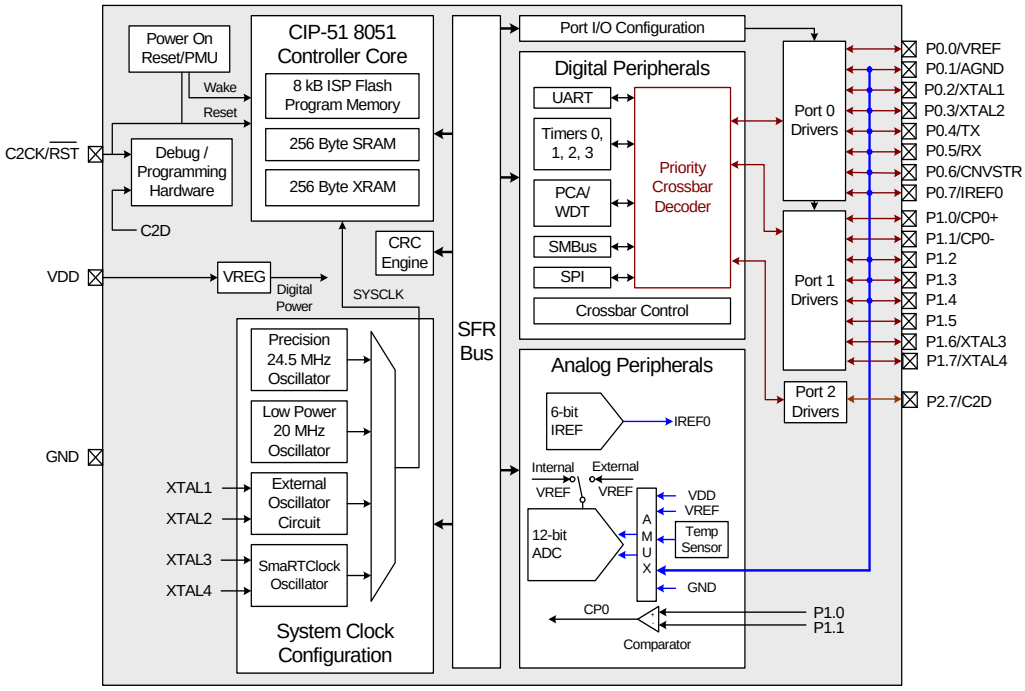



**Figure 1.6. C8051F986 Block Diagram**

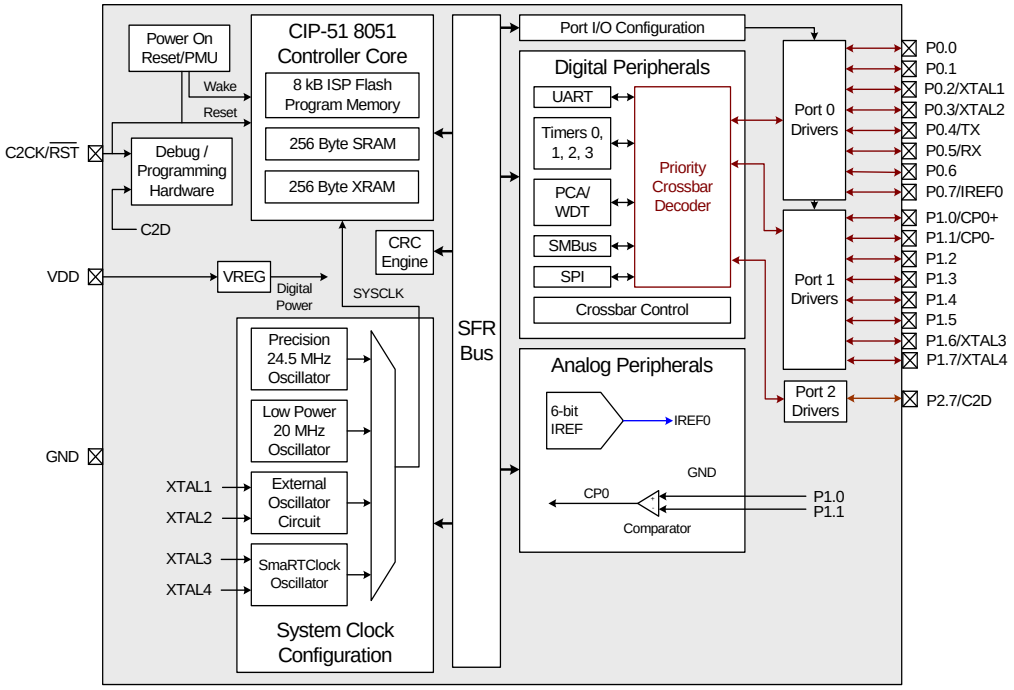



**Figure 1.7. C8051F987 Block Diagram**

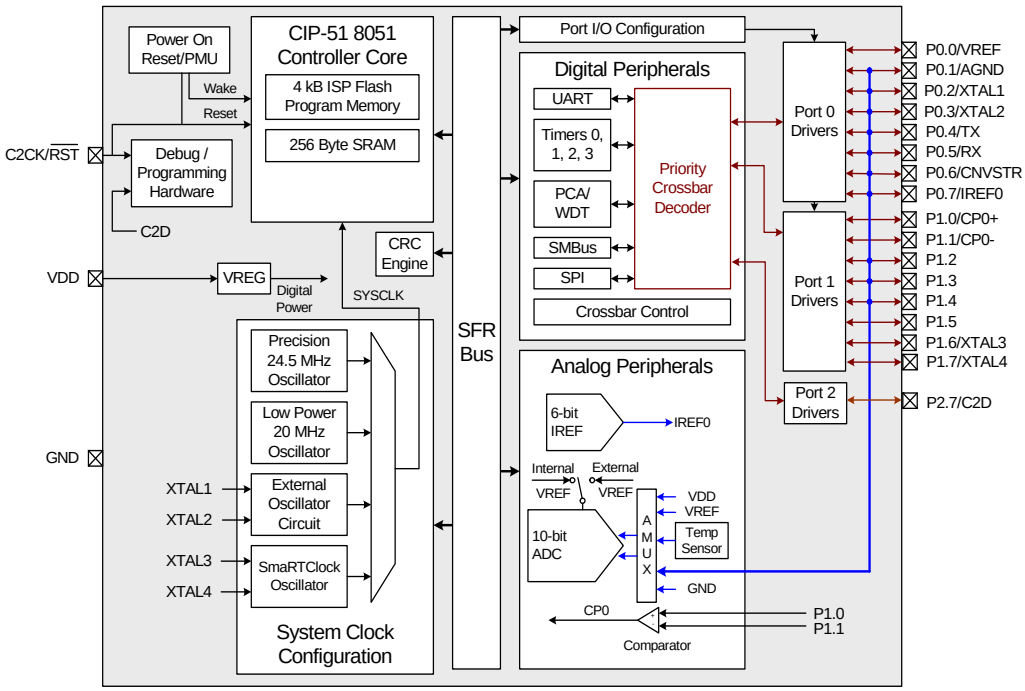



**Figure 1.8. C8051F988 Block Diagram**

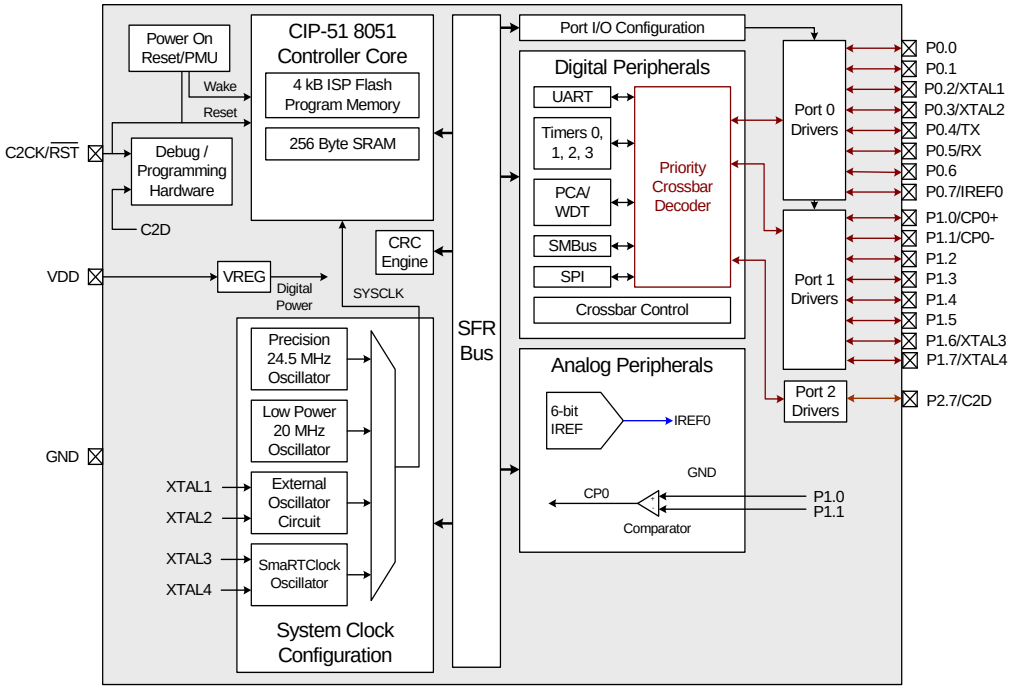



**Figure 1.9. C8051F989 Block Diagram**

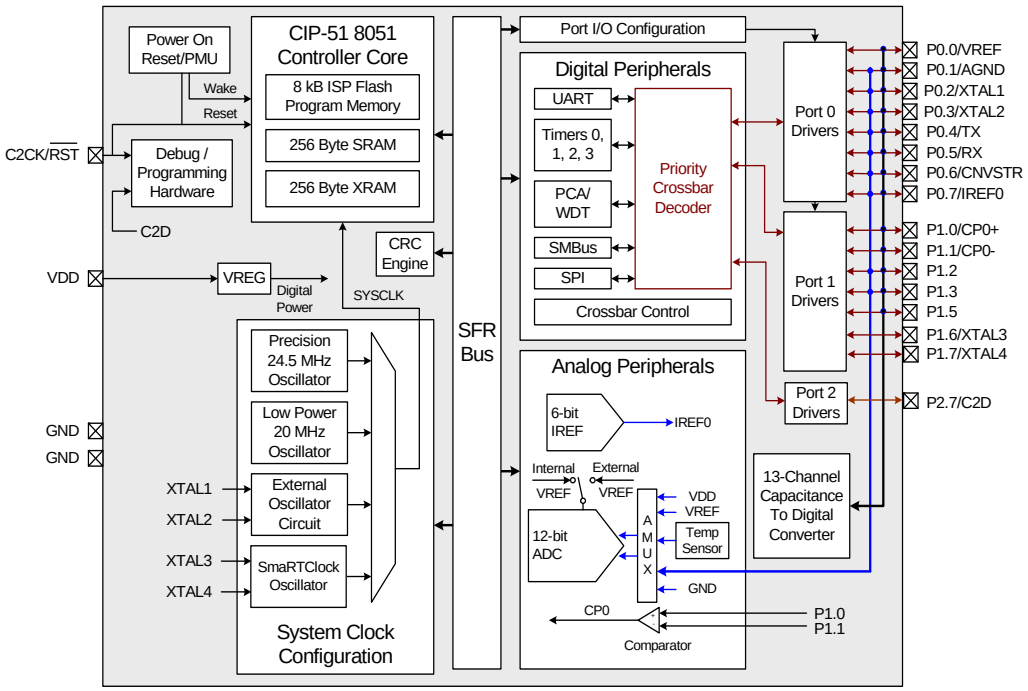



**Figure 1.10. C8051F990 Block Diagram**

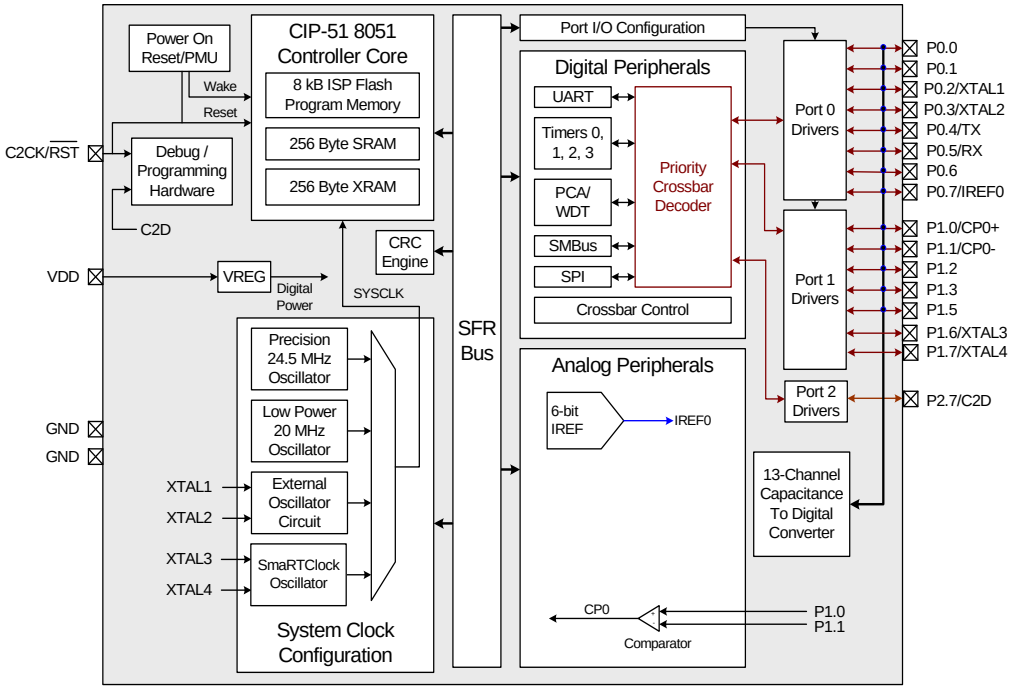



**Figure 1.11. C8051F991 Block Diagram**

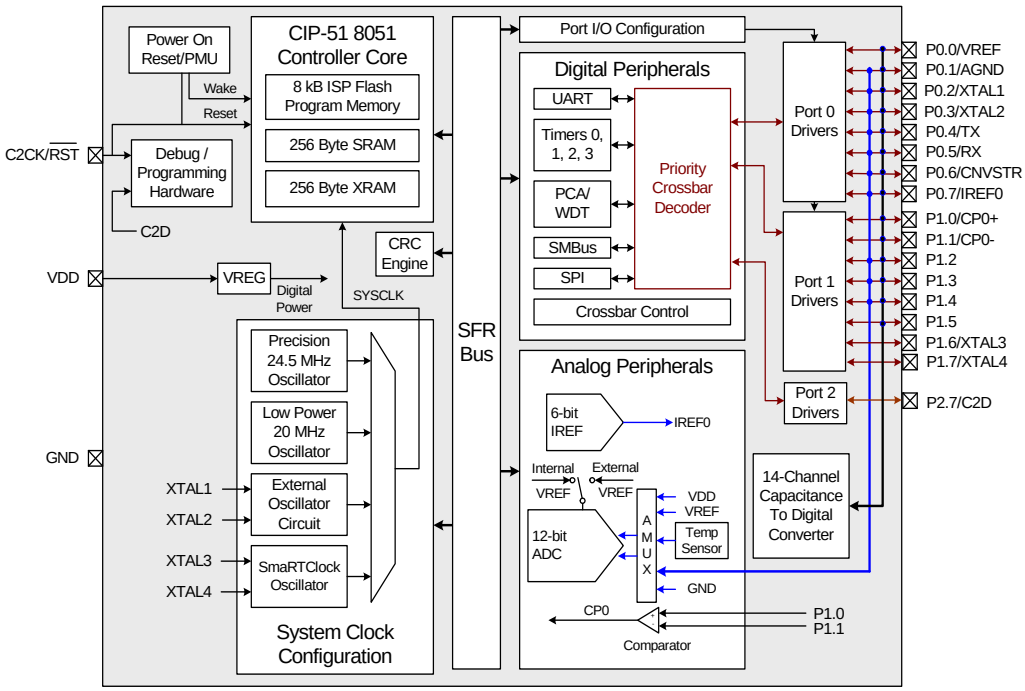



**Figure 1.12. C8051F996 Block Diagram**

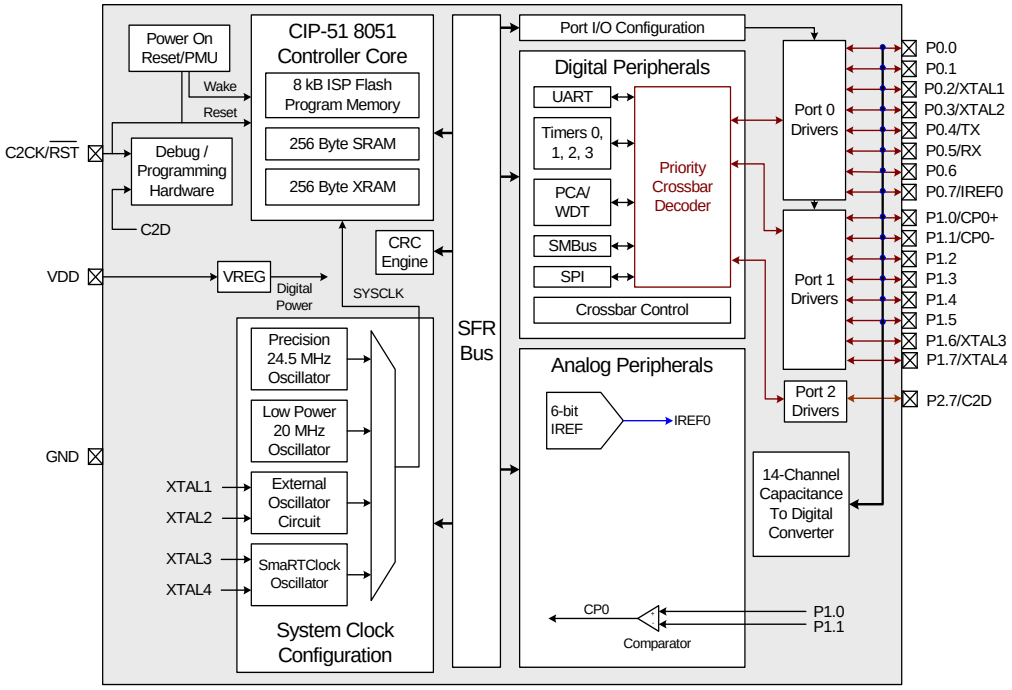



**Figure 1.13. C8051F997 Block Diagram**

## 1.1. CIP-51™ Microcontroller Core

### 1.1.1. Fully 8051 Compatible

The C8051F99x-C8051F98x family utilizes Silicon Labs' proprietary CIP-51 microcontroller core. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The CIP-51 core offers all the peripherals included with a standard 8052.

### 1.1.2. 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 with a maximum system clock of 12-to-24 MHz. By contrast, the CIP-51 core executes 70% of its instructions in one or two system clock cycles, with only four instructions taking more than four system clock cycles.

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

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

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

### 1.1.3. Additional Features

The C8051F99x-C8051F98x SoC family includes several key enhancements to the CIP-51 core and peripherals to improve performance and ease of use in end applications.

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

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

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

## 1.2. Port Input/Output

Digital and analog resources are available through 16 or 17 I/O pins. Port pins are organized as three byte-wide ports. Port pins P0.0–P1.7 can be defined as digital or analog I/O. Digital I/O pins can be assigned to one of the internal digital resources or used as general purpose I/O (GPIO). Analog I/O pins are used by the internal analog resources. P2.7 can be used as GPIO and is shared with the C2 Interface Data signal (C2D). See Section “27. C2 Interface” on page 319 for more details.

The designer has complete control over which digital and analog functions are assigned to individual Port pins, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a Priority Crossbar Decoder. See Section “21.3. Priority Crossbar Decoder” on page 219 for more information on the Crossbar.

All Port I/Os can tolerate voltages up to the supply rail when used as digital inputs or open-drain outputs. For Port I/Os configured as push-pull outputs, current is sourced from the VDD supply. Port I/Os used for analog functions can operate up to the VDD supply voltage. See Section “21.1. Port I/O Modes of Operation” on page 216 for more information on Port I/O operating modes and the electrical specifications chapter for detailed electrical specifications.

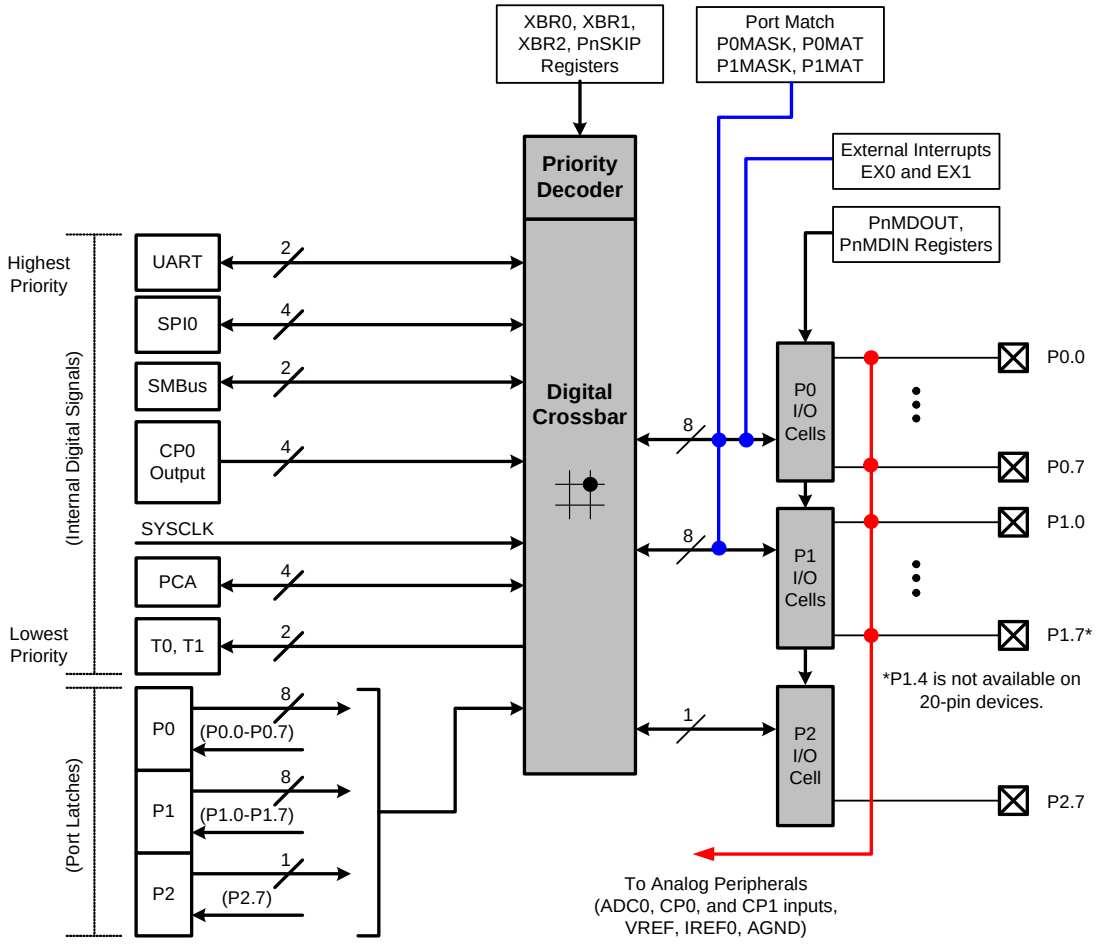



**Figure 1.14. Port I/O Functional Block Diagram**

## 1.3. Serial Ports

The C8051F99x-C8051F98x Family includes an SMBus/$I^2C$ interface, a full-duplex UART with enhanced baud rate configuration, and an Enhanced SPI interface. Each of the serial buses is fully implemented in hardware and makes extensive use of the CIP-51's interrupts, thus requiring very little CPU intervention.

## 1.4. Programmable Counter Array

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

Each capture/compare module can be configured to operate in a variety of modes: edge-triggered capture, software timer, high-speed output, pulse width modulator (8, 9, 10, 11, or 16-bit), or frequency output. Additionally, Capture/Compare Module 2 offers watchdog timer (WDT) capabilities. Following a system reset, Module 2 is configured and enabled in WDT mode. The PCA Capture/Compare Module I/O and External Clock Input may be routed to Port I/O via the Digital Crossbar.

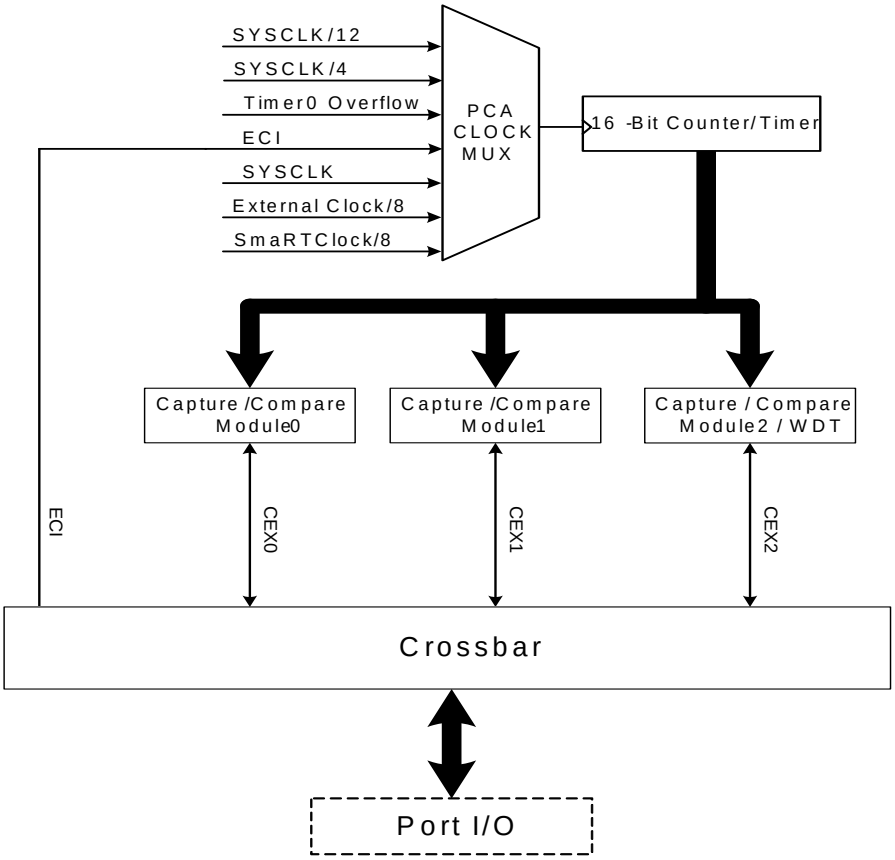



**Figure 1.15. PCA Block Diagram**

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

C8051F99x-C8051F98x devices have a 300 ksps, 10-bit or 75 ksps 12-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and programmable window detector. ADC0 also has an autonomous low power Burst Mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. It also has a 16-bit accumulator that can automatically average the ADC results, providing an effective 11, 12, or 13 bit ADC result without any additional CPU intervention.

The ADC can sample the voltage at select GPIO pins (see Figure 1.17) and has an on-chip attenuator that allows it to measure voltages up to twice the voltage reference. Additional ADC inputs include an on-chip temperature sensor, the VDD supply voltage, and the internal digital supply voltage.

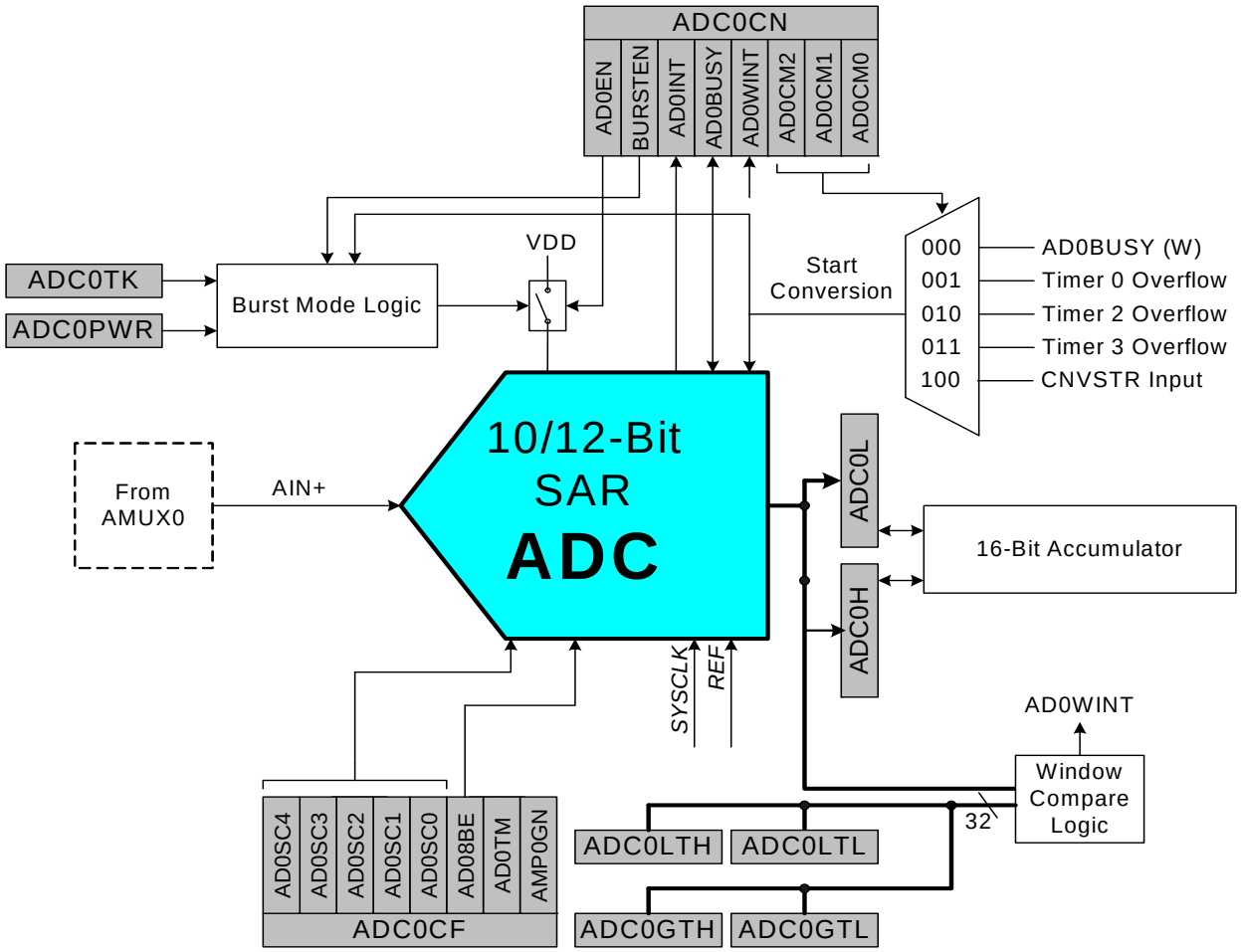



**Figure 1.16. ADC0 Functional Block Diagram**

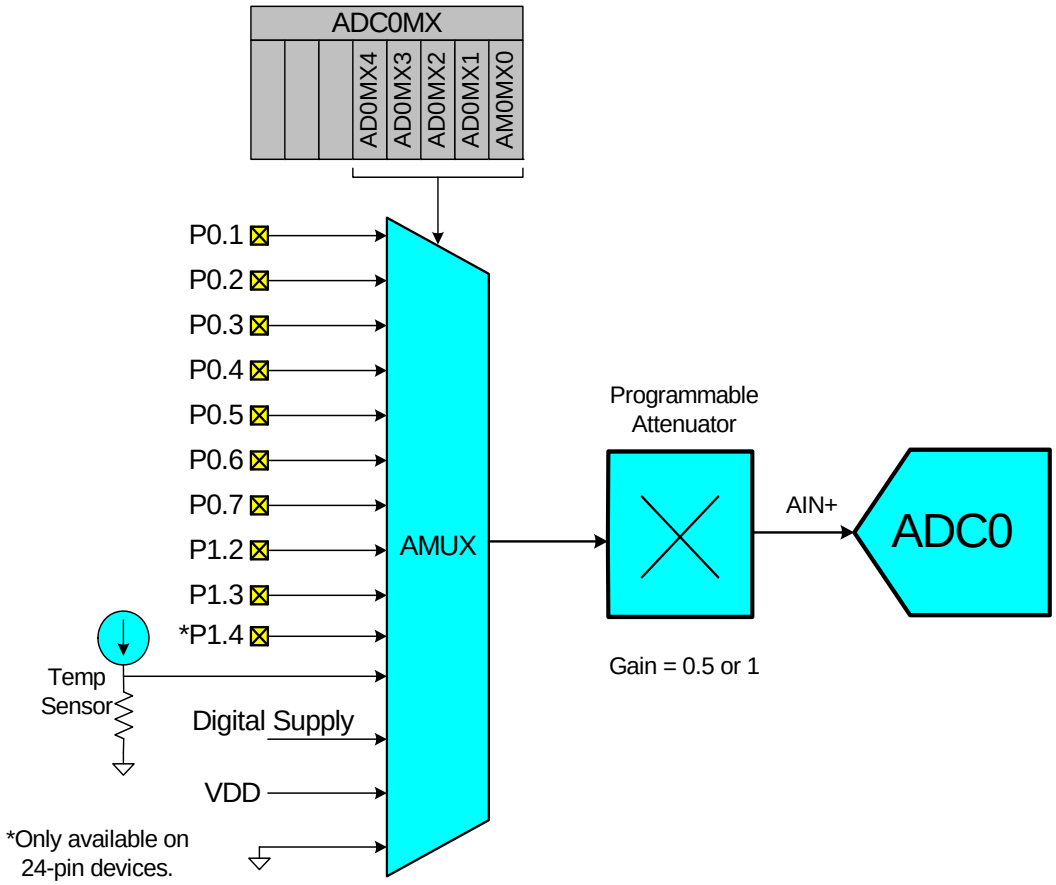



**Figure 1.17. ADC0 Multiplexer Block Diagram**

## 1.6. Programmable Current Reference (IREF0)

C8051F99x-C8051F98x devices include an on-chip programmable current reference (source or sink) with two output current settings: low power mode and high current mode. The maximum current output in low power mode is 63 µA (1 µA steps) and the maximum current output in high current mode is 504 µA (8 µA steps).

## 1.7. Comparator

C8051F99x-C8051F98x devices include an on-chip programmable voltage comparator: Comparator 0 (CPT0) which is shown in Figure 1.18.

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), or an asynchronous “raw” output (CP0A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the Comparator to operate and generate an output when the device is in some low power modes.

The comparator inputs may be connected to Port I/O pins or to other internal signals. Port pins may also be used to directly sense capacitive touch switches.

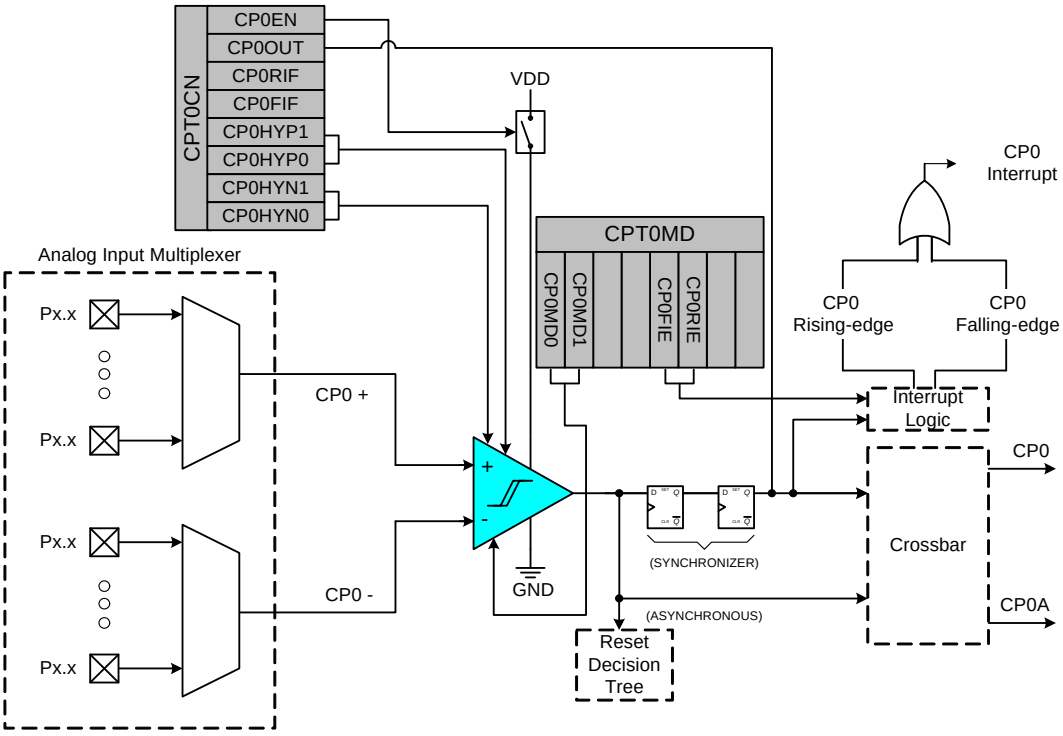



**Figure 1.18. Comparator 0 Functional Block Diagram**

## 2. Ordering Information

**Table 2.1. Product Selection Guide**

| Ordering Part Number | MIPS (Peak) | Flash Memory (kB) | RAM (bytes) | SmaRTClock Real Time Clock | SMBus/$I^2C$, UART, Enhanced SPI | Timers (16-bit) | Programmable Counter Array | Digital Port I/Os | Analog to Digital Converter Inputs | ADC with internal voltage reference and temperature sensor | Capacitive Touch Inputs | Programmable Current Reference | Analog Comparators | Lead-free (RoHS Compliant) | Package |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C8051F980-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 16 | 9 | 12-bit | — | ✓ | 1 | ✓ | QFN-20 |
| C8051F981-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 16 | — | — | — | ✓ | 1 | ✓ | QFN-20 |
| C8051F982-C-GM | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 16 | 9 | 10-bit | — | ✓ | 1 | ✓ | QFN-20 |
| C8051F983-C-GM | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 16 | — | — | — | ✓ | 1 | ✓ | QFN-20 |
| C8051F985-C-GM | 25 | 2 | 512 | ✓ | ✓ | 4 | ✓ | 16 | — | — | — | ✓ | 1 | ✓ | QFN-20 |
| C8051F986-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 12-bit | — | ✓ | 1 | ✓ | QFN-24 |
| C8051F986-C-GU | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 12-bit | — | ✓ | 1 | ✓ | QSOP-24 |
| C8051F987-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | — | ✓ | 1 | ✓ | QFN-24 |
| C8051F987-C-GU | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | — | ✓ | 1 | ✓ | QSOP-24 |
| C8051F988-C-GM | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 10-bit | — | ✓ | 1 | ✓ | QFN-24 |
| C8051F988-C-GU | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 10-bit | — | ✓ | 1 | ✓ | QSOP-24 |
| C8051F989-C-GM | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | — | ✓ | 1 | ✓ | QFN-24 |
| C8051F989-C-GU | 25 | 4 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | — | ✓ | 1 | ✓ | QSOP-24 |
| C8051F990-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 16 | 9 | 12-bit | 13 | ✓ | 1 | ✓ | QFN-20 |
| C8051F991-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 16 | — | — | 13 | ✓ | 1 | ✓ | QFN-20 |
| C8051F996-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 12-bit | 14 | ✓ | 1 | ✓ | QFN-24 |
| C8051F996-C-GU | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | 10 | 12-bit | 14 | ✓ | 1 | ✓ | QSOP-24 |
| C8051F997-C-GM | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | 14 | ✓ | 1 | ✓ | QFN-24 |
| C8051F997-C-GU | 25 | 8 | 512 | ✓ | ✓ | 4 | ✓ | 17 | — | — | 14 | ✓ | 1 | ✓ | QSOP-24 |

**Note:** Starting with silicon revision C, the ordering part numbers have been updated to include the silicon revision and use this format: "C8051F990-C-GM". Package marking diagrams are included as Figure 3.4, Figure 3.5, and Figure 3.6 to identify the silicon revision.

## 3. Pinout and Package Definitions

**Table 3.1. Pin Definitions for the C8051F99x-C8051F98x**

| Name | Pin Numbers | | | Type | Description |
|---|---|---|---|---|---|
| | ‘F980/1/2 ‘F983/5 ‘F990/1 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GU | | |
| $V_{DD}$ | 4 | 3 | 6 | P In | Power Supply Voltage. Must be 1.8 to 3.6 V. |
| GND | 3, 12 | 2 | 5 | G | Required Ground. |
| $\overline{RST}$/ | 5 | 6 | 9 | 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 to 5 kΩ pullup to $V_{DD}$ is recommended. See Section “18. Reset Sources” on page 181 Section for a complete description. |
| C2CK | | | | D I/O | Clock signal for the C2 Debug Interface. |
| P2.7/ | 6 | 7 | 10 | D I/O | Port 2.7. This pin can only be used as GPIO. The Crossbar cannot route signals to this pin and it cannot be configured as an analog input. See Port I/O Section for a complete description. |
| C2D | | | | D I/O | Bi-directional data signal for the C2 Debug Interface. |
| P1.6/ | 8 | 9 | 12 | D I/O | Port 1.6. See Port I/O Section for a complete description. |
| XTAL3 | | | | A In | SmaRTClock Oscillator Crystal Input. See Section 20 for a complete description. |
| P1.7/ | 7 | 8 | 11 | D I/O | Port 1.7. See Port I/O Section for a complete description. |
| XTAL4 | | | | A Out | SmaRTClock Oscillator Crystal Output. See Section 20 for a complete description. |
| P0.0/ | 2 | 24 | 3 | D I/O or A In | Port 0.0. See Port I/O Section for a complete description. |
| $V_{REF}$* | | | | A In | External $V_{REF}$ Input. See Section “5.9. Voltage and Ground Reference Options” on page 88. |

***Note:** Available only on the C8051F980/2/6/8 and C8051F990/6 devices.

**Table 3.1. Pin Definitions for the C8051F99x-C8051F98x (Continued)**

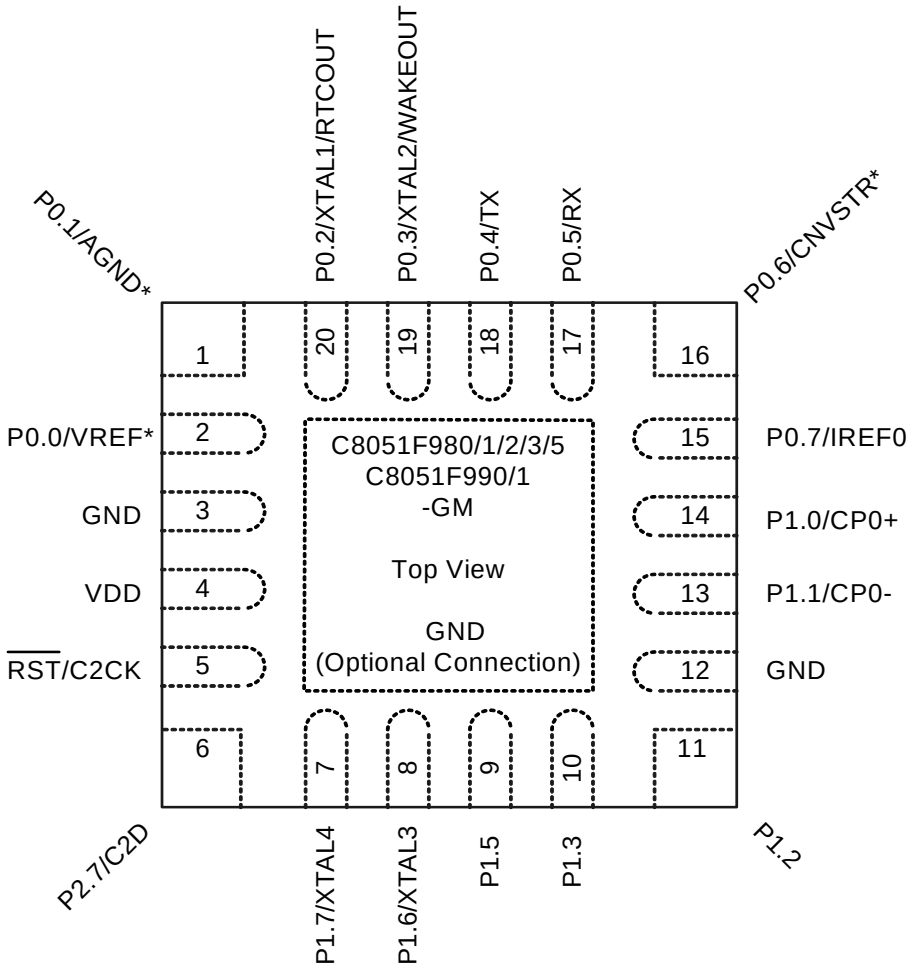
| Name | Pin Numbers | | | Type | Description |
|---|---|---|---|---|---|
| | ‘F980/1/2 ‘F983/5 ‘F990/1 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GU | | |
| P0.1/ | 1 | 23 | 2 | D I/O or A In | Port 0.1. See Port I/O Section for a complete description. |
| AGND* | | | | G | Optional Analog Ground. See Section “5.9. Voltage and Ground Reference Options” on page 88. |
| P0.2/ | 20 | 22 | 1 | D I/O or A In | Port 0.2. See Port I/O Section for a complete description. |
| XTAL1/ | | | | A In | External Clock Input. This pin is the external oscillator return for a crystal or resonator. See Section “19. Clocking Sources” on page 188. |
| RTCOUT | | | | D Out | Buffered SmaRTClock oscillator output. |
| P0.3/ | 19 | 21 | 24 | D I/O or A In | Port 0.3. See Section “21. Port Input/Output” on page 215 for a complete description. |
| XTAL2/ | | | | A Out | External Clock Output. This pin is the excitation driver for an external crystal or resonator. |
| | | | | D In | External Clock Input. This pin is the external clock input in external CMOS clock mode. |
| | | | | A In | External Clock Input. This pin is the external clock input in capacitor or RC oscillator configurations. See Section “19. Clocking Sources” on page 188 for complete details. |
| WAKEOUT | | | | D Out | Wake-up request signal to wake up external devices. |
| P0.4/ | 18 | 20 | 23 | D I/O or A In | Port 0.4. See Section “21. Port Input/Output” on page 215 for a complete description. |
| TX | | | | D Out | UART TX Pin. See Section “21. Port Input/Output” on page 215. |
| P0.5/ | 17 | 19 | 22 | D I/O or A In | Port 0.5. See Section “21. Port Input/Output” on page 215 for a complete description. |
| RX | | | | D In | UART RX Pin. See Section “21. Port Input/Output” on page 215. |

***Note:** Available only on the C8051F980/2/6/8 and C8051F990/6 devices.

**Table 3.1. Pin Definitions for the C8051F99x-C8051F98x (Continued)**

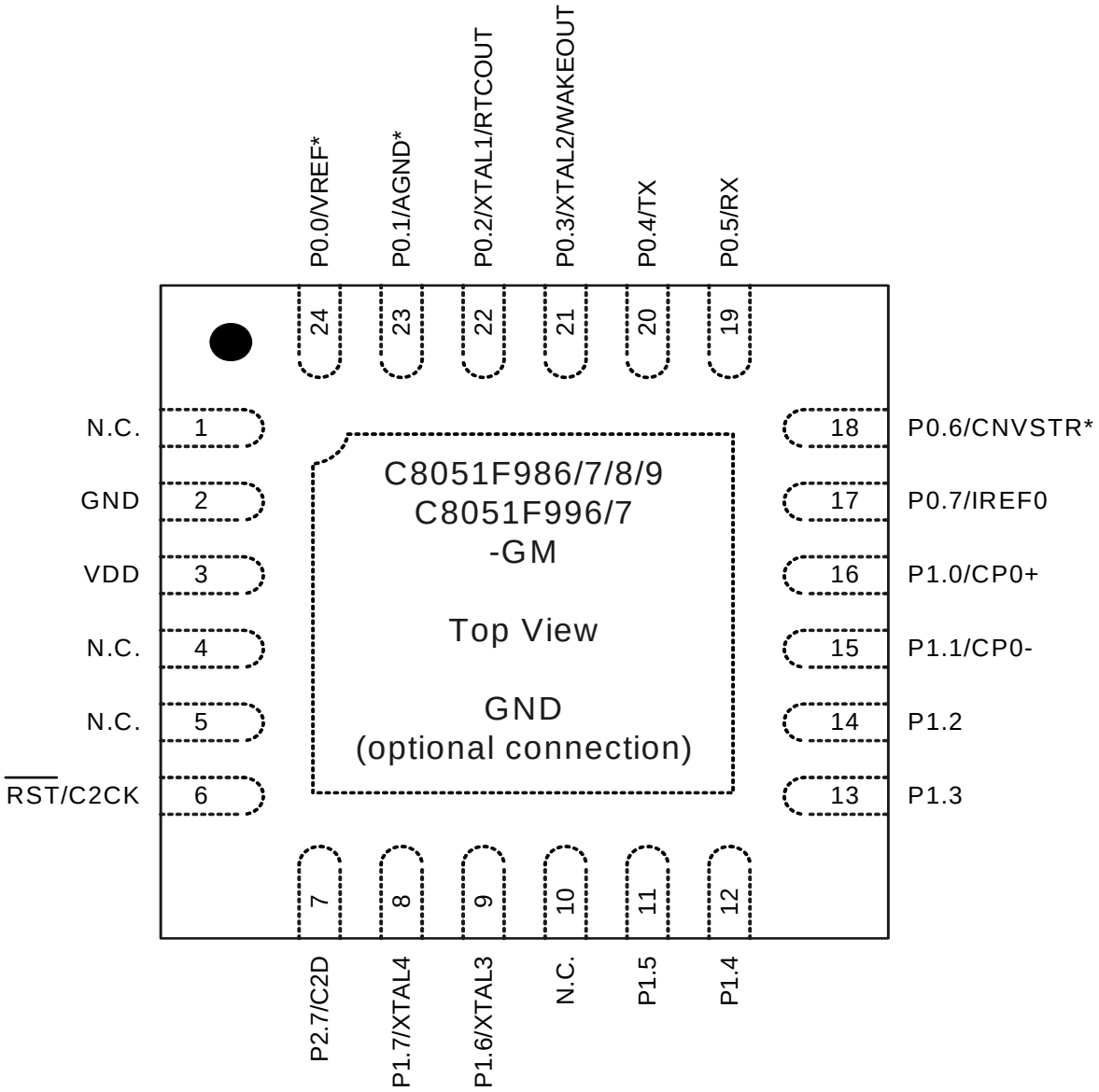
| Name | Pin Numbers | | | Type | Description |
|---|---|---|---|---|---|
| | ‘F980/1/2 ‘F983/5 ‘F990/1 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GM | ‘F986/7 ‘F988/9 ‘F996/7 -GU | | |
| P0.6/ | 16 | 18 | 21 | D I/O or A In | Port 0.6. See Section “21. Port Input/Output” on page 215 for a complete description. |
| CNVSTR* | | | | D In | External Convert Start Input for ADC0. See Section “5.7. ADC0 Analog Multiplexer” on page 83 for a complete description. |
| P0.7/ | 15 | 17 | 20 | D I/O or A In | Port 0.7. See Section “21. Port Input/Output” on page 215 for a complete description. |
| IREF0 | | | | A Out | IREF0 Output. See IREF Section for complete description. |
| P1.0 | 14 | 16 | 19 | D I/O or A In | Port 1.0. See Section “21. Port Input/Output” on page 215 for a complete description. May also be used as SCK for SPI1. |
| CP0+ | | | | A In | Comparator0 positive input. See Comparator Section for complete description. |
| P1.1 | 13 | 15 | 18 | D I/O or A In | Port 1.1. See Section “21. Port Input/Output” on page 215 for a complete description. |
| CP0- | | | | A In | Comparator0 negative input. See Comparator Section for complete description. |
| P1.2 | 11 | 14 | 17 | D I/O or A In | Port 1.2. See Section “21. Port Input/Output” on page 215 for a complete description. |
| P1.3 | 10 | 13 | 16 | D I/O or A In | Port 1.3. See Section “21. Port Input/Output” on page 215 for a complete description. |
| P1.4 | — | 12 | 15 | D I/O or A In | Port 1.4. See Section “21. Port Input/Output” on page 215 for a complete description. |
| P1.5 | 9 | 11 | 14 | D I/O or A In | Port 1.5. See Section “21. Port Input/Output” on page 215 for a complete description. |

***Note:** Available only on the C8051F980/2/6/8 and C8051F990/6 devices.



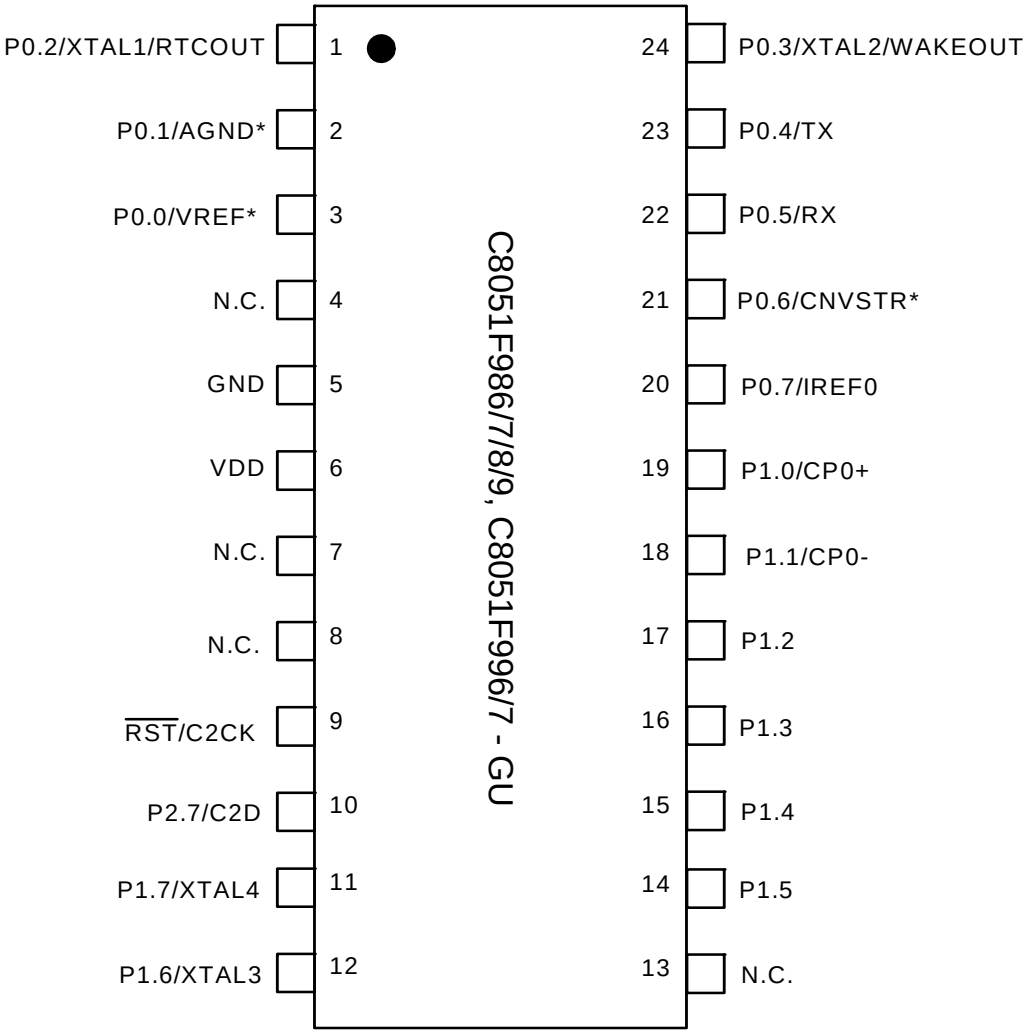

***Note:** Signal only available on ‘F980, ‘F982 and ‘F990 devices.

**Figure 3.1. QFN-20 Pinout Diagram (Top View)**




***Note:** Signal only available on ‘F986, ‘F988, and ‘F996 devices.

**Figure 3.2. QFN-24 Pinout Diagram (Top View)**




*__Note:__ Signal only available on ‘F986, ‘F988, and ‘F996 devices.

**Figure 3.3. QSOP-24 Pinout Diagram (Top View)**

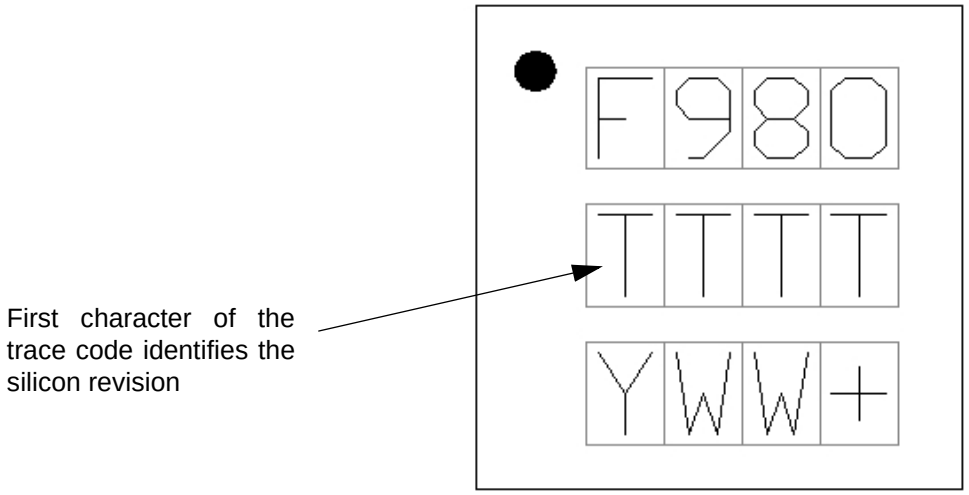



**Figure 3.4. QFN-20 Package Marking Diagram**

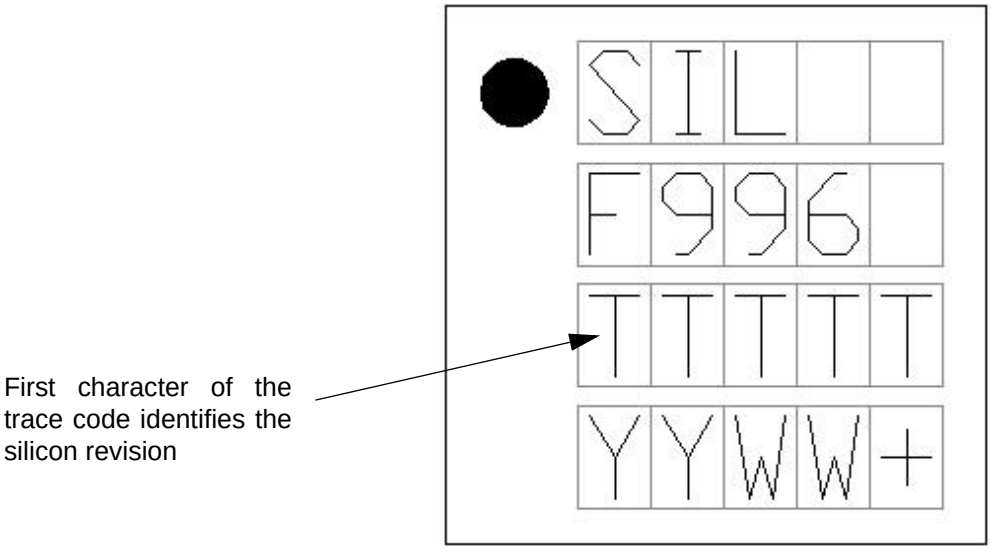



**Figure 3.5. QFN-24 Package Marking Diagram**

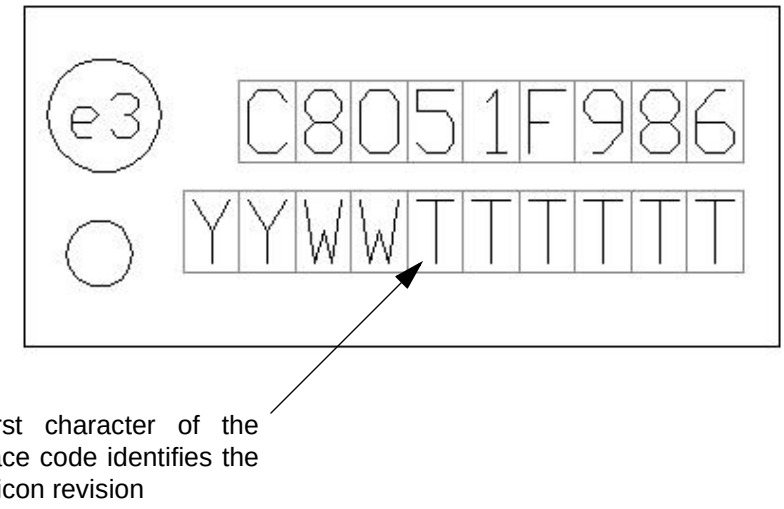



**Figure 3.6. QSOP-24 Package Marking Diagram**

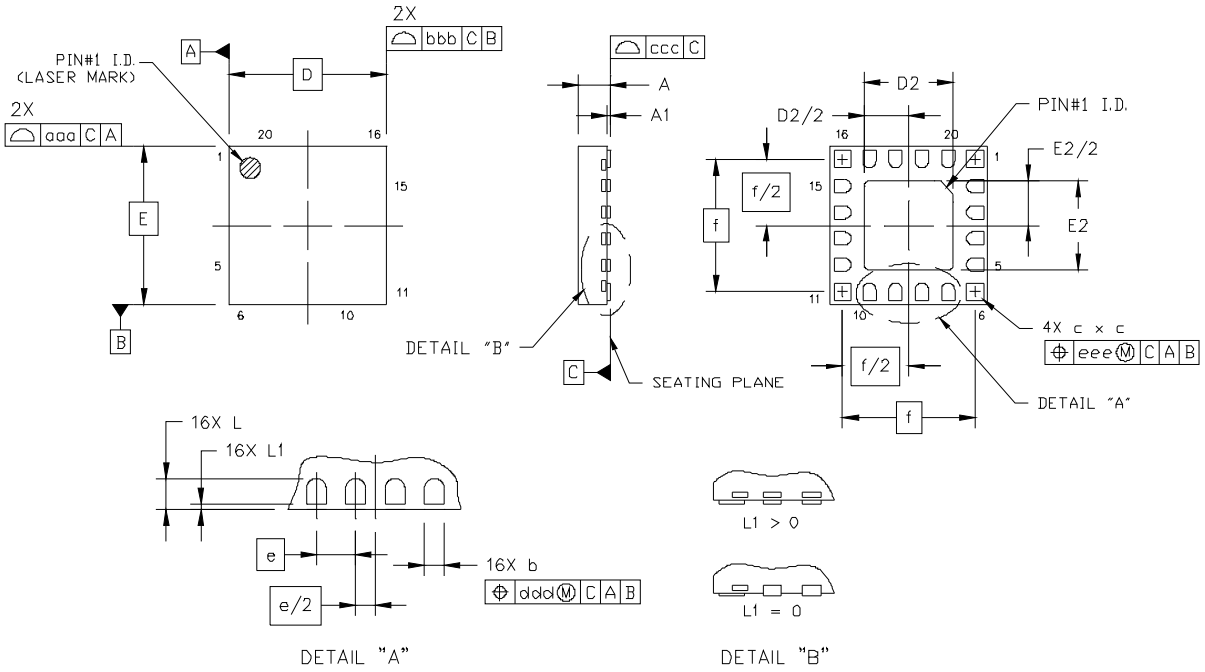



**Figure 3.7. QFN-20 Package Drawing**

**Table 3.2. QFN-20 Package Dimensions**

| Dimension | Min | Typ | Max | | Dimension | Min | Typ | Max |
|---|---|---|---|---|---|---|---|---|
| A | 0.50 | 0.55 | 0.60 | | f | 2.53 BSC | | |
| A1 | 0.00 | 0.02 | 0.05 | | L | 0.35 | 0.40 | 0.45 |
| b | 0.20 | 0.25 | 0.30 | | L1 | 0.00 | — | 0.10 |
| c | 0.27 | 0.32 | 0.375 | | aaa | — | — | 0.05 |
| D | 3.00 BSC | | | | bbb | — | — | 0.05 |
| D2 | 1.6 | 1.70 | 1.8 | | ccc | | — | 0.08 |
| e | 0.50 BSC | | | | ddd | — | — | 0.10 |
| E | 3.00 BSC | | | | eee | — | — | 0.10 |
| E2 | 1.6 | 1.70 | 1.8 | | | | | |

**Notes:**
1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.

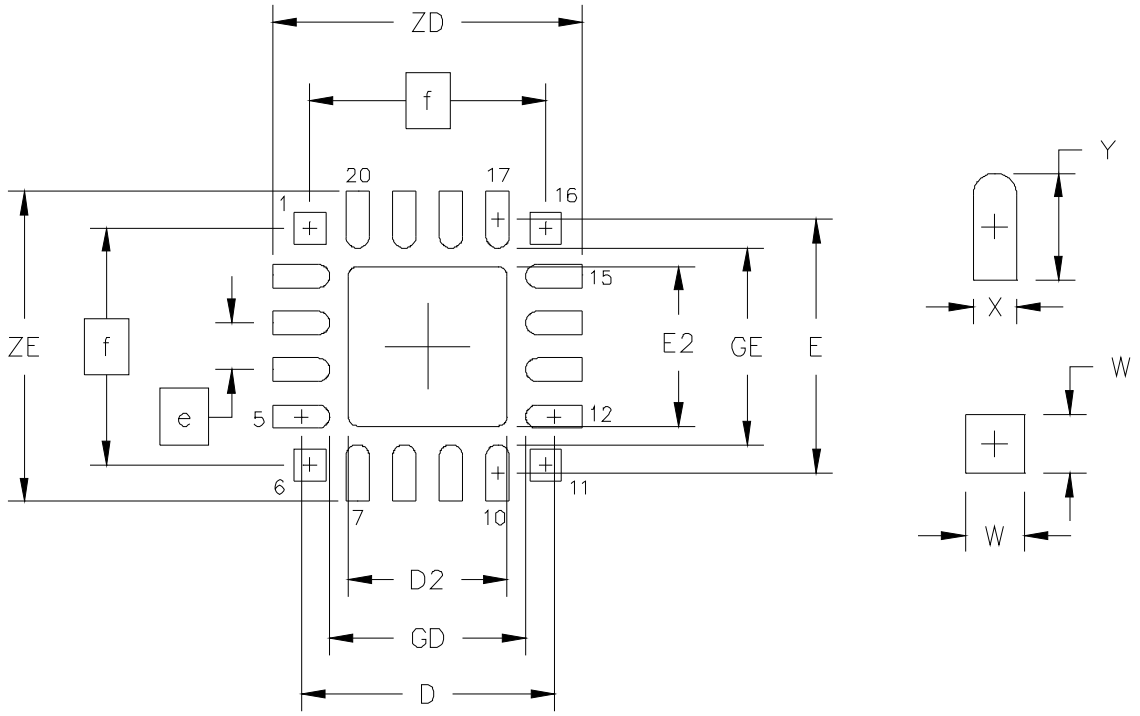



**Figure 3.8. Typical QFN-20 Landing Diagram**

**Table 3.3. PCB Land Pattern**

| Dimension | Min | Max |
|---|---|---|
| D | 2.71 REF | |
| D2 | 1.60 | 1.80 |
| e | 0.50 BSC | |
| E | 2.71 REF | |
| E2 | 1.60 | 1.80 |
| f | 2.53 REF | |
| GD | 2.10 | — |
| GE | 2.10 | — |
| W | — | 0.34 |
| X | — | 0.28 |
| Y | 0.61 REF | |
| ZE | — | 3.31 |
| ZD | — | 3.31 |

**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 IPC-SM-782 guidelines.
4. All dimensions shown are at Maximum Material Condition (MMC). Least Material Condition (LMC) is calculated based on a Fabrication Allowance of 0.05 mm.

**Solder Mask Design**

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

**Stencil Design**

1. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
2. The stencil thickness should be 0.125 mm (5 mils).
3. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pads.
4. A 1.45 x 1.45 mm square aperture should be used for the center pad. This provides approximately 70% solder paste coverage on the pad, which is optimum to assure correct component stand-off.

**Card Assembly**

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

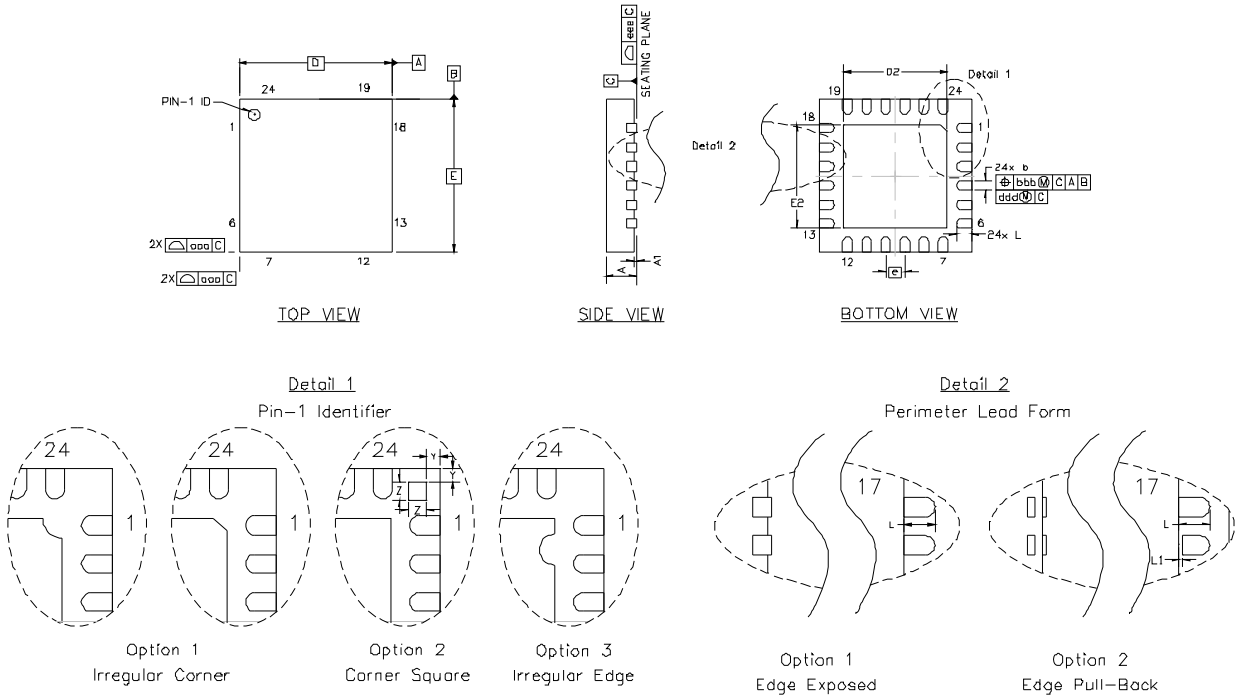



**Figure 3.9. QFN-24 Package Drawing**

**Table 3.4. QFN-24 Package Dimensions**

| Dimension | Min | Typ | Max | | Dimension | Min | Typ | Max |
|---|---|---|---|---|---|---|---|---|
| A | 0.70 | 0.75 | 0.80 | | L | 0.30 | 0.40 | 0.50 |
| A1 | 0.00 | 0.02 | 0.05 | | L1 | 0.00 | — | 0.15 |
| b | 0.18 | 0.25 | 0.30 | | aaa | — | — | 0.15 |
| D | 4.00 BSC | | | | bbb | — | — | 0.10 |
| D2 | 2.55 | 2.70 | 2.80 | | ddd | — | — | 0.05 |
| e | 0.50 BSC | | | | eee | — | — | 0.08 |
| E | 4.00 BSC | | | | Z | — | 0.24 | — |
| E2 | 2.55 | 2.70 | 2.80 | | Y | — | 0.18 | — |

**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 WGGD 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-020 specification for Small Body Components.

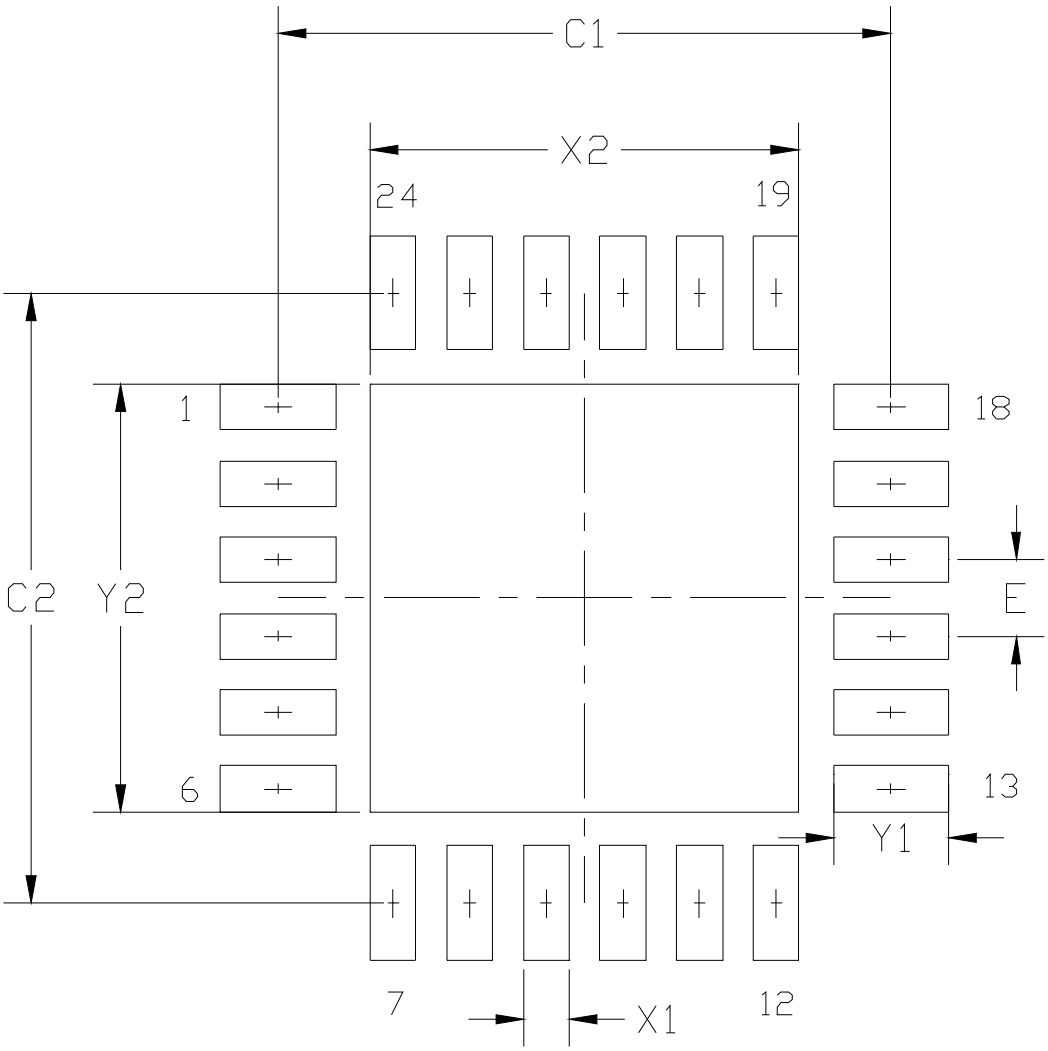



**Figure 3.10. Typical QFN-24 Landing Diagram**

**Table 3.5. PCB Land Pattern**

| Dimension | MIN | MAX |
|---|---|---|
| C1 | 3.90 | 4.00 |
| C2 | 3.90 | 4.00 |
| E | 0.50 BSC | |
| X1 | 0.20 | 0.30 |
| X2 | 2.70 | 2.80 |
| Y1 | 0.65 | 0.75 |
| Y2 | 2.70 | 2.80 |

**Notes:**

**General**

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on the IPC-7351 guidelines.

**Solder Mask Design**

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

**Stencil Design**

1. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
2. The stencil thickness should be 0.125 mm (5 mils).
3. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pads.
4. A 2x2 array of 1.10 mm x 1.10 mm openings on 1.30 mm pitch should be used for the center ground pad.

**Card Assembly**

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

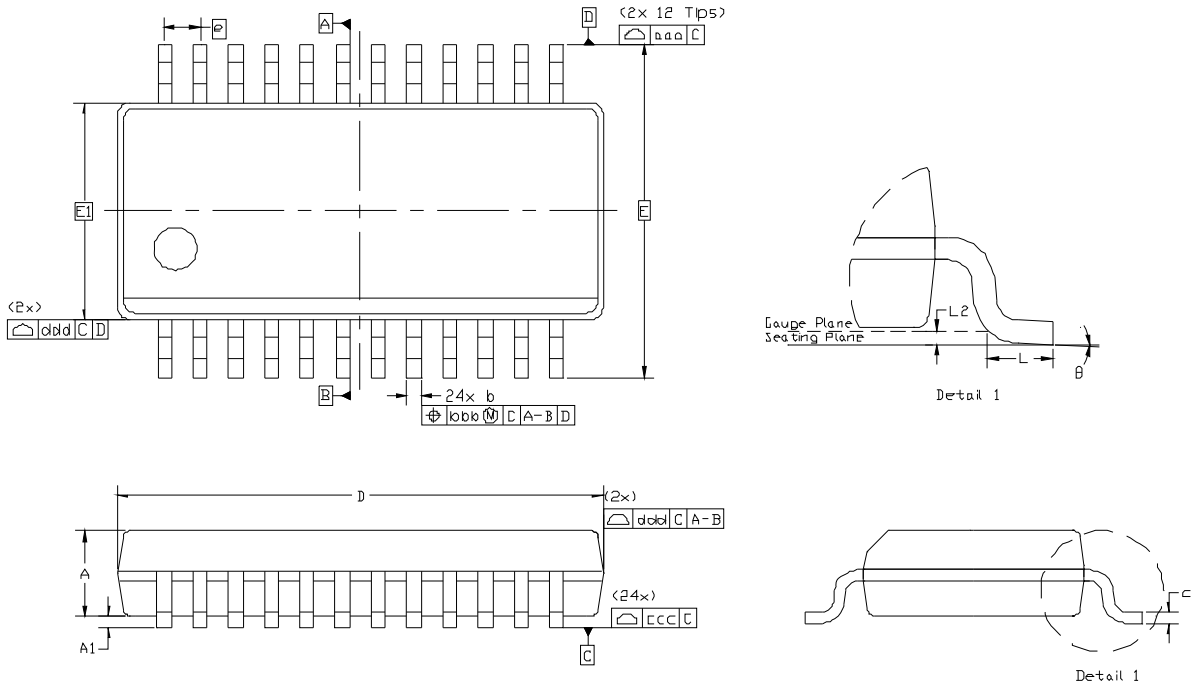



**Figure 3.11. QSOP-24 Package Diagram**

**Table 3.6. QSOP-24 Package Dimensions**

| Dimension | Min | Typ | Max | Dimension | Min | Typ | Max |
|---|---|---|---|---|---|---|---|
| A | — | — | 1.75 | L | 0.40 | — | 1.27 |
| A1 | 0.10 | — | 0.25 | L2 | 0.25 BSC | | |
| b | 0.20 | — | 0.30 | θ | 0º | — | 8º |
| c | 0.10 | — | 0.25 | aaa | 0.20 | | |
| D | 8.65 BSC. | | | bbb | 0.18 | | |
| E | 6.00 BSC | | | ccc | 0.10 | | |
| E1 | 3.90 BSC | | | ddd | 0.10 | | |
| e | 0.635 BSC | | | | | | |

**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 JEDEC outline MO-147, variation AE.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

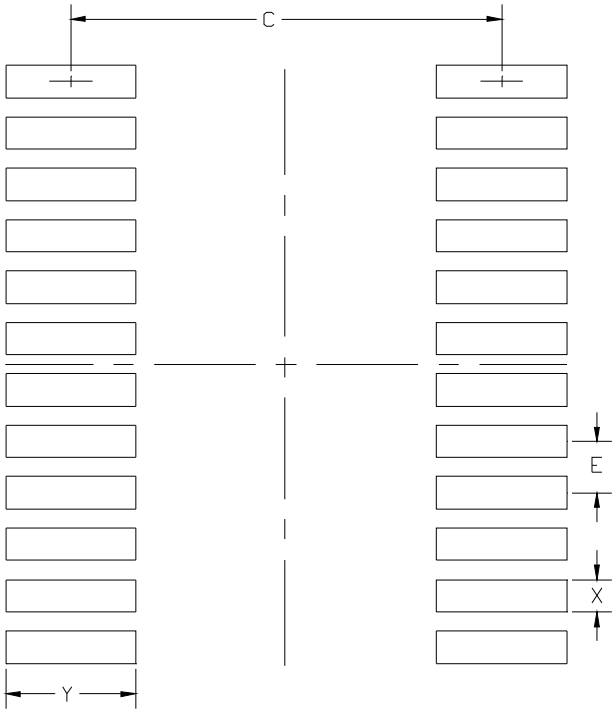



**Figure 3.12. QSOP-24 Landing Diagram**

**Table 3.7. PCB Land Pattern**

| Dimension | MIN | MAX |
|---|---|---|
| C | 5.20 | 5.30 |
| E | 0.635 BSC | |
| X | 0.30 | 0.40 |
| Y | 1.50 | 1.60 |

**Notes:**

**General**

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on the IPC-7351 guidelines.

**Solder Mask Design**

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

**Stencil Design**

1. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
2. The stencil thickness should be 0.125 mm (5 mils).
3. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pads.

**Card Assembly**

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

# 4. Electrical Characteristics

Throughout the Electrical Characteristics chapter, “VDD” refers to the Supply Voltage.

## 4.1. Absolute Maximum Specifications

**Table 4.1. Absolute Maximum Ratings**

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Ambient Temperature under Bias | | –55 | — | 125 | °C |
| Storage Temperature | | –65 | — | 150 | °C |
| Voltage on any Port I/O Pin or $\overline{RST}$ with Respect to GND | | –0.3 | — | $V_{DD}$ + 0.3 | V |
| Voltage on $V_{DD}$ with Respect to GND | | –0.3 | — | 4.0 | V |
| Maximum Total Current through $V_{DD}$ or GND | | — | — | 500 | mA |
| Maximum Current through $\overline{RST}$ or any Port Pin | | — | — | 100 | mA |
| Maximum Total Current through all Port Pins | | — | — | 200 | 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.

## 4.2. Electrical Characteristics

### Table 4.2. Global Electrical Characteristics

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Supply Voltage ($V_{DD}$) | | 1.8 | 2.4 | 3.6 | V |
| Minimum RAM Data Retention Voltage[1] | not in sleep mode<br>in sleep mode | —<br>— | 1.4<br>0.3 | —<br>0.45 | V |
| SYSCLK (System Clock)[2] | | 0 | — | 25 | MHz |
| $T_{SYSH}$ (SYSCLK High Time) | | 18 | — | — | ns |
| $T_{SYSL}$ (SYSCLK Low Time) | | 18 | — | — | ns |
| Specified Operating Temperature Range | | –40 | — | +85 | °C |
| **Digital Supply Current—CPU Active (Normal Mode, fetching instructions from Flash)** | | | | | |
| $I_{DD}$ [3, 4, 5] | $V_{DD}$ = 1.8–3.6 V, F = 24.5 MHz (includes precision oscillator current) | — | 3.6 | 4.5 | mA |
| | $V_{DD}$ = 1.8–3.6 V, F = 20 MHz (includes low power oscillator current) | — | 3.1 | — | mA |
| | $V_{DD}$ = 1.8 V, F = 1 MHz<br>$V_{DD}$ = 3.6 V, F = 1 MHz (includes external oscillator/GPIO current) | —<br>— | 225<br>290 | —<br>— | µA<br>µA |
| | $V_{DD}$ = 1.8–3.6 V, F = 32.768 kHz (includes SmaRTClock oscillator current) | — | 84 | — | µA |
| $I_{DD}$ Frequency Sensitivity[1, 3, 5] | $V_{DD}$ = 1.8–3.6 V, T = 25 °C, F < 14 MHz (Flash oneshot active, see Section 14.6) | — | 174 | — | µA/MHz |
| | $V_{DD}$ = 1.8–3.6 V, T = 25 °C, F > 14 MHz (Flash oneshot bypassed, see Section 14.6) | — | 88 | — | µA/MHz |

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

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Digital Supply Current—CPU Inactive (Idle Mode, not fetching instructions from Flash)** | | | | | |
| $I_{DD}$[4, 6] | $V_{DD}$ = 1.8–3.6 V, F = 24.5 MHz (includes precision oscillator current) | — | 1.8 | 3.0 | mA |
| | $V_{DD}$ = 1.8–3.6 V, F = 20 MHz (includes low power oscillator current) | — | 1.4 | — | mA |
| | $V_{DD}$ = 1.8 V, F = 1 MHz<br>$V_{DD}$ = 3.6 V, F = 1 MHz (includes external oscillator/GPIO current) | —<br>— | 145<br>180 | —<br>— | µA<br>µA |
| | $V_{DD}$ = 1.8–3.6 V, F = 32.768 kHz (includes SmaRTClock oscillator current) | — | 82 | — | µA |
| $I_{DD}$ Frequency Sensitivity[1,6] | $V_{DD}$ = 1.8–3.6 V, T = 25 °C | — | 67 | — | µA/MHz |
| **Digital Supply Current—Suspend and Sleep Mode** | | | | | |
| Digital Supply Current (Suspend Mode) | $V_{DD}$ = 1.8–3.6 V | — | 77 | — | µA |
| Digital Supply Current (Sleep Mode, SmaRTClock running, 32.768 kHz crystal) | 1.8 V, T = 25 °C<br>3.0 V, T = 25 °C<br>3.6 V, T = 25 °C<br>1.8 V, T = 85 °C<br>3.0 V, T = 85 °C<br>3.6 V, T = 85 °C<br>(includes SmaRTClock oscillator and $V_{DD}$ Supply Monitor) | —<br>—<br>—<br>—<br>—<br>— | 0.60<br>0.70<br>0.80<br>0.80<br>0.90<br>1.00 | —<br>—<br>—<br>—<br>—<br>— | µA |
| Digital Supply Current (Sleep Mode, SmaRTClock running, internal LFO) | 1.8 V, T = 25 °C<br>3.0 V, T = 25 °C<br>3.6 V, T = 25 °C<br>1.8 V, T = 85 °C<br>3.0 V, T = 85 °C<br>3.6 V, T = 85 °C<br>(includes SmaRTClock oscillator and $V_{DD}$ Supply Monitor) | —<br>—<br>—<br>—<br>—<br>— | 0.30<br>0.40<br>0.50<br>0.50<br>0.70<br>0.80 | —<br>—<br>—<br>—<br>—<br>— | µA |
| Digital Supply Current (Sleep Mode) | 1.8 V, T = 25 °C<br>3.0 V, T = 25 °C<br>3.6 V, T = 25 °C<br>1.8 V, T = 85 °C<br>3.0 V, T = 85 °C<br>3.6 V, T = 85 °C<br>(includes $V_{DD}$ supply monitor) | —<br>—<br>—<br>—<br>—<br>— | 0.05<br>0.07<br>0.08<br>0.20<br>0.24<br>0.28 | —<br>—<br>—<br>—<br>—<br>— | µA |
| Digital Supply Current (Sleep Mode, VDD Supply Monitor Disabled) | 1.8 V, T = 25 °C<br>3.0 V, T = 25 °C<br>3.6 V, T = 25 °C<br>1.8 V, T = 85 °C<br>3.0 V, T = 85 °C<br>3.6 V, T = 85 °C | —<br>—<br>—<br>—<br>—<br>— | 0.005<br>0.01<br>0.02<br>0.15<br>0.19<br>0.23 | —<br>—<br>—<br>—<br>—<br>— | µA |

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

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|

**Notes:**

1. Based on device characterization data; Not production tested.
2. SYSCLK must be at least 32 kHz to enable debugging.
3. Digital Supply Current depends upon the particular code being executed. The values in this table are obtained with the CPU executing an “sjmp $” loop, which is the compiled form of a while(1) loop in C. One iteration requires 3 CPU clock cycles, and the Flash memory is read on each cycle. The supply current will vary slightly based on the physical location of the sjmp instruction and the number of Flash address lines that toggle as a result. In the worst case, current can increase by up to 30% if the sjmp loop straddles a 64-byte Flash address boundary (e.g., 0x007F to 0x0080). Real-world code with larger loops and longer linear sequences will have few transitions across the 64-byte address boundaries.
4. Includes oscillator and regulator supply current.
5. IDD can be estimated for frequencies $\leq$ 14 MHz by simply multiplying the frequency of interest by the frequency sensitivity number for that range, then adding an offset of 84 µA. When using these numbers to estimate $I_{DD}$ for > 14 MHz, the estimate should be the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 3.0 V; F = 20 MHz, $I_{DD}$ = 3.6 mA – (25 MHz – 20 MHz) x 0.088 mA/MHz = 3.16 mA assuming the same oscillator setting.
6. Idle IDD can be estimated by taking the current at 25 MHz minus the difference in current indicated by the frequency sensitivity number. For example: $V_{DD}$ = 3.0 V; F = 5 MHz, Idle $I_{DD}$ = 1.75 mA – (25 MHz – 5 MHz) x 0.067 mA/MHz = 0.41 mA.

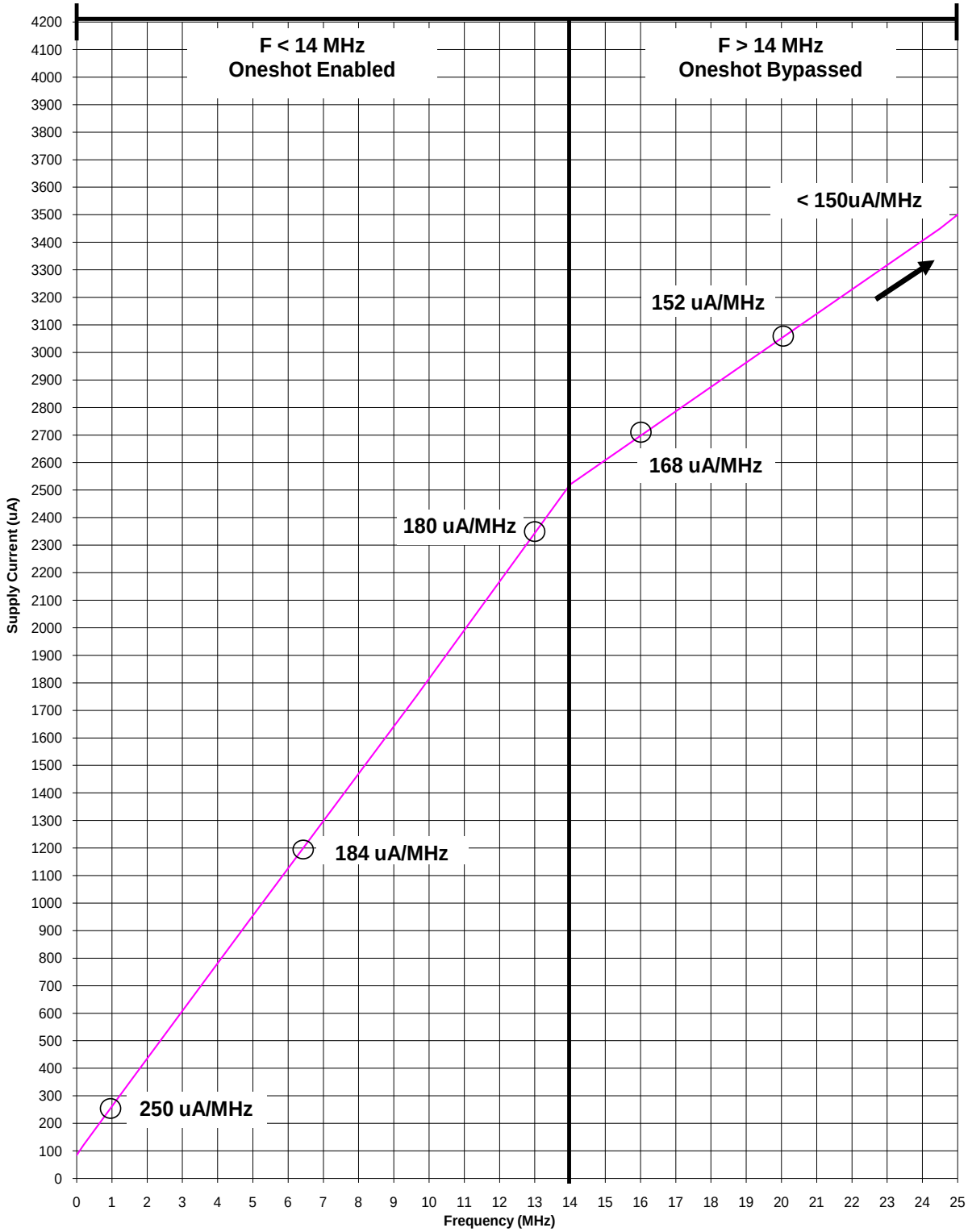



**Figure 4.1. Active Mode Current (External CMOS Clock)**

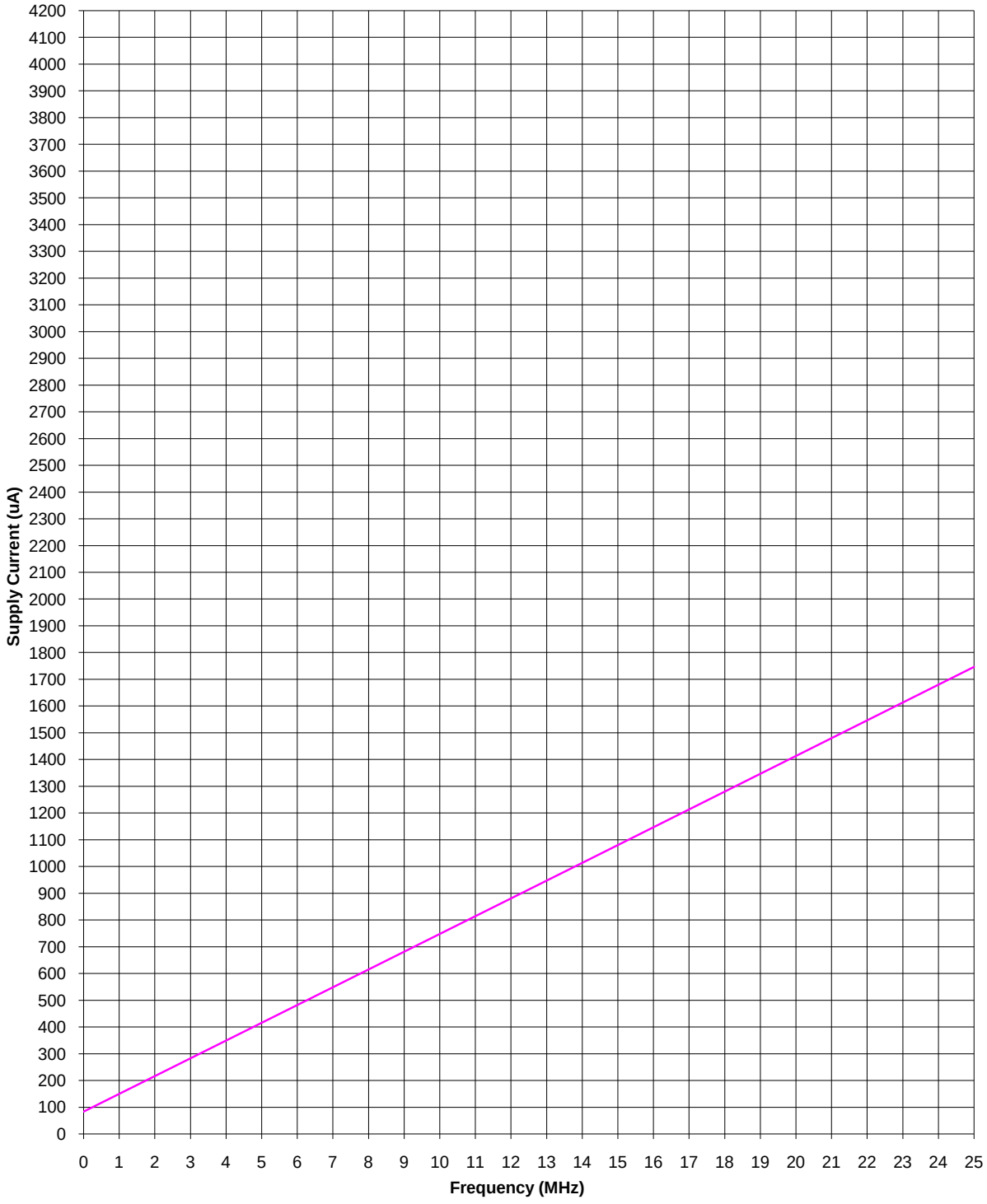



**Figure 4.2. Idle Mode Current (External CMOS Clock)**

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

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

| Parameters | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Output High Voltage | High Drive Strength, PnDRV.n = 1 | | | | V |
| | IOH = –3 mA, Port I/O push-pull | $V_{DD}$ – 0.7 | — | — | |
| | IOH = –10 µA, Port I/O push-pull | $V_{DD}$ – 0.1 | — | — | |
| | IOH = –10 mA, Port I/O push-pull | | See Chart | | |
| | Low Drive Strength, PnDRV.n = 0 | | | | |
| | IOH = –1 mA, Port I/O push-pull | $V_{DD}$ – 0.7 | — | — | |
| | IOH = –10 µA, Port I/O push-pull | $V_{DD}$ – 0.1 | — | — | |
| | IOH = –3 mA, Port I/O push-pull | — | See Chart | — | |
| Output Low Voltage | High Drive Strength, PnDRV.n = 1 | | | | V |
| | $I_{OL}$ = 8.5 mA | — | — | 0.6 | |
| | $I_{OL}$ = 10 µA | — | — | 0.1 | |
| | $I_{OL}$ = 25 mA | — | See Chart | — | |
| | Low Drive Strength, PnDRV.n = 0 | | | | |
| | $I_{OL}$ = 1.4 mA | — | — | 0.6 | |
| | $I_{OL}$ = 10 µA | — | — | 0.1 | |
| | $I_{OL}$ = 4 mA | — | See Chart | — | |
| Input High Voltage | $V_{DD}$ = 2.0 to 3.6 V | $V_{DD}$ – 0.6 | — | — | V |
| | $V_{DD}$ = 0.9 to 2.0 V | 0.7 x VDD | — | — | V |
| Input Low Voltage | $V_{DD}$ = 2.0 to 3.6 V | — | — | 0.6 | V |
| | $V_{DD}$ = 0.9 to 2.0 V | — | — | 0.3 x VDD | V |
| Input Leakage Current | Weak Pullup Off | — | — | ±1 | µA |
| | Weak Pullup On, $V_{IN}$ = 0 V, $V_{DD}$ = 1.8 V | — | 4 | — | |
| | Weak Pullup On, Vin = 0 V, $V_{DD}$ = 3.6 V | — | 20 | 35 | |

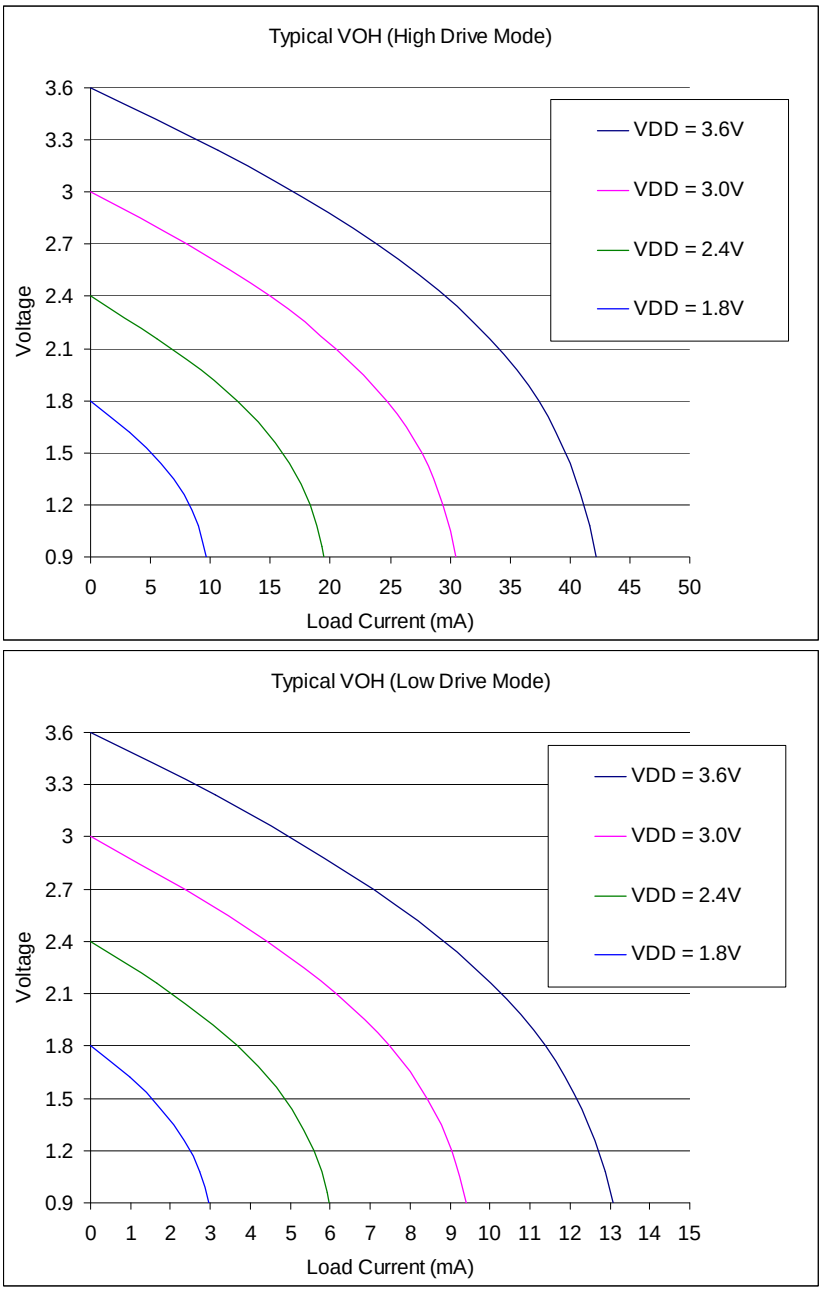



**Figure 4.3. Typical VOH Curves, 1.8–3.6 V**

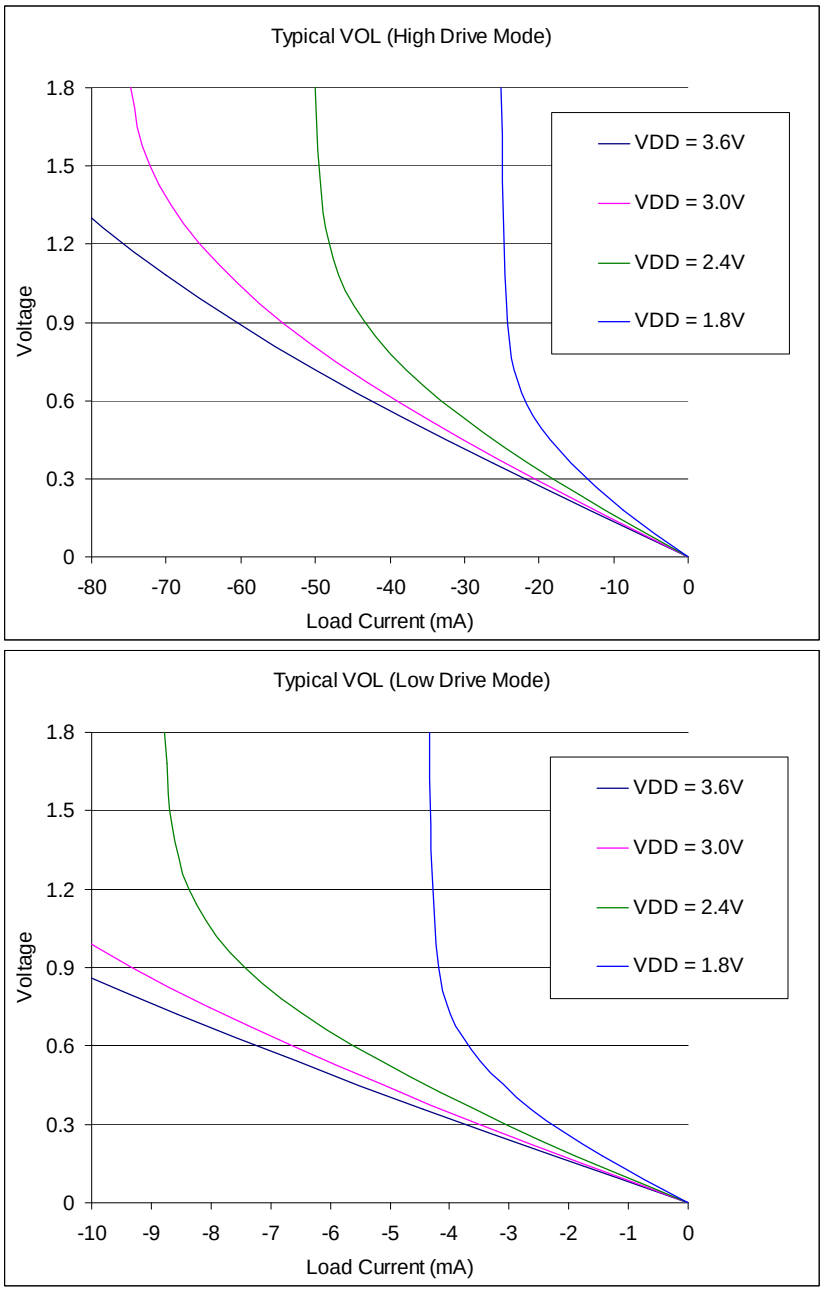



**Figure 4.4. Typical VOL Curves, 1.8–3.6 V**

**Table 4.4. Reset Electrical Characteristics**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| $\overline{RST}$ Output Low Voltage | $I_{OL}$ = 1.4 mA, | — | — | 0.6 | V |
| $\overline{RST}$ Input High Voltage | $V_{DD}$ = 12.0 to 3.6 V | $V_{DD}$ – 0.6 | — | — | V |
| | $V_{DD}$ = 10.9 to 2.0 V | 0.7 x $V_{DD}$ | — | — | V |
| $\overline{RST}$ Input Low Voltage | $V_{DD}$ = 12.0 to 3.6 V | — | — | 0.6 | V |
| | $V_{DD}$ = 10.9 to 2.0 V | — | — | 0.3 x $V_{DD}$ | V |
| $\overline{RST}$ Input Pullup Current | $\overline{RST}$ = 10.0 V, $V_{DD}$ = 1.8 V | — | 4 | — | µA |
| | $\overline{RST}$ = 10.0 V, $V_{DD}$ = 13.6 V | — | 20 | 30 | |
| $V_{DD}$ Monitor Threshold ($V_{RST}$) | Early Warning | 1.8 | 1.85 | 1.9 | V |
| | Reset Trigger (all power modes except Sleep) | 1.7 | 1.75 | 1.8 | |
| $V_{DD}$ Ramp Time for Power On | $V_{DD}$ Ramp from 0–1.8 V | — | — | 3 | ms |
| POR Monitor Threshold ($V_{POR}$) | Brownout Condition ($V_{DD}$ Falling) | 0.75 | 1.0 | 1.3 | V |
| | Recovery from Brownout ($V_{DD}$ Rising) | — | 1.75 | — | |
| Missing Clock Detector Timeout | Time from last system clock rising edge to reset initiation | 100 | 650 | 1000 | µs |
| Minimum System Clock w/ Missing Clock Detector Enabled | System clock frequency which triggers a missing clock detector timeout | — | 7 | 10 | kHz |
| Reset Time Delay | Delay between release of any reset source and code execution at location 0x0000 | — | 10 | — | µs |
| Minimum $\overline{RST}$ Low Time to Generate a System Reset | | 15 | — | — | µs |
| $V_{DD}$ Monitor Turn-on Time | | — | 300 | — | ns |
| $V_{DD}$ Monitor Supply Current | | — | 7 | — | µA |

### Table 4.5. Power Management Electrical Specifications

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Idle Mode Wake-up Time | | 2 | — | 3 | SYSCLKs |
| Suspend Mode Wake-up Time | CLKDIV = 0x00<br>Low Power or Precision Osc. | — | 400 | — | ns |
| Sleep Mode Wake-up Time | | — | 2 | — | µs |

### Table 4.6. Flash Electrical Characteristics

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Flash Size | C8051F980/1/6/7, C8051F990/1/6/7 | 8192 | — | — | bytes |
| | C8051F982/3/8/9 | 4096 | — | — | bytes |
| | C8051F985 | 2048 | — | — | bytes |
| Endurance | | 20 k | 100k | — | Erase/Write Cycles |
| Erase Cycle Time | | 28 | 32 | 36 | ms |
| Write Cycle Time | | 57 | 64 | 71 | µs |

### Table 4.7. Internal Precision Oscillator Electrical Characteristics

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

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

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

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

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

### Table 4.9. SmaRTClock Characteristics

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Oscillator Frequency (LFO) | | 13.1 | 16.4 | 19.7 | kHz |

### Table 4.10. ADC0 Electrical Characteristics

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **DC Accuracy** | | | | | |
| Resolution | 12-bit mode<br>10-bit mode | | 12<br>10 | | bits |
| Integral Nonlinearity | 12-bit mode[1]<br>10-bit mode | —<br>— | ±1<br>±0.5 | ±1.5<br>±1 | LSB |
| Differential Nonlinearity (Guaranteed Monotonic) | 12-bit mode[1]<br>10-bit mode | —<br>— | ±0.8<br>±0.5 | ±1<br>±1 | LSB |
| Offset Error | 12-bit mode<br>10-bit mode | —<br>— | ±<1<br>±<1 | ±2<br>±2 | LSB |
| Full Scale Error | 12-bit mode[2]<br>10-bit mode | —<br>— | ±1<br>±1 | ±4<br>±2.5 | LSB |
| **Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, maximum sampling rate)** | | | | | |
| Signal-to-Noise Plus Distortion[3] | 12-bit mode<br>10-bit mode | 62<br>54 | 65<br>58 | —<br>— | dB |
| Signal-to-Distortion[3] | 12-bit mode<br>10-bit mode | —<br>— | 76<br>73 | —<br>— | dB |
| Spurious-Free Dynamic Range[3] | 12-bit mode<br>10-bit mode | —<br>— | 82<br>75 | —<br>— | dB |
| **Conversion Rate** | | | | | |
| SAR Conversion Clock | Normal Power Mode<br>Low Power Mode | —<br>— | —<br>— | 8.33<br>4.4 | MHz |
| Conversion Time in SAR Clocks | 10-bit Mode<br>8-bit Mode | 13<br>11 | —<br>— | —<br>— | clocks |
| Track/Hold Acquisition Time | Initial Acquisition<br>Subsequent Acquisitions (DC input, burst mode) | 1.5<br>1.1 | —<br>— | —<br>— | us |
| Throughput Rate | 12-bit mode<br>10-bit mode | —<br>— | —<br>— | 75<br>300 | ksps |

1. INL and DNL specifications for 12-bit mode do not include the first or last four ADC codes.
2. The maximum code in 12-bit mode is 0xFFFC. The Full Scale Error is referenced from the maximum code.
3. Performance in 8-bit mode is similar to 10-bit mode.

**Table 4.10. ADC0 Electrical Characteristics (Continued)**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Analog Inputs** | | | | | |
| ADC Input Voltage Range | Single Ended (AIN+ – GND) | 0 | — | VREF | V |
| Absolute Pin Voltage with respect to GND | Single Ended | 0 | — | VDD | V |
| Sampling Capacitance | 1x Gain<br>0.5x Gain | — | 16<br>13 | — | pF |
| Input Multiplexer Impedance | | — | 5 | — | kΩ |
| **Power Specifications** | | | | | |
| Power Supply Current ($V_{DD}$ supplied to ADC0) | Normal Power Mode:<br>Conversion Mode (300 ksps)<br>Tracking Mode (0 ksps)<br>Low Power Mode:<br>Conversion Mode (150 ksps)<br>Tracking Mode (0 ksps) | <br>—<br>—<br><br>—<br>— | <br>650<br>740<br><br>370<br>400 | <br>—<br>—<br><br>—<br>— | µA |
| Power Supply Rejection | Internal High Speed VREF<br>External VREF | —<br>— | 67<br>74 | —<br>— | dB |

1. INL and DNL specifications for 12-bit mode do not include the first or last four ADC codes.
2. The maximum code in 12-bit mode is 0xFFFC. The Full Scale Error is referenced from the maximum code.
3. Performance in 8-bit mode is similar to 10-bit mode.

**Table 4.11. Temperature Sensor Electrical Characteristics**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Linearity | | — | ±1 | — | °C |
| Slope | | — | 3.40 | — | mV/°C |
| Slope Error* | | — | 40 | — | µV/°C |
| Offset | Temp = 25 °C | — | 1025 | — | mV |
| Offset Error* | Temp = 25 °C | — | 18 | — | mV |
| Temperature Sensor Turn-On Time | | — | 1.7 | — | µs |
| Supply Current | | — | 35 | — | µA |

***Note:** Represents one standard deviation from the mean.

**Table 4.12. Voltage Reference Electrical Characteristics**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| **Internal High-Speed Reference (REFSL[1:0] = 11)** | | | | | |
| Output Voltage | –40 to +85 °C,<br>$V_{DD}$ = 1.8–3.6 V | 1.62 | 1.65 | 1.68 | V |
| VREF Turn-on Time | | — | — | 1.5 | µs |
| Supply Current | Normal Power Mode<br>Low Power Mode | —<br>— | 260<br>140 | —<br>— | µA |
| **External Reference (REFSL[1:0] = 00, REFOE = 0)** | | | | | |
| Input Voltage Range | | 0 | — | $V_{DD}$ | V |
| Input Current | Sample Rate = 300 ksps;<br>VREF = 3.0 V | — | 5.25 | — | µA |

### Table 4.13. IREF0 Electrical Characteristics

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

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

**Note:** Refer to “PWM Enhanced Mode” on page 91 for information on how to improve IREF0 resolution.

**Table 4.14. Comparator Electrical Characteristics**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Response Time: Mode 0, $V_{DD}$ = 2.4 V, $V_{CM}$* = 1.2 V | CP0+ – CP0– = 100 mV | — | 120 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 110 | — | ns |
| Response Time: Mode 1, $V_{DD}$ = 2.4 V, $V_{CM}$* = 1.2 V | CP0+ – CP0– = 100 mV | — | 180 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 220 | — | ns |
| Response Time: Mode 2, $V_{DD}$ = 2.4 V, $V_{CM}$* = 1.2 V | CP0+ – CP0– = 100 mV | — | 350 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 600 | — | ns |
| Response Time: Mode 3, $V_{DD}$ = 2.4 V, $V_{CM}$* = 1.2 V | CP0+ – CP0– = 100 mV | — | 1240 | — | ns |
| | CP0+ – CP0– = –100 mV | — | 3200 | — | ns |
| Common-Mode Rejection Ratio | | — | 1.5 | — | mV/V |
| Inverting or Non-Inverting Input Voltage Range | | –0.25 | — | $V_{DD}$ + 0.25 | V |
| Input Capacitance | | — | 12 | — | pF |
| Input Bias Current | | — | 1 | — | nA |
| Input Offset Voltage | | –7 | — | +7 | mV |
| **Power Supply** | | | | | |
| Power Supply Rejection | | — | 0.1 | — | mV/V |
| Power-up Time | VDD = 3.6 V | — | 0.6 | — | µs |
| | VDD = 3.0 V | — | 1.0 | — | µs |
| | VDD = 2.4 V | — | 1.8 | — | µs |
| | VDD = 1.8 V | — | 10 | — | µs |
| Supply Current at DC | Mode 0 | — | 23 | — | µA |
| | Mode 1 | — | 8.8 | — | µA |
| | Mode 2 | — | 2.6 | — | µA |
| | Mode 3 | — | 0.4 | — | µA |

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

**Table 4.14. Comparator Electrical Characteristics (Continued)**
$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise noted.

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

**Table 4.15. VREG0 Electrical Characteristics**
$V_{DD}$ = 1.8 to 3.6 V, –40 to +85 °C unless otherwise specified.

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Input Voltage Range | | 1.8 | — | 3.6 | V |
| Bias Current | Normal, idle, suspend, or stop mode | — | 20 | — | µA |

**Table 4.16. Capacitive Sense Electrical Characteristics**

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Single Conversion Time[1] | 12-bit Mode | 20 | 25 | 40 | µs |
| | 13-bit Mode (default) | 21 | 27 | 42.5 | |
| | 14-bit Mode | 23 | 29 | 45 | |
| | 16-bit Mode | 26 | 33 | 50 | |
| Number of Channels | 24-pin Packages | | 17 | | Channels |
| | 20-pin Packages | | 16 | | |
| Capacitance per Code | Default Configuration | — | 1 | — | fF |
| Maximum External Capacitive Load | CS0CG = 111b (Default) | — | 45 | — | pF |
| | CS0CG = 000b | — | 500 | — | pF |
| Maximum External Series Impedance | CS0CG = 111b (Default) | — | 50 | — | kΩ |
| Power Supply Current | CS module bias current, 25 °C | — | 50 | 60 | µA |
| | CS module alone, maximum code output, 25 °C | — | 90 | 125 | µA |
| | Wake-on-CS threshold (suspend mode with regulator and CS module on)[3] | — | 130 | 180 | µA |

**Notes:**

1. Conversion time is specified with the default configuration.
2. RMS Noise is equivalent to one standard deviation. Peak-to-peak noise encompasses ±3.3 standard deviations. The RMS noise value is specified with the default configuration.
3. Includes only current from regulator, CS module, and MCU in suspend mode.

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

The ADC0 on C8051F980/6 and C8051F990/6 devices is a 300 ksps, 10-bit or 75 ksps, 12-bit successive-approximation-register (SAR) ADC with integrated track-and-hold and programmable window detector. ADC0 also has an autonomous low power Burst Mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. It also has a 16-bit accumulator that can automatically oversample and average the ADC results. See Section 5.4 for more details on using the ADC in 12-bit mode. C8051F982 and C8051F988 devices only support the 10-bit mode.

The ADC is fully configurable under software control via Special Function Registers. The ADC0 operates in Single-ended mode and may be configured to measure various different signals using the analog multiplexer described in “5.7. ADC0 Analog Multiplexer” on page 83. The voltage reference for the ADC is selected as described in “5.9. Voltage and Ground Reference Options” on page 88.

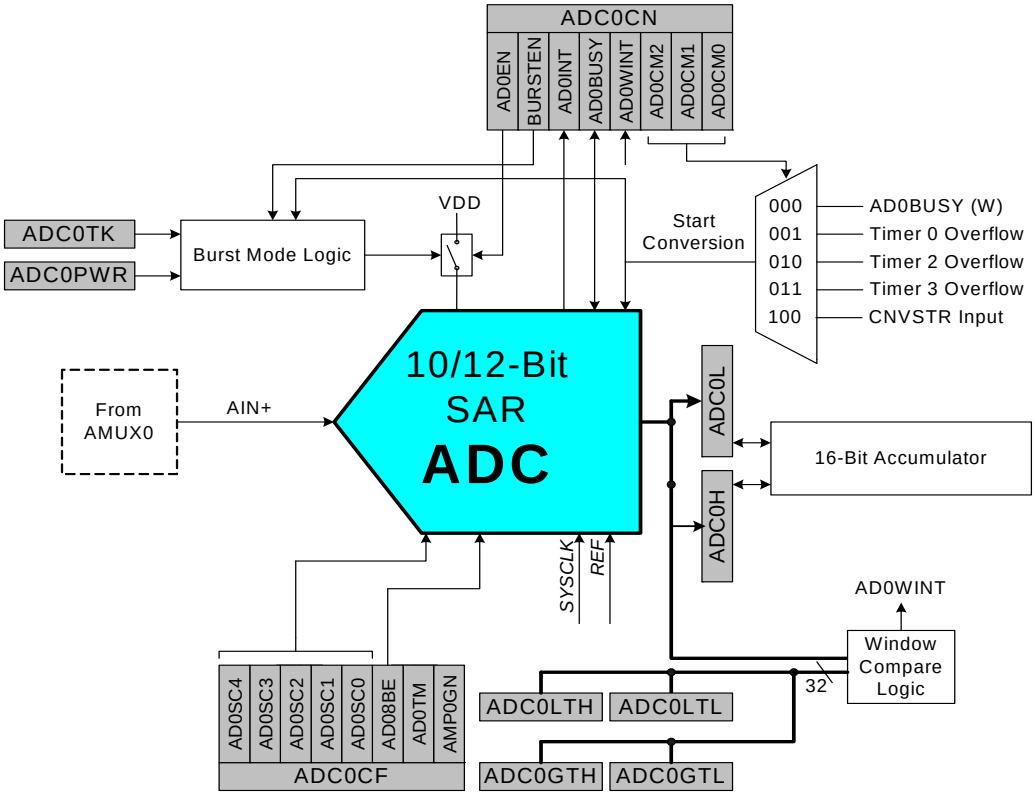



**Figure 5.1. ADC0 Functional Block Diagram**

## 5.1. Output Code Formatting

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

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

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

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

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

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

## 5.2. Modes of Operation

ADC0 has a maximum conversion speed of 300 ksps in 10-bit mode. The ADC0 conversion clock (SARCLK) is a divided version of the system clock when burst mode is disabled (BURSTEN = 0), or a divided version of the low power oscillator when burst mode is enabled (BURSEN = 1). The clock divide value is determined by the AD0SC bits in the ADC0CF register.

### 5.2.1. Starting a Conversion

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

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

Writing a 1 to AD0BUSY provides software control of ADC0 whereby conversions are performed "on-demand". During conversion, the AD0BUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the ADC0 interrupt flag (AD0INT). 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. 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 “25. Timers” on page 278 for timer configuration.

**Important Note About Using CNVSTR:** The CNVSTR input pin also functions as Port pin P0.6. When the CNVSTR input is used as the ADC0 conversion source, Port pin P0.6 should be skipped by the Digital Crossbar. To configure the Crossbar to skip P0.6, set to 1 Bit 6 in register P0SKIP. See “21. Port Input/Output” on page 215 for details on Port I/O configuration.

### 5.2.2. Tracking Modes

Each ADC0 conversion must be preceded by a minimum tracking time in order for the converted result to be accurate. The minimum tracking time is given in Table 4.10. The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state when Burst Mode is disabled, the ADC0 input is continuously tracked, except when a conversion is in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power track-and-hold mode. In this mode, each conversion is preceded by a tracking period of 3 SAR clocks (after the start-of-conversion signal). When the CNVSTR signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR (see Figure 5.2). Tracking can also be disabled (shutdown) when the device is in low power standby or sleep modes. Low-power track-and-hold mode is also useful when AMUX settings are frequently changed, due to the settling time requirements described in “5.2.4. Settling Time Requirements” on page 71.

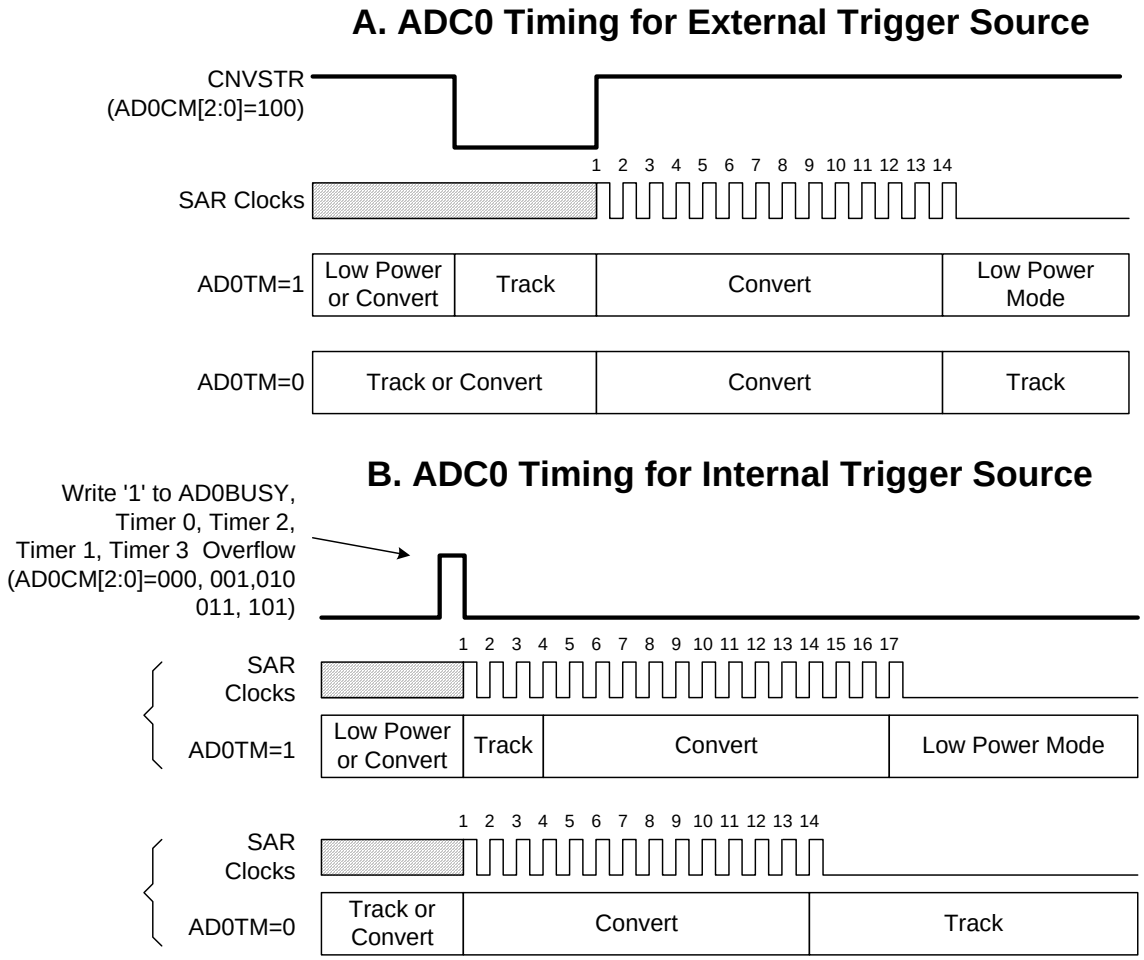



**Figure 5.2. 10-Bit ADC Track and Conversion Example Timing (BURSTEN = 0)**

### 5.2.3. Burst Mode

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

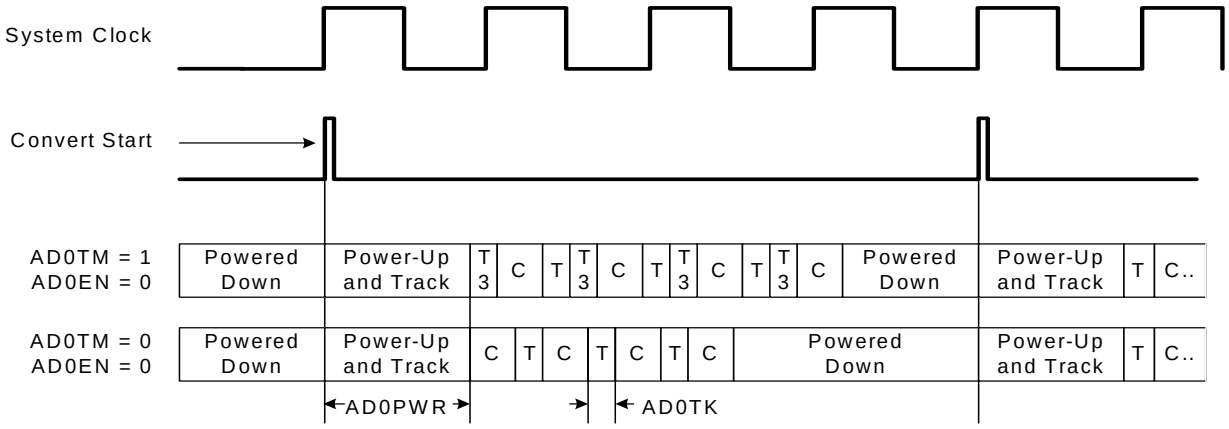
Burst Mode is enabled by setting BURSTEN to logic 1. When in Burst Mode, AD0EN controls the ADC0 idle power state (i.e. the state ADC0 enters when not tracking or performing conversions). If AD0EN is set to logic 0, ADC0 is powered down after each burst. If AD0EN is set to logic 1, ADC0 remains enabled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

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

In Burst Mode, tracking is determined by the settings in AD0PWR and AD0TK. The default settings for these registers will work in most applications without modification; however, settling time requirements may need adjustment in some applications. Refer to “5.2.4. Settling Time Requirements” on page 71 for more details.

**Notes:**

- Setting AD0TM to 1 will insert an additional 3 SAR clocks of tracking before each conversion, regardless of the settings of AD0PWR and AD0TK.
- When using Burst Mode, care must be taken to issue a convert start signal no faster than once every four SYSCLK periods. This includes external convert start signals.




T = Tracking set by AD0TK
T3 = Tracking set by AD0TM (3 SAR clocks)
C = Converting

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

### 5.2.4. Settling Time Requirements

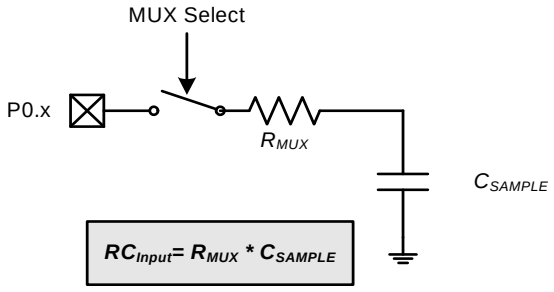
A minimum amount of tracking time is required before each conversion can be performed, to allow the sampling capacitor voltage to settle. This tracking time is determined by the AMUX0 resistance, the ADC0 sampling capacitance, any external source resistance, and the accuracy required for the conversion.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

Figure 5.4 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 5.1. When measuring the Temperature Sensor output or $V_{DD}$ with respect to GND, $R_{TOTAL}$ reduces to $R_{MUX}$. See Table 4.10 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

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

**Equation 5.1. ADC0 Settling Time Requirements**

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




**Note:** The value of CSAMPLE depends on the PGA Gain. See Table 4.10 for details.

**Figure 5.4. ADC0 Equivalent Input Circuits**

### 5.2.5. Gain Setting

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

## 5.3. 8-Bit Mode

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

## 5.4. 12-Bit Mode (C8051F980/6 and C8051F990/6 devices only)

C8051F980/6 and C8051F990/6 devices have an enhanced SAR converter that provides 12-bit resolution while retaining the 10- and 8-bit operating modes of the other devices in the family. When configured for 12-bit conversions, the ADC performs four 10-bit conversions using four different reference voltages and combines the results into a single 12-bit value. Unlike simple averaging techniques, this method provides true 12-bit resolution of AC or DC input signals without depending on noise to provide dithering. The converter also employs a hardware Dynamic Element Matching algorithm that reconfigures the largest elements of the internal DAC for each of the four 10-bit conversions to cancel any matching errors, enabling the converter to achieve 12-bit linearity performance to go along with its 12-bit resolution. For best performance, the Low Power Oscillator should be selected as the system clock source while taking 12-bit ADC measurements.

The 12-bit mode is enabled by setting the AD012BE bit (ADC0AC.7) to logic 1 and configuring Burst Mode for four conversions as described in Section 5.2.3. The conversion can be initiated using any of the methods described in Section 5.2.1, and the 12-bit result will appear in the ADC0H and ADC0L registers. Since the 12-bit result is formed from a combination of four 10-bit results, the maximum output value is 4 x (1023) = 4092, rather than the max value of (2^12 – 1) = 4095 that is produced by a traditional 12-bit converter. To further increase resolution, the burst mode repeat value may be configured to any multiple of four conversions. For example, if a repeat value of 16 is selected, the ADC0 output will be a 14-bit number (sum of four 12-bit numbers) with 13 effective bits of resolution.

## 5.5. Low Power Mode

The SAR converter provides a low power mode that allows a significant reduction in operating current when operating at low SAR clock frequencies. Low power mode is enabled by setting the AD0LPM bit (ADC0PWR.7) to 1. In general, low power mode is recommended when operating with SAR conversion clock frequency at 4 MHz or less. See the Electrical Characteristics chapter for details on power consumption and the maximum clock frequencies allowed in each mode. Setting the Low Power Mode bit reduces the bias currents in both the SAR converter and in the High-Speed Voltage Reference.

**Table 5.1. Representative Conversion Times and Energy Consumption for the SAR ADC with 1.65 V High-Speed VREF**

| | Normal Power Mode | | | Low Power Mode | | |
|---|---|---|---|---|---|---|
| | **8 bit** | **10 bit** | **12 bit** | **8 bit** | **10 bit** | **12 bit** |
| **Highest nominal SAR clock frequency** | 8.17 MHz (24.5/3) | 8.17 MHz (24.5/3) | 6.67 MHz (20.0/3) | 4.08 MHz (24.5/6) | 4.08 MHz (24.5/6) | 4.00 MHz (20.0/5) |
| **Total number of conversion clocks required** | 11 | 13 | 52 (13 x 4) | 11 | 13 | 52 (13*4) |
| **Total tracking time (min)** | 1.5 µs | 1.5 µs | 4.8 µs (1.5+3 x 1.1) | 1.5 µs | 1.5 µs | 4.8 µs (1.5+3 x 1.1) |
| **Total time for one conversion** | 2.85 µs | 3.09 µs | 12.6 µs | 4.19 µs | 4.68 µs | 17.8 µs |
| **ADC Throughput** | 351 ksps | 323 ksps | 79 ksps | 238 ksps | 214 ksps | 56 ksps |
| **Energy per conversion** | 8.2 nJ | 8.9 nJ | 36.5 nJ | 6.5 nJ | 7.3 nJ | 27.7 nJ |

**Note:** This table assumes that the 24.5 MHz precision oscillator is used for 8- and 10-bit modes, and the 20 MHz low power oscillator is used for 12-bit mode. The values in the table assume that the oscillators run at their nominal frequencies. The maximum SAR clock values given in Table 4.10 allow for maximum oscillation frequencies of 25.0 MHz and 22 MHz for the precision and low-power oscillators, respectively, when using the given SAR clock divider values. Energy calculations are for the ADC subsystem only and do not include CPU current. 12-bit mode is only available on C8051F980/6 and C8051F990/6 devices.

## SFR Definition 5.1. ADC0CN: ADC0 Control

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

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

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

## SFR Definition 5.2. ADC0CF: ADC0 Configuration

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

SFR Page = 0x0; SFR Address = 0x97

| Bit | Name | Function |
|---|---|---|
| 7:3 | AD0SC[4:0] | **ADC0 SAR Conversion Clock Divider.**<br>SAR Conversion clock is derived from FCLK by the following equation, where *AD0SC* refers to the 5-bit value held in bits AD0SC[4:0]. SAR Conversion clock requirements are given in Table 4.10.<br>BURSTEN = 0: FCLK is the current system clock.<br>BURSTEN = 1: FCLK is the 20 MHz low power oscillator, independent of the system clock.<br>$AD0SC = \frac{FCLK}{CLK_{SAR}} - 1$ *<br>*Round the result up.<br>or<br>$CLK_{SAR} = \frac{FCLK}{AD0SC + 1}$ |
| 2 | AD08BE | **ADC0 8-Bit Mode Enable.**<br>0: ADC0 operates in 10-bit mode (normal operation).<br>1: ADC0 operates in 8-bit mode. |
| 1 | AD0TM | **ADC0 Track Mode.**<br>Selects between Normal or Delayed Tracking Modes.<br>0: Normal Track Mode: When ADC0 is enabled, conversion begins immediately following the start-of-conversion signal.<br>1: Delayed Track Mode: When ADC0 is enabled, conversion begins 3 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time. |
| 0 | AMP0GN | **ADC0 Gain Control.**<br>0: The on-chip PGA gain is 0.5.<br>1: The on-chip PGA gain is 1. |

## SFR Definition 5.3. ADC0AC: ADC0 Accumulator Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD012BE | AD0AE | AD0SJST[2:0] | | | AD0RPT[2:0] | | |
| Type | R/W | W | R/W | | | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xBA

| Bit | Name | Function |
|---|---|---|
| 7 | AD012BE | **ADC0 12-Bit Mode Enable.**<br>Enables 12-bit Mode on C8051F980/6 and C8051F990/6 devices.<br>0: 12-bit Mode Disabled.<br>1: 12-bit Mode Enabled. |
| 6 | AD0AE | **ADC0 Accumulate Enable.**<br>Enables multiple conversions to be accumulated when burst mode is disabled.<br>0: ADC0H:ADC0L contain the result of the latest conversion when Burst Mode is disabled.<br>1: ADC0H:ADC0L contain the accumulated conversion results when Burst Mode is disabled. Software must write 0x0000 to ADC0H:ADC0L to clear the accumulated result.<br>This bit is write-only. Always reads 0b. |
| 5:3 | AD0SJST[2:0] | **ADC0 Accumulator Shift and Justify.**<br>Specifies the format of data read from ADC0H:ADC0L.<br>000: Right justified. No shifting applied.<br>001: Right justified. Shifted right by 1 bit.<br>010: Right justified. Shifted right by 2 bits.<br>011: Right justified. Shifted right by 3 bits.<br>100: Left justified. No shifting applied.<br>All remaining bit combinations are reserved. |
| 2:0 | AD0RPT[2:0] | **ADC0 Repeat Count.**<br>Selects the number of conversions to perform and accumulate in Burst Mode. This bit field must be set to 000 if Burst Mode is disabled.<br>000: Perform and Accumulate 1 conversion.<br>001: Perform and Accumulate 4 conversions.<br>010: Perform and Accumulate 8 conversions.<br>011: Perform and Accumulate 16 conversions.<br>100: Perform and Accumulate 32 conversions.<br>101: Perform and Accumulate 64 conversions.<br>All remaining bit combinations are reserved. |

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

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD0LPM | | | | AD0PWR[3:0] | | | |
| Type | R/W | R | R | R | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

SFR Page = All; SFR Address = 0xBB

| Bit | Name | Function |
|---|---|---|
| 7 | AD0LPM | **ADC0 Low Power Mode Enable.**<br>Enables Low Power Mode Operation.<br>0: Low Power Mode disabled.<br>1: Low Power Mode enabled. |
| 6:4 | Unused | Read = 0000b; Write = Don't Care. |
| 3:0 | AD0PWR[3:0] | **ADC0 Burst Mode Power-Up Time.**<br>Sets the time delay required for ADC0 to power up from a low power state.<br>For BURSTEN = 0:<br>ADC0 power state controlled by AD0EN.<br>For BURSTEN = 1 and AD0EN = 1:<br>ADC0 remains enabled and does not enter a low power state after all conversions are complete.<br>Conversions can begin immediately following the start-of-conversion signal.<br>For BURSTEN = 1 and AD0EN = 0:<br>ADC0 enters a low power state after all conversions are complete.<br>Conversions can begin a programmed delay after the start-of-conversion signal.<br><br>The ADC0 Burst Mode Power-Up time is programmed according to the following equation:<br><br>$AD0PWR = \frac{Tstartup}{400ns} - 1$<br>or<br>$Tstartup = (AD0PWR + 1)400ns$<br><br>**Note:** Setting AD0PWR to 0x04 provides a typical tracking time of 2 us for the first sample taken after the start of conversion. |

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

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

SFR Page = All; SFR Address = 0xBC

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

**Notes:**

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

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

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

SFR Page = 0x0; SFR Address = 0xBE

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

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

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

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

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

## 5.6. Programmable Window Detector

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

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

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

SFR Page = 0x0; SFR Address = 0xC4

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

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

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

SFR Page = 0x0; SFR Address = 0xC3

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

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

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

SFR Page = 0x0; SFR Address = 0xC6

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

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

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

SFR Page = 0x0; SFR Address = 0xC5

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

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

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

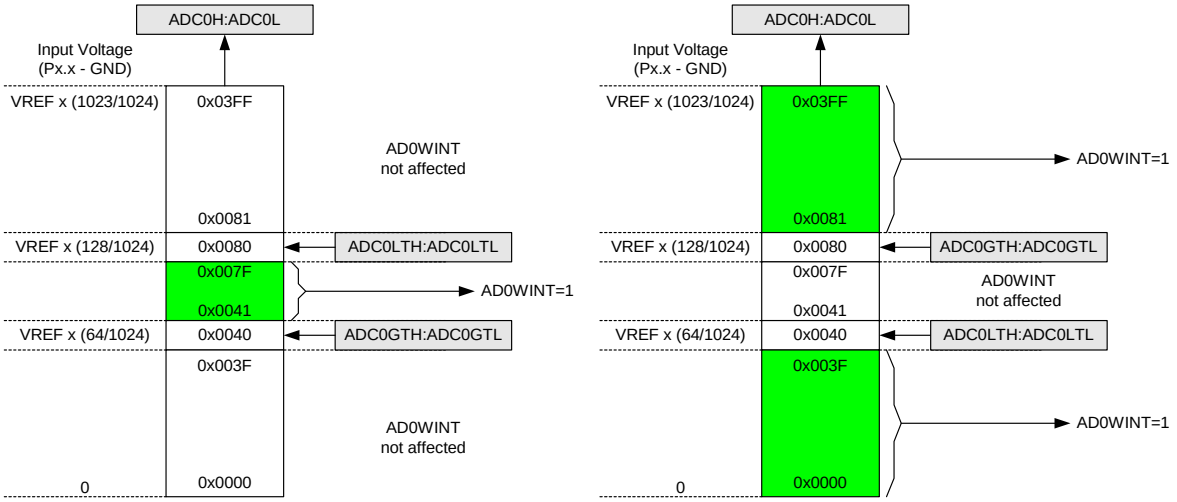



**Figure 5.5. ADC Window Compare Example: Right-Justified Single-Ended Data**

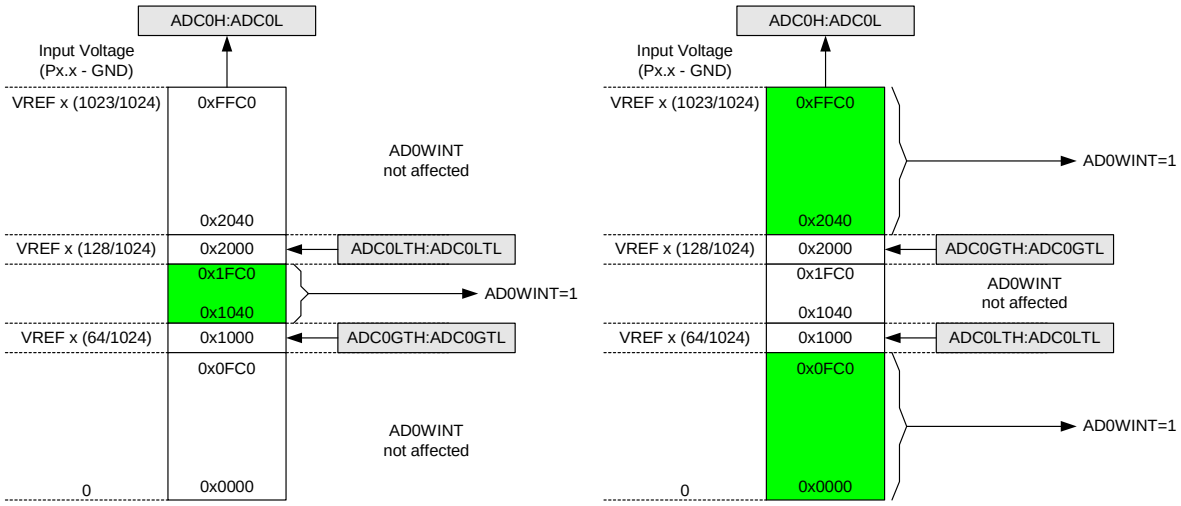



**Figure 5.6. ADC Window Compare Example: Left-Justified Single-Ended Data**

### 5.6.2. ADC0 Specifications

See “4. Electrical Characteristics” on page 48 for a detailed listing of ADC0 specifications.

## 5.7. ADC0 Analog Multiplexer

ADC0 on C8051F99x-C8051F98x has an analog multiplexer, referred to as AMUX0.

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

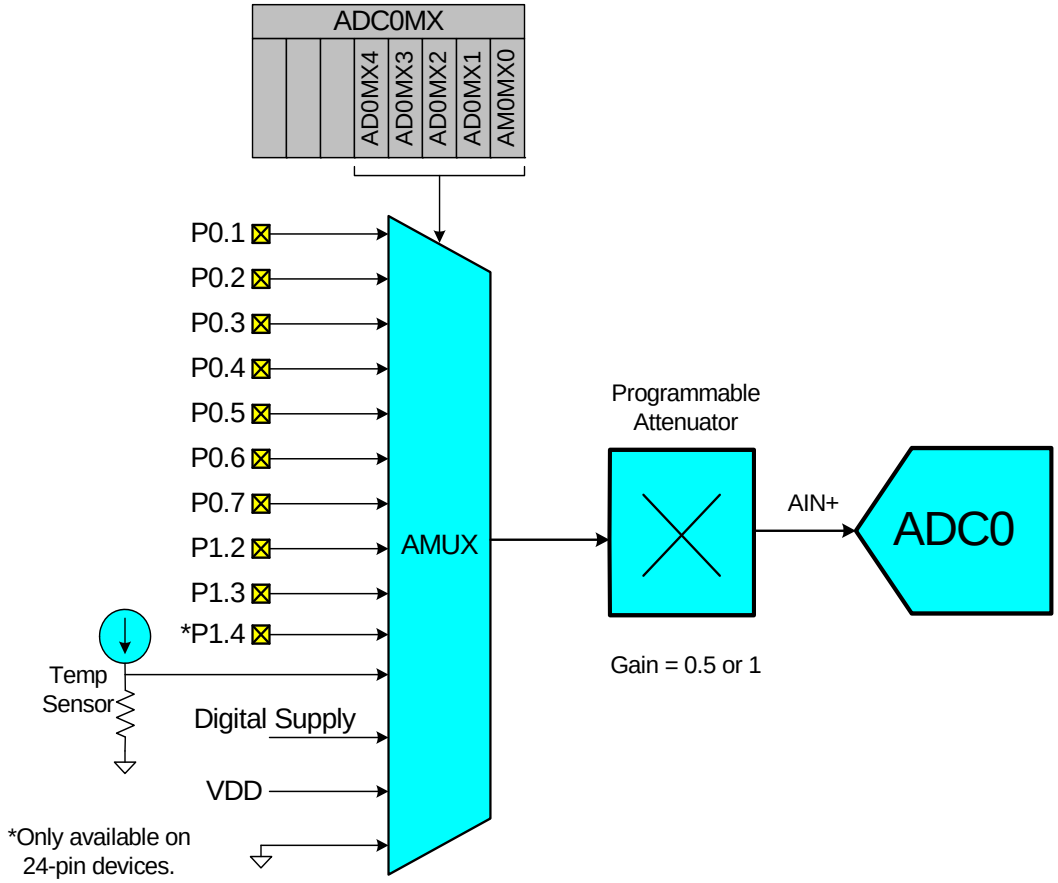



**Figure 5.7. ADC0 Multiplexer Block Diagram**

**Important Note About ADC0 Input Configuration:** Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to 0 the corresponding bit in register PnMDIN and disable the digital driver (PnMDOUT = 0 and Port Latch = 1). To force the Crossbar to skip a Port pin, set to 1 the corresponding bit in register PnSKIP. See Section “21. Port Input/Output” on page 215 for more Port I/O configuration details.

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

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

SFR Page = 0x0; SFR Address = 0x96

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

| | | | |
|---|---|---|---|
| 00000: | Reserved. | 10000: | Reserved. |
| 00001: | P0.1 | 10001: | Reserved. |
| 00010: | P0.2 | 10010: | Reserved. |
| 00011: | P0.3 | 10011: | Reserved. |
| 00100: | P0.4 | 10100: | Reserved. |
| 00101: | P0.5 | 10101: | Reserved. |
| 00110: | P0.6 | 10110: | Reserved. |
| 00111: | P0.7 | 10111: | Reserved. |
| 01000: | Reserved. | 11000: | Reserved. |
| 01001: | Reserved. | 11001: | Reserved. |
| 01010: | P1.2 | 11010: | Reserved. |
| 01011: | P1.3 | 11011: | Temperature Sensor |
| 01100: | P1.4 (only available on 24-pin devices) | 11100: | $V_{DD}$ Supply Voltage |
| 01101: | Reserved. | 11101: | Digital Supply Voltage (VREG0 Output, 1.7 V Typical) |
| 01110: | Reserved. | | |
| 01111: | Reserved. | 11110: | $V_{DD}$ Supply Voltage |
| | | 11111: | Ground |

## 5.8. Temperature Sensor

An on-chip temperature sensor is included on the C8051F99x-C8051F98x which can be directly accessed via the ADC multiplexer in single-ended configuration. To use the ADC to measure the temperature sensor, the ADC mux channel should select the temperature sensor. The temperature sensor transfer function is shown in Figure 5.8. The output voltage ($V_{TEMP}$) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register REF0CN enables/disables the temperature sensor, as described in SFR Definition 5.15. While disabled, the temperature sensor defaults to a high impedance state and any ADC measurements performed on the sensor will result in meaningless data. Refer to Table 4.10 for the slope and offset parameters of the temperature sensor.

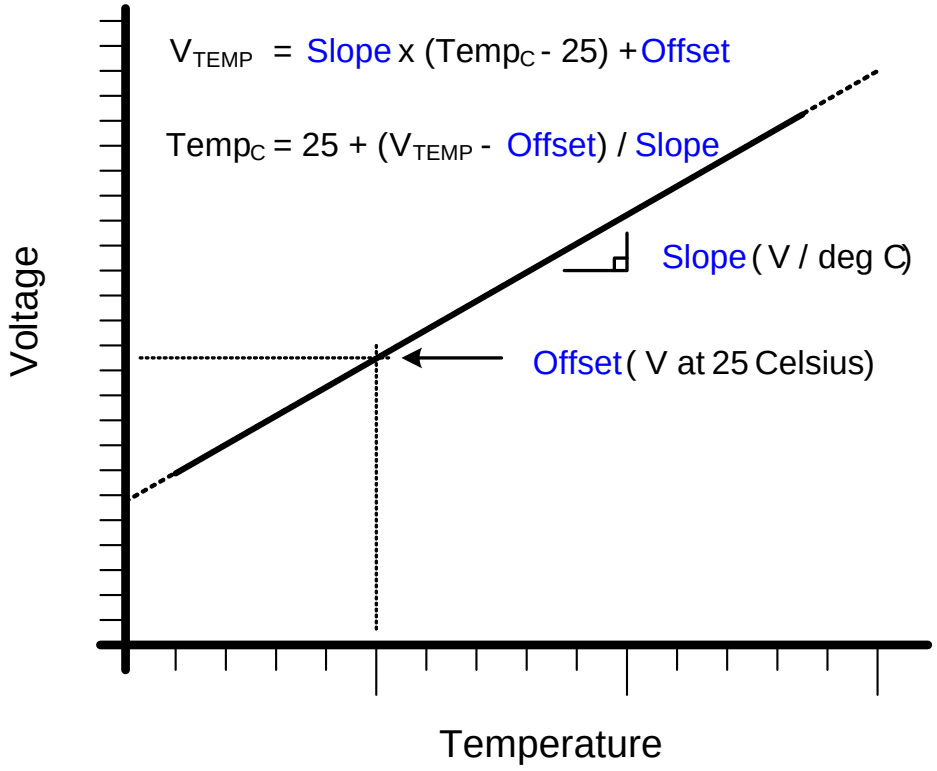



**Figure 5.8. Temperature Sensor Transfer Function**

### 5.8.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 4.11 for linearity specifications). For absolute temperature measurements, offset and/or gain calibration is recommended. Typically a 1-point (offset) calibration includes the following steps:

1. Control/measure the ambient temperature (this temperature must be known).
2. Power the device, and delay for a few seconds to allow for self-heating.
3. Perform an ADC conversion with the temperature sensor selected as the positive input and GND selected as the negative input.
4. Calculate the offset characteristics, and store this value in non-volatile memory for use with subsequent temperature sensor measurements.

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

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

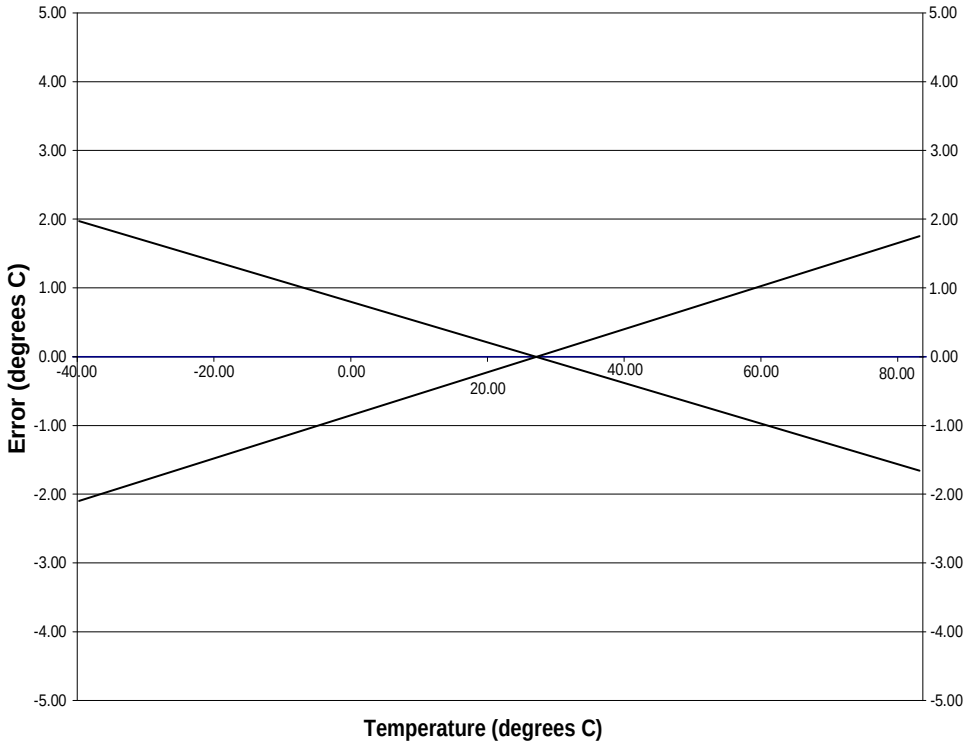



**Figure 5.9. Temperature Sensor Error with 1-Point Calibration ($V_{REF}$ = 1.65 V)**

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

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

SFR Page = 0xF; SFR Address = 0x8E

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

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

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

SFR Page = 0xF; SFR Address = 0x8D

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

## 5.9. Voltage and Ground Reference Options

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

The voltage and ground reference options are configured using the REF0CN SFR described on page 90. Electrical specifications are can be found in the Electrical Specifications Chapter.

**Important Note About the $V_{REF}$ and AGND Inputs:** Port pins are used as the external $V_{REF}$ and AGND inputs. When using an external voltage reference or the internal precision reference, P0.0/VREF should be configured as an analog input and skipped by the Digital Crossbar. When using AGND as the ground reference to ADC0, P0.1/AGND should be configured as an analog input and skipped by the Digital Crossbar. Refer to Section “21. Port Input/Output” on page 215 for complete Port I/O configuration details. The external reference voltage must be within the range $0 \le V_{REF} \le VDD$ and the external ground reference must be at the same DC voltage potential as GND.

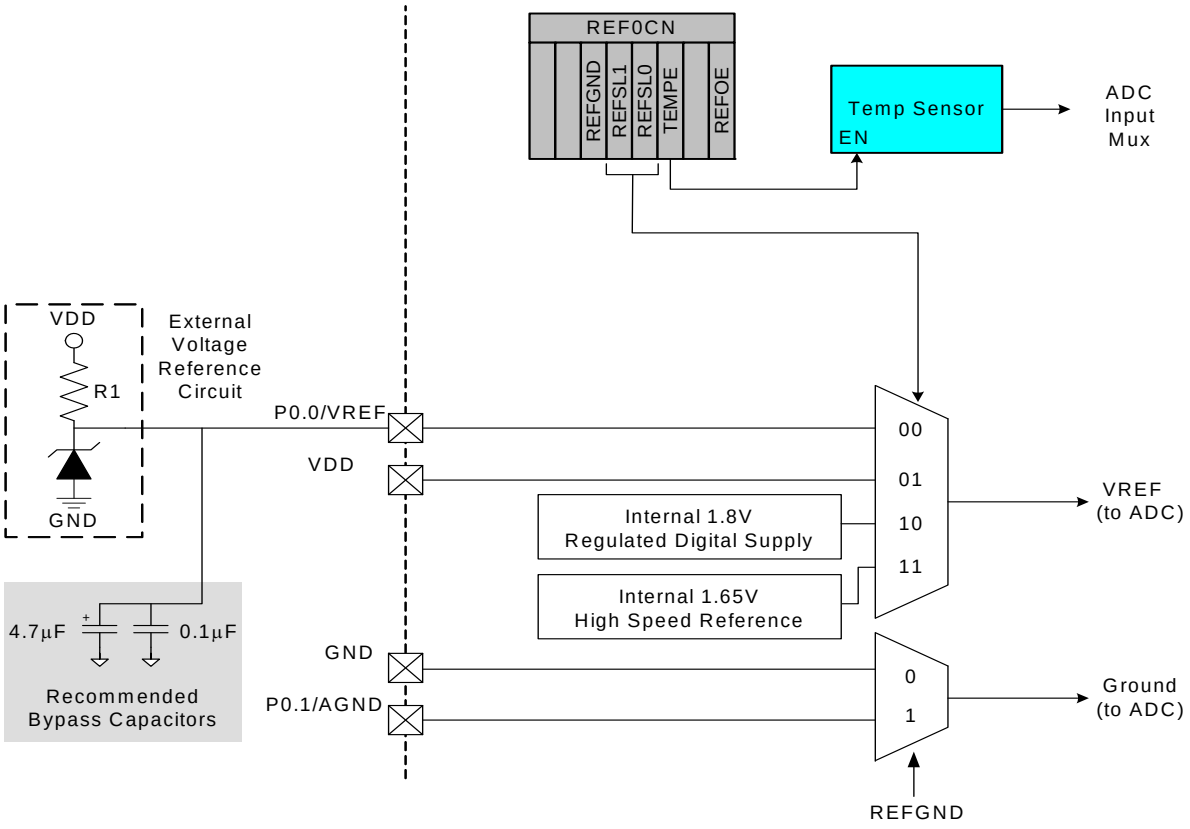



**Figure 5.10. Voltage Reference Functional Block Diagram**

## 5.10. External Voltage Reference

To use an external voltage reference, REFSL[1:0] should be set to 00. Bypass capacitors should be added as recommended by the manufacturer of the external voltage reference. If the manufacturer does not provide recommendations, a 4.7 µF in parallel with a 0.1 µF capacitor is recommended.

## 5.11. Internal Voltage Reference

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

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

## 5.12. Analog Ground Reference

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

## 5.13. Temperature Sensor Enable

The TEMPE bit in register REF0CN enables/disables the temperature sensor. While disabled, the temperature sensor defaults to a high impedance state and any ADC0 measurements performed on the sensor result in meaningless data. See Section “5.8. Temperature Sensor” on page 85 for details on temperature sensor characteristics when it is enabled.

## SFR Definition 5.15. REF0CN: Voltage Reference Control

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

SFR Page = 0x0; SFR Address = 0xD1

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

## 5.14. Voltage Reference Electrical Specifications

See Table 4.12 on page 61 for detailed Voltage Reference Electrical Specifications.

# 6. Programmable Current Reference (IREF0)

C8051F99x-C8051F98x devices include an on-chip programmable current reference (source or sink) with two output current settings: Low Power Mode and High Current Mode. The maximum current output in Low Power Mode is 63 µA (1 µA steps) and the maximum current output in High Current Mode is 504 µA (8 µA steps).

The current source/sink is controlled though the IREF0CN special function register. It is enabled by setting the desired output current to a non-zero value. It is disabled by writing 0x00 to IREF0CN. The port I/O pin associated with ISRC0 should be configured as an analog input and skipped in the Crossbar. See Section "21. Port Input/Output" on page 215 for more details.

## SFR Definition 6.1. IREF0CN: Current Reference Control

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

SFR Page = 0x0; SFR Address = 0xD6

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

## 6.1. PWM Enhanced Mode

The precision of the current reference can be increased by fine tuning the IREF0 output using a PWM signal generated by the PCA. This mode allows the IREF0DAT bits to perform a course adjustment on the IREF0 output. Any available PCA channel can perform a fine adjustment on the IREF0 output. When enabled (PWMEN = 1), the CEX signal selected using the PWMSS bit field is internally routed to IREF0 to control the on time of a current source having the weight of 2 LSBs. With the two least significant bits of IREF0DAT set to 00b, applying a 100% duty cycle on the CEX signal will be equivalent to setting the two LSBs of IREF0DAT to 10b. PWM enhanced mode is enabled and setup using the IREF0CF register.

### SFR Definition 6.2. IREF0CF: Current Reference Configuration

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

SFR Page = All; SFR Address = 0xB9

| Bit | Name | Function |
|---|---|---|
| 7 | PWMEN | **PWM Enhanced Mode Enable.**<br>Enables the PWM Enhanced Mode.<br>0: PWM Enhanced Mode disabled.<br>1: PWM Enhanced Mode enabled. |
| 6:3 | Unused | Read = 0000b, Write = don't care. |
| 2:0 | PWMSS[2:0] | **PWM Source Select.**<br>Selects the PCA channel to use for the fine-tuning control signal.<br>000: CEX0 selected as fine-tuning control signal.<br>001: CEX1 selected as fine-tuning control signal.<br>010: CEX2 selected as fine-tuning control signal.<br>All Other Values: Reserved. |

## 6.2. IREF0 Specifications

See Table 4.13 on page 62 for a detailed listing of IREF0 specifications.

# 7. Comparator

C8051F99x-C8051F98x devices include an on-chip programmable voltage comparator: Comparator 0 (CPT0) shown in Figure 7.1.

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

## 7.1. Comparator Inputs

Each Comparator performs an analog comparison of the voltage levels at its positive (CP0+) and negative (CP0-) input. The analog input multiplexers are completely under software control and configured using SFR registers. See Section “7.6. Comparator0 Analog Multiplexer” on page 98 for details on how to select and configure Comparator inputs.

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

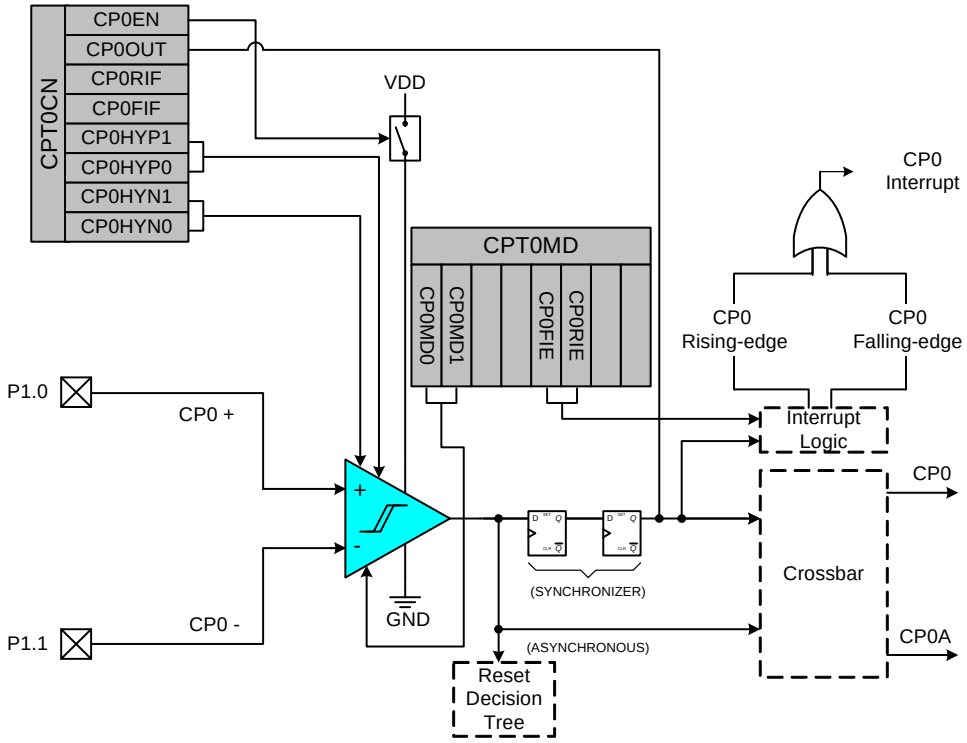



**Figure 7.1. Comparator 0 Functional Block Diagram**

## 7.2. Comparator Outputs

When a comparator is enabled, its output is a logic 1 if the voltage at the positive input is higher than the voltage at the negative input. When disabled, the comparator output is a logic 0. The comparator output is synchronized with the system clock as shown in Figure 7.2. The synchronous “latched” output (CP0) can be polled in software (CP0OUT bit), used as an interrupt source, or routed to a Port pin (configured for digital I/O) through the Crossbar.

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

When using a Comparator as an interrupt source, Comparator interrupts can be generated on rising-edge and/or falling-edge comparator output transitions. Two independent interrupt flags (CP0RIF and CP0FIF) allow software to determine which edge caused the Comparator interrupt. The comparator rising-edge and falling-edge interrupt flags are set by hardware when a corresponding edge is detected regardless of the interrupt enable state. Once set, these bits remain set until cleared by software.

The rising-edge and falling-edge interrupts can be individually enabled using the CP0RIE and CP0FIE interrupt enable bits in the CPT0MD register. In order for the CP0RIF and/or CP0FIF interrupt flags to generate an interrupt request to the CPU, the Comparator must be enabled as an interrupt source and global interrupts must be enabled. See the Interrupt Handler chapter for additional information.

## 7.3. Comparator Response Time

Comparator response time may be configured in software via the CPT0MD register described on “CPT0MD: Comparator 0 Mode Selection” on page 97. Four response time settings are available: Mode 0 (Fastest Response Time), Mode 1, Mode 2, and Mode 3 (Lowest Power). Selecting a longer response time reduces the Comparator active supply current. The Comparator also has a low power shutdown state, which is entered any time the comparator is disabled. Comparator rising edge and falling edge response times are typically not equal. See Table 4.14 on page 63 for complete comparator timing and supply current specifications.

## 7.4. Comparator Hysteresis

The Comparator features software-programmable hysteresis that can be used to stabilize the comparator output while a transition is occurring on the input. Using the CPT0CN register, the user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage (i.e., the comparator negative input).

Figure 7.2 shows that when positive hysteresis is enabled, the comparator output does not transition from logic 0 to logic 1 until the comparator positive input voltage has exceeded the threshold voltage by an amount equal to the programmed hysteresis. It also shows that when negative hysteresis is enabled, the comparator output does not transition from logic 1 to logic 0 until the comparator positive input voltage has fallen below the threshold voltage by an amount equal to the programmed hysteresis.

The amount of positive hysteresis is determined by the settings of the CP0HYP bits in the CPT0CN register and the amount of negative hysteresis voltage is determined by the settings of the CP0HYN bits in the same register. Settings of 20, 10, 5, or 0 mV can be programmed for both positive and negative hysteresis. See Section “Table 4.14. Comparator Electrical Characteristics” on page 63 for complete comparator hysteresis specifications.

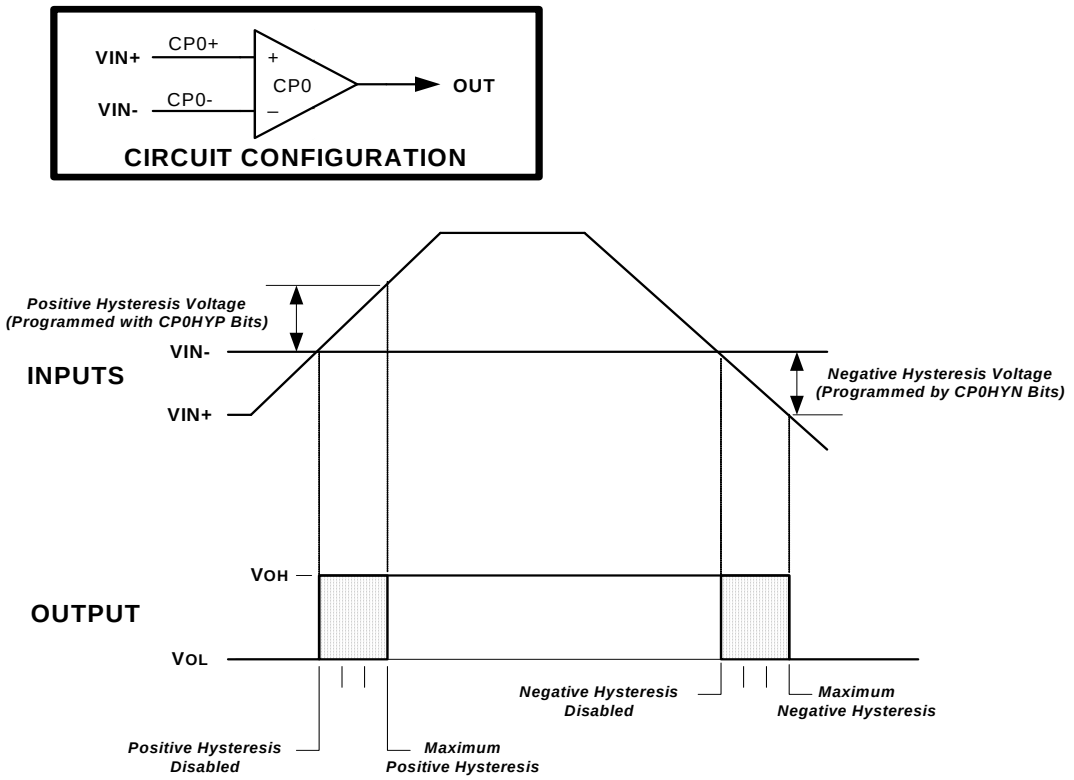



**Figure 7.2. Comparator Hysteresis Plot**

## 7.5. Comparator Register Descriptions

The SFRs used to enable and configure the comparator are described in the following register descriptions. The comparator must be enabled by setting the CP0EN bit to logic 1 before it can be used. From an enabled state, a comparator can be disabled and placed in a low power state by clearing the CP0EN bit to logic 0.

**Important Note About Comparator Settings:** False rising and falling edges can be detected by the Comparator while powering on or if changes are made to the hysteresis or response time control bits. Therefore, it is recommended that the rising-edge and falling-edge flags be explicitly cleared to logic 0 a short time after the comparator is enabled or its mode bits have been changed. The Comparator Power Up Time is specified in Section “Table 4.14. Comparator Electrical Characteristics” on page 63.

## SFR Definition 7.1. CPT0CN: Comparator 0 Control

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

SFR Page= 0x0; SFR Address = 0x9B

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

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

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

SFR Page = 0x0; SFR Address = 0x9D

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

## 7.6. Comparator0 Analog Multiplexer

Comparator0 on C8051F99x-C8051F98x devices has an analog input multiplexer to connect Port I/O pins and internal signals the comparator inputs; CP0+/CP0- are the positive and negative input multiplexers for Comparator0.

The comparator input multiplexers directly support capacitive touch switches. When the Capacitive Touch Sense Compare input is selected on the positive or negative multiplexer, any Port I/O pin connected to the other multiplexer can be directly connected to a capacitive touch switch with no additional external components. The Capacitive Touch Sense Compare provides the appropriate reference level for detecting when the capacitive touch switches have charged or discharged through the on-chip Rsense resistor. The Comparator outputs can be routed to Timer2 or Timer3 for capturing sense capacitor's charge and discharge time. See Section "25. Timers" on page 278 for details.

Any of the following may be selected as comparator inputs: Port I/O pins, Capacitive Touch Sense Compare, VDD Supply Voltage, Regulated Digital Supply Voltage (Output of VREG0) or ground. The Comparator's supply voltage divided by 2 is also available as an input; the resistors used to divide the voltage only draw current when this setting is selected. The Comparator input multiplexers are configured using the CPT0MX register described in SFR Definition 7.3.

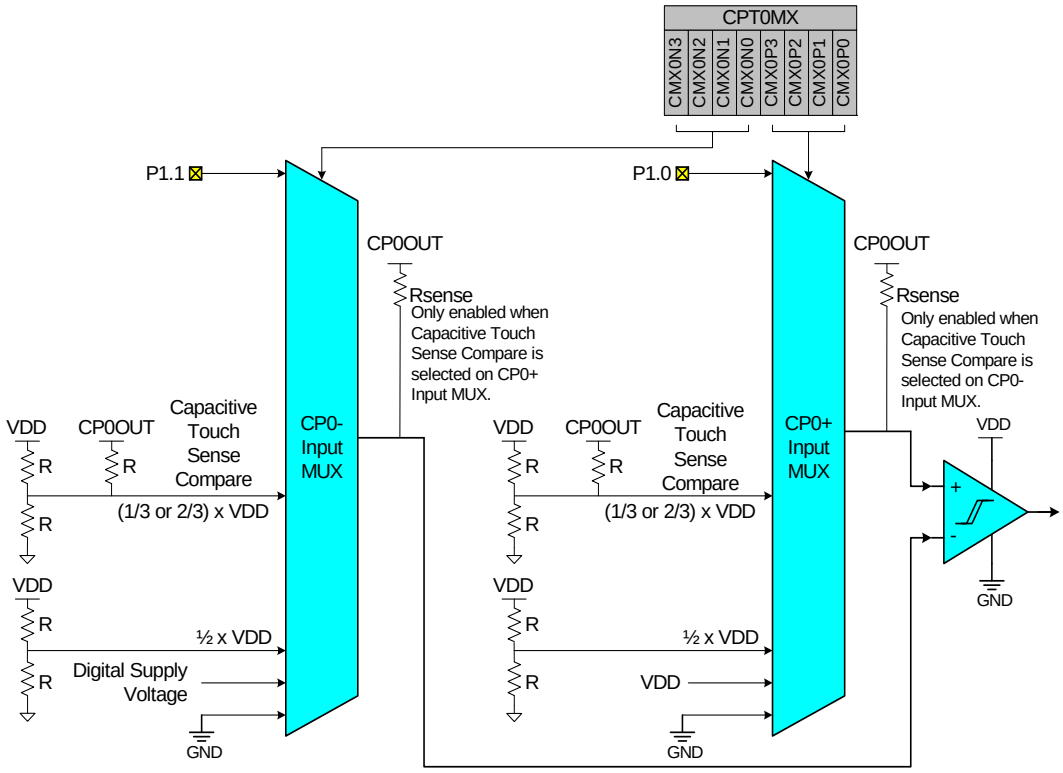



**Figure 7.3. CP0 Multiplexer Block Diagram**

**Important Note About Comparator Input Configuration:** Port pins selected as comparator inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set to 0 the corresponding bit in register PnMDIN and disable the digital driver (PnMDOUT = 0 and Port Latch = 1). To force the Crossbar to skip a Port pin, set to 1 the corresponding bit in register PnSKIP. See Section "21. Port Input/Output" on page 215 for more Port I/O configuration details.

## SFR Definition 7.3. CPT0MX: Comparator0 Input Channel Select

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

SFR Page = 0x0; SFR Address = 0x9F

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

## 8. Capacitive Sense (CS0)

The Capacitive Sense subsystem uses a capacitance-to-digital circuit to determine the capacitance on a port pin. The module can take measurements from different port pins using the module's analog multiplexer. The module is enabled only when the CS0EN bit (CS0CN) is set to 1. Otherwise the module is in a low-power shutdown state. The module can be configured to take measurements on one port pin or a group of port pins, using auto-scan. A selectable gain circuit allows the designer to adjust the maximum allowable capacitance. An accumulator is also included, which can be configured to average multiple conversions on an input channel. Interrupts can be generated when CS0 completes a conversion or when the measured value crosses a threshold defined in CS0THH:L.

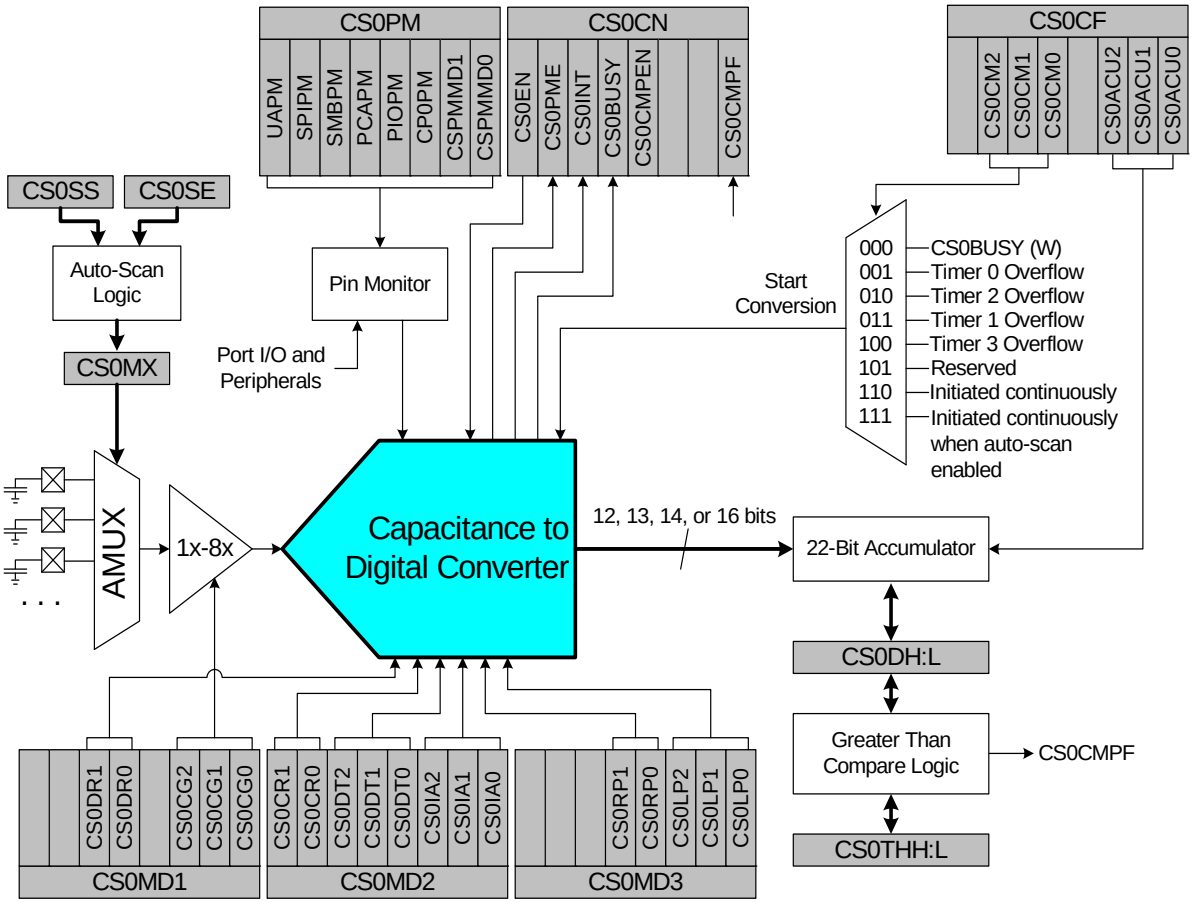



**Figure 8.1. CS0 Block Diagram**

## 8.1. Configuring Port Pins as Capacitive Sense Inputs

In order for a port pin to be measured by CS0, that port pin must be configured as an analog input (see “21. Port Input/Output” ). Configuring the input multiplexer to a port pin not configured as an analog input will cause the capacitive sense comparator to output incorrect measurements.

## 8.2. Initializing the Capacitive Sensing Peripheral

The following procedure is recommended for properly initializing the CS0 peripheral:

1. Enable the CS0 block (CS0EN = 1) before performing any other initializations.
2. Initialize the Start of Conversion Mode Select bits (CS0CM[2:0]) to the desired mode.
3. Continue initializing all remaining CS0 registers.

## 8.3. Capacitive Sense Start-Of-Conversion Sources

A capacitive sense conversion can be initiated in one of eight ways, depending on the programmed state of the CS0 start of conversion bits (CS0CF6:4). Conversions may be initiated by one of the following:

1. Writing a 1 to the CS0BUSY bit of register CS0CN
2. Timer 0 overflow
3. Timer 2 overflow
4. Timer 1 overflow
5. Timer 3 overflow
6. Convert continuously
7. Convert continuously with auto-scan enabled
8. Perform a single scan of all enabled channels

Conversions can be configured to be initiated continuously through one of two methods. CS0 can be configured to convert at a single channel continuously or it can be configured to convert continuously with auto-scan enabled. When configured to convert continuously, conversions will begin after the CS0BUSY bit in CS0CF has been set. An interrupt will be generated if CS0 conversion complete interrupts are enabled by setting the ECSCPT bit (EIE2.0).

Single Scan Mode allows all channels enabled in the CS0SCAN0 and CS0SCAN1 registers to be scanned in a single pass. An end of scan interrupt can be enabled to trigger once all selected channels have been converted. See Section “8.9. Automatic Scanning (Method 2—CS0SMEN = 1)” on page 104 for more details about this mode.

The CS0 module uses a method of successive approximation to determine the value of an external capacitance. The number of bits the CS0 module converts is adjustable using the CS0CR bits in register CS0MD2. Conversions are 13 bits long by default, but they can be adjusted to 12, 13, 14, or 16 bits depending on the needs of the application. Unconverted bits will be set to 0. Shorter conversion lengths produce faster conversion rates, and vice-versa. Applications can take advantage of faster conversion rates when the unconverted bits fall below the noise floor.

**Note:** CS0 conversion complete interrupt behavior depends on the settings of the CS0 accumulator. If CS0 is configured to accumulate multiple conversions on an input channel, a CS0 conversion complete interrupt will be generated only after the last conversion completes.

## 8.4. CS0 Multiple Channel Enable

CS0 has the capability of measuring the total capacitance of multiple channels using a single conversion. When the multiple channel feature is enabled (CS0MCEN = 1), Channels selected by CS0SCAN0/1 are internally shorted together and the combined node is selected as the CS0 input. This mode can be used to detect a capacitance change on multiple channels using a single conversion and is useful for implementing “wake-on-multiple channels”.

## 8.5. CS0 Gain Adjustment

The gain of the CS0 circuit can be adjusted in integer increments from 1x to 8x (8x is the default). High gain gives the best sensitivity and resolution for small capacitors, such as those typically implemented as touch-sensitive PCB features. To measure larger capacitance values, the gain should be lowered accordingly. The bits CS0CG[2:0] in register CS0MD set the gain value.

## 8.6. Wake from Suspend

CS0 has the capability of waking the device from a low power suspend mode upon detection of a “touch” using the digital comparator. When the CS0SMEN is set to 1, CS0 may also wake up the device after an end of scan event when CS0CM[2:0] are set to 101b or after each conversion when CS0CM[2:0] are set to 110b or 111b. If the accumulate feature is enabled, the device wakes up after all samples have been accumulated. The CS0WOI bit in the CS0MD1 register can be used to configure desire wake from suspend behavior.

## 8.7. Using CS0 in Applications that Utilize Sleep Mode

To achieve maximum power efficiency, CS0 should be enabled only when taking a conversion and disabled at all other times. CS0 must be disabled by software prior to entering Sleep Mode.

## 8.8. Automatic Scanning (Method 1—CS0SMEN = 0)

CS0 can be configured to automatically scan a sequence of contiguous CS0 input channels by configuring and enabling auto-scan. Using auto-scan with the CS0 comparator interrupt enabled allows a system to detect a change in measured capacitance without requiring any additional dedicated MCU resources.

Auto-scan is enabled by setting the CS0 start-of-conversion bits (CS0CF6:4) to 111b. After enabling auto-scan, the starting and ending channels should be set to appropriate values in CS0SS and CS0SE, respectively. Writing to CS0SS when auto-scan is enabled will cause the value written to CS0SS to be copied into CS0MX. After being enabled, writing a 1 to CS0BUSY will start auto-scan conversions. When auto-scan completes the number of conversions defined in the CS0 accumulator bits (CS0CF1:0), auto-scan configures CS0MX to the next sequential port pin configured as an analog input and begins a conversion on that channel. All other pins between CS0SS and CS0SE which are set as analog inputs are grounded during the conversion. This scan sequence continues until CS0MX reaches the ending input channel value defined in CS0SE. After one or more conversions have been taken at this channel, auto-scan configures CS0MX back to the starting input channel. For an example system configured to use auto-scan, please see Figure “8.2 Auto-Scan Example” on page 103.

**Note:** Auto-scan attempts one conversion on a CS0MX channel regardless of whether that channel’s port pin has been configured as an analog input. Auto-scan will also complete the current rotation when the device is halted for debugging.

If auto-scan is enabled when the device enters suspend mode, auto-scan will remain enabled and running. This feature allows the device to wake from suspend through CS0 greater-than comparator event on any configured capacitive sense input included in the auto-scan sequence of inputs.

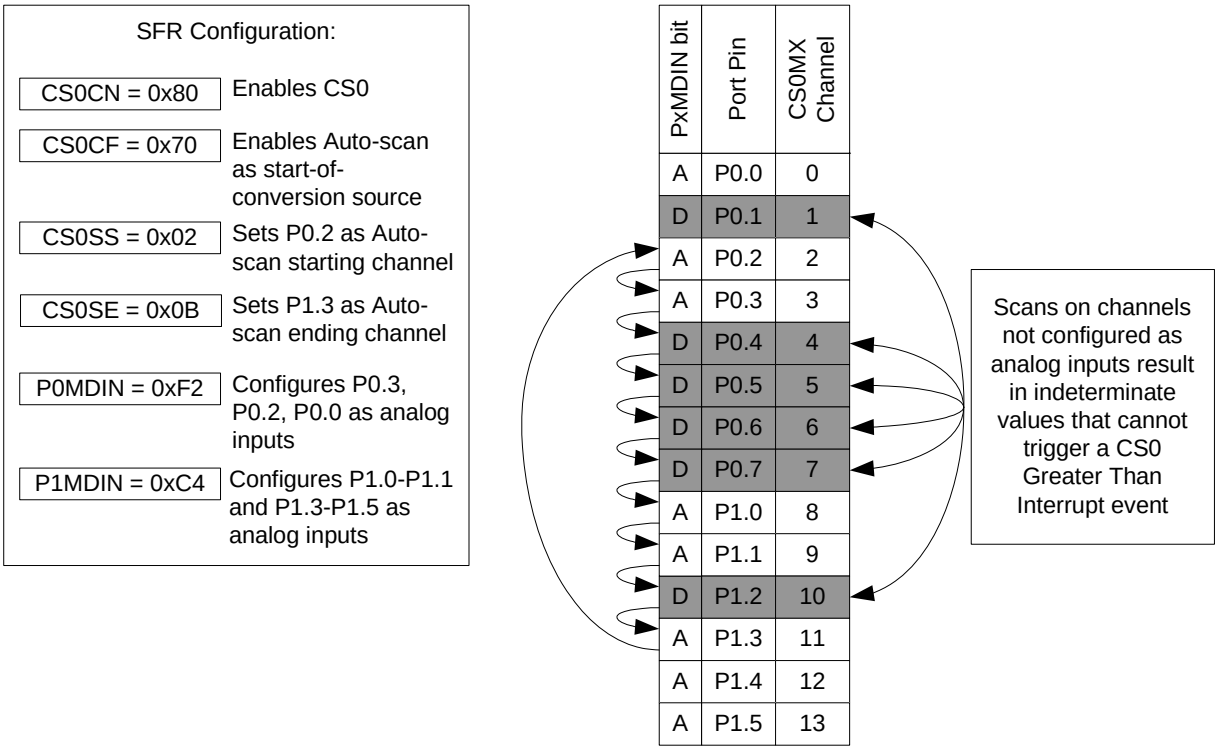



**Figure 8.2. Auto-Scan Example**

## 8.9. Automatic Scanning (Method 2—CS0SMEN = 1)

When CS0SMEN is enabled, CS0 uses an alternate autoscanning method that uses the contents of CS0SCAN0 and CS0SCAN1 to determine which channels to include in the scan. This maximizes flexibility for application development and can result in more power efficient scanning. The following procedure can be used to configure the device for Automatic Scanning with CS0SMEN = 1.

1. Set the CS0SMEN bit to 1.
2. Select the start of conversion mode (CS0CM[2:0]) if not already configured. Mode 101b is the mode of choice for most systems.
3. Configure the CS0SCAN0 and CS0SCAN1 registers to enable channels in the scan.
4. Configure the CS0THH:CS0THL digital comparator threshold and polarity.
5. Enable wake from suspend on end of scan (CS0WOI = 1) if this functionality is desired.
6. Set CS0SS to point to the first channel in the scan. Note: CS0SS uses the same bit mapping as the CS0MX register.
7. Issue a start of conversion (BUSY = 1).
8. Enable the CS0 Wakeup Source and place the device in Suspend mode (optional).

If using Mode 101b, scanning will stop once a “touch” has been detected using the digital comparator. The CS0MX register will contain the channel mux value of the channel that caused the interrupt. Setting the busy bit when servicing the interrupt will cause the scan to continue where it left off. Scanning will also stop after all channels have been sampled and no “touches” have been detected. If the CS0WOI bit is set, a wake from suspend event will be generated. Note: When automatic scanning is enabled, the contents of the CS0MX register are only valid when the digital comparator interrupt is set and BUSY = 0.

## 8.10. CS0 Comparator

The CS0 comparator compares the latest capacitive sense conversion result with the value stored in CS0THH:CS0THL. If the result is less than or equal to the stored value, the CS0CMPF bit(CS0CN:0) is set to 0. If the result is greater than the stored value, CS0CMPF is set to 1.

If the CS0 conversion accumulator is configured to accumulate multiple conversions, a comparison will not be made until the last conversion has been accumulated.

An interrupt will be generated if CS0 greater-than comparator interrupts are enabled by setting the ECSDC bit (EIE2.5) when the comparator sets CS0CMPF to 1.

If auto-scan is running when the comparator sets the CS0CMPF bit, no further auto-scan initiated conversions will start until firmware sets CS0BUSY to 1.

A CS0 greater-than comparator event can wake a device from suspend mode. This feature is useful in systems configured to continuously sample one or more capacitive sense channels. The device will remain in the low-power suspend state until the captured value of one of the scanned channels causes a CS0 greater-than comparator event to occur. It is not necessary to have CS0 comparator interrupts enabled in order to wake a device from suspend with a greater-than event.

For a summary of behavior with different CS0 comparator, auto-scan, and auto accumulator settings, please see Table 8.1.

## 8.11. CS0 Conversion Accumulator

CS0 can be configured to accumulate multiple conversions on an input channel. The number of samples to be accumulated is configured using the CS0ACU2:0 bits (CS0CF2:0). The accumulator can accumulate 1, 4, 8, 16, 32, or 64 samples. After the defined number of samples have been accumulated, the result is divided by either 1, 4, 8, 16, 32, or 64 (depending on the CS0ACU[2:0] setting) and copied to the CS0DH:CS0DL SFRs.

**Table 8.1. Operation with Auto-scan and Accumulate**

| Auto-Scan Enabled | Accumulator Enabled | CS0 Conversion Complete Interrupt Behavior | CS0 Greater Than Interrupt Behavior | CS0MX Behavior |
|---|---|---|---|---|
| **N** | **N** | CS0INT Interrupt serviced after 1 conversion completes | Interrupt serviced after 1 conversion completes if value in CS0DH:CS0DL is greater than CS0THH:CS0THL | CS0MX unchanged. |
| **N** | **Y** | CS0INT Interrupt serviced after *M* conversions complete | Interrupt serviced after *M* conversions complete if value in CS0DH:CS0DL (post accumulate and divide) is greater than CS0THH:CS0THL | CS0MX unchanged. |
| **Y** | **N** | CS0INT Interrupt serviced after 1 conversion completes | Interrupt serviced after conversion completes if value in CS0DH:CS0DL is greater than CS0THH:CS0THL; Auto-Scan stopped | If greater-than comparator detects conversion value is greater than CS0THH:CS0THL, CS0MX is left unchanged; otherwise, CS0MX updates to the next channel (CS0MX + 1) and wraps back to CS0SS after passing CS0SE. |
| **Y** | **Y** | CS0INT Interrupt serviced after *M* conversions complete | Interrupt serviced after *M* conversions complete if value in CS0DH:CS0DL (post accumulate and divide) is greater than CS0THH:CS0THL; Auto-Scan stopped | If greater-than comparator detects conversion value is greater than CS0THH:CS0THL, CS0MX is left unchanged; otherwise, CS0MX updates to the next channel (CS0MX + 1) and wraps back to CS0SS after passing CS0SE. |
| **Note:** M = Accumulator setting (1x, 4x, 8x, 16x, 32x, 64x). | | | | |

## 8.12. CS0 Pin Monitor

The CS0 module provides accurate conversions in all operating modes of the CPU, peripherals and I/O ports. Pin monitoring circuits are provided to improve interference immunity from high-current output pin switching. The CS0 Pin Monitor register (CS0PM, SFR Definition 8.14) controls the operation of these pin monitors.

Conversions in the CS0 module are immune to any change on digital inputs and immune to most output switching. Even high-speed serial data transmission will not affect CS0 operation as long as the output load is limited. Output changes that switch large loads such as LEDs and heavily-loaded communications lines can affect conversion accuracy. For this reason, the CS0 module includes pin monitoring circuits that will, if enabled, automatically adjust conversion timing if necessary to eliminate any effect from high-current output pin switching.

The pin monitor enable bit should be set for any output signal that is expected to drive a large load.

Example: The SMBus in a system is heavily loaded with multiple slaves and a long PCB route. Set the SMBus pin monitor enable, SMBPM = 1.

Example: Timer2 controls an LED on Port 1, pin 3 to provide variable dimming. Set the Port SFR write monitor enable, PIOPM = 1.

Example: The SPI bus is used to communicate to a nearby host. The pin monitor is not needed because the output is not heavily loaded, SPIPM remains = 0, the default reset state.

Pin monitors should not be enabled unless they are required. The pin monitor works by repeating any portion of a conversion that may have been corrupted by a change on an output pin. Setting pin monitor enables bits will slow CS0 conversions.

The frequency of CS0 retry operations can be limited by setting the CSPMMD bits. In the default (reset) state, all converter retry requests will be performed. This is the recommended setting for all applications. The number of retries per conversion can be limited to either two or four retries by changing CSPMMD. Limiting the number of retries per conversion ensures that even in circumstances where extremely frequent high-power output switching occurs, conversions will be completed, though there may be some loss of accuracy due to switching noise.

Activity of the pin monitor circuit can be detected by reading the Pin Monitor Event bit, CS0PME, in register CS0CN. This bit will be set if any CS0 converter retries have occurred. It remains set until cleared by software or a device reset.

## 8.13. Adjusting CS0 For Special Situations

There are several configuration options in the CS0 module designed to modify the operation of the circuit and address special situations. In particular, any circuit with more than 500 ohms of series impedance between the sensor and the device pin may require adjustments for optimal performance. Typical applications which may require adjustments include:

- Touch panel sensors fabricated using a resistive conductor such as indium-tin-oxide (ITO).
- Circuits using a high-value series resistor to isolate the sensor element for high ESD protection.

**Most systems will require no fine tuning, and the default settings for CS0DT, CS0DR, CS0IA, CS0RP and CS0LP should be used.**

## SFR Definition 8.1. CS0CN: Capacitive Sense Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0EN | CS0EOS | CS0INT | CS0BUSY | CS0CMPEN | Reserved | CS0PME | CS0CMPF |
| Type | R/W | R | R/W | R/W | R/W | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xB0

| Bit | Name | Description |
|---|---|---|
| 7 | CS0EN | **CS0 Enable.**<br>0: CS0 disabled and in low-power mode.<br>1: CS0 enabled and ready to convert. |
| 6 | CS0EOS | **CS0 End of Scan Interrupt Flag.**<br>0: CS0 has not completed a scan since the last time CS0EOS was cleared.<br>1: CS0 has completed a scan.<br>This bit is not automatically cleared by hardware. |
| 5 | CS0INT | **CS0 Interrupt Flag.**<br>0: CS0 has not completed a data conversion since the last time CS0INT was cleared.<br>1: CS0 has completed a data conversion.<br>This bit is not automatically cleared by hardware. |
| 4 | CS0BUSY | **CS0 Busy.**<br>Read:<br>0: CS0 conversion is complete or a conversion is not currently in progress.<br>1: CS0 conversion is in progress.<br>Write:<br>0: No effect.<br>1: Initiates CS0 conversion if CS0CM[2:0] = 000b, 110b, or 111b. |
| 3 | CS0CMPEN | **CS0 Digital Comparator Enable Bit.**<br>Enables the digital comparator, which compares accumulated CS0 conversion output to the value stored in CS0THH:CS0THL.<br>0: CS0 digital comparator disabled.<br>1: CS0 digital comparator enabled. |
| 2 | Reserved | Read = Varies. |
| 1 | CS0PME | **CS0 Pin Monitor Event.**<br>Set if any converter re-tries have occurred due to a pin monitor event. This bit remains set until cleared by firmware. |
| 0 | CS0CMPF | **CS0 Digital Comparator Interrupt Flag.**<br>0: CS0 result is smaller than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared.<br>1: CS0 result is greater than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared. |

## SFR Definition 8.2. CS0CF: Capacitive Sense Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0SMEN | CS0CM[2:0] | | | CS0MCEN | CS0ACU[2:0] | | |
| Type | R/W | R/W | | | R | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xAA

| Bit | Name | Description |
|---|---|---|
| 7 | CS0SMEN | **CS0 Channel Scan Masking Enable.**<br>0: The CS0SCAN0 and CS0SCAN1 register contents are ignored.<br>1: The CS0SCAN0 and CS0SCAN1 registers are used to determine which channels will be included in the scan. |
| 6:4 | CS0CM[2:0] | **CS0 Start of Conversion Mode Select.**<br>000: Conversion initiated on every write of 1 to CS0BUSY.<br>001: Conversion initiated on overflow of Timer 0.<br>010: Conversion initiated on overflow of Timer 2.<br>011: Conversion initiated on overflow of Timer 1.<br>100: Conversion initiated on overflow of Timer 3.<br>**When CS0SMEN = 0**<br>101: Reserved.<br>110: Conversion initiated continuously on the channel selected by CS0MX after writing 1 to CS0BUSY.<br>111: Conversions initiated continuously on channels from CS0SS to CS0SE after writing 1 to CS0BUSY.<br>**When CS0SMEN = 1**<br>101: Single Scan Mode, scans the channels selected by CS0SCAN0/1 once.<br>110: Conversion initiated continuously on the channel selected by CS0MX after writing 1 to CS0BUSY.<br>111: Auto Scan Mode, continuously scans the channels selected by CS0SCAN0/1. |
| 3 | CS0MCEN | **CS0 Multiple Channel Enable.**<br>0: Multiple channel feature is disabled.<br>1: Channels selected by CS0SCAN0/1 are internally shorted together and the combined node is selected as the CS0 input. This mode can be used to detect a capacitance change on multiple channels using a single conversion. |
| 2:0 | CS0ACU[2:0] | **CS0 Accumulator Mode Select.**<br>000: Accumulate 1 sample.<br>001: Accumulate 4 samples.<br>010: Accumulate 8 samples.<br>011: Accumulate 16 samples<br>100: Accumulate 32 samples.<br>101: Accumulate 64 samples.<br>11x: Reserved. |

## SFR Definition 8.3. CS0DH: Capacitive Sense Data High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0DH[7:0] | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xEE

| Bit | Name | Description |
|---|---|---|
| 7:0 | CS0DH | **CS0 Data High Byte.**<br>Stores the high byte of the last completed 16-bit Capacitive Sense conversion. |

## SFR Definition 8.4. CS0DL: Capacitive Sense Data Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0DL[7:0] | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xED

| Bit | Name | Description |
|---|---|---|
| 7:0 | CS0DL | **CS0 Data Low Byte.**<br>Stores the low byte of the last completed 16-bit Capacitive Sense conversion. |

## SFR Definition 8.5. CS0SCAN0: Capacitive Sense Channel Scan Mask 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0SCAN0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD2

| Bit | Name | Description |
|---|---|---|
| 7:0 | CS0SCAN[7:0] | **Capacitive Sense Channel Scan Mask for Port 0.**<br>The selected channels are included in the Auto Scan when scan masking is enabled (CS0SMEN = 1). |

## SFR Definition 8.6. CS0SCAN1: Capacitive Sense Channel Scan Mask 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | CS0SCAN[5:0] | | | | | |
| Type | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD3

| Bit | Name | Description |
|---|---|---|
| 7:6 | Unused | Read = 00b; Write = Don't care |
| 5:0 | CS0SCAN[5:0] | **Capacitive Sense Channel Scan Mask for Port 0.**<br>The selected channels are included in the Auto Scan when scan masking is enabled (CS0SMEN = 1). |

## SFR Definition 8.7. CS0SS: Capacitive Sense Auto-Scan Start Channel

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | CS0SS[4:0] | | | | |
| Type | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xDD

| Bit | Name | Description |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't care |
| 4:0 | CS0SS[4:0] | **Starting Channel for Auto-Scan.**<br>Sets the first CS0 channel to be selected by the mux for Capacitive Sense conversion when auto-scan is enabled and active. All channels detailed in CS0MX SFR Definition 8.15 are possible choices for this register.<br><br>When auto-scan is enabled, a write to CS0SS will also update CS0MX. |

## SFR Definition 8.8. CS0SE: Capacitive Sense Auto-Scan End Channel

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | CS0SE[4:0] | | | | |
| Type | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xDE

| Bit | Name | Description |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't care |
| 4:0 | CS0SE[4:0] | **Ending Channel for Auto-Scan.**<br>Sets the last CS0 channel to be selected by the mux for Capacitive Sense conversion when auto-scan is enabled and active. All channels detailed in CS0MX SFR Definition 8.15 are possible choices for this register. |

## SFR Definition 8.9. CS0THH: Capacitive Sense Comparator Threshold High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0THH[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFE

| Bit | Name | Description |
|---|---|---|
| 7:0 | CS0THH[7:0] | **CS0 Comparator Threshold High Byte.**<br>High byte of the 16-bit value compared to the Capacitive Sense conversion result. |

## SFR Definition 8.10. CS0THL: Capacitive Sense Comparator Threshold Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0THL[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFD

| Bit | Name | Description |
|---|---|---|
| 7:0 | CS0THL[7:0] | **CS0 Comparator Threshold Low Byte.**<br>Low byte of the 16-bit value compared to the Capacitive Sense conversion result. |

## SFR Definition 8.11. CS0MD1: Capacitive Sense Mode 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | CS0POL | CS0DR | | CS0WOI | CS0CG[2:0] | | |
| Type | R/W | | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xAF

| Bit | Name | Description |
|---|---|---|
| 7 | Reserved | Must write 0. |
| 6 | CS0POL | **CS0 Digital Comparator Polarity Select.**<br>0: The digital comparator generates an interrupt if the conversion is greater than the threshold.<br>1: The digital comparator generates an interrupt if the conversion is less than or equal to the threshold. |
| 5:4 | CS0DR[1:0] | **CS0 Double Reset Select.**<br>These bits adjust the secondary CS0 reset time. For most touch-sensitive switches, the default (fastest) value is sufficient. See the discussion in Section 8.13 for more information.<br>00: No additional time is used for secondary reset.<br>01: An additional 0.75 µs is used for secondary reset.<br>10: An additional 1.5 µs is used for secondary reset.<br>11: An additional 2.25 µs is used for secondary reset. |
| 3 | CS0WOI | **CS0 Wake on Interrupt Configuration.**<br>0: Wake-up event generated on digital comparator interrupt only.<br>1: Wake-up event generated on end of scan or digital comparator interrupt. |
| 2:0 | CS0CG[2:0] | **CS0 Capacitance Gain Select.**<br>These bits select the gain applied to the capacitance measurement. Lower gain values increase the size of the capacitance that can be measured with the CS0 module. The capacitance gain is equivalent to CS0CG[2:0] + 1.<br>000: Gain = 1x<br>001: Gain = 2x<br>010: Gain = 3x<br>011: Gain = 4x<br>100: Gain = 5x<br>101: Gain = 6x<br>110: Gain = 7x<br>111: Gain = 8x (default) |

## SFR Definition 8.12. CS0MD2: Capacitive Sense Mode 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CS0CR[1:0] | | CS0DT[2:0] | | | CS0IA[2:0] | | |
| Type | R/W | | R/W | | | R/W | | |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xF3

| Bit | Name | Description |
|---|---|---|
| 7:6 | CS0CR[1:0] | **CS0 Conversion Rate.**<br>These bits control the conversion rate of the CS0 module. See the electrical specifications table for specific timing.<br>00: Conversions last 12 internal CS0 clocks and are 12 bits in length.<br>01: Conversions last 13 internal CS0 clocks and are 13 bits in length.<br>10: Conversions last 14 internal CS0 clocks and are 14 bits in length.<br>11: Conversions last 16 internal CS0 clocks.and are 16 bits in length. |
| 5:3 | CS0DT[2:0] | **CS0 Discharge Time.**<br>These bits adjust the primary CS0 reset time. For most touch-sensitive switches, the default (fastest) value is sufficient. See the discussion in Section 8.13 for more information.<br>000: Discharge time is 0.75 µs (recommended for most switches)<br>001: Discharge time is 1.0 µs<br>010: Discharge time is 1.2 µs<br>011: Discharge time is 1.5 µs<br>100: Discharge time is 2 µs<br>101: Discharge time is 3 µs<br>110: Discharge time is 6 µs<br>111: Discharge time is 12 µs |
| 2:0 | CS0IA[2:0] | **CS0 Output Current Adjustment.**<br>These bits allow the user to adjust the output current used to charge up the capacitive sensor element. For most touch-sensitive switches, the default (highest) current is sufficient. See the discussion in Section 8.13 for more information.<br>000: Full Current<br>001: 1/8 Current<br>010: 1/4 Current<br>011: 3/8 Current<br>100: 1/2 Current<br>101: 5/8 Current<br>110: 3/4 Current<br>111: 7/8 Current |

## SFR Definition 8.13. CS0MD3: Capacitive Sense Mode 3

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | CS0RP[1:0] | | CS0LP[2:0] | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0xF3

| Bit | Name | Description |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't care |
| 4:3 | CS0RP[1:0] | **CS0 Ramp Selection.**<br>These bits are used to compensate CS0 conversions for circuits requiring slower ramp times. For most touch-sensitive switches, the default (fastest) value is sufficient. See the discussion in Section 8.13 for more information.<br>00: Ramp time is less than 1.5 µs.<br>01: Ramp time is between 1.5 µs and 3 µs.<br>10: Ramp time is between 3 µs and 6 µs.<br>11: Ramp time is greater than 6 µs. |
| 2:0 | CS0LP[2:0] | **CS0 Low Pass Filter Selection.**<br>These bits set the internal corner frequency of the CS0 low-pass filter. Higher values of CS0LP result in a lower internal corner frequency.<br><br>For most touch-sensitive switches, the default setting of 000b should be used. If the CS0RP bits are adjusted from their default value, the CS0LP bits should normally be set to 001b. Settings higher than 001b will result in attenuated readings from the CS0 module and should be used only under special circumstances. See the discussion in Section 8.13 for more information. |

## SFR Definition 8.14. CS0PM: Capacitive Sense Pin Monitor

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | UAPM | SPIPM | SMBPM | PCAPM | PIOPM | CP0PM | CSPMMD[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0xDE;

| Bit | Name | Description |
|---|---|---|
| 7 | UAPM | **UART Pin Monitor Enable.**<br>Enables monitoring of the UART TX pin. |
| 6 | SPIPM | **SPI Pin Monitor Enable.**<br>Enables monitoring SPI output pins. |
| 5 | SMBPM | **SMBus Pin Monitor Enable.**<br>Enables monitoring of the SMBus pins. |
| 4 | PCAPM | **PCA Pin Monitor Enable.**<br>Enables monitoring of PCA output pins. |
| 3 | PIOPM | **Port I/O Pin Monitor Enable.**<br>Enables monitoring of writes to the port latch registers. |
| 2 | CP0PM | **CP0 Pin Monitor Enable.**<br>Enables monitoring of the comparator CP0 output. |
| 1:0 | CSPMMD[1:0] | **CS0 Pin Monitor Mode.**<br>Selects the operation to take when a monitored signal changes state.<br>00: Always retry bit cycles on a pin state change.<br>01: Retry up to twice on consecutive bit cycles.<br>10: Retry up to four times on consecutive bit cycles.<br>11: Reserved. |

## 8.14. Capacitive Sense Multiplexer

The input multiplexer can be controlled through two methods. The CS0MX register can be written to through firmware, or the register can be configured automatically using the modules auto-scan functionality (see “8.8. Automatic Scanning (Method 1—CS0SMEN = 0)” ).

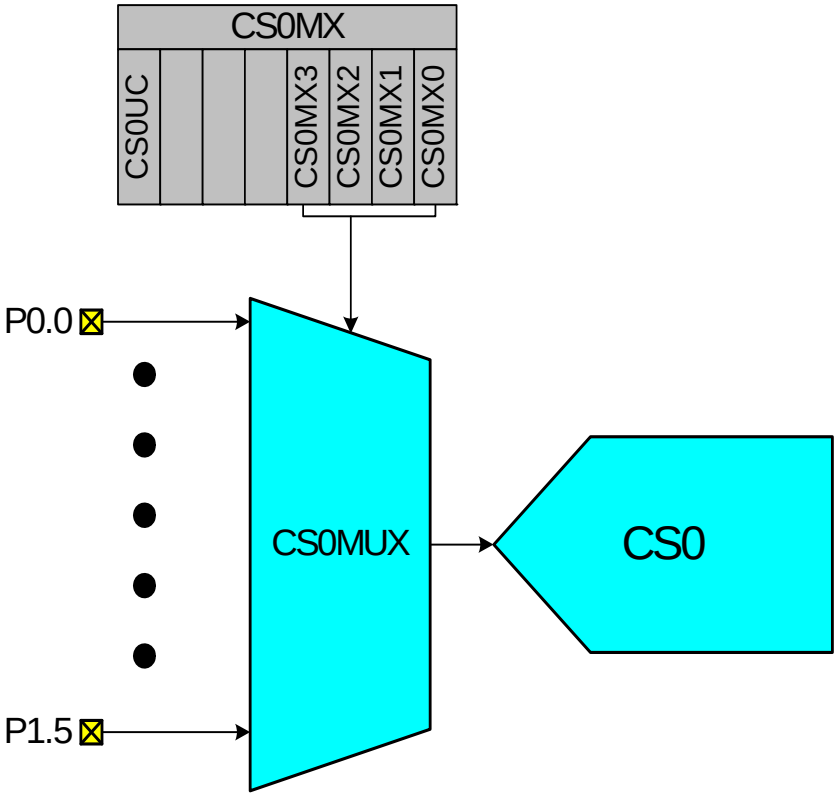



**Figure 8.3. CS0 Multiplexer Block Diagram**

## SFR Definition 8.15. CS0MX: Capacitive Sense Mux Channel Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | Reserved | Reserved | Reserved | CS0MX[3:0] | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xAB

| Bit | Name | Description | |
|---|---|---|---|
| 7:5 | Reserved | Read = 0000b; Write = 0000b. | |
| 4:0 | CS0MX[4:0] | **CS0 Mux Channel Select.**<br>Selects one of the 14 input channels for Capacitive Sense conversion. | |
| | | **Value** | **Channel** |
| | | **0000** | P0.0 |
| | | **0001** | P0.1 |
| | | **0010** | P0.2 |
| | | **0011** | P0.3 |
| | | **0100** | P0.4 |
| | | **0101** | P0.5 |
| | | **0110** | P0.6 |
| | | **0111** | P0.7 |
| | | **1000** | P1.0 |
| | | **1001** | P1.1 |
| | | **1010** | P1.2 |
| | | **1011** | P1.3 |
| | | **1100** | P1.4 (24-pin packages only) |
| | | **1101** | P1.5 |
| | | **1110** | Reserved |
| | | **1111** | Reserved |

# 9. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 27) and interfaces directly with the analog and digital subsystems providing a complete data acquisition or control-system solution in a single integrated circuit.

The CIP-51 Microcontroller core implements the standard 8051 organization and peripherals as well as additional custom peripherals and functions to extend its capability (see Figure 9.1 for a block diagram). The CIP-51 includes the following features:

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

## 9.1. Performance

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

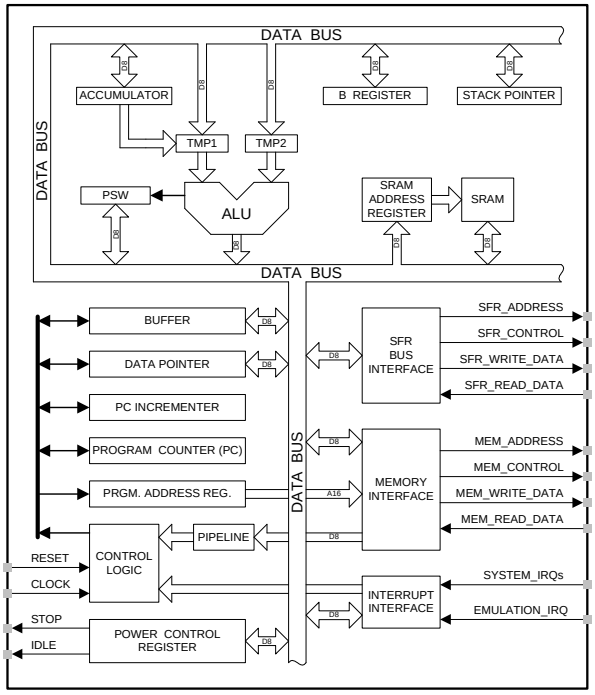



**Figure 9.1. CIP-51 Block Diagram**

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

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

## 9.2. Programming and Debugging Support

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

The on-chip debug support logic facilitates full speed in-circuit debugging, allowing the setting of hardware breakpoints, starting, stopping and single stepping through program execution (including interrupt service routines), examination of the program's call stack, and reading/writing the contents of registers and memory. This method of on-chip debugging is completely non-intrusive, requiring no RAM, Stack, timers, or other on-chip resources. C2 details can be found in Section “27. C2 Interface” on page 319.

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

## 9.3. Instruction Set

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

### 9.3.1. Instruction and CPU Timing

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

Due to the pipelined architecture of the CIP-51, most instructions execute in the same number of clock cycles as there are program bytes in the instruction. Conditional branch instructions take one less clock cycle to complete when the branch is not taken as opposed to when the branch is taken. Table 9.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

**Table 9.1. CIP-51 Instruction Set Summary**

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

**Table 9.1. CIP-51 Instruction Set Summary (Continued)**

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

**Table 9.1. CIP-51 Instruction Set Summary (Continued)**

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

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

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

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

**rel**—8-bit, signed (twos 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 2 kB 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 8 kB program memory space.

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

## 9.4. CIP-51 Register Descriptions

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

### SFR Definition 9.1. DPL: Data Pointer Low Byte

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

SFR Page = All; SFR Address = 0x82

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

### SFR Definition 9.2. DPH: Data Pointer High Byte

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

SFR Page = All; SFR Address = 0x83

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

## SFR Definition 9.3. SP: Stack Pointer

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

SFR Page = All; SFR Address = 0x81

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

## SFR Definition 9.4. ACC: Accumulator

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

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

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

## SFR Definition 9.5. B: B Register

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

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

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

## SFR Definition 9.6. PSW: Program Status Word

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

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

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

## 10. Memory Organization

The memory organization of the CIP-51 System Controller is similar to that of a standard 8051. There are two separate memory spaces: program memory and data memory. Program and data memory share the same address space but are accessed via different instruction types. The memory organization of the C8051F99x-C8051F98x device family is shown in Figure 10.1.

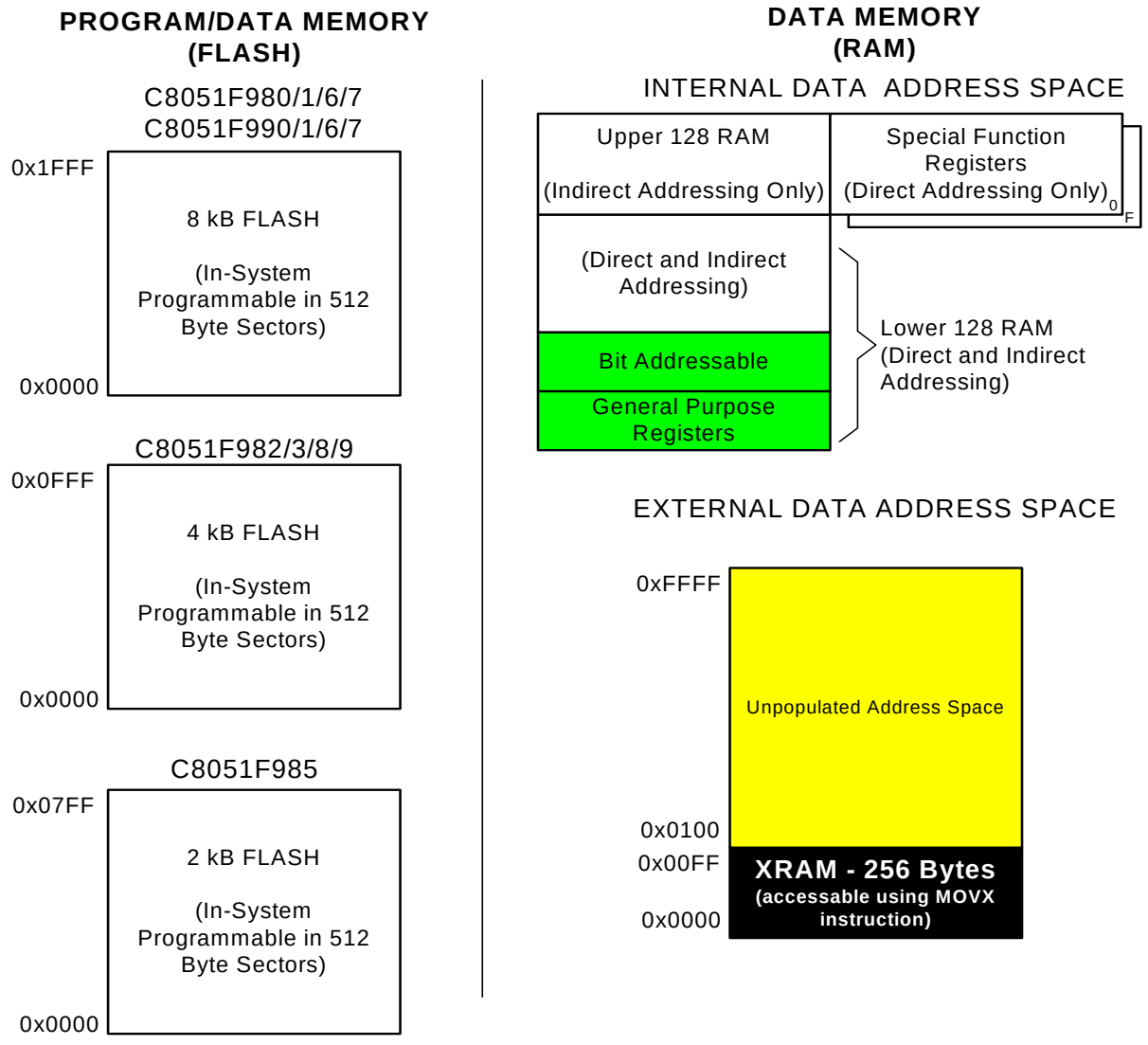



**Figure 10.1. C8051F99x-C8051F98x Memory Map**

## 10.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051F99x-C8051F98x devices implement 8 kB (C8051F980/1/6/7, C8051F990/1/6/7), 4 kB (C8051F982/3/8/9), or 2 kB (C8051F985) of this program memory space as in-system, re-programmable Flash memory, organized in a contiguous block from addresses 0x0000 to 0x1FFF (8 kB devices), 0x0FFF (4 kB devices), or 0x07FF (2 kB devices). The last byte of this contiguous block of addresses serves as the security lock byte for the device. Any addresses above the lock byte are reserved.

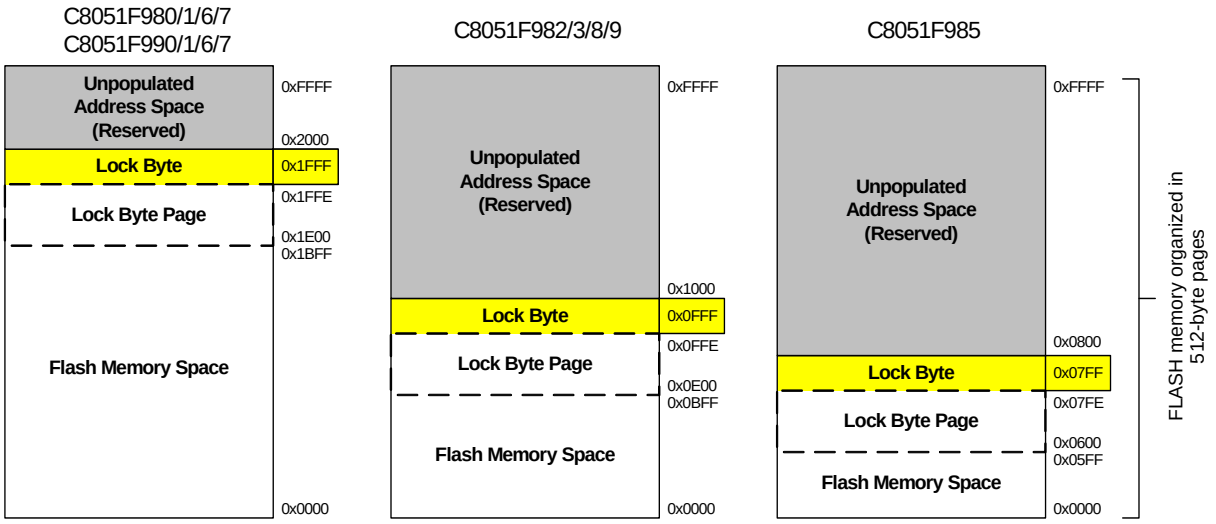



**Figure 10.2. Flash Program Memory Map**

### 10.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F99x-C8051F98x devices, the MOVX instruction is normally used to read and write on-chip XRAM, but can be re-configured to write and erase on-chip Flash memory space. MOVC instructions are always used to read Flash memory, while MOVX write instructions are used to erase and write Flash. This Flash access feature provides a mechanism for the C8051F99x-C8051F98x to update program code and use the program memory space for non-volatile data storage. Refer to Section “14. Flash Memory” on page 150 for further details.

## 10.2. Data Memory

The C8051F99x-C8051F98x device family include 512 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. The remainder of this memory is on-chip “external” memory. The data memory map is shown in Figure 10.1 for reference.

### 10.2.1. Internal RAM

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

The upper 128 bytes of data memory are accessible only by indirect addressing. This region occupies the same address space as the Special Function Registers (SFR) but is physically separate from the SFR

space. The addressing mode used by an instruction when accessing locations above 0x7F determines whether the CPU accesses the upper 128 bytes of data memory space or the SFRs. Instructions that use direct addressing will access the SFR space. Instructions using indirect addressing above 0x7F access the upper 128 bytes of data memory. Figure 10.1 illustrates the data memory organization of the C8051F99x-C8051F98x.

#### 10.2.1.1.General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 9.6). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

#### 10.2.1.2.Bit Addressable Locations

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

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

```
MOV    C, 22.3h
```

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

#### 10.2.1.3.Stack

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

### 10.2.2. External RAM

There are 256 bytes of on-chip RAM mapped into the external data memory space. All of these address locations may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using MOVX indirect addressing mode (such as @R1).

# 11. On-Chip XRAM

The C8051F99x-C8051F98x MCUs include on-chip RAM mapped into the external data memory space (XRAM). The external memory space may be accessed using the external move instruction (MOVX) with the target address specified in either the data pointer (DPTR), or with the target address low byte in R0 or R1. On C8051F99x-C8051F98x devices, the target address high byte is a don't care.

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

**Important Note**: MOVX write operations can be configured to target Flash memory, instead of XRAM. See Section "14. Flash Memory" on page 150 for more details. The MOVX instruction accesses XRAM by default.

**Important Note**: On device reset or upon waking up from Sleep mode, address 0x0000 of external memory may be overwritten by an indeterminate value. The indeterminate value is 0x00 in most situations. A dummy variable should be placed at address 0x0000 in external memory to ensure that the application firmware does not store any data that needs to be retained during sleep or reset at this memory location.

## 11.1. Accessing XRAM

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

### 11.1.1. 16-Bit MOVX Example

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

```
MOV    DPTR, #0034h        ; load DPTR with 16-bit address to read (0x0034)
MOVX   A, @DPTR            ; load contents of 0x0034 into accumulator A
```

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

### 11.1.2. 8-Bit MOVX Example

The 8-bit form of the MOVX instruction uses the contents of R0 or R1 to determine the 8-bits of the effective address to be accessed. The following series of instructions read the contents of the byte at address 0x0034 into the accumulator A.

```
MOV    R0, #34h            ; load low byte of address into R0 (or R1)
MOVX   a, @R0              ; load contents of 0x0034 into accumulator A
```

## 12. 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 C8051F99x-C8051F98x's resources and peripherals. The CIP-51 controller core duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the C8051F99x-C8051F98x. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 12.1 and Table 12.2 list the SFRs implemented in the C8051F99x-C8051F98x device family.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g., P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the data sheet, as indicated in Table 12.3, for a detailed description of each register.

**Table 12.1. Special Function Register (SFR) Memory Map (Page 0x0)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | SPI0CN | PCA0L | PCA0H | PCA0CPL0 | PCA0CPH0 | CS0THL | CS0THH | VDM0CN |
| F0 | B | P0MDIN | P1MDIN | CS0MD2 | SMB0ADR | SMB0ADM | EIP1 | EIP2 |
| E8 | ADC0CN | PCA0CPL1 | PCA0CPH1 | PCA0CPL2 | PCA0CPH2 | CS0DL | CS0DH | RSTSRC |
| E0 | ACC | XBR0 | XBR1 | XBR2 | IT01CF | FLWR | EIE1 | EIE2 |
| D8 | PCA0CN | PCA0MD | PCA0CPM0 | PCA0CPM1 | PCA0CPM2 | CS0SS | CS0SE | PCA0PWM |
| D0 | PSW | REF0CN | CS0SCAN0 | CS0SCAN1 | P0SKIP | P1SKIP | IREF0CN | P0MAT |
| C8 | TMR2CN | REG0CN | TMR2RLL | TMR2RLH | TMR2L | TMR2H | PMU0FL | P1MAT |
| C0 | SMB0CN | SMB0CF | SMB0DAT | ADC0GTL | ADC0GTH | ADC0LTL | ADC0LTH | P0MASK |
| B8 | IP | IREF0CN | ADC0AC | ADC0PWR | ADC0TK | ADC0L | ADC0H | P1MASK |
| B0 | CS0CN | OSCXCN | OSCICN | OSCICL | | PMU0CF | FLSCL | FLKEY |
| A8 | IE | CLKSEL | CS0CF | CS0MX | RTC0ADR | RTC0DAT | RTC0KEY | CS0MD1 |
| A0 | P2 | SPI0CFG | SPI0CKR | SPI0DAT | P0MDOUT | P1MDOUT | P2MDOUT | SFRPAGE |
| 98 | SCON0 | SBUF0 | CRC0CNT | CPT0CN | CRC0FLIP | CPT0MD | CRC0AUTO | CPT0MX |
| 90 | P1 | TMR3CN | TMR3RLL | TMR3RLH | TMR3L | TMR3H | ADC0MX | ADC0CF |
| 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)

## 12.1. SFR Paging

To accommodate more than 128 SFRs in the 0x80 to 0xFF address space, SFR paging has been implemented. By default, all SFR accesses target SFR Page 0x0 to allow access to the registers listed in Table 12.1. During device initialization, some SFRs located on SFR Page 0xF may need to be accessed. Table 12.2 lists the SFRs accessible from SFR Page 0x0F. Some SFRs are accessible from both pages, including the SFRPAGE register. SFRs only accessible from Page 0xF are in **bold**.

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

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

**Table 12.2. Special Function Register (SFR) Memory Map (Page 0xF)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | | | | | | | | |
| F0 | B | | | **CS0MD3** | | | EIP1 | EIP2 |
| E8 | | | | | | | | |
| E0 | ACC | | **REVID** | **DEVICEID** | | FLWR | EIE1 | EIE2 |
| D8 | | | | | | | **CS0PM** | |
| D0 | PSW | | | | | | | |
| C8 | | | | | | | | |
| C0 | | | | | | | | |
| B8 | | **IREF0CF** | | ADC0PWR | ADC0TK | | | |
| B0 | | | | | | **PMU0MD** | | |
| A8 | IE | CLKSEL | | | | | | |
| A0 | P2 | | | | | | | SFRPAGE |
| 98 | | **P0DRV** | CRC0CNT | **P1DRV** | CRC0FLIP | **P2DRV** | CRC0AUTO | |
| 90 | P1 | | | | | | | |
| 88 | | | | | | **TOFFL** | **TOFFH** | |
| 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)

## SFR Definition 12.1. SFR Page: SFR Page

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

SFR Page = All; SFR Address = 0xA7

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

### Table 12.3. Special Function Registers

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

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| ACC | 0xE0 | All | Accumulator | 126 |
| ADC0AC | 0xBA | 0x0 | ADC0 Accumulator Configuration | 76 |
| ADC0CF | 0x97 | 0x0 | ADC0 Configuration | 75 |
| ADC0CN | 0xE8 | 0x0 | ADC0 Control | 74 |
| ADC0GTH | 0xC4 | 0x0 | ADC0 Greater-Than Compare High | 80 |
| ADC0GTL | 0xC3 | 0x0 | ADC0 Greater-Than Compare Low | 80 |
| ADC0H | 0xBE | 0x0 | ADC0 High | 79 |
| ADC0L | 0xBD | 0x0 | ADC0 Low | 79 |
| ADC0LTH | 0xC6 | 0x0 | ADC0 Less-Than Compare Word High | 81 |
| ADC0LTL | 0xC5 | 0x0 | ADC0 Less-Than Compare Word Low | 81 |
| ADC0MX | 0x96 | 0x0 | AMUX0 Channel Select | 84 |
| ADC0PWR | 0xBB | All | ADC0 Burst Mode Power-Up Time | 77 |
| ADC0TK | 0xBC | All | ADC0 Tracking Control | 78 |
| B | 0xF0 | All | B Register | 126 |
| CKCON | 0x8E | 0x0 | Clock Control | 279 |
| CLKSEL | 0xA9 | All | Clock Select | 193 |
| CPT0CN | 0x9B | 0x0 | Comparator0 Control | 96 |
| CPT0MD | 0x9D | 0x0 | Comparator0 Mode Selection | 97 |
| CPT0MX | 0x9F | 0x0 | Comparator0 Mux Selection | 99 |
| CRC0AUTO | 0x9E | All | CRC0 Automatic Control | 177 |
| CRC0CN | 0x84 | All | CRC0 Control | 175 |
| CRC0CNT | 0x9A | All | CRC0 Automatic Flash Sector Count | 178 |
| CRC0DAT | 0x86 | All | CRC0 Data | 176 |
| CRC0FLIP | 0x9C | All | CRC0 Flip | 179 |
| CRC0IN | 0x85 | All | CRC0 Input | 176 |
| CS0CF | 0xAA | 0x0 | CS0 Configuration | 108 |
| CS0CN | 0xB0 | 0x0 | CS0 Control | 107 |

**Table 12.3. Special Function Registers (Continued)**

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

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| CS0DH | 0xEE | 0x0 | CS0 Data High Byte | 109 |
| CS0DL | 0xED | 0x0 | CS0 Data Low Byte | 109 |
| CS0MD1 | 0xAF | 0x0 | CS0 Mode1 | 113 |
| CS0MD2 | 0xF3 | 0x0 | CS0 Mode 2 | 114 |
| CS0MD3 | 0xF3 | 0xF | CS0 Mode 3 | 115 |
| CS0MX | 0xAB | 0x0 | CS0 Mux Channel Select | 118 |
| CS0PM | 0xDE | 0xF | CS0 Power Management | 116 |
| CS0SCAN0 | 0xD2 | 0x0 | CS0 Scan Channel Enable 0 | 110 |
| CS0SCAN1 | 0xD3 | 0x0 | CS0 Scan Channel Enable 1 | 110 |
| CS0SE | 0xDE | 0x0 | CS0 Auto-Scan End Channel | 111 |
| CS0SS | 0xDD | 0x0 | CS0 Auto-Scan Start Channel | 111 |
| CS0THH | 0xFE | 0x0 | CS0 Comparator Threshold High Byte | 112 |
| CS0THL | 0xFD | 0x0 | CS0 Comparator Threshold Low Byte | 112 |
| DEVICEID | 0xE3 | 0xF | Device ID | 154 |
| DPH | 0x83 | All | Data Pointer High | 125 |
| DPL | 0x82 | All | Data Pointer Low | 125 |
| EIE1 | 0xE6 | All | Extended Interrupt Enable 1 | 144 |
| EIE2 | 0xE7 | All | Extended Interrupt Enable 2 | 146 |
| EIP1 | 0xF6 | All | Extended Interrupt Priority 1 | 145 |
| EIP2 | 0xF7 | All | Extended Interrupt Priority 2 | 147 |
| FLKEY | 0xB7 | All | Flash Lock And Key | 160 |
| FLSCL | 0xB6 | 0x0 | Flash Scale Register | 161 |
| FLWR | 0xE5 | All | Flash Write Only Register | 161 |
| IE | 0xA8 | All | Interrupt Enable | 142 |
| IP | 0xB8 | All | Interrupt Priority | 143 |
| IREF0CF | 0xB9 | All | Current Reference IREF0 Configuration | 92 |
| IREF0CN | 0xD6 | 0x0 | Current Reference IREF0 Control | 91 |
| IT01CF | 0xE4 | 0x0 | INT0/INT1 Configuration | 149 |
| OSCICL | 0xB3 | 0x0 | Internal Oscillator Calibration | 195 |
| OSCICN | 0xB2 | 0x0 | Internal Oscillator Control | 194 |
| OSCXCN | 0xB1 | 0x0 | External Oscillator Control | 196 |
| P0 | 0x80 | All | Port 0 Latch | 228 |
| P0DRV | 0x99 | 0xF | Port 0 Drive Strength | 230 |
| P0MASK | 0xC7 | 0x0 | Port 0 Mask | 225 |
| P0MAT | 0xD7 | 0x0 | Port 0 Match | 225 |
| P0MDIN | 0xF1 | 0x0 | Port 0 Input Mode Configuration | 229 |
| P0MDOUT | 0xA4 | 0x0 | Port 0 Output Mode Configuration | 229 |
| P0SKIP | 0xD4 | 0x0 | Port 0 Skip | 228 |
| P1 | 0x90 | All | Port 1 Latch | 231 |
| P1DRV | 0x9B | 0xF | Port 1 Drive Strength | 233 |
| P1MASK | 0xBF | 0x0 | Port 1 Mask | 226 |
| P1MAT | 0xCF | 0x0 | Port 1 Match | 226 |
| P1MDIN | 0xF2 | 0x0 | Port 1 Input Mode Configuration | 232 |
| P1MDOUT | 0xA5 | 0x0 | Port 1 Output Mode Configuration | 228 |

**Table 12.3. Special Function Registers (Continued)**

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

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| P1SKIP | 0xD5 | 0x0 | Port 1 Skip | 231 |
| P2 | 0xA0 | All | Port 2 Latch | 233 |
| P2DRV | 0x9D | 0xF | Port 2 Drive Strength | 234 |
| P2MDOUT | 0xA6 | 0x0 | Port 2 Output Mode Configuration | 234 |
| PCA0CN | 0xD8 | 0x0 | PCA0 Control | 313 |
| PCA0CPH0 | 0xFC | 0x0 | PCA0 Capture 0 High | 318 |
| PCA0CPH1 | 0xEA | 0x0 | PCA0 Capture 1 High | 318 |
| PCA0CPH2 | 0xEC | 0x0 | PCA0 Capture 2 High | 318 |
| PCA0CPL0 | 0xFB | 0x0 | PCA0 Capture 0 Low | 318 |
| PCA0CPL1 | 0xE9 | 0x0 | PCA0 Capture 1 Low | 318 |
| PCA0CPL2 | 0xEB | 0x0 | PCA0 Capture 2 Low | 318 |
| PCA0CPM0 | 0xDA | 0x0 | PCA0 Module 0 Mode Register | 316 |
| PCA0CPM1 | 0xDB | 0x0 | PCA0 Module 1 Mode Register | 316 |
| PCA0CPM2 | 0xDC | 0x0 | PCA0 Module 2 Mode Register | 316 |
| PCA0H | 0xFA | 0x0 | PCA0 Counter High | 317 |
| PCA0L | 0xF9 | 0x0 | PCA0 Counter Low | 317 |
| PCA0MD | 0xD9 | 0x0 | PCA0 Mode | 314 |
| PCA0PWM | 0xDF | 0x0 | PCA0 PWM Configuration | 315 |
| PCON | 0x87 | All | Power Control | 171 |
| PMU0CF | 0xB5 | 0x0 | PMU0 Configuration | 168 |
| PMU0FL | 0xCE | 0x0 | PMU0 Flag Register | 169 |
| PMU0MD | 0xB5 | 0xF | PMU0 Mode | 170 |
| PSCTL | 0x8F | All | Program Store R/W Control | 159 |
| PSW | 0xD0 | All | Program Status Word | 127 |
| REF0CN | 0xD1 | 0x0 | Voltage Reference Control | 90 |
| REG0CN | 0xC9 | 0x0 | Voltage Regulator (REG0) Control | 180 |
| REVID | 0xE2 | 0xF | Revision ID | 155 |
| RSTSRC | 0xEF | 0x0 | Reset Source Configuration/Status | 187 |
| RTC0ADR | 0xAC | 0x0 | RTC0 Address | 202 |
| RTC0DAT | 0xAD | 0x0 | RTC0 Data | 202 |
| RTC0KEY | 0xAE | 0x0 | RTC0 Key | 201 |
| SBUF0 | 0x99 | 0x0 | UART0 Data Buffer | 263 |
| SCON0 | 0x98 | 0x0 | UART0 Control | 262 |
| SFRPAGE | 0xA7 | All | SFR Page | 134 |
| SMB0ADM | 0xF5 | 0x0 | SMBus Slave Address Mask | 247 |
| SMB0ADR | 0xF4 | 0x0 | SMBus Slave Address | 247 |
| SMB0CF | 0xC1 | 0x0 | SMBus0 Configuration | 242 |
| SMB0CN | 0xC0 | 0x0 | SMBus0 Control | 244 |
| SMB0DAT | 0xC2 | 0x0 | SMBus0 Data | 248 |
| SP | 0x81 | All | Stack Pointer | 126 |
| SPI0CFG | 0xA1 | 0x0 | SPI0 Configuration | 272 |
| SPI0CKR | 0xA2 | 0x0 | SPI0 Clock Rate Control | 274 |
| SPI0CN | 0xF8 | 0x0 | SPI0 Control | 273 |
| SPI0DAT | 0xA3 | 0x0 | SPI0 Data | 274 |

**Table 12.3. Special Function Registers (Continued)**

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

| Register | Address | SFR Page | Description | Page |
|---|---|---|---|---|
| TCON | 0x88 | 0x0 | Timer/Counter Control | 284 |
| TH0 | 0x8C | 0x0 | Timer/Counter 0 High | 287 |
| TH1 | 0x8D | 0x0 | Timer/Counter 1 High | 287 |
| TL0 | 0x8A | 0x0 | Timer/Counter 0 Low | 286 |
| TL1 | 0x8B | 0x0 | Timer/Counter 1 Low | 286 |
| TMOD | 0x89 | 0x0 | Timer/Counter Mode | 285 |
| TMR2CN | 0xC8 | 0x0 | Timer/Counter 2 Control | 291 |
| TMR2H | 0xCD | 0x0 | Timer/Counter 2 High | 293 |
| TMR2L | 0xCC | 0x0 | Timer/Counter 2 Low | 293 |
| TMR2RLH | 0xCB | 0x0 | Timer/Counter 2 Reload High | 292 |
| TMR2RLL | 0xCA | 0x0 | Timer/Counter 2 Reload Low | 292 |
| TMR3CN | 0x91 | 0x0 | Timer/Counter 3 Control | 297 |
| TMR3H | 0x95 | 0x0 | Timer/Counter 3 High | 299 |
| TMR3L | 0x94 | 0x0 | Timer/Counter 3 Low | 299 |
| TMR3RLH | 0x93 | 0x0 | Timer/Counter 3 Reload High | 298 |
| TMR3RLL | 0x92 | 0x0 | Timer/Counter 3 Reload Low | 298 |
| TOFFH | 0x8E | 0xF | Temperature Offset High | 87 |
| TOFFL | 0x8D | 0xF | Temperature Offset Low | 87 |
| VDM0CN | 0xFF | 0x0 | VDD Monitor Control | 184 |
| XBR0 | 0xE1 | 0x0 | Port I/O Crossbar Control 0 | 222 |
| XBR1 | 0xE2 | 0x0 | Port I/O Crossbar Control 1 | 223 |
| XBR2 | 0xE3 | 0x0 | Port I/O Crossbar Control 2 | 224 |

# 13. Interrupt Handler

The C8051F99x-C8051F98x microcontroller family includes an extended interrupt system supporting multiple interrupt sources and two priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Refer to Table 13.1, “Interrupt Summary,” on page 140 for a detailed listing of all interrupt sources supported by the device. Refer to the data sheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

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

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

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

## 13.1. Enabling Interrupt Sources

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

## 13.2. MCU Interrupt Sources and Vectors

The CPU services interrupts by generating an LCALL to a predetermined address (the interrupt vector address) to begin execution of an interrupt service routine (ISR). The interrupt vector addresses associated with each interrupt source are listed in Table 13.1 on page 140. Software should ensure that the interrupt vector for each enabled interrupt source contains a valid interrupt service routine.

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

## 13.3. Interrupt Priorities

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

If two interrupts are recognized simultaneously, the interrupt with the higher priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate. See Table 13.1 on page 140 to determine the fixed priority order used to arbitrate between simultaneously recognized interrupts.

## 13.4. Interrupt Latency

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

**Table 13.1. Interrupt Summary**

| Interrupt Source | Interrupt Vector | Priority Order | Pending Flag | Bit addressable? | Cleared by HW? | Enable Flag | Priority Control |
|---|---|---|---|---|---|---|---|
| Reset | 0x0000 | Top | None | N/A | N/A | Always Enabled | Always Highest |
| External Interrupt 0 ($\overline{INT0}$) | 0x0003 | 0 | IE0 (TCON.1) | Y | Y | EX0 (IE.0) | PX0 (IP.0) |
| Timer 0 Overflow | 0x000B | 1 | TF0 (TCON.5) | Y | Y | ET0 (IE.1) | PT0 (IP.1) |
| External Interrupt 1 ($\overline{INT1}$) | 0x0013 | 2 | IE1 (TCON.3) | Y | Y | EX1 (IE.2) | PX1 (IP.2) |
| Timer 1 Overflow | 0x001B | 3 | TF1 (TCON.7) | Y | Y | ET1 (IE.3) | PT1 (IP.3) |
| UART0 | 0x0023 | 4 | RI0 (SCON0.0)<br>TI0 (SCON0.1) | Y | N | ES0 (IE.4) | PS0 (IP.4) |
| Timer 2 Overflow | 0x002B | 5 | TF2H (TMR2CN.7)<br>TF2L (TMR2CN.6) | Y | N | ET2 (IE.5) | PT2 (IP.5) |
| SPI0 | 0x0033 | 6 | SPIF (SPI0CN.7)<br>WCOL (SPI0CN.6)<br>MODF (SPI0CN.5)<br>RXOVRN (SPI0CN.4) | Y | N | ESPI0 (IE.6) | PSPI0 (IP.6) |
| SMB0 | 0x003B | 7 | SI (SMB0CN.0) | Y | N | ESMB0 (EIE1.0) | PSMB0 (EIP1.0) |
| SmaRTClock Alarm | 0x0043 | 8 | ALRM (RTC0CN.2)[2] | N | N | EARTC0 (EIE1.1) | PARTC0 (EIP1.1) |
| ADC0 Window Comparator | 0x004B | 9 | AD0WINT (ADC0CN.3) | Y | N | EWADC0 (EIE1.2) | PWADC0 (EIP1.2) |
| ADC0 End of Conversion | 0x0053 | 10 | AD0INT (ADC0CN.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) |
| Reserved | 0x006B | 13 | | | | | |
| Timer 3 Overflow | 0x0073 | 14 | TF3H (TMR3CN.7)<br>TF3L (TMR3CN.6) | N | N | ET3 (EIE1.7) | PT3 (EIP1.7) |
| Supply Monitor Early Warning | 0x007B | 15 | VDDOK (VDM0CN.5)[1] | N | N | EWARN (EIE2.0) | PWARN (EIP2.0) |
| Port Match | 0x0083 | 16 | None | | | EMAT (EIE2.1) | PMAT (EIP2.1) |
| SmaRTClock Oscillator Fail | 0x008B | 17 | OSCFAIL (RTC0CN.5)[2] | N | N | ERTC0F (EIE2.2) | PFRTC0F (EIP2.2) |
| Reserved | 0x0093 | 18 | | | | | |
| CS0 Conversion Complete | 0x009B | 19 | CS0INT (CS0CN.5) | Y | N | ECSCPT (EIE2.4) | PCSCPT (EIP2.4) |
| CS0 Digital Comparator | 0x00A3 | 20 | CS0CMPF (CS0CN.0) | Y | N | ECSDC (EIE2.5) | PCSDC (EIP2.5) |
| CS0 End of Scan | 0x00AB | 21 | CS0EOS (CS0CN.6) | Y | N | ECSEOS (EIE2.6) | PCSEOS (EIP2.6) |

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

## 13.5. Interrupt Register Descriptions

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

## SFR Definition 13.1. IE: Interrupt Enable

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

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

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

## SFR Definition 13.2. IP: Interrupt Priority

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

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

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

## SFR Definition 13.3. EIE1: Extended Interrupt Enable 1

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

SFR Page = All; SFR Address = 0xE6

| Bit | Name | Function |
|---|---|---|
| 7 | ET3 | **Enable Timer 3 Interrupt.**<br>This bit sets the masking of the Timer 3 interrupt.<br>0: Disable Timer 3 interrupts.<br>1: Enable interrupt requests generated by the TF3L or TF3H flags. |
| 6 | Unused | Read = 0b. Write = Don't care. |
| 5 | ECP0 | **Enable Comparator0 (CP0) Interrupt.**<br>This bit sets the masking of the CP0 interrupt.<br>0: Disable CP0 interrupts.<br>1: Enable interrupt requests generated by the CP0RIF or CP0FIF flags. |
| 4 | EPCA0 | **Enable Programmable Counter Array (PCA0) Interrupt.**<br>This bit sets the masking of the PCA0 interrupts.<br>0: Disable all PCA0 interrupts.<br>1: Enable interrupt requests generated by PCA0. |
| 3 | EADC0 | **Enable ADC0 Conversion Complete Interrupt.**<br>This bit sets the masking of the ADC0 Conversion Complete interrupt.<br>0: Disable ADC0 Conversion Complete interrupt.<br>1: Enable interrupt requests generated by the AD0INT flag. |
| 2 | EWADC0 | **Enable Window Comparison ADC0 Interrupt.**<br>This bit sets the masking of ADC0 Window Comparison interrupt.<br>0: Disable ADC0 Window Comparison interrupt.<br>1: Enable interrupt requests generated by ADC0 Window Compare flag (AD0WINT). |
| 1 | ERTC0A | **Enable SmaRTClock Alarm Interrupts.**<br>This bit sets the masking of the SmaRTClock Alarm interrupt.<br>0: Disable SmaRTClock Alarm interrupts.<br>1: Enable interrupt requests generated by a SmaRTClock Alarm. |
| 0 | ESMB0 | **Enable SMBus (SMB0) Interrupt.**<br>This bit sets the masking of the SMB0 interrupt.<br>0: Disable all SMB0 interrupts.<br>1: Enable interrupt requests generated by SMB0. |

## SFR Definition 13.4. EIP1: Extended Interrupt Priority 1

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

SFR Page = All; SFR Address = 0xF6

| Bit | Name | Function |
|---|---|---|
| 7 | PT3 | **Timer 3 Interrupt Priority Control.**<br>This bit sets the priority of the Timer 3 interrupt.<br>0: Timer 3 interrupts set to low priority level.<br>1: Timer 3 interrupts set to high priority level. |
| 6 | Unused | Read = 0b. Write = Don't care. |
| 5 | PCP0 | **Comparator0 (CP0) Interrupt Priority Control.**<br>This bit sets the priority of the CP0 interrupt.<br>0: CP0 interrupt set to low priority level.<br>1: CP0 interrupt set to high priority level. |
| 4 | PPCA0 | **Programmable Counter Array (PCA0) Interrupt Priority Control.**<br>This bit sets the priority of the PCA0 interrupt.<br>0: PCA0 interrupt set to low priority level.<br>1: PCA0 interrupt set to high priority level. |
| 3 | PADC0 | **ADC0 Conversion Complete Interrupt Priority Control.**<br>This bit sets the priority of the ADC0 Conversion Complete interrupt.<br>0: ADC0 Conversion Complete interrupt set to low priority level.<br>1: ADC0 Conversion Complete interrupt set to high priority level. |
| 2 | PWADC0 | **ADC0 Window Comparator Interrupt Priority Control.**<br>This bit sets the priority of the ADC0 Window interrupt.<br>0: ADC0 Window interrupt set to low priority level.<br>1: ADC0 Window interrupt set to high priority level. |
| 1 | PRTC0A | **SmaRTClock Alarm Interrupt Priority Control.**<br>This bit sets the priority of the SmaRTClock Alarm interrupt.<br>0: SmaRTClock Alarm interrupt set to low priority level.<br>1: SmaRTClock Alarm interrupt set to high priority level. |
| 0 | PSMB0 | **SMBus (SMB0) Interrupt Priority Control.**<br>This bit sets the priority of the SMB0 interrupt.<br>0: SMB0 interrupt set to low priority level.<br>1: SMB0 interrupt set to high priority level. |

## SFR Definition 13.5. EIE2: Extended Interrupt Enable 2

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

SFR Page = All;SFR Address = 0xE7

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 0b. Write = Don't care. |
| 6 | ECSEOS | **Enable Capacitive Sense End of Scan Interrupt.**<br>0: Disable Capacitive Sense End of Scan interrupt.<br>1: Enable interrupt requests generated by CS0EOS. |
| 5 | ECSDC | **Enable Capacitive Sense Digital Comparator Interrupt.**<br>0: Disable Capacitive Sense Digital Comparator interrupt.<br>1: Enable interrupt requests generated by CS0CMPF. |
| 4 | ECSCPT | **Enable Capacitive Sense Conversion Complete Interrupt.**<br>0: Disable Capacitive Sense Conversion Complete interrupt.<br>1: Enable interrupt requests generated by CS0INT. |
| 3 | Unused | Read = 0b. Write = Don't care. |
| 2 | ERTC0F | **Enable SmaRTClock Oscillator Fail Interrupt.**<br>This bit sets the masking of the SmaRTClock Alarm interrupt.<br>0: Disable SmaRTClock Alarm interrupts.<br>1: Enable interrupt requests generated by SmaRTClock Alarm. |
| 1 | EMAT | **Enable Port Match Interrupts.**<br>This bit sets the masking of the Port Match Event interrupt.<br>0: Disable all Port Match interrupts.<br>1: Enable interrupt requests generated by a Port Match. |
| 0 | EWARN | **Enable Supply Monitor Early Warning Interrupt.**<br>This bit sets the masking of the Supply Monitor Early Warning interrupt.<br>0: Disable the Supply Monitor Early Warning interrupt.<br>1: Enable interrupt requests generated by the Supply Monitor. |

## SFR Definition 13.6. EIP2: Extended Interrupt Priority 2

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

SFR Page = All; SFR Address = 0xF7

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 0b. Write = Don't care. |
| 6 | PCSEOS | **Capacitive Sense End of Scan Interrupt Priority Control.**<br>0: Capacitive Sense End of Scan interrupt set to low priority level.<br>1: Capacitive Sense End of Scan interrupt set to high priority level. |
| 5 | PCSDC | **Capacitive Sense Digital Comparator Interrupt Priority Control.**<br>0: Capacitive Sense Digital Comparator interrupt set to low priority level.<br>1: Capacitive Sense Digital Comparator interrupt set to high priority level. |
| 4 | PCSCPT | **Capacitive Sense Conversion Complete Interrupt Priority Control.**<br>0: Capacitive Sense Conversion Complete interrupt set to low priority level.<br>1: Capacitive Sense Conversion Complete interrupt set to high priority level. |
| 3 | Unused | Read = 0b. Write = Don't care. |
| 2 | PRTC0F | **SmaRTClock Oscillator Fail Interrupt Priority Control.**<br>This bit sets the priority of the SmaRTClock Alarm interrupt.<br>0: SmaRTClock Alarm interrupt set to low priority level.<br>1: SmaRTClock Alarm interrupt set to high priority level. |
| 1 | PMAT | **Port Match Interrupt Priority Control.**<br>This bit sets the priority of the Port Match Event interrupt.<br>0: Port Match interrupt set to low priority level.<br>1: Port Match interrupt set to high priority level. |
| 0 | PWARN | **Supply Monitor Early Warning Interrupt Priority Control.**<br>This bit sets the priority of the Supply Monitor Early Warning interrupt.<br>0: Supply Monitor Early Warning interrupt set to low priority level.<br>1: Supply Monitor Early Warning interrupt set to high priority level. |

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

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL ($\overline{\text{INT0}}$ Polarity) and IN1PL ($\overline{\text{INT1}}$ Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (Section “25.1. Timer 0 and Timer 1” on page 280) select level or edge sensitive. The table below lists the possible configurations.

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

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

$\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ are assigned to Port pins as defined in the IT01CF register (see SFR Definition 13.7). Note that $\overline{\text{INT0}}$ and $\overline{\text{INT0}}$ Port pin assignments are independent of any Crossbar assignments. $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ will monitor their assigned Port pins without disturbing the peripheral that was assigned the Port pin via the Crossbar. To assign a Port pin only to $\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register PnSKIP (see Section “21.3. Priority Crossbar Decoder” on page 219 for complete details on configuring the Crossbar).

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

## SFR Definition 13.7. IT01CF: $\overline{\text{INT0}}$/$\overline{\text{INT1}}$ Configuration

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

SFR Page = 0x0; SFR Address = 0xE4

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

# 14. Flash Memory

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

## 14.1. 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 “27. C2 Interface” on page 319.

The Flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before programming Flash memory using MOVX, Flash programming operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target Flash memory); and (2) Writing the Flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software. For detailed guidelines on programming Flash from firmware, please see Section “14.5. Flash Write and Erase Guidelines” on page 156.

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

### 14.1.1. 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 14.4.

### 14.1.2. Flash Erase Procedure

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 Flash page, perform the following steps:

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

Steps 4–6 must be repeated for each 512-byte page to be erased.

**Notes:**

1. Flash security settings may prevent erasure of some Flash pages, such as the reserved area and the page containing the lock bytes. For a summary of Flash security settings and restrictions affecting Flash erase operations, please see Section “14.3. Security Options” on page 152.
2. 8-bit MOVX instructions cannot be used to erase or write to Flash memory at addresses higher than 0x00FF.

### 14.1.3. Flash Write Procedure

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

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

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

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

**Notes:**

1. Flash security settings may prevent writes to some areas of Flash, such as the reserved area. For a summary of Flash security settings and restrictions affecting Flash write operations, please see Section “14.3. Security Options” on page 152.
2. 8-bit MOVX instructions cannot be used to erase or write to Flash memory at addresses higher than 0x00FF.

## 14.2. 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. MOVX read instructions always target XRAM.

## 14.3. 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. See **Section "10. Memory Organization" on page 128** for the location of the security byte. 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 1s complement number represented by the Security Lock Byte. **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).**

| | |
|---|---|
| Security Lock Byte: | 1111 1011b |
| ones Complement: | 0000 0100b |
| Flash pages locked: | 5 (First four Flash pages + Lock Byte Page) |

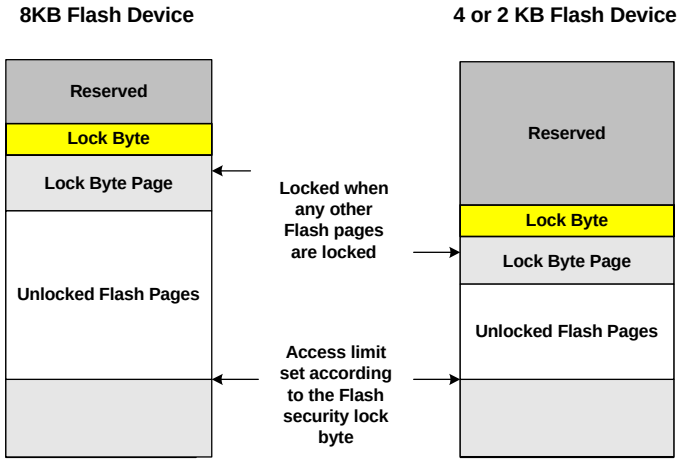



**Figure 14.1. Flash Program Memory Map (8 kB and smaller devices)**

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 14.1 summarizes the Flash security features of the C8051F99x-C8051F98x devices.

**Table 14.1. Flash Security Summary**

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

- C2DE—C2 Device Erase (Erases all Flash pages including the page containing the Lock Byte)
- FEDR—Not permitted; Causes Flash Error Device Reset (FERROR bit in RSTSRC is 1 after reset)
- All prohibited operations that are performed via the C2 interface are ignored (do not cause device reset).
- Locking any Flash page also locks the page containing the Lock Byte.
- Once written to, the Lock Byte cannot be modified except by performing a C2 Device Erase.
- If user code writes to the Lock Byte, the Lock does not take effect until the next device reset.
- The scratchpad is locked when all other Flash pages are locked.
- The scratchpad is erased when a Flash Device Erase command is performed.

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

In many applications, user software may need to determine the MCU part number at run time in order to determine the hardware capabilities. The part number can be determined by reading the value of the DEVICEID Special Function Register.

The value of the DEVICEID register can be decoded as follows:

0xD0—C8051F990
0xD1—C8051F991
0xD6—C8051F996
0xD2—C8051F997

0xD3—C8051F980
0xD4—C8051F981
0xD5—C8051F982
0xD7—C8051F983
0xD8—C8051F985
0xD9—C8051F986
0xDA—C8051F987
0xDB—C8051F988
0xDC—C8051F989

### SFR Definition 14.1. DEVICEID: Device Identification

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

SFR Page = 0xF; SFR Address = 0xE3

| Bit | Name | Function |
|---|---|---|
| 7:0 | DEVICEID[7:0] | **Device Identification.**<br>These bits contain a value that can be decoded to determine the device part number. |

## SFR Definition 14.2. REVID: Revision Identification

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

SFR Page = 0xF; SFR Address = 0xE2

| Bit | Name | Function |
|---|---|---|
| 7:0 | REVID[7:0] | **Revision Identification.**<br>These bits contain a value that can be decoded to determine the silicon revision. For example, 0x00 for Rev A, 0x01 for Rev B, 0x02 for Rev C, etc. |

## 14.5. Flash Write and Erase Guidelines

Any system which contains routines which write or erase Flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of $V_{DD}$, 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 $V_{DD}$ Monitor must be enabled and enabled as a reset source on C8051F99x-C8051F98x devices for the Flash to be successfully modified. **If either the $V_{DD}$ Monitor or the $V_{DD}$ 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.

### 14.5.1. $V_{DD}$ Maintenance and the $V_{DD}$ Monitor

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

**Notes:**

1. On C8051F99x-C8051F98x devices, both the $V_{DD}$ Monitor and the $V_{DD}$ Monitor reset source must be enabled to write or erase Flash without generating a Flash Error Device Reset.
2. On C8051F99x-C8051F98x devices, both the $V_{DD}$ Monitor and the $V_{DD}$ Monitor reset source are enabled by hardware after a power-on reset.

4. As an added precaution, explicitly enable the $V_{DD}$ Monitor and enable the $V_{DD}$ Monitor as a reset source inside the functions that write and erase Flash memory. The $V_{DD}$ Monitor enable instructions should be placed just after the instruction to set PSWE to a 1, but before the Flash write or erase operation instruction.
5. Make certain that all writes to the RSTSRC (Reset Sources) register use direct assignment operators and explicitly DO NOT use the bit-wise operators (such as AND or OR). For example, "RSTSRC = 0x02" is correct, but "RSTSRC |= 0x02" is incorrect.
6. Make certain that all writes to the RSTSRC register explicitly set the PORSF bit to a 1. Areas to check are initialization code which enables other reset sources, such as the Missing Clock Detector or Comparator, for example, and instructions which force a Software Reset. A global search on "RSTSRC" can quickly verify this.

### 14.5.2. PSWE Maintenance

1. 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.
2. Minimize the number of variable accesses while PSWE is set to a 1. Handle pointer address updates and loop maintenance outside the "PSWE = 1;... PSWE = 0;" area. Code examples showing this can be found in :AN201: "Writing to Flash from Firmware", available from the Silicon Laboratories website.
3. 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.
4. 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.
5. 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.

### 14.5.3. System Clock

1. 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.
2. If operating from the external oscillator, switch to the internal oscillator during Flash write or erase operations. The external oscillator can continue to run, and the CPU can switch back to the external oscillator after the Flash operation has completed.

Additional Flash recommendations and example code can be found in “AN201: Writing to Flash from Firmware", available from the Silicon Laboratories web site.

## 14.6. Minimizing Flash Read Current

The Flash memory in the C8051F99x-C8051F98x devices is responsible for a substantial portion of the total digital supply current when the device is executing code. Below are suggestions to minimize Flash read current.

1. Use idle, suspend, or sleep modes while waiting for an interrupt, rather than polling the interrupt flag. Idle Mode is particularly well-suited for use in implementing short pauses, since the wake-up time is no more than three system clock cycles. See the Power Management chapter for details on the various low-power operating modes.
2. C8051F99x-C8051F98x devices have a one-shot timer that saves power when operating at system clock frequencies of 14 MHz or less. The one-shot timer generates a minimum-duration enable signal for the Flash sense amps on each clock cycle in which the Flash memory is accessed. This allows the Flash to remain in a low power state for the remainder of the long clock cycle.
   At clock frequencies above 14 MHz, the system clock cycle becomes short enough that the one-shot timer no longer provides a power benefit. Disabling the one-shot timer at higher frequencies reduces power consumption. The one-shot is enabled by default, and it can be disabled (bypassed) by setting the BYPASS bit (FLSCL.6) to logic 1. To re-enable the one-shot, clear the BYPASS bit to logic 0.
3. Flash read current depends on the number of address lines that toggle between sequential Flash read operations. In most cases, the difference in power is relatively small (on the order of 5%).

   The Flash memory is organized in rows of 64 bytes. A substantial current increase can be detected when the read address jumps from one row in the Flash memory to another. Consider a 3-cycle loop (e.g., SJMP $, or while(1);) which straddles a Flash row boundary. The Flash address jumps from one row to another on two of every three clock cycles. This can result in a current increase of up 30% when compared to the same 3-cycle loop contained entirely within a single row.

   To minimize the power consumption of small loops, it is best to locate them within a single row, if possible. To check if a loop is contained within a Flash row, divide the starting address of the first instruction in the loop by 64. If the remainder (result of modulo operation) plus the length of the loop is less than 63, then the loop fits inside a single Flash row. Otherwise, the loop will be straddling two adjacent Flash rows. If a loop executes in 20 or more clock cycles, then the transitions from one row to another will occur on relatively few clock cycles, and any resulting increase in operating current will be negligible.

## SFR Definition 14.3. PSCTL: Program Store R/W Control

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

SFR Page =All; SFR Address = 0x8F

| Bit | Name | Function |
|---|---|---|
| 7:2 | Unused | Read = 000000b, Write = don't care. |
| 1 | PSEE | **Program Store Erase Enable.**<br>Setting this bit (in combination with PSWE) allows an entire page of Flash program memory to be erased. If this bit is logic 1 and Flash writes are enabled (PSWE is logic 1), a write to Flash memory using the MOVX instruction will erase the entire page that contains the location addressed by the MOVX instruction. The value of the data byte written does not matter.<br>0: Flash program memory erasure disabled.<br>1: Flash program memory erasure enabled. |
| 0 | PSWE | **Program Store Write Enable.**<br>Setting this bit allows writing a byte of data to the Flash program memory using the MOVX write instruction. The Flash location should be erased before writing data.<br>0: Writes to Flash program memory disabled.<br>1: Writes to Flash program memory enabled; the MOVX write instruction targets Flash memory. |

## SFR Definition 14.4. FLKEY: Flash Lock and Key

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

SFR Page = All; SFR Address = 0xB7

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

## SFR Definition 14.5. FLSCL: Flash Scale

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

SFR Page = 0x0; SFR Address = 0xB6

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Always Write to 0. |
| 6 | BYPASS | **Flash Read Timing One-Shot Bypass.**<br>0: The one-shot determines the Flash read time. This setting should be used for operating frequencies less than 14 MHz.<br>1: The system clock determines the Flash read time. This setting should be used for frequencies greater than 14 MHz. |
| 5:0 | Reserved | Reserved. Always Write to 000000b. |

**Note:** Operations which clear the BYPASS bit do not need to be immediately followed by a benign 3-byte instruction. For code compatibility with C8051F930/31/20/21 devices, a benign 3-byte instruction whose third byte is a don't care should follow the clear operation. See the C8051F93x-C8051F92x data sheet for more details.

## SFR Definition 14.6. FLWR: Flash Write Only

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

SFR Page = All; SFR Address = 0xE5

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

## 15. Power Management

C8051F99x-C8051F98x devices support 5 power modes: Normal, Idle, Stop, Suspend, and Sleep. The power management unit (PMU0) allows the device to enter and wake-up from the available power modes. A brief description of each power mode is provided in Table 15.1. Detailed descriptions of each mode can be found in the following sections.

**Table 15.1. Power Modes**

| Power Mode | Description | Wake-Up Sources | Power Savings |
|---|---|---|---|
| Normal | Device fully functional | N/A | Excellent MIPS/mW |
| Idle | All peripherals fully functional. Very easy to wake up. | Any Interrupt | Good<br>No Code Execution |
| Stop | Legacy 8051 low power mode. A reset is required to wake up. | Any Reset | Good<br>No Code Execution<br>Precision Oscillator Disabled |
| Suspend | Similar to Stop Mode, but very fast wake-up time and code resumes execution at the next instruction. | CS0, SmaRTClock, Port Match, Comparator0, $\overline{\text{RST}}$ pin | Very Good<br>No Code Execution<br>All Internal Oscillators Disabled<br>System Clock Gated |
| Sleep | Ultra Low Power and flexible wake-up sources. Code resumes execution at the next instruction. Comparator0 only functional in two-cell mode. | SmaRTClock, Port Match, Comparator0, $\overline{\text{RST}}$ pin | Excellent<br>Power Supply Gated<br>All Oscillators except SmaRT-Clock Disabled |

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

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

## 15.1. Normal Mode

The MCU is fully functional in normal mode. Figure 15.1 shows the on-chip power distribution to various peripherals. There are two supply voltages powering various sections of the chip: $V_{DD}$ and the 1.8 V internal core supply. All analog peripherals are directly powered from the $V_{DD}$ pin. All digital peripherals and the CIP-51 core are powered from the 1.8 V internal core supply. RAM, PMU0 and the SmaRTClock are always powered directly from the $V_{DD}$ pin in sleep mode and powered from the core supply in all other power modes.

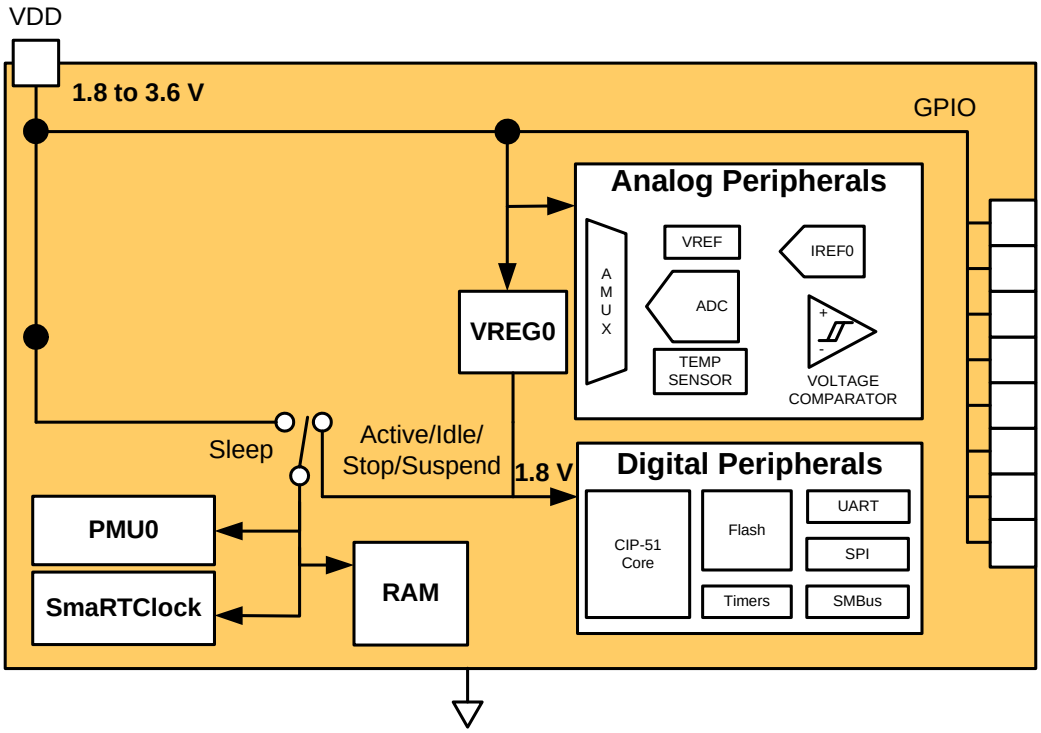



**Figure 15.1. C8051F99x-C8051F98x Power Distribution**

## 15.2. Idle Mode

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

**Note:** To ensure the MCU enters a low power state upon entry into Idle Mode, the one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6).

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

If enabled, the Watchdog Timer (WDT) will eventually cause an internal watchdog reset and thereby terminate the Idle mode. This feature protects the system from an unintended permanent shutdown in the event of an inadvertent write to the PCON register. If this behavior is not desired, the WDT may be disabled by software prior to entering the idle mode if the WDT was initially configured to allow this operation. This provides the opportunity for additional power savings, allowing the system to remain in the idle mode indefinitely, waiting for an external stimulus to wake up the system. Refer to Section “18.6. PCA Watchdog Timer Reset” on page 185 for more information on the use and configuration of the WDT.

## 15.3. Stop Mode

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

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

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

**Note:** To ensure the MCU enters a low power state upon entry into Stop Mode, the one-shot circuit should be enabled by clearing the BYPASS bit (FLSCL.6).

## 15.4. Suspend Mode

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

The following wake-up sources can be configured to wake the device from Suspend Mode:

- CS0 End of Conversion or End of Scan
- SmaRTClock Oscillator Fail
- SmaRTClock Alarm
- Port Match Event
- Comparator0 Rising Edge

**Note:** Upon wake-up from suspend mode, PMU0 requires two system clocks in order to update the PMU0CF wake-up flags. All flags will read back a value of '0' during the first two system clocks following a wake-up from suspend mode.

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

## 15.5. Sleep Mode

Setting the Sleep Mode Select bit (PMU0CF.7) turns off the internal 1.8 V regulator (VREG0) and switches the power supply of all on-chip RAM to the VDD pin (see Figure 15.1). Power to most digital logic on the chip is disconnected; only PMU0 and the SmaRTClock remain powered. Analog peripherals remain powered in two-cell mode. Only the Comparators remain functional when the device enters Sleep Mode. All other analog peripherals (CS0, ADC0, IREF0, External Oscillator, etc.) should be disabled prior to entering Sleep Mode. The system clock source must be set to the low power internal oscillator prior to entering Sleep Mode.

**Important Note**: The precision internal oscillator may potentially lock up after exiting Sleep mode. Systems using Sleep Mode should switch to the low power oscillator prior to entering Sleep Mode:

1. Switch the system clock to the low power oscillator (CLKSEL = 0x04).
2. Turn off the Precision Oscillator (OSCICN &= ~0x80).
3. Enter Sleep.
4. Exit Sleep.
5. Turn on the Precision Oscillator (OSCICN |= 0x80).
6. Switch the system clock to the Precision Oscillator (CLKSEL = 0x00).

GPIO pins configured as digital outputs will retain their output state during sleep mode. In two-cell mode, they will maintain the same current drive capability in sleep mode as they have in normal mode.

GPIO pins configured as digital inputs can be used during sleep mode as wakeup sources using the port match feature. In two-cell mode, they will maintain the same input level specs in sleep mode as they have in normal mode.

C8051F99x-C8051F98x devices support a wakeup request for external devices. Upon exit from sleep mode, the wake-up request signal is driven low, allowing other devices in the system to wake up from their low power modes.

RAM and SFR register contents are preserved in sleep mode as long as the voltage on $V_{DD}$ does not fall below $V_{POR}$. The PC counter and all other volatile state information is preserved allowing the device to resume code execution upon waking up from sleep mode.

**Important Note**: On device reset or upon waking up from Sleep mode, address 0x0000 of external memory may be overwritten by an indeterminate value. The indeterminate value is 0x00 in most situations. A dummy variable should be placed at address 0x0000 in external memory to ensure that the application firmware does not store any data that needs to be retained during sleep or reset at this memory location.

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

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

The comparator requires a supply voltage of at least 1.8 V to operate properly. On C8051F99x-C8051F98x devices, the POR supply monitor can be disabled to save power by writing 1 to the MONDIS (PMU0MD.5) bit. When the POR supply monitor is disabled, all reset sources will trigger a full POR and will re-enable the POR supply monitor.

**Important Note**: The POR Supply Monitor should not be disabled if the supply voltage is greater than 2.4 V. The lowest power sleep mode current, 10 nA typical, can only be achieved when the supply voltage is less than 2.4 V. The lowest power sleep mode for voltages above 2.4 V is 50 nA typical with the POR Supply Monitor enabled.

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

## 15.6. Configuring Wakeup Sources

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

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

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

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

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

Upon exit from suspend or sleep mode, the wake-up flags in the PMU0CF and PMU0FL registers can be read to determine the event which caused the device to wake up. After waking up, the wake-up flags will continue to be updated if any of the wake-up events occur. Wake-up flags are always updated, even if they are not enabled as wake-up sources.

All wake-up flags enabled as wake-up sources in PMU0CF and PMU0FL must be cleared before the device can enter suspend or sleep mode. After clearing the wake-up flags, each of the enabled wake-up events should be checked in the individual peripherals to ensure that a wake-up event did not occur while the wake-up flags were being cleared.

## SFR Definition 15.1. PMU0CF: Power Management Unit Configuration[1,2,3]

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

SFR Page = 0x0; SFR Address = 0xB5

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

**Notes:**

1. Read-modify-write operations (ORL, ANL, etc.) should not be used on this register. Wake-up sources must be re-enabled each time the SLEEP or SUSPEND bits are written to 1.
2. The Low Power Internal Oscillator cannot be disabled and the MCU cannot be placed in Suspend or Sleep Mode if any wake-up flags are set to 1. Software should clear all wake-up sources after each reset and after each wake-up from Suspend or Sleep Modes.
3. PMU0 requires two system clocks to update the wake-up source flags after waking from Suspend mode. The wake-up source flags will read ‘0’ during the first two system clocks following the wake from Suspend mode.

## SFR Definition 15.2. PMU0FL: Power Management Unit Flag[1,2]

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | | | | | | CS0WK |
| Type | R | R | R | R | R | R | R | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Varies |

SFR Page = 0x0; SFR Address = 0xCE

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:1 | Unused | **Unused** | Don't Care. | 0000000b |
| 0 | CS0WK | **CS0 Wake-up Source Enable and Flag** | 0: Disable wake-up on CS0 event.<br>1: Enable wake-up on CS0 event. | Set to 1 if CS0 event caused the last wake-up. |

**Notes:**

1. The Low Power Internal Oscillator cannot be disabled and the MCU cannot be placed in Suspend or Sleep Mode if any wake-up flags are set to 1. Software should clear all wake-up sources after each reset and after each wake-up from Suspend or Sleep Modes.
2. PMU0 requires two system clocks to update the wake-up source flags after waking from Suspend mode. The wake-up source flags will read '0' during the first two system clocks following the wake from Suspend mode.

## SFR Definition 15.3. PMU0MD: Power Management Unit Mode

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

SFR Page = 0xF; SFR Address = 0xB5

| Bit | Name | Function |
|---|---|---|
| 7 | RTCOE | **Buffered SmaRTClock Output Enable.**<br>Enables the buffered SmaRTClock oscillator output on P0.2.<br>0: Buffered SmaRTClock output not enabled.<br>1: Buffered SmaRTClock output is enabled. |
| 6 | WAKEOE | **Wakeup Request Output Enable.**<br>Enables the Sleep Mode wake-up request signal on P0.3.<br>0: Wake-up request signal is not enabled.<br>1: Wake-up request signal is enabled. |
| 5 | MONDIS* | **POR Supply Monitor Disable.**<br>Writing a 1 to this bit disables the POR supply monitor. |
| 4:0 | Unused | Read = 00000b. Write = Don't Care. |

**Notes:** The POR Supply Monitor should not be disabled if the supply voltage is greater than 2.4 V. The lowest power sleep mode current, 10 nA typical, can only be achieved when the supply voltage is less than 2.4 V. The lowest power sleep mode for voltages above 2.4 V is 50 nA typical with the POR Supply Monitor enabled.

## SFR Definition 15.4. PCON: Power Management Control Register

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

SFR Page = All; SFR Address = 0x87

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

## 15.8. Power Management Specifications

See Table 4.5 on page 58 for detailed Power Management Specifications.

# 16. Cyclic Redundancy Check Unit (CRC0)

C8051F99x-C8051F98x devices include a cyclic redundancy check unit (CRC0) that can perform a CRC using a 16-bit polynomial. CRC0 accepts a stream of 8-bit data written to the CRC0IN register. CRC0 posts the 16-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 16.1. CRC0 also has a bit reverse register for quick data manipulation.

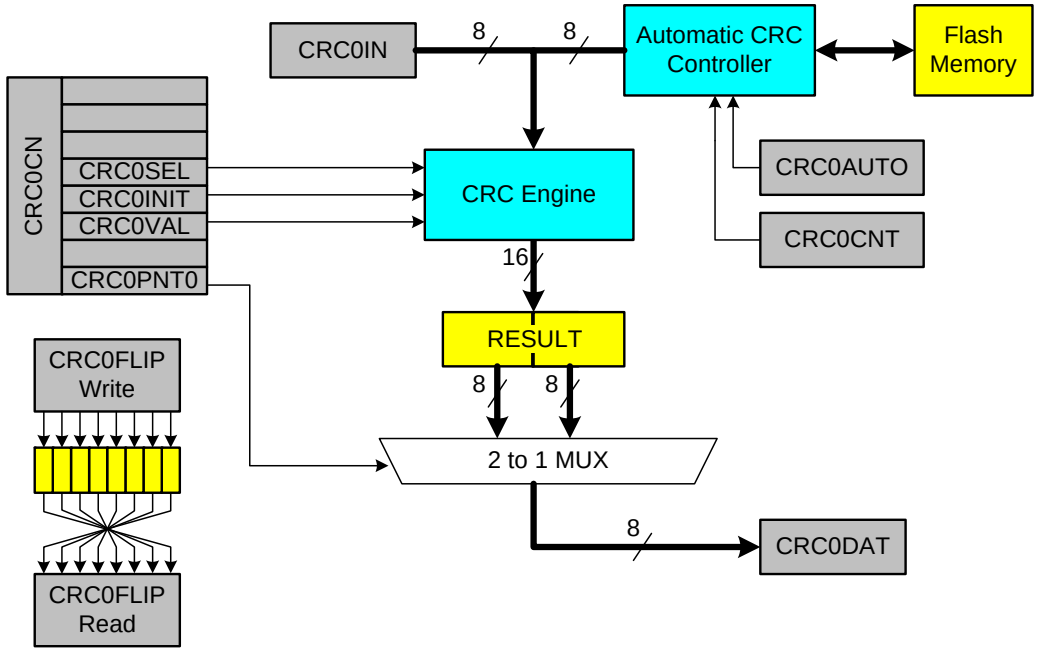



**Figure 16.1. CRC0 Block Diagram**

## 16.1. CRC Algorithm

The C8051F99x-C8051F98x CRC unit generates a CRC result equivalent to the following algorithm:

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

2a. If the MSB of the CRC result is set, shift the CRC result and XOR the result with the selected polynomial.

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

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

The 16-bit C8051F99x-C8051F98x 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;
}
```

Table 16.1 lists several input values and the associated outputs using the 16-bit C8051F99x-C8051F98x CRC algorithm:

**Table 16.1. Example 16-bit CRC Outputs**

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

## 16.2. Preparing for a CRC Calculation

To prepare CRC0 for a CRC calculation, software should set the initial value of the result. The polynomial used for the CRC computation is 0x1021. The CRC0 result may be initialized to one of two values: 0x0000 or 0xFFFF. The following steps can be used to initialize CRC0.

1. Select the initial result value (Set CRC0VAL to 0 for 0x0000 or 1 for 0xFFFF).
2. Set the result to its initial value (Write 1 to CRC0INIT).

## 16.3. Performing a CRC Calculation

Once CRC0 is initialized, the input data stream is sequentially written to CRC0IN, one byte at a time. The CRC0 result is automatically updated after each byte is written. The CRC engine may also be configured to automatically perform a CRC on one or more 256 byte blocks. The following steps can be used to automatically perform a CRC on Flash memory.

1. Prepare CRC0 for a CRC calculation as shown above.
2. Write the index of the starting page to CRC0AUTO.
3. Set the AUTOEN bit in CRC0AUTO.
4. Write the number of 256 byte blocks to perform in the CRC calculation to CRC0CNT.
5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute any additional code until the CRC operation completes. **See the note in SFR Definition 16.1. CRC0CN: CRC0 Control for more information on how to properly initiate a CRC calculation.**
6. Clear the AUTOEN bit in CRC0AUTO.
7. Read the CRC result using the procedure below.

## 16.4. Accessing the CRC0 Result

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

## SFR Definition 16.1. CRC0CN: CRC0 Control

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

SFR Page = All; SFR Address = 0x84

| Bit | Name | Function |
|---|---|---|
| 7:4 | Unused | Read = 0001b; Write = Don't Care. |
| 3 | CRC0INIT | **CRC0 Result Initialization Bit.**<br>Writing a 1 to this bit initializes the entire CRC result based on CRC0VAL. |
| 2 | CRC0VAL | **CRC0 Set Value Initialization Bit.**<br>This bit selects the set value of the CRC result.<br>0: CRC result is set to 0x00000000 on write of 1 to CRC0INIT.<br>1: CRC result is set to 0xFFFFFFFF on write of 1 to CRC0INIT. |
| 1 | Unused | Read = 0b; Write = Don't Care. |
| 0 | CRC0PNT | **CRC0 Result Pointer.**<br>Specifies the byte of the CRC result to be read/written on the next access to CRC0DAT. The value of these bits will auto-increment upon each read or write.<br>0: CRC0DAT accesses bits 7–0 of the 16-bit CRC result.<br>1: CRC0DAT accesses bits 15–8 of the 16-bit CRC result. |

**Note:** Upon initiation of an automatic CRC calculation, the three cycles following a write to CRC0CN that initiate a CRC operation must only contain instructions which execute in the same number of cycles as the number of bytes in the instruction. An example of such an instruction is a 3-byte MOV that targets the CRC0FLIP register. When programming in C, the dummy value written to CRC0FLIP should be a non-zero value to prevent the compiler from generating a 2-byte MOV instruction.

## SFR Definition 16.2. CRC0IN: CRC0 Data Input

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

SFR Page = All; SFR Address = 0x85

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0IN[7:0] | **CRC0 Data Input.**<br>Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm described in Section 16.1 |

## SFR Definition 16.3. CRC0DAT: CRC0 Data Output

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

SFR Page = All; SFR Address = 0x86

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

## SFR Definition 16.4. CRC0AUTO: CRC0 Automatic Control

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

SFR Page = All; SFR Address = 0x9E

| Bit | Name | Function |
|---|---|---|
| 7 | AUTOEN | **Automatic CRC Calculation Enable.**<br>When AUTOEN is set to 1, any write to CRC0CN will initiate an automatic CRC starting at Flash sector CRC0ST and continuing for CRC0CNT sectors. |
| 6 | CRCDONE | **CRCDONE Automatic CRC Calculation Complete.**<br>Set to 0 when a CRC calculation is in progress. Code execution is stopped during a CRC calculation; therefore, reads from firmware will always return 1. |
| 5 | Unused | Read = 0b; Write = Don't Care. |
| 4:0 | CRC0ST[4:0] | **Automatic CRC Calculation Starting Block.**<br>These bits specify the Flash block to start the automatic CRC calculation. The starting address of the first Flash block included in the automatic CRC calculation is CRC0ST x Block Size.<br>**Note:** The block size is 256 bytes. |

## SFR Definition 16.5. CRC0CNT: CRC0 Automatic Flash Sector Count

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

SFR Page = All; SFR Address = 0x9A

| Bit | Name | Function |
|---|---|---|
| 7:5 | Unused | Read = 000b; Write = Don't Care. |
| 4:0 | CRC0CNT[4:0] | **Automatic CRC Calculation Block Count.**<br>These bits specify the number of Flash blocks to include in an automatic CRC calculation. The last address of the last Flash block included in the automatic CRC calculation is (CRC0ST+CRC0CNT) x Block Size - 1.<br><br>**Notes:**<br>**1.** The block size is 256 bytes.<br>**2.** The maximum number of blocks that may be computed in a single operation is 31. To compute a CRC on all 32 blocks, perform one operation on 31 blocks, then perform a second operation on 1 block without clearing the CRC result. |

## 16.5. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 16.2. Each byte of data written to CRC0FLIP is read back bit reversed. For example, if 0xC0 is written to CRC0FLIP, the data read back is 0x03. Bit reversal is a useful mathematical function used in algorithms such as the FFT.

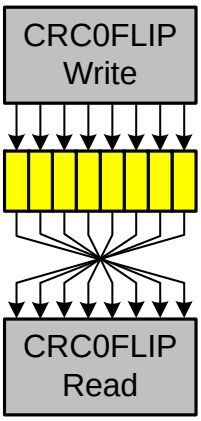



**Figure 16.2. Bit Reverse Register**

### SFR Definition 16.6. CRC0FLIP: CRC0 Bit Flip

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

SFR Page = All; SFR Address = 0x9C

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

# 17. Voltage Regulator (VREG0)

C8051F99x-C8051F98x devices include an internal voltage regulator (VREG0) to regulate the internal core supply to 1.8 V from a VDD supply of 1.8 to 3.6 V. Electrical characteristics for the on-chip regulator are specified in the Electrical Specifications chapter.

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

The internal regulator (VREG0) is disabled when the device enters Sleep Mode and remains enabled when the device enters Suspend Mode. See Section “15. Power Management” on page 162 for complete details about low power modes.

## SFR Definition 17.1. REG0CN: Voltage Regulator Control

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

SFR Page = 0x0; SFR Address = 0xC9

| Bit | Name | Function |
|---|---|---|
| 7 | Unused | Read = 0b. Write = Don’t care. |
| 6:5 | Reserved | Read = 00b. Must Write 00b. |
| 4 | OSCBIAS | **Precision Oscillator Bias.**<br>When set to 1, the bias used by the precision oscillator is forced on. If the precision oscillator is not being used, this bit may be cleared to 0 to reduce supply current in all non-Sleep power modes. |
| 3:1 | Unused | Read = 000b. Write = Don’t care. |
| 0 | Reserved | Read = 0b. Must Write 0b. |

## 17.1. Voltage Regulator Electrical Specifications

See Table 4.15 on page 64 for detailed Voltage Regulator Electrical Specifications.

# 18. Reset Sources

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

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

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

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

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to an internal oscillator. Refer to Section “19. Clocking Sources” on page 188 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 “26.4. Watchdog Timer Mode” on page 311 details the use of the Watchdog Timer). Program execution begins at location 0x0000.

**Important Note**: On device reset or upon waking up from Sleep mode, address 0x0000 of external memory may be overwritten by an indeterminate value. The indeterminate value is 0x00 in most situations. A dummy variable should be placed at address 0x0000 in external memory to ensure that the application firmware does not store any data that needs to be retained during sleep or reset at this memory location.

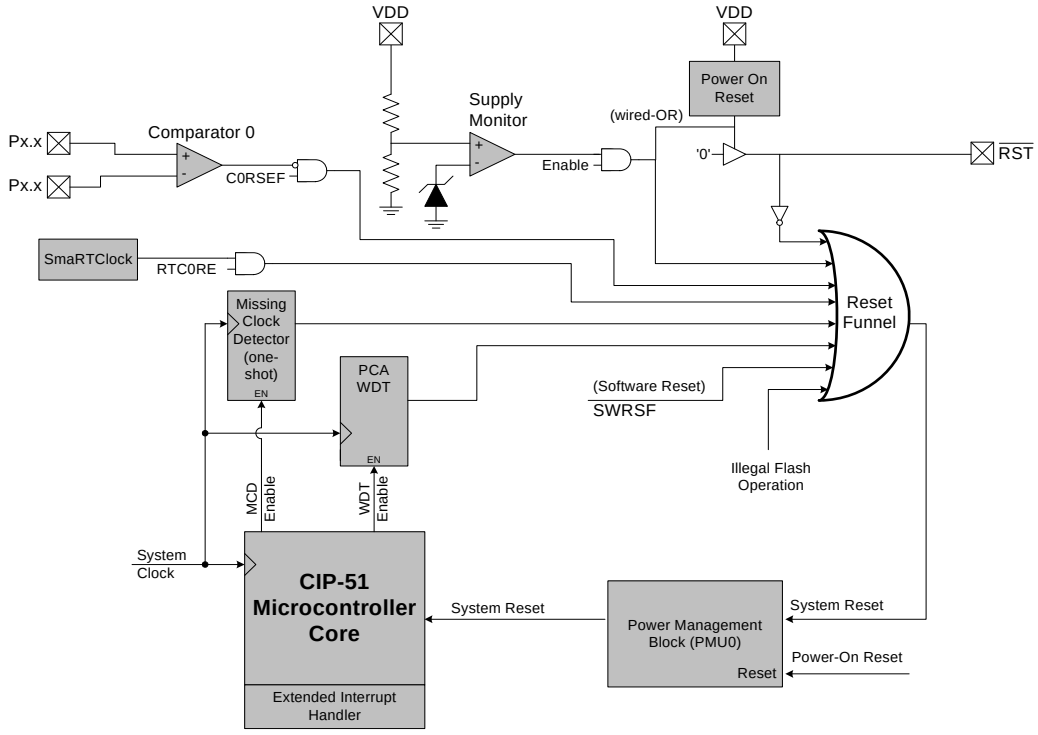



**Figure 18.1. Reset Sources**

## 18.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin voltage tracks $V_{DD}$ (through a weak pull-up) until the device is released from reset. After VDD settles above VPOR, a 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 $V_{POR}$). Figure 18.2 plots the power-on and $V_{DD}$ monitor reset timing. For valid ramp times (less than 3 ms), the power-on reset delay ($T_{PORDelay}$) is typically 7 ms ($V_{DD}$ = 1.8 V) or 15 ms ($V_{DD}$ = 3.6 V).

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

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

The POR supply monitor can be disabled to save power by writing 1 to the MONDIS (PMU0MD.5) bit. When the POR supply monitor is disabled, all reset sources will trigger a full POR and will re-enable the POR supply monitor.

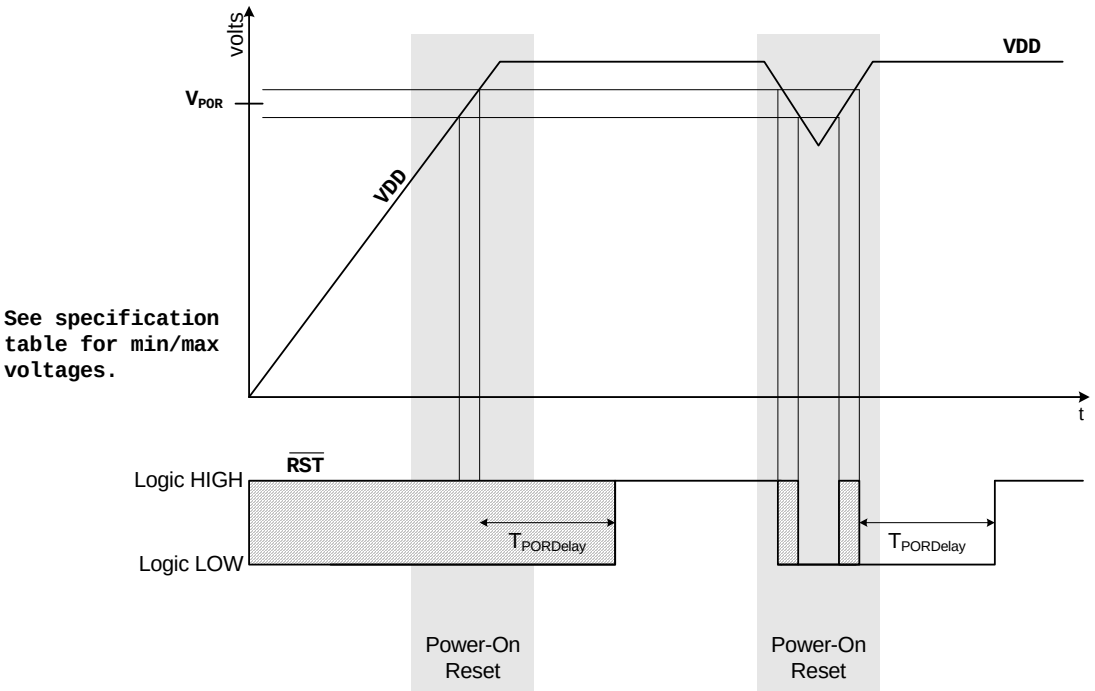



**Figure 18.2. Power-Fail Reset Timing Diagram**

## 18.2. Power-Fail Reset

C8051F99x-C8051F98x devices have a $V_{DD}$ Supply Monitor that is enabled and selected as a reset source after each power-on or power-fail reset. When enabled and selected as a reset source, any power down transition or power irregularity that causes $V_{DD}$ to drop below $V_{RST}$ will cause the $\overline{RST}$ pin to be driven low and the CIP-51 will be held in a reset state (see Figure 18.2). When $V_{DD}$ returns to a level above $V_{RST}$, the CIP-51 will be released from the reset state.

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

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

To allow software early notification that a power failure is about to occur, the VDDOK bit is cleared when the $V_{DD}$ supply falls below the $V_{WARN}$ threshold. The VDDOK bit can be configured to generate an interrupt. See Section "13. Interrupt Handler" on page 138 for more details.

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

**Important Notes**:

- The Power-on Reset (POR) delay is not incurred after a $V_{DD}$ supply monitor reset. See Section "4. Electrical Characteristics" on page 48 for complete electrical characteristics of the $V_{DD}$ monitor.
- Software should take care not to inadvertently disable the $V_{DD}$ Monitor as a reset source when writing to RSTSRC to enable other reset sources or to trigger a software reset. All writes to RSTSRC should explicitly set PORSF to 1 to keep the $V_{DD}$ Monitor enabled as a reset source.
- The $V_{DD}$ supply monitor must be enabled before selecting it as a reset source**.** Selecting the $V_{DD}$ supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the $V_{DD}$ supply monitor and selecting it as a reset source. See Section "4. Electrical Characteristics" on page 48 for minimum $V_{DD}$ Supply Monitor turn-on time. **No delay should be introduced in systems where software contains routines that erase or write Flash memory.** The procedure for enabling the $V_{DD}$ supply monitor and selecting it as a reset source is shown below:

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

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

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

SFR Page = 0x0; SFR Address = 0xFF

| Bit | Name | Function |
|---|---|---|
| 7 | VDMEN | **$V_{DD}$ Supply Monitor Enable.**<br>This bit turns the $V_{DD}$ supply monitor circuit on/off. The VDD Supply Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 18.2).<br>0: $V_{DD}$ Supply Monitor Disabled.<br>1: $V_{DD}$ Supply Monitor Enabled. |
| 6 | VDDSTAT | **$V_{DD}$ Supply Status.**<br>This bit indicates the current power supply status.<br>0: $V_{DD}$ is at or below the $V_{RST}$ threshold.<br>1: $V_{DD}$ is above the $V_{RST}$ threshold. |
| 5 | VDDOK | **$V_{DD}$ Supply Status (Early Warning).**<br>This bit indicates the current $V_{DD}$ power supply status.<br>0: $V_{DD}$ is at or below the $VDD_{WARN}$ threshold.<br>1: $V_{DD}$ is above the $VDD_{WARN}$ threshold. |
| 4 | Unused | Read = 0b. Write = Don't Care. |
| 3 | VDDOKIE | **$V_{DD}$ Early Warning Interrupt Enable.**<br>Enables the $V_{DD}$ Early Warning Interrupt.<br>0: $V_{DD}$ Early Warning Interrupt is disabled.<br>1: $V_{DD}$ Early Warning Interrupt is enabled. |
| 2:0 | Unused | Read = 000b. Write = Don't Care. |

## 18.3. External Reset

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

## 18.4. Missing Clock Detector Reset

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

## 18.5. Comparator0 Reset

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

## 18.6. PCA Watchdog Timer Reset

The programmable watchdog timer (WDT) function of the programmable counter array (PCA) can be used to prevent software from running out of control during a system malfunction. The PCA WDT function can be enabled or disabled by software as described in Section “26.4. Watchdog Timer Mode” on page 311; the WDT is enabled and clocked by SYSCLK / 12 following any reset. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit (RSTSRC.5) is set to 1. The PCA Watchdog Timer reset source is automatically disabled when the device is in the low power suspend or sleep mode. Upon exit from either low power state, the enabled/disabled state of this reset source is restored to its previous value.The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 18.7. Flash Error Reset

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

- A Flash write or erase is attempted above user code space. This occurs when PSWE is set to 1 and a MOVX write operation targets an address above the Lock Byte address.
- A Flash read is attempted above user code space. This occurs when a MOVC operation targets an address above the Lock Byte address.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above the Lock Byte address.
- A Flash read, write or erase attempt is restricted due to a Flash security setting (see Section “14.3. Security Options” on page 152).
- A Flash write or erase is attempted while the $V_{DD}$ Monitor is disabled.

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

## 18.8. SmaRTClock (Real Time Clock) Reset

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

## 18.9. Software Reset

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

## SFR Definition 18.2. RSTSRC: Reset Source

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

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

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

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

## 19. Clocking Sources

C8051F99x-C8051F98x devices include a programmable precision internal oscillator, an external oscillator drive circuit, a low power internal oscillator, and a SmaRTClock real time clock oscillator. The precision internal oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 19.1. The external oscillator can be configured using the OSCXCN register. The low power internal oscillator is automatically enabled and disabled when selected and deselected as a clock source. SmaRTClock operation is described in the SmaRTClock oscillator chapter.

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

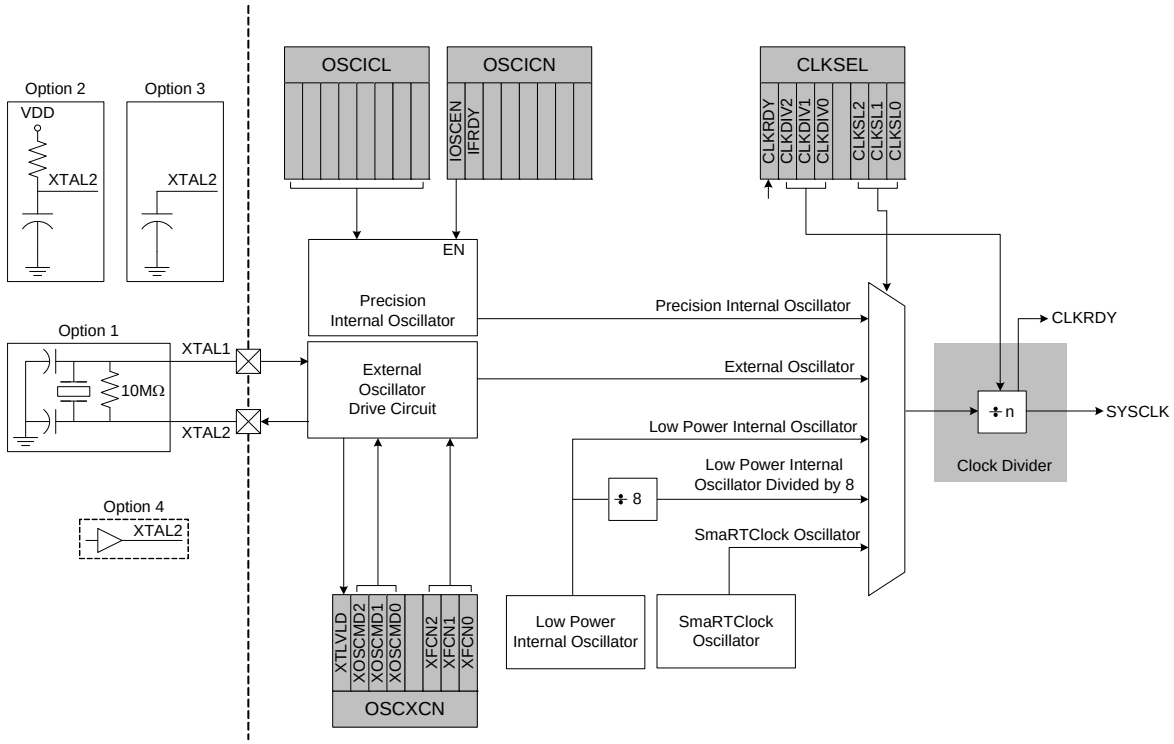



**Figure 19.1. Clocking Sources Block Diagram**

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

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

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

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

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

## 19.1. Programmable Precision Internal Oscillator

All C8051F99x-C8051F98x devices include a programmable precision internal oscillator that may be selected as the system clock. OSCICL is factory calibrated to obtain a 24.5 MHz frequency. See Section “4. Electrical Characteristics” on page 48 for complete oscillator specifications.

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

**Important Note**: The precision internal oscillator may potentially lock up after exiting Sleep mode. Systems using Sleep Mode should switch to the low power oscillator prior to entering Sleep Mode:

1. Switch the system clock to the low power oscillator (CLKSEL = 0x04).
2. Turn off the Precision Oscillator (OSCICN &= ~0x80).
3. Enter Sleep.
4. Exit Sleep.
5. Turn on the Precision Oscillator (OSCICN |= 0x80).
6. Switch the system clock to the Precision Oscillator (CLKSEL = 0x00).

## 19.2. Low Power Internal Oscillator

All C8051F99x-C8051F98x devices include a low power internal oscillator that defaults as the system clock after a system reset. The low power internal oscillator frequency is 20 MHz ± 10% and is automatically enabled when selected as the system clock and disabled when not in use. See Section “4. Electrical Characteristics” on page 48 for complete oscillator specifications.

## 19.3. External Oscillator Drive Circuit

All C8051F99x-C8051F98x devices include an external oscillator circuit that may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. Figure 19.1 shows a block diagram of the four external oscillator options. The external oscillator is enabled and configured using the OSCXCN register.

The external oscillator output may be selected as the system clock or used to clock some of the digital peripherals (e.g., Timers, PCA, etc.). See the data sheet chapters for each digital peripheral for details. See Section “4. Electrical Characteristics” on page 48 for complete oscillator specifications.

### 19.3.1. External Crystal Mode

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

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

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

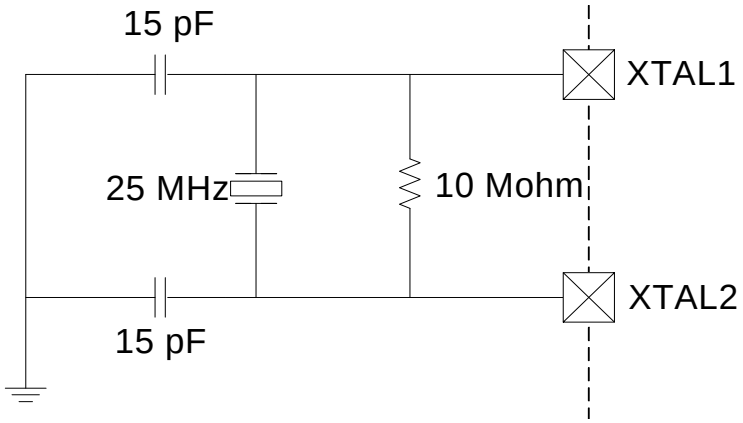



**Figure 19.2. 25 MHz External Crystal Example**

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

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

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

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

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

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

### 19.3.2. External RC Mode

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

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

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

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

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

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

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

Referencing Table 19.2, the recommended XFCN setting is 010.

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

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

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

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

### 19.3.3. External Capacitor Mode

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

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

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

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

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

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

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

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

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

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

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

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

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

### 19.3.4. External CMOS Clock Mode

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

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

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

The clocking sources on C8051F99x-C8051F98x devices are enabled and configured using the OSCICN, OSCICL, OSCXCN and the SmaRTClock internal registers. See Section “20. SmaRTClock (Real Time Clock)” on page 197 for SmaRTClock register descriptions. The system clock source for the MCU can be selected using the CLKSEL register. To minimize active mode current, the oneshot timer which sets Flash read time should by bypassed when the system clock is greater than 10 MHz. See the FLSCL register description for details.

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

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

### SFR Definition 19.1. CLKSEL: Clock Select

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

SFR Page = All; SFR Address = 0xA9

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

## SFR Definition 19.2. OSCICN: Internal Oscillator Control

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

SFR Page = 0x0; SFR Address = 0xB2

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

**Notes:**

1. Read-modify-write operations such as ORL and ANL must be used to set or clear the enable bit of this register.
2. OSCBIAS (REG0CN.4) must be set to 1 before enabling the precision internal oscillator.

## SFR Definition 19.3. OSCICL: Internal Oscillator Calibration

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

SFR Page = 0x0; SFR Address = 0xB3

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

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

## SFR Definition 19.4. OSCXCN: External Oscillator Control

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

SFR Page = 0x0; SFR Address = 0xB1

| Bit | Name | Function |
|---|---|---|
| 7 | XCLKVLD | **External Oscillator Valid Flag.**<br>Provides External Oscillator status and is valid at all times for all modes of operation except External CMOS Clock Mode and External CMOS Clock Mode with divide by 2. In these modes, XCLKVLD always returns 0.<br>0: External Oscillator is unused or not yet stable.<br>1: External Oscillator is running and stable. |
| 6:4 | XOSCMD | **External Oscillator Mode Bits.**<br>Configures the external oscillator circuit to the selected mode.<br>00x: External Oscillator circuit disabled.<br>010: External CMOS Clock Mode.<br>011: External CMOS Clock Mode with divide by 2 stage.<br>100: RC Oscillator Mode.<br>101: Capacitor Oscillator Mode.<br>110: Crystal Oscillator Mode.<br>111: Crystal Oscillator Mode with divide by 2 stage. |
| 3 | Reserved | Read = 0b. Must Write 0b. |
| 2:0 | XFCN | **External Oscillator Frequency Control Bits.**<br>Controls the external oscillator bias current.<br>000-111: See Table 19.1 on page 190 (Crystal Mode) or Table 19.2 on page 191 (RC or C Mode) for recommended settings. |

## 20. SmaRTClock (Real Time Clock)

C8051F99x-C8051F98x devices include an ultra low power 32-bit SmaRTClock Peripheral (Real Time Clock) with alarm. The SmaRTClock has a dedicated 32 kHz oscillator that can be configured for use with or without a crystal. No external resistor or loading capacitors are required. The on-chip loading capacitors are programmable to 16 discrete levels allowing compatibility with a wide range of crystals. The SmaRTClock can operate directly from a 1.8–3.6 V battery voltage and remains operational even when the device goes into its lowest power down mode. The SmaRTClock output can be buffered and routed to a GPIO pin to provide an accurate, low frequency clock to other devices while the MCU is in its lowest power down mode (see "PMU0MD: Power Management Unit Mode" on page 170 for more details). C8051F99x-C8051F98x devices also support an ultra low power internal LFO that reduces sleep mode current.

The SmaRTClock allows a maximum of 36 hour 32-bit independent time-keeping when used with a 32.768 kHz Watch Crystal. The SmaRTClock provides an Alarm and Missing SmaRTClock events, which could be used as reset or wakeup sources. See Section "18. Reset Sources" on page 181 and Section "15. Power Management" on page 162 for details on reset sources and low power mode wake-up sources, respectively.

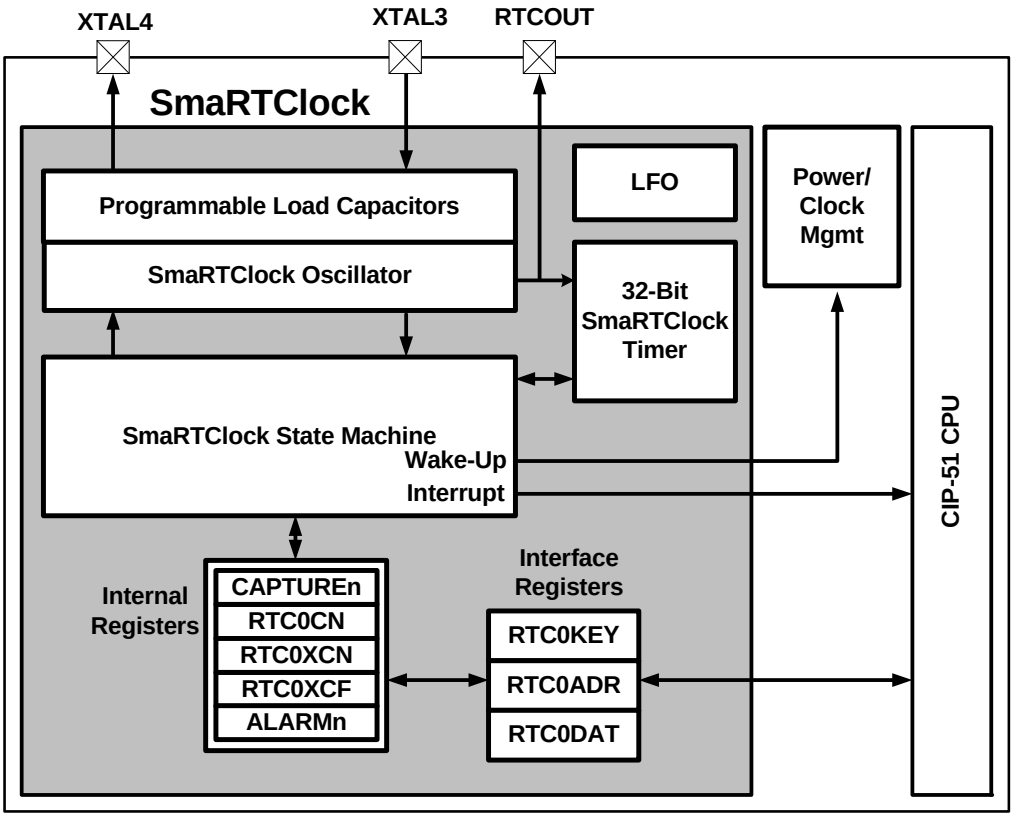



**Figure 20.1. SmaRTClock Block Diagram**

## 20.1. SmaRTClock Interface

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

### Table 20.1. SmaRTClock Internal Registers

| SmaRTClock Address | SmaRTClock Register | Register Name | Description |
|---|---|---|---|
| 0x00–0x03 | CAPTUREn | SmaRTClock Capture Registers | Four Registers used for setting the 32-bit SmaRTClock timer or reading its current value. |
| 0x04 | RTC0CN | SmaRTClock Control Register | Controls the operation of the SmaRTClock State Machine. |
| 0x05 | RTC0XCN | SmaRTClock Oscillator Control Register | Controls the operation of the SmaRTClock Oscillator. |
| 0x06 | RTC0XCF | SmaRTClock Oscillator Configuration Register | Controls the value of the progammable oscillator load capacitance and enables/disables AutoStep. |
| 0x08–0x0B | ALARMn | SmaRTClock Alarm Registers | Four registers used for setting or reading the 32-bit SmaRTClock alarm value. |

### 20.1.1. SmaRTClock Lock and Key Functions

The SmaRTClock Interface has an RTC0KEY register for legacy reasons, however, all writes to this register are ignored. The SmaRTClock interface is always unlocked on C8051F99x-C8051F98x.

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

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

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

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

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

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

### 20.1.3. RTC0ADR Short Strobe Feature

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

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

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

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

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

### 20.1.4. SmaRTClock Interface Autoread Feature

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

### 20.1.5. RTC0ADR Autoincrement Feature

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

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

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

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

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

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

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

SFR Page = 0x0; SFR Address = 0xAE

| Bit | Name | Function |
|---|---|---|
| 7:0 | RTC0ST | **SmaRTClock Interface Status.**<br>Provides lock status when read.<br><br>**Read:**<br>**0x02:** SmaRTClock Interface is unlocked.<br><br>**Write:**<br>Writes to RTC0KEY have no effect. |

## SFR Definition 20.2. RTC0ADR: SmaRTClock Address

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

SFR Page = 0x0; SFR Address = 0xAC

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

## SFR Definition 20.3. RTC0DAT: SmaRTClock Data

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

SFR Page= 0x0; SFR Address = 0xAD

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

## 20.2. SmaRTClock Clocking Sources

The SmaRTClock peripheral is clocked from its own timebase, independent of the system clock. The SmaRTClock timebase can be derived from an external CMOS clock, the internal LFO, or the SmaRTClock oscillator circuit, which has two modes of operation: Crystal Mode, and Self-Oscillate Mode. The oscillation frequency is 32.768 kHz in Crystal Mode and can be programmed in the range of 10 kHz to 40 kHz in Self-Oscillate Mode. The internal LFO frequency is 16.4 kHz ±20%. The frequency of the SmaRTClock oscillator can be measured with respect to another oscillator using an on-chip timer. See Section “25. Timers” on page 278 for more information on how this can be accomplished.

**Note:** The SmaRTClock timebase can be selected as the system clock and routed to a port pin. See Section “19. Clocking Sources” on page 188 for information on selecting the system clock source and Section “21. Port Input/Output” on page 215 for information on how to route the system clock to a port pin. The SmaRTClock timebase can also be routed to a port pin while the device is in its ultra low power sleep mode. See the PMU0MD register description for details.

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

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

1. Configure the XTAL3 and XTAL4 pins for Analog I/O.
2. Set SmaRTClock to Crystal Mode (XMODE = 1).
3. Disable Automatic Gain Control (AGCEN) and enable Bias Doubling (BIASX2) for fast crystal startup.
4. Set the desired loading capacitance (RTC0XCF).
5. Enable power to the SmaRTClock oscillator circuit (RTC0EN = 1).
6. Wait 20 ms.
7. Poll the SmaRTClock Clock Valid Bit (CLKVLD) until the crystal oscillator stabilizes.
8. Poll the SmaRTClock Load Capacitance Ready Bit (LOADRDY) until the load capacitance reaches its programmed value.
9. Enable Automatic Gain Control (AGCEN) and disable Bias Doubling (BIASX2) for maximum power savings.
10. Enable the SmaRTClock missing clock detector.
11. Wait 2 ms.
12. Clear the PMU0CF wake-up source flags.

In Crystal Mode, the SmaRTClock oscillator may be driven by an external CMOS clock. The CMOS clock should be applied to XTAL3. XTAL34 should be left floating. In this mode, the external CMOS clock is ac coupled into the SmaRTClock and should have a minimum voltage swing of 400 mV. The CMOS clock signal voltage should not exceed VDD or drop below GND. Bias levels closer to VDD will result in lower I/O power consumption because the XTAL3 pin has a built-in weak pull-up. The SmaRTClock oscillator should be configured to its lowest bias setting with AGC disabled. The CLKVLD bit is indeterminate when using a CMOS clock, however, the OSCFAIL bit may be checked 2 ms after SmaRTClock oscillator is powered on to ensure that there is a valid clock on XTAL3.

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

When using Self-Oscillate Mode, the XTAL3 and XTAL4 pins are internally shorted together. The following steps show how to configure SmaRTClock for use in Self-Oscillate Mode:

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

### 20.2.3. Using the Low Frequency Oscillator (LFO)

The low frequency oscillator provides an ultra low power, on-chip clock source to the SmaRTClock. The typical frequency of oscillation is 16.4 kHz ±20%. No external components are required to use the LFO and the XTAL3 and XTAL4 pins do not need to be shorted together.

The following steps show how to configure SmaRTClock for use with the LFO:

1. Enable and select the Low Frequency Oscillator (LFOEN = 1).
2. The LFO starts oscillating instantaneously.

When the LFO is enabled, the SmaRTClock oscillator increments bit 1 of the 32-bit timer (instead of bit 0). This effectively multiplies the LFO frequency by 2, making the RTC timebase behave as if a 32.768 kHz crystal is connected at the output.

### 20.2.4. Programmable Load Capacitance

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

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

**Table 20.2. SmaRTClock Load Capacitance Settings**

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

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

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

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

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

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

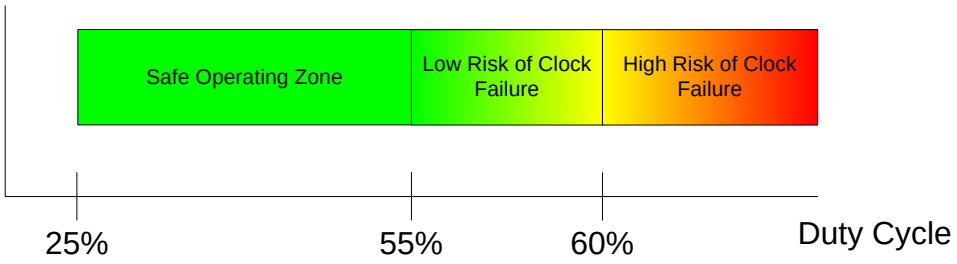



**Figure 20.2. Interpreting Oscillation Robustness (Duty Cycle) Test Results**

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

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

**Table 20.3. SmaRTClock Bias Settings**

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

### 20.2.6. Missing SmaRTClock Detector

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

A SmaRTClock Missing Clock detector timeout can trigger an interrupt, wake the device from a low power mode, or reset the device. See Section “13. Interrupt Handler” on page 138, Section “15. Power Management” on page 162, and Section “18. Reset Sources” on page 181 for more information.

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

### 20.2.7. SmaRTClock Oscillator Crystal Valid Detector

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

**Notes:**

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

## 20.3. SmaRTClock Timer and Alarm Function

The SmaRTClock timer is a 32-bit counter that, when running (RTC0TR = 1), is incremented every SmaRTClock oscillator cycle. The timer has an alarm function that can be set to generate an interrupt, wake the device from a low power mode, or reset the device at a specific time. See Section “13. Interrupt Handler” on page 138, Section “15. Power Management” on page 162, and Section “18. Reset Sources” on page 181 for more information.

The SmaRTClock timer includes an Auto Reset feature, which automatically resets the timer to zero one SmaRTClock cycle after the alarm signal is deasserted. When using Auto Reset, the Alarm match value should always be set to 2 counts less than the desired match value. When using the LFO in combination with Auto Reset, the right-justified Alarm match value should be set to 4 counts less than the desired match value. Auto Reset can be enabled by writing a 1 to ALRM (RTC0CN.2).

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

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

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

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

1. Write 1 to RTC0CAP. This will transfer the contents of the timer to the CAPTUREn registers.
2. Poll RTC0CAP until it is cleared to 0 by hardware.
3. A snapshot of the timer value can be read from the CAPTUREn registers

### 20.3.2. Setting a SmaRTClock Alarm

The SmaRTClock alarm function compares the 32-bit value of SmaRTClock Timer to the value of the ALARMn registers. An alarm event is triggered if the SmaRTClock timer is **equal to** the ALARMn registers. If Auto Reset is enabled, the 32-bit timer will be cleared to zero one SmaRTClock cycle after the alarm event.

The SmaRTClock alarm event can be configured to reset the MCU, wake it up from a low power mode, or generate an interrupt. See Section “13. Interrupt Handler” on page 138, Section “15. Power Management” on page 162, and Section “18. Reset Sources” on page 181 for more information.

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

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

**Notes:**

1. The ALRM bit, which is used as the SmaRTClock Alarm Event flag, is cleared by disabling SmaRTClock Alarm Events (RTC0AEN = 0).
2. If AutoReset is disabled, disabling (RTC0AEN = 0) then Re-enabling Alarm Events (RTC0AEN = 1) after a SmaRTClock Alarm without modifying ALARMn registers will automatically schedule the next alarm after 2^32 SmaRTClock cycles (approximately 36 hours using a 32.768 kHz crystal).
3. The SmaRTClock Alarm Event flag will remain asserted for a maximum of one SmaRTClock cycle. See Section “15. Power Management” on page 162 for information on how to capture a SmaRTClock Alarm event using a flag which is not automatically cleared by hardware.

### 20.3.3. Software Considerations for using the SmaRTClock Timer and Alarm

The SmaRTClock timer and alarm have two operating modes to suit varying applications. The two modes are described below:

**Mode 1:**
The first mode uses the SmaRTClock timer as a perpetual timebase which is never reset to zero. Every 36 hours, the timer is allowed to overflow without being stopped or disrupted. The alarm interval is software managed and is added to the ALRMn registers by software after each alarm. This allows the alarm match value to always stay ahead of the timer by one software managed interval. If software uses 32-bit unsigned addition to increment the alarm match value, then it does not need to handle overflows since both the timer and the alarm match value will overflow in the same manner.

This mode is ideal for applications which have a long alarm interval (e.g., 24 or 36 hours) and/or have a need for a perpetual timebase. An example of an application that needs a perpetual timebase is one whose wake-up interval is constantly changing. For these applications, software can keep track of the number of timer overflows in a 16-bit variable, extending the 32-bit (36 hour) timer to a 48-bit (272 year) perpetual timebase.

**Mode 2:**
The second mode uses the SmaRTClock timer as a general purpose up counter which is auto reset to zero by hardware after each alarm. The alarm interval is managed by hardware and stored in the ALRMn registers. Software only needs to set the alarm interval once during device initialization. After each alarm, software should keep a count of the number of alarms that have occurred in order to keep track of time.

This mode is ideal for applications that require minimal software intervention and/or have a fixed alarm interval. This mode is the most power efficient since it requires less CPU time per alarm.

## Internal Register Definition 20.4. RTC0CN: SmaRTClock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RTC0EN | MCLKEN | OSCFAIL | RTC0TR | RTC0AEN | ALRM | RTC0SET | RTC0CAP |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | Varies | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Address = 0x04

| Bit | Name | Function | | |
|---|---|---|---|---|
| 7 | RTC0EN | **SmaRTClock Enable.**<br>Enables/disables the SmaRTClock oscillator and associated bias currents.<br>0: SmaRTClock oscillator disabled.<br>1: SmaRTClock oscillator enabled. | | |
| 6 | MCLKEN | **Missing SmaRTClock Detector Enable.**<br>Enables/disables the missing SmaRTClock detector.<br>0: Missing SmaRTClock detector disabled.<br>1: Missing SmaRTClock detector enabled. | | |
| 5 | OSCFAIL | **SmaRTClock Oscillator Fail Event Flag.**<br>Set by hardware when a missing SmaRTClock detector timeout occurs. Must be cleared by software. The value of this bit is not defined when the SmaRTClock oscillator is disabled. | | |
| 4 | RTC0TR | **SmaRTClock Timer Run Control.**<br>Controls if the SmaRTClock timer is running or stopped (holds current value).<br>0: SmaRTClock timer is stopped.<br>1: SmaRTClock timer is running. | | |
| 3 | RTC0AEN | **SmaRTClock Alarm Enable.**<br>Enables/disables the SmaRTClock alarm function. Also clears the ALRM flag.<br>0: SmaRTClock alarm disabled.<br>1: SmaRTClock alarm enabled. | | |
| 2 | ALRM | **SmaRTClock Alarm Event Flag and Auto Reset Enable.**<br>Reads return the state of the alarm event flag.<br>Writes enable/disable the Auto Reset function. | Read:<br>0: SmaRTClock alarm event flag is de-asserted.<br>1: SmaRTClock alarm event flag is asserted. | Write:<br>0: Disable Auto Reset.<br>1: Enable Auto Reset. |
| 1 | RTC0SET | **SmaRTClock Timer Set.**<br>Writing 1 initiates a SmaRTClock timer set operation. This bit is cleared to 0 by hardware to indicate that the timer set operation is complete. | | |
| 0 | RTC0CAP | **SmaRTClock Timer Capture.**<br>Writing 1 initiates a SmaRTClock timer capture operation. This bit is cleared to 0 by hardware to indicate that the timer capture operation is complete. | | |
| **Note:** The ALRM flag will remain asserted for a maximum of one SmaRTClock cycle. See Section “Power Management” on page 162 for information on how to capture a SmaRTClock Alarm event using a flag which is not automatically cleared by hardware. | | | | |

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

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AGCEN | XMODE | BIASX2 | CLKVLD | LFOEN | | | |
| Type | R/W | R/W | R/W | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Address = 0x05

| Bit | Name | Function |
|---|---|---|
| 7 | AGCEN | **SmaRTClock Oscillator Automatic Gain Control (AGC) Enable.**<br>0: AGC disabled.<br>1: AGC enabled. |
| 6 | XMODE | **SmaRTClock Oscillator Mode.**<br>Selects Crystal or Self Oscillate Mode.<br>0: Self-Oscillate Mode selected.<br>1: Crystal Mode selected. |
| 5 | BIASX2 | **SmaRTClock Oscillator Bias Double Enable.**<br>**Enables/disables the Bias Double feature.**<br>0: Bias Double disabled.<br>1: Bias Double enabled. |
| 4 | CLKVLD | **SmaRTClock Oscillator Crystal Valid Indicator.**<br>Indicates if oscillation amplitude is sufficient for maintaining oscillation.<br>0: Oscillation has not started or oscillation amplitude is too low to maintain oscillation.<br>1: Sufficient oscillation amplitude detected. |
| 3 | LFOEN | **Low Frequency Oscillator Enable and Select.**<br>Overrides XMODE and selects the internal low frequency oscillator (LFO) as the SmaRTClock oscillator source.<br>0: XMODE determines SmaRTClock oscillator source.<br>1: LFO enabled and selected as SmaRTClock oscillator source. |
| 2:0 | Unused | Read = 000b; Write = Don't Care. |

## Internal Register Definition 20.6. RTC0XCF: SmaRTClock Oscillator Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AUTOSTP | LOADRDY | | | LOADCAP | | | |
| Type | R/W | R | R | R | R/W | | | |
| Reset | 0 | 0 | 0 | 0 | Varies | Varies | Varies | Varies |

SmaRTClock Address = 0x06

| Bit | Name | Function |
|---|---|---|
| 7 | AUTOSTP | **Automatic Load Capacitance Stepping Enable.**<br>Enables/disables automatic load capacitance stepping.<br>0: Load capacitance stepping disabled.<br>1: Load capacitance stepping enabled. |
| 6 | LOADRDY | **Load Capacitance Ready Indicator.**<br>Set by hardware when the load capacitance matches the programmed value.<br>0: Load capacitance is currently stepping.<br>1: Load capacitance has reached it programmed value. |
| 5:4 | Unused | Read = 00b; Write = Don't Care. |
| 3:0 | LOADCAP | **Load Capacitance Programmed Value.**<br>Holds the user's desired value of the load capacitance. See Table 20.2 on page 205. |

## Internal Register Definition 20.7. CAPTUREn: SmaRTClock Timer Capture

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CAPTURE[31:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Addresses: CAPTURE0 = 0x00; CAPTURE1 = 0x01; CAPTURE2 =0x02; CAPTURE3: 0x03.

| Bit | Name | Function |
|---|---|---|
| 7:0 | CAPTURE[31:0] | **SmaRTClock Timer Capture.**<br>These 4 registers (CAPTURE3–CAPTURE0) are used to read or set the 32-bit SmaRTClock timer. Data is transferred to or from the SmaRTClock timer when the RTC0SET or RTC0CAP bits are set. |
| **Note:** The least significant bit of the timer capture value is in CAPTURE0.0. | | |

## Internal Register Definition 20.8. ALARMn: SmaRTClock Alarm Programmed Value

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ALARM[31:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SmaRTClock Addresses: ALARM0 = 0x08; ALARM1 = 0x09; ALARM2 = 0x0A; ALARM3 = 0x0B

| Bit | Name | Function |
|---|---|---|
| 7:0 | ALARM[31:0] | **SmaRTClock Alarm Programmed Value.**<br>These 4 registers (ALARM3–ALARM0) are used to set an alarm event for the SmaRTClock timer. The SmaRTClock alarm should be disabled (RTC0AEN=0) when updating these registers. |
| **Note:** The least significant bit of the alarm programmed value is in ALARM0.0. | | |

## 21. Port Input/Output

Digital and analog resources are available through 16 or 17 I/O pins. Port pins are organized as three byte-wide ports. Port pins P0.0–P1.7 can be defined as digital or analog I/O. Digital I/O pins can be assigned to one of the internal digital resources or used as general purpose I/O (GPIO). Analog I/O pins are used by the internal analog resources. P2.7 can be used as GPIO and is shared with the C2 Interface Data signal (C2D). See Section “27. C2 Interface” on page 319 for more details.

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

All Port I/Os can tolerate voltages up to the supply rail when used as digital inputs or open-drain outputs. For Port I/Os configured as push-pull outputs, current is sourced from the $V_{DD}$ supply. Port I/Os used for analog functions can operate up to the $V_{DD}$ supply voltage. See Section 21.1 for more information on Port I/O operating modes and the electrical specifications chapter for detailed electrical specifications.

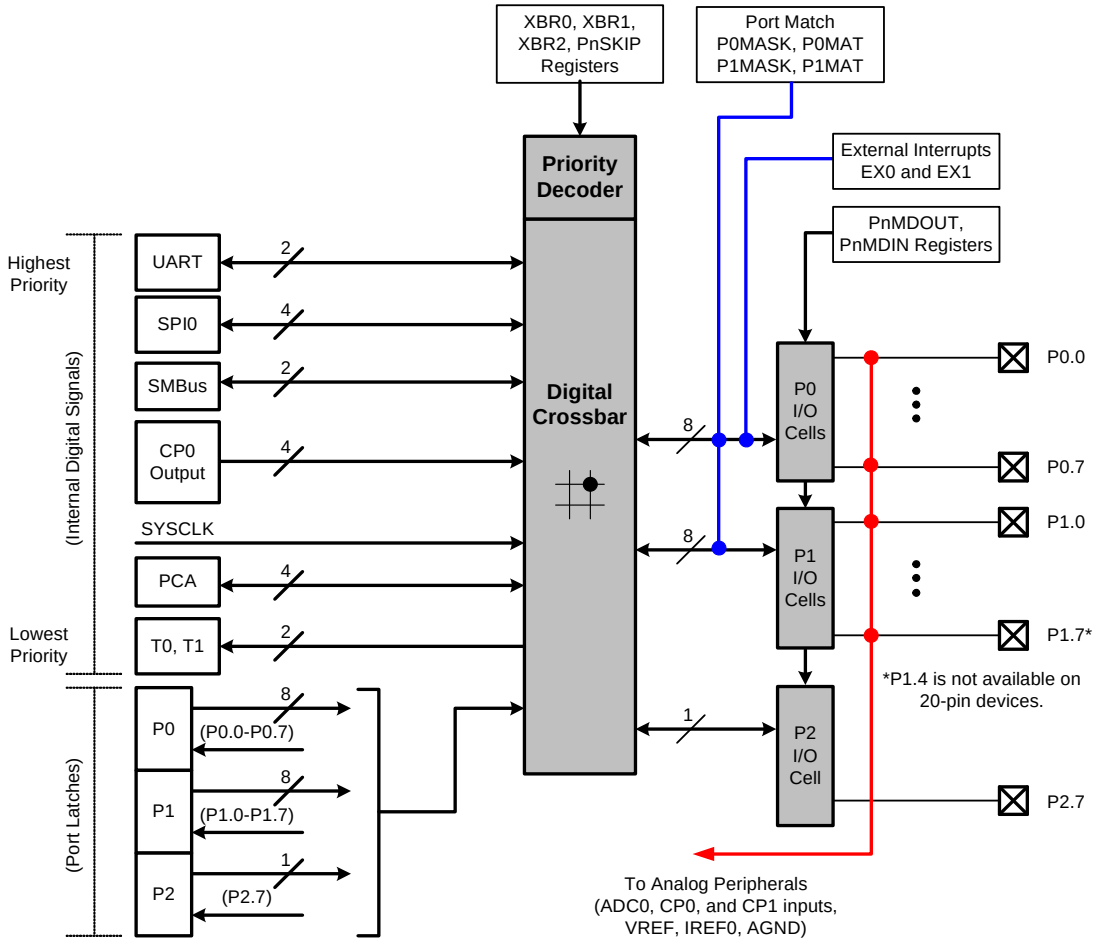



**Figure 21.1. Port I/O Functional Block Diagram**

## 21.1. Port I/O Modes of Operation

Port pins P0.0–P1.7 use the Port I/O cell shown in Figure 21.2. Each Port I/O cell can be configured by software for analog I/O or digital I/O using the PnMDIN registers. On reset, all Port I/O cells default to a digital high impedance state with weak pull-ups enabled.

### 21.1.1. Port Pins Configured for Analog I/O

Any pins to be used as Comparator or ADC input, oscillator input/output, or AGND, VREF, or Current Reference output should be configured for analog I/O (PnMDIN.n = 0). When a pin is configured for analog I/O, its weak pullup and digital receiver are disabled. In most cases, software should also disable the digital output drivers. Port pins configured for analog I/O will always read back a value of 0 regardless of the actual voltage on the pin.

Configuring pins as analog I/O saves power and isolates the Port pin from digital interference. Port pins configured as digital inputs may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

### 21.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external digital event capture functions, or as GPIO should be configured as digital I/O (PnMDIN.n = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

Push-pull outputs (PnMDOUT.n = 1) drive the Port pad to the VDD or GND supply rails based on the output logic value of the Port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the Port pad to GND when the output logic value is 0 and become high impedance inputs (both high and low drivers turned off) when the output logic value is 1.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the Port pad to the VDD supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to GND to minimize power consumption and may be globally disabled by setting WEAKPUD to 1. The user must ensure that digital I/O are always internally or externally pulled or driven to a valid logic state. Port pins configured for digital I/O always read back the logic state of the Port pad, regardless of the output logic value of the Port pin.

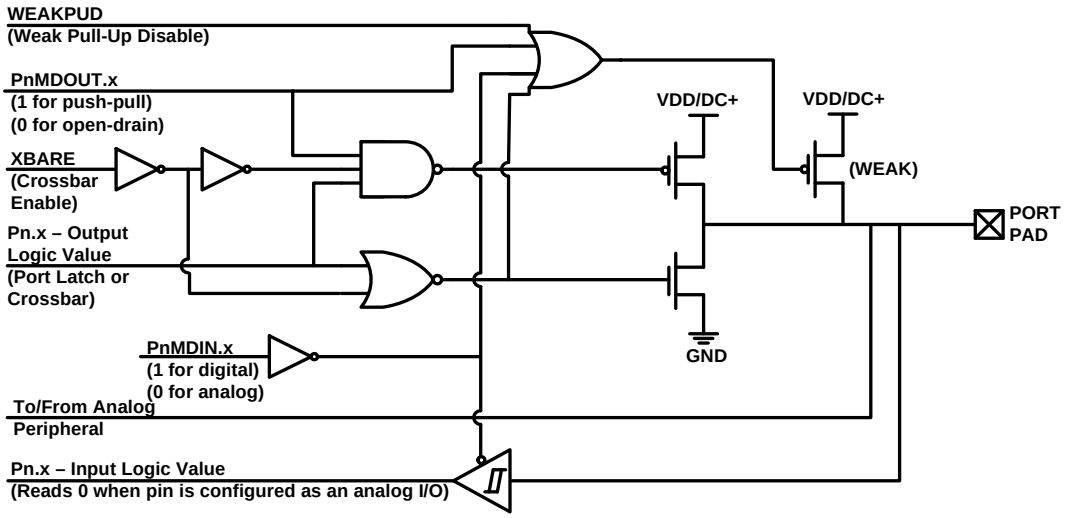



**Figure 21.2. Port I/O Cell Block Diagram**

### 21.1.3. Interfacing Port I/O to 5 V Logic

All Port I/O have internal ESD protection diodes to prevent the pin voltage from exceeding the $V_{DD}$ supply. The Port I/O pins are not 5V tolerant and require level translators to interface to 5V logic.

### 21.1.4. Increasing Port I/O Drive Strength

Port I/O output drivers support a high and low drive strength; the default is low drive strength. The drive strength of a Port I/O can be configured using the PnDRV registers. See Section “4. Electrical Characteristics” on page 48 for the difference in output drive strength between the two modes.

## 21.2. Assigning Port I/O Pins to Analog and Digital Functions

Port I/O pins P0.0–P1.7 can be assigned to various analog, digital, and external interrupt functions. The Port pins assuaged to analog functions should be configured for analog I/O and Port pins assuaged to digital or external interrupt functions should be configured for digital I/O.

### 21.2.1. Assigning Port I/O Pins to Analog Functions

Table 21.1 shows all available analog functions that need Port I/O assignments. **Port pins selected for these analog functions should have their digital drivers disabled (PnMDOUT.n = 0 and Port Latch = 1) and their corresponding bit in PnSKIP set to 1.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. Table 21.1 shows the potential mapping of Port I/O to each analog function.

**Table 21.1. Port I/O Assignment for Analog Functions**

| Analog Function | Potentially Assignable Port Pins | Registers used for Assignment |
|---|---|---|
| ADC Input | P0.1–P0.7, P1.2–P1.4 | ADC0MX, PnSKIP |
| Comparator0 Input | P1.0, P1.1 | CPT0MX, PnSKIP |
| Voltage Reference (VREF0) | P0.0 | REF0CN, PnSKIP |
| Analog Ground Reference (AGND) | P0.1 | REF0CN, PnSKIP |
| Current Reference (IREF0) | P0.7 | IREF0CN, PnSKIP |
| External Oscillator Input (XTAL1) | P0.2 | OSCXCN, PnSKIP |
| External Oscillator Output (XTAL2) | P0.3 | OSCXCN, PnSKIP |
| SmaRTClock Oscillator Input (XTAL3) | P1.6 | RTC0CN, PnSKIP |
| SmaRTClock Oscillator Output (XTAL4) | P1.7 | RTC0CN, PnSKIP |

### 21.2.2. Assigning Port I/O Pins to Digital Functions

Any Port pins not assigned to analog functions may be assigned to digital functions or used as GPIO. Most digital functions rely on the Crossbar for pin assignment; however, some digital functions bypass the Crossbar in a manner similar to the analog functions listed above. **Port pins used by these digital functions and any Port pins selected for use as GPIO should have their corresponding bit in PnSKIP set to 1.** Table 21.2 shows all available digital functions and the potential mapping of Port I/O to each digital function.

**Table 21.2. Port I/O Assignment for Digital Functions**

| Digital Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| UART0, SPI0, SMBus, CP0 Outputs, System Clock Output, PCA0, Timer0 and Timer1 External Inputs. | Any Port pin available for assignment by the Crossbar. This includes P0.0–P1.7 pins which have their PnSKIP bit set to 0.<br>**Note:** The Crossbar will always assign UART0 and SPI1 pins to fixed locations. | XBR0, XBR1, XBR2 |
| Any pin used for GPIO | P0.0–P1.7, P2.7 | P0SKIP, P1SKIP |

### 21.2.3. Assigning Port I/O Pins to External Digital Event Capture Functions

External digital event capture functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The digital event capture functions do not require dedicated pins and will function on both GPIO pins (PnSKIP = 1) and pins in use by the Crossbar (PnSKIP = 0). External digital even capture functions cannot be used on pins configured for analog I/O. Table 21.3 shows all available external digital event capture functions.

**Table 21.3. Port I/O Assignment for External Digital Event Capture Functions**

| Digital Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| External Interrupt 0 | P0.0–P0.7 | IT01CF |
| External Interrupt 1 | P0.0–P0.7 | IT01CF |
| Port Match | P0.0–P1.7 | P0MASK, P0MAT<br>P1MASK, P1MAT |

## 21.3. Priority Crossbar Decoder

The Priority Crossbar Decoder (Figure 21.3) 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 is always at pins 4 and 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 the VREF signal, external oscillator pins (XTAL1, XTAL2), the ADC's external conversion start signal (CNVSTR), EMIF control signals, and any selected ADC or Comparator inputs. The PnSKIP registers may also be used to skip pins to be used as GPIO. The Crossbar skips selected pins as if they were already assigned, and moves to the next unassigned pin. Figure 21.3 shows all the possible pins available to each peripheral. Figure 21.4 shows the Crossbar Decoder priority in an example configuration with no Port pins skipped. Figure 21.5 shows the same Crossbar example with pins P0.2, P0.3, P1.0, and P1.1 skipped.

| | P0 | | | | | | | | P1 | | | | | | | | P2 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SF Signals | VREF | AGND | XTAL1 | XTAL2 | | | CNVSTR | IREF0 | CP0+ | CP0- | | | | | XTAL3 | XTAL4 | P2.0 - P2.6 not available on C8051F98x-C8051F99x devices | | | | | | | C2D |
| PIN I/O | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| TX0 | | | | | | | | | | | | | | | | | | | | | | | | Pin not available for Crossbar functions |
| RX0 | | | | | | | | | | | | | | | | | | | | | | | | |
| SCK | | | | | | | | | | | | | | | | | | | | | | | | |
| MISO | | | | | | | | | | | | | | | | | | | | | | | | |
| MOSI | | | | | | | | | | | | | | | | | | | | | | | | |
| NSS* | | | | | | | | | | | | | | | | | | | | | | | | |
| SDA | | | | | | | | | | | | | | | | | | | | | | | | |
| SCL | | | | | | | | | | | | | | | | | | | | | | | | |
| CP0 | | | | | | | | | | | | | | | | | | | | | | | | |
| CP0A | | | | | | | | | | | | | | | | | | | | | | | | |
| SYSCLK | | | | | | | | | | | | | | | | | | | | | | | | |
| CEX0 | | | | | | | | | | | | | | | | | | | | | | | | |
| CEX1 | | | | | | | | | | | | | | | | | | | | | | | | |
| CEX2 | | | | | | | | | | | | | | | | | | | | | | | | |
| ECI | | | | | | | | | | | | | | | | | | | | | | | | |
| T0 | | | | | | | | | | | | | | | | | | | | | | | | |
| T1 | | | | | | | | | | | | | | | | | | | | | | | | |

Port pin potentially available 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.

*NSS is only pinned out in 4-wire SPI mode

**Figure 21.3. Peripheral Availability on Port I/O Pins**

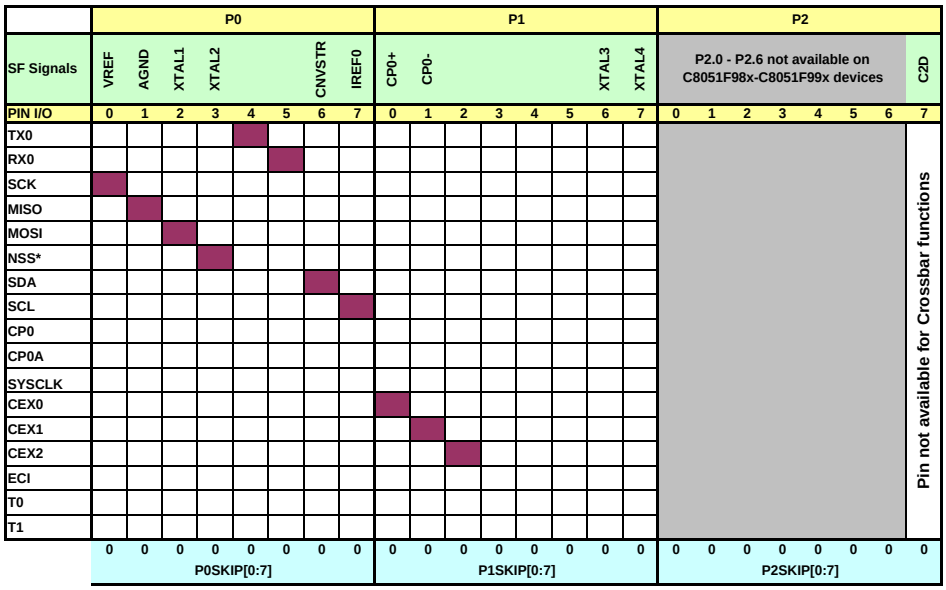



**Figure 21.4. Crossbar Priority Decoder in Example Configuration (No Pins Skipped)**

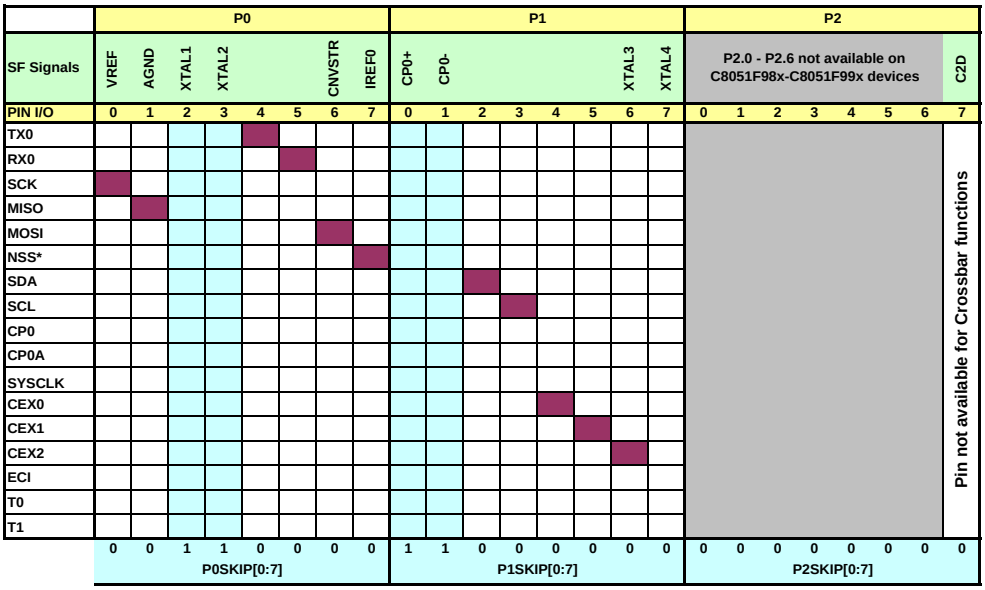



**Figure 21.5. Crossbar Priority Decoder in Example Configuration (4 Pins Skipped)**

Registers XBR0, XBR1, and XBR2 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 either 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 after the prioritized functions have been assigned.

**Notes:**

1. The Crossbar must be enabled (XBARE = 1) before any Port pin is used as a digital output. Port output drivers are disabled while the Crossbar is disabled.
2. When SMBus is selected in the Crossbar, the pins associated with SDA and SCL will automatically be forced into open-drain output mode regardless of the PnMDOUT setting.
3. SPI0 can be operated in either 3-wire or 4-wire modes, depending on the state of the NSSMD1-NSSMD0 bits in register SPI0CN. The NSS signal is only routed to a Port pin when 4-wire mode is selected. When SPI0 is selected in the Crossbar, the SPI0 mode (3-wire or 4-wire) will affect the pinout of all digital functions lower in priority than SPI0.
4. For given XBRn, PnSKIP, and SPInCN register settings, one can determine the I/O pin-out of the device using Figure 21.3.
5. On 20-pin devices, P1.4 should be skipped in the Crossbar. It is not available as a device pin.

## SFR Definition 21.1. XBR0: Port I/O Crossbar Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | CP0AE | CP0E | SYSCKE | SMB0E | SPI0E | URT0E |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE1

| Bit | Name | Function |
|---|---|---|
| 7:6 | Unused | Read = 00b. Write = Don't Care. |
| 5 | CP0AE | **Comparator0 Asynchronous Output Enable.**<br>0: Asynchronous CP0 output unavailable at Port pin.<br>1: Asynchronous CP0 output routed to Port pin. |
| 4 | CP0E | **Comparator0 Output Enable.**<br>0: CP1 output unavailable at Port pin.<br>1: CP1 output routed to Port pin. |
| 3 | SYSCKE | **$\overline{\text{SYSCLK}}$ Output Enable.**<br>0: $\overline{\text{SYSCLK}}$ output unavailable at Port pin.<br>1: $\overline{\text{SYSCLK}}$ output routed to Port pin. |
| 2 | SMB0E | **SMBus I/O Enable.**<br>0: SMBus I/O unavailable at Port pin.<br>1: SDA and SCL routed to Port pins. |
| 1 | SPI0E | **SPI0 I/O Enable.**<br>0: SPI0 I/O unavailable at Port pin.<br>1: SCK, MISO, and MOSI (for SPI0) routed to Port pins.<br>NSS (for SPI0) routed to Port pin only if SPI0 is configured to 4-wire mode. |
| 0 | URT0E | **UART0 Output Enable.**<br>0: UART I/O unavailable at Port pin.<br>1: TX0 and RX0 routed to Port pins P0.4 and P0.5. |
| **Note:** SPI0 can be assigned either 3 or 4 Port I/O pins. | | |

## SFR Definition 21.2. XBR1: Port I/O Crossbar Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | | | T1E | T0E | ECIE | PCA0ME[2:0] | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xE2

| Bit | Name | Function |
|---|---|---|
| 7:6 | Unused | Read = 00b; Write = Don't Care. |
| 5 | T1E | **Timer1 Input Enable.**<br>0: T1 input unavailable at Port pin.<br>1: T1 input routed to Port pin. |
| 4 | T0E | **Timer0 Input Enable.**<br>0: T0 input unavailable at Port pin.<br>1: T0 input routed to Port pin. |
| 3 | ECIE | **PCA0 External Counter Input (ECI) Enable.**<br>0: PCA0 external counter input unavailable at Port pin.<br>1: PCA0 external counter input routed to Port pin. |
| 2:0 | PCA0ME | **PCA0 Module I/O Enable.**<br>000: All PCA0 I/O unavailable at Port pin.<br>001: CEX0 routed to Port pin.<br>010: CEX0, CEX1 routed to Port pins.<br>011: CEX0, CEX1, CEX2 routed to Port pins.<br>100: Reserved.<br>101: Reserved.<br>110: Reserved.<br>111: Reserved. |

## SFR Definition 21.3. XBR2: Port I/O Crossbar Register 2

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

SFR Page = 0x0; SFR Address = 0xE3

| Bit | Name | Function |
|---|---|---|
| 7 | WEAKPUD | **Port I/O Weak Pullup Disable.**<br>0: Weak Pullups enabled (except for Port I/O pins configured for analog mode).<br>1: Weak Pullups disabled. |
| 6 | XBARE | **Crossbar Enable.**<br>0: Crossbar disabled.<br>1: Crossbar enabled. |
| 5:0 | Unused | Read = 000000b; Write = Don't Care. |
| **Note:** The Crossbar must be enabled (XBARE = 1) to use any Port pin as a digital output. | | |

## 21.4. Port Match

Port match functionality allows system events to be triggered by a logic value change on P0 or P1. A software controlled value stored in the PnMAT registers specifies the expected or normal logic values of P0 and P1. A Port mismatch event occurs if the logic levels of the Port's input pins no longer match the software controlled value. This allows Software to be notified if a certain change or pattern occurs on P0 or P1 input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which P0 and P1 pins should be compared against the PnMAT registers. A Port mismatch event is generated if (P0 & P0MASK) does not equal (PnMAT & P0MASK) or if (P1 & P1MASK) does not equal (PnMAT & P1MASK).

A Port mismatch event may be used to generate an interrupt or wake the device from a low power mode. See Section "13. Interrupt Handler" on page 138 and Section "15. Power Management" on page 162 for more details on interrupt and wake-up sources.

### SFR Definition 21.4. P0MASK: Port0 Mask Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MASK[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xC7

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MASK[7:0] | **Port0 Mask Value.**<br>Selects the P0 pins to be compared with the corresponding bits in P0MAT.<br>0: P0.n pin pad logic value is ignored and cannot cause a Port Mismatch event.<br>1: P0.n pin pad logic value is compared to P0MAT.n. |

### SFR Definition 21.5. P0MAT: Port0 Match Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page= 0x0; SFR Address = 0xD7

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MAT[7:0] | **Port 0 Match Value.**<br>Match comparison value used on Port 0 for bits in P0MASK which are set to 1.<br>0: P0.n pin logic value is compared with logic LOW.<br>1: P0.n pin logic value is compared with logic HIGH. |

## SFR Definition 21.6. P1MASK: Port1 Mask Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MASK[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xBF

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MASK[7:0] | **Port 1 Mask Value.**<br>Selects P1 pins to be compared to the corresponding bits in P1MAT.<br>0: P1.n pin logic value is ignored and cannot cause a Port Mismatch event.<br>1: P1.n pin logic value is compared to P1MAT.n. |

## SFR Definition 21.7. P1MAT: Port1 Match Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xCF

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MAT[7:0] | **Port 1 Match Value.**<br>Match comparison value used on Port 1 for bits in P1MASK which are set to 1.<br>0: P1.n pin logic value is compared with logic LOW.<br>1: P1.n pin logic value is compared with logic HIGH. |

## 21.5. Special Function Registers for Accessing and Configuring Port I/O

All Port I/O are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. The read-modify-write instructions when operating on a Port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SETB, when the destination is an individual bit in a Port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

Each Port has a corresponding PnSKIP register which allows its individual Port pins to be assigned to digital functions or skipped by the Crossbar. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to 1.

The Port input mode of the I/O pins is defined using the Port Input Mode registers (PnMDIN). Each Port cell can be configured for analog or digital I/O. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is P2.7, which can only be used for digital I/O.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). Each Port Output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the PnMDOUT settings.

The drive strength of the output drivers are controlled by the Port Drive Strength (PnDRV) registers. The default is low drive strength. See Section “4. Electrical Characteristics” on page 48 for the difference in output drive strength between the two modes.

## SFR Definition 21.8. P0: Port0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = All; SFR Address = 0x80; Bit-Addressable

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:0 | P0[7:0] | **Port 0 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P0.n Port pin is logic LOW.<br>1: P0.n Port pin is logic HIGH. |

## SFR Definition 21.9. P0SKIP: Port0 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0SKIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page= 0x0; SFR Address = 0xD4

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0SKIP[7:0] | **Port 0 Crossbar Skip Enable Bits.**<br>These bits select Port 0 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar.<br>0: Corresponding P0.n pin is not skipped by the Crossbar.<br>1: Corresponding P0.n pin is skipped by the Crossbar. |

## SFR Definition 21.10. P0MDIN: Port0 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDIN[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page= 0x0; SFR Address = 0xF1

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDIN[7:0] | **Analog Configuration Bits for P0.7–P0.0 (respectively).**<br>Port pins configured for analog mode have their weak pullup, and digital receiver disabled. The digital driver is not explicitly disabled.<br>0: Corresponding P0.n pin is configured for analog mode.<br>1: Corresponding P0.n pin is not configured for analog mode. |

## SFR Definition 21.11. P0MDOUT: Port0 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDOUT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA4

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDOUT[7:0] | **Output Configuration Bits for P0.7–P0.0 (respectively).**<br>These bits control the digital driver even when the corresponding bit in register P0MDIN is logic 0.<br>0: Corresponding P0.n Output is open-drain.<br>1: Corresponding P0.n Output is push-pull. |

## SFR Definition 21.12. P0DRV: Port0 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0DRV[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x99

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0DRV[7:0] | **Drive Strength Configuration Bits for P0.7–P0.0 (respectively).**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: Corresponding P0.n Output has low output drive strength.<br>1: Corresponding P0.n Output has high output drive strength. |

## SFR Definition 21.13. P1: Port1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = All; SFR Address = 0x90; Bit-Addressable

| Bit | Name | Description | Write | Read |
|---|---|---|---|---|
| 7:0 | P1[6:0] | **Port 1 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P1.n Port pin is logic LOW.<br>1: P1.n Port pin is logic HIGH. |

## SFR Definition 21.14. P1SKIP: Port1 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1SKIP[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD5

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1SKIP[6:0] | **Port 1 Crossbar Skip Enable Bits.**<br>These bits select Port 1 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar.<br>0: Corresponding P1.n pin is not skipped by the Crossbar.<br>1: Corresponding P1.n pin is skipped by the Crossbar. |

## SFR Definition 21.15. P1MDIN: Port1 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDIN[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xF2

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDIN[7:0] | **Analog Configuration Bits for P1.7–P1.0 (respectively).**<br>Port pins configured for analog mode have their weak pullup and digital receiver disabled. The digital driver is not explicitly disabled.<br>0: Corresponding P1.n pin is configured for analog mode.<br>1: Corresponding P1.n pin is not configured for analog mode. |

## SFR Definition 21.16. P1MDOUT: Port1 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDOUT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA5

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDOUT[7:0] | **Output Configuration Bits for P1.7–P1.0 (respectively).**<br>These bits control the digital driver even when the corresponding bit in register P1MDIN is logic 0.<br>0: Corresponding P1.n Output is open-drain.<br>1: Corresponding P1.n Output is push-pull. |

## SFR Definition 21.17. P1DRV: Port1 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1DRV[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x9B

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1DRV[7:0] | **Drive Strength Configuration Bits for P1.7–P1.0 (respectively).**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: Corresponding P1.n Output has low output drive strength.<br>1: Corresponding P1.n Output has high output drive strength. |

## SFR Definition 21.18. P2: Port2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2 | | | | | | | |
| Type | R/W | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = All; SFR Address = 0xA0; Bit-Addressable

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7 | P2 | **Port 2 Data.**<br>Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O. | 0: Set output latch to logic LOW.<br>1: Set output latch to logic HIGH. | 0: P2.7 Port pin is logic LOW.<br>1: P2.7 Port pin is logic HIGH. |
| 6:0 | Unused | Read = 0000000b; Write = Don't Care. | | |

## SFR Definition 21.19. P2MDOUT: Port2 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2MDOUT | | | | | | | |
| Type | R/W | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xA6

| Bit | Name | Function |
|---|---|---|
| 7 | P2MDOUT | **Output Configuration Bits for P2.7.**<br>These bits control the digital driver.<br>0: P2.7 Output is open-drain.<br>1: P2.7 Output is push-pull. |
| 6:0 | Unused | Read = 0000000b; Write = Don't Care. |

## SFR Definition 21.20. P2DRV: Port2 Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P2DRV | | | | | | | |
| Type | R/W | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0xF; SFR Address = 0x9D

| Bit | Name | Function |
|---|---|---|
| 7 | P2DRV | **Drive Strength Configuration Bits for P2.7.**<br>Configures digital I/O Port cells to high or low output drive strength.<br>0: P2.7 Output has low output drive strength.<br>1: P2.7 Output has high output drive strength. |
| 6:0 | Unused | Read = 0000000b; Write = Don't Care. |

## 22. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compatible with the System Management Bus Specification, version 1.1, and compatible with the $I^2C$ serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/20th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. The SMBus peripheral can be fully driven by software (i.e. software accepts/rejects slave addresses, and generates ACKs), or hardware slave address recognition and automatic ACK generation can be enabled to minimize software overhead. A block diagram of the SMBus peripheral and the associated SFRs is shown in Figure 22.1.

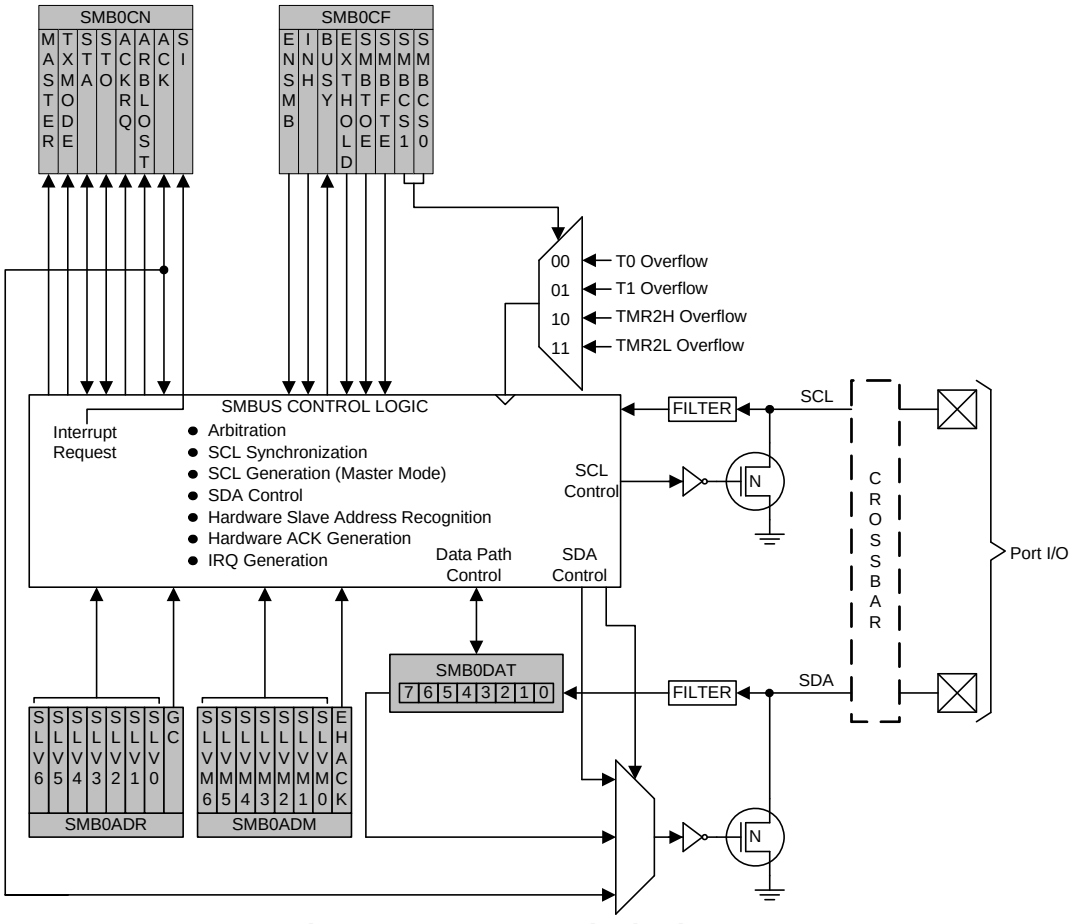



**Figure 22.1. SMBus Block Diagram**

## 22.1. Supporting Documents

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

- The $I^2C$-Bus and How to Use It (including specifications), Philips Semiconductor.
- The $I^2C$-Bus Specification—Version 2.0, Philips Semiconductor.
- System Management Bus Specification—Version 1.1, SBS Implementers Forum.

## 22.2. SMBus Configuration

Figure 22.2 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.

**Note:** The port pins on C8051F99x-C8051F98x devices are not 5 V tolerant, therefore, the device may only be used in SMBus networks where the supply voltage does not exceed $V_{DD}$.

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.

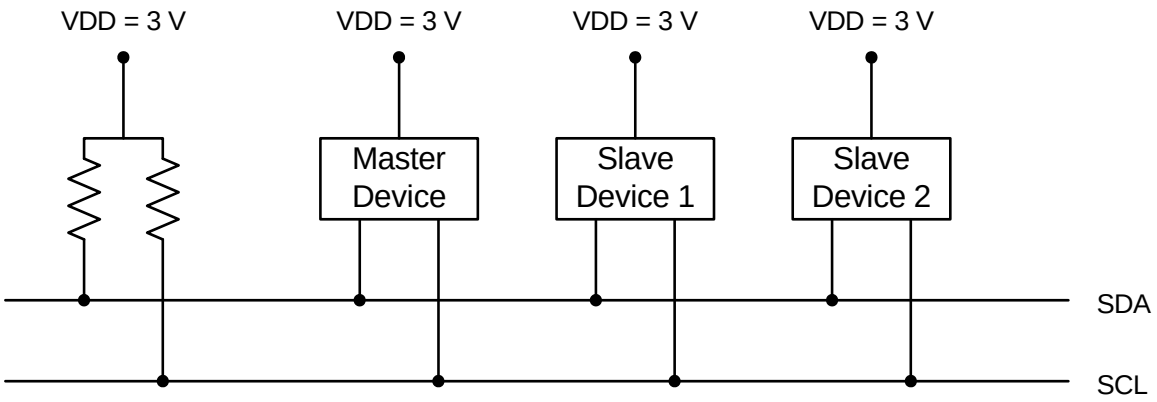



**Figure 22.2. Typical SMBus Configuration**

## 22.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. Note that it is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 22.3). 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 22.3 illustrates a typical SMBus transaction.

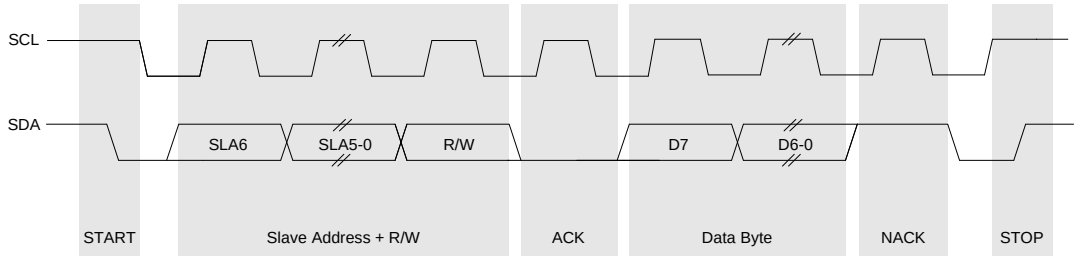



**Figure 22.3. SMBus Transaction**

### 22.3.1. Transmitter vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

### 22.3.2. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see Section “22.3.5. SCL High (SMBus Free) Timeout” on page 238). 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.

### 22.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to $I^2C$, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

### 22.3.4. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

When the SMBTOE bit in SMB0CF is set, Timer 3 is used to detect SCL low timeouts. Timer 3 is forced to reload when SCL is high, and allowed to count when SCL is low. With Timer 3 enabled and configured to overflow after 25 ms (and SMBTOE set), the Timer 3 interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

### 22.3.5. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMBFTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. Note that a clock source is required for free timeout detection, even in a slave-only implementation.

## 22.4. Using the SMBus

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgement is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e. sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e., receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgement is enabled, these interrupts are always generated after the ACK cycle. See Section 22.5 for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. The SMB0CN register is described in Section 22.4.2; Table 22.5 provides a quick SMB0CN decoding reference.

### 22.4.1. 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 22.1. SMBus Clock Source Selection**

| SMBCS1 | SMBCS0 | SMBus Clock Source |
|---|---|---|
| 0 | 0 | Timer 0 Overflow |
| 0 | 1 | Timer 1 Overflow |
| 1 | 0 | Timer 2 High Byte Overflow |
| 1 | 1 | Timer 2 Low Byte Overflow |

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 22.1. 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 “25. Timers” on page 278.

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

**Equation 22.1. 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 22.1. 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 22.2.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

**Equation 22.2. Typical SMBus Bit Rate**

Figure 22.4 shows the typical SCL generation described by Equation 22.2. 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 22.1.

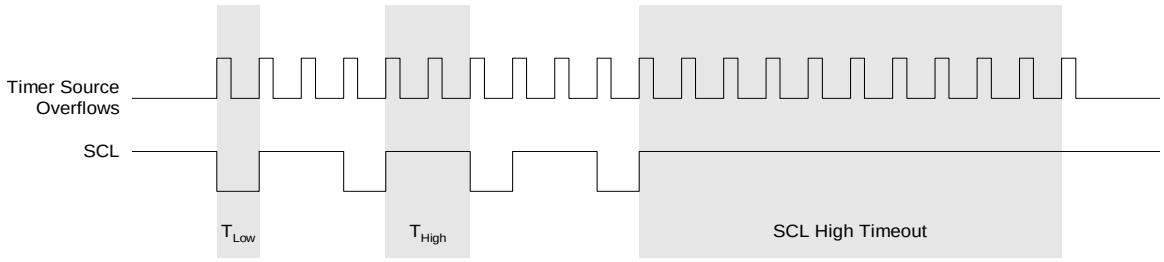



**Figure 22.4. Typical SMBus SCL Generation**

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 22.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

**Table 22.2. Minimum SDA Setup and Hold Times**

| EXTHOLD | Minimum SDA Setup Time | Minimum SDA Hold Time |
|---|---|---|
| 0 | $T_{low}$ – 4 system clocks<br>or<br>1 system clock + s/w delay* | 3 system clocks |
| 1 | 11 system clocks | 12 system clocks |
| ***Note:** Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgement, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero. | | |

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “22.3.4. SCL Low Timeout” on page 238). 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 22.4).

## SFR Definition 22.1. SMB0CF: SMBus Clock/Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ENSMB | INH | BUSY | EXTHOLD | SMBTOE | SMBFTE | SMBCS[1:0] | |
| Type | R/W | R/W | R | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC1

| Bit | Name | Function |
|---|---|---|
| 7 | ENSMB | **SMBus Enable.**<br>This bit enables the SMBus interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins. |
| 6 | INH | **SMBus Slave Inhibit.**<br>When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected. |
| 5 | BUSY | **SMBus Busy Indicator.**<br>This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed. |
| 4 | EXTHOLD | **SMBus Setup and Hold Time Extension Enable.**<br>This bit controls the SDA setup and hold times according to Table 22.2.<br>0: SDA Extended Setup and Hold Times disabled.<br>1: SDA Extended Setup and Hold Times enabled. |
| 3 | SMBTOE | **SMBus SCL Timeout Detection Enable.**<br>This bit enables SCL low timeout detection. If set to logic 1, the SMBus forces Timer 3 to reload while SCL is high and allows Timer 3 to count when SCL goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication. |
| 2 | SMBFTE | **SMBus Free Timeout Detection Enable.**<br>When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods. |
| 1:0 | SMBCS[1:0] | **SMBus Clock Source Selection.**<br>These two bits select the SMBus clock source, which is used to generate the SMBus bit rate. The selected device should be configured according to Equation 22.1.<br>00: Timer 0 Overflow<br>01: Timer 1 Overflow<br>10:Timer 2 High Byte Overflow<br>11: Timer 2 Low Byte Overflow |

### 22.4.2. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information (see SFR Definition 22.2). The higher four bits of SMB0CN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost; see Table 22.3 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.

#### 22.4.2.1.Software ACK Generation

When the EHACK bit in register SMB0ADM is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during the last ACK cycle. ACKRQ is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQ is set, software should write the desired outgoing value to the ACK bit before clearing SI. A NACK will be generated if software does not write the ACK bit before clearing SI. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however SCL will remain low until SI is cleared. If a received slave address is not acknowledged, further slave events will be ignored until the next START is detected.

#### 22.4.2.2.Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. More detail about automatic slave address recognition can be found in Section 22.4.3. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 22.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 22.5 for SMBus status decoding using the SMB0CN register.

## SFR Definition 22.2. SMB0CN: SMBus Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | MASTER | TXMODE | STA | STO | ACKRQ | ARBLOST | ACK | SI |
| Type | R | R | R/W | R/W | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC0; Bit-Addressable

| Bit | Name | Description | Read | Write |
|---|---|---|---|---|
| 7 | MASTER | **SMBus Master/Slave Indicator.** This read-only bit indicates when the SMBus is operating as a master. | 0: SMBus operating in slave mode.<br>1: SMBus operating in master mode. | N/A |
| 6 | TXMODE | **SMBus Transmit Mode Indicator.** This read-only bit indicates when the SMBus is operating as a transmitter. | 0: SMBus in Receiver Mode.<br>1: SMBus in Transmitter Mode. | N/A |
| 5 | STA | **SMBus Start Flag.** | 0: No Start or repeated Start detected.<br>1: Start or repeated Start detected. | 0: No Start generated.<br>1: When Configured as a Master, initiates a START or repeated START. |
| 4 | STO | **SMBus Stop Flag.** | 0: No Stop condition detected.<br>1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode). | 0: No STOP condition is transmitted.<br>1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle.<br>Cleared by Hardware. |
| 3 | ACKRQ | **SMBus Acknowledge Request.** | 0: No Ack requested<br>1: ACK requested | N/A |
| 2 | ARBLOST | **SMBus Arbitration Lost Indicator.** | 0: No arbitration error.<br>1: Arbitration Lost | N/A |
| 1 | ACK | **SMBus Acknowledge.** | 0: NACK received.<br>1: ACK received. | 0: Send NACK<br>1: Send ACK |
| 0 | SI | **SMBus Interrupt Flag.**<br>This bit is set by hardware under the conditions listed in Table 15.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled. | **0: No interrupt pending**<br>1: Interrupt Pending | 0: Clear interrupt, and initiate next state machine event.<br>1: Force interrupt. |

**Table 22.3. Sources for Hardware Changes to SMB0CN**

| Bit | Set by Hardware When: | Cleared by Hardware When: |
|---|---|---|
| MASTER | ■ A START is generated. | ■ A STOP is generated.<br>■ Arbitration is lost. |
| TXMODE | ■ START is generated.<br>■ SMB0DAT is written before the start of an SMBus frame. | ■ A START is detected.<br>■ Arbitration is lost.<br>■ SMB0DAT is not written before the start of an SMBus frame. |
| STA | ■ A START followed by an address byte is received. | ■ Must be cleared by software. |
| STO | ■ A STOP is detected while addressed as a slave.<br>■ Arbitration is lost due to a detected STOP. | ■ A pending STOP is generated. |
| ACKRQ | ■ A byte has been received and an ACK response value is needed (only when hardware ACK is not enabled). | ■ After each ACK cycle. |
| ARBLOST | ■ A repeated START is detected as a MASTER when STA is low (unwanted repeated START).<br>■ SCL is sensed low while attempting to generate a STOP or repeated START condition.<br>■ SDA is sensed low while transmitting a 1 (excluding ACK bits). | ■ Each time SI is cleared. |
| ACK | ■ The incoming ACK value is low (ACKNOWLEDGE). | ■ The incoming ACK value is high (NOT ACKNOWLEDGE). |
| SI | ■ A START has been generated.<br>■ Lost arbitration.<br>■ A byte has been transmitted and an ACK/NACK received.<br>■ A byte has been received.<br>■ A START or repeated START followed by a slave address + R/W has been received.<br>■ A STOP has been received. | ■ Must be cleared by software. |

### 22.4.3. Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave). More detail on automatic hardware ACK generation can be found in Section 22.4.2.2.

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register (SFR Definition 22.3) and the SMBus Slave Address Mask register (SFR Definition 22.4). A single address or range of addresses (including the General Call Address 0x00) can be specified using these two registers. The most-significant seven bits of the two registers are used to define which addresses will be ACKed. A 1 in bit positions of the slave address mask SLVM[6:0] enable a comparison between the received slave address and the hardware's slave address SLV[6:0] for those bits. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address. Additionally, if the GC bit in register SMB0ADR is set to 1, hardware will recognize the General Call Address (0x00). Table 22.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions.

**Table 22.4. Hardware Address Recognition Examples (EHACK = 1)**

| Hardware Slave Address SLV[6:0] | Slave Address Mask SLVM[6:0] | GC bit | Slave Addresses Recognized by Hardware |
|---|---|---|---|
| 0x34 | 0x7F | 0 | 0x34 |
| 0x34 | 0x7F | 1 | 0x34, 0x00 (General Call) |
| 0x34 | 0x7E | 0 | 0x34, 0x35 |
| 0x34 | 0x7E | 1 | 0x34, 0x35, 0x00 (General Call) |
| 0x70 | 0x73 | 0 | 0x70, 0x74, 0x78, 0x7C |

## SFR Definition 22.3. SMB0ADR: SMBus Slave Address

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

SFR Page = 0x0; SFR Address = 0xF4

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLV[6:0] | **SMBus Hardware Slave Address.**<br>Defines the SMBus Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM[6:0] are checked against the incoming address. This allows multiple addresses to be recognized. |
| 0 | GC | **General Call Address Enable.**<br>When hardware address recognition is enabled (EHACK = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware.<br>0: General Call Address is ignored.<br>1: General Call Address is recognized. |

## SFR Definition 22.4. SMB0ADM: SMBus Slave Address Mask

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

SFR Page = 0x0; SFR Address = 0xF5

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLVM[6:0] | **SMBus Slave Address Mask.**<br>Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM[6:0] enables comparisons with the corresponding bit in SLV[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address). |
| 0 | EHACK | **Hardware Acknowledge Enable.**<br>Enables hardware acknowledgement of slave address and received data bytes.<br>0: Firmware must manually acknowledge all incoming address and data bytes.<br>1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled. |

### 22.4.4. 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 22.5. SMB0DAT: SMBus Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SMB0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC2

| Bit | Name | Function |
|---|---|---|
| 7:0 | SMB0DAT[7:0] | **SMBus Data.**<br>The SMB0DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can read from or write to this register whenever the SI serial interrupt flag (SMB0CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register. |

## 22.5. SMBus Transfer Modes

The SMBus interface may be configured to operate as master and/or slave. At any particular time, it will be operating in one of the following four modes: Master Transmitter, Master Receiver, Slave Transmitter, or Slave Receiver. The SMBus interface enters Master Mode any time a START is generated, and remains in Master Mode until it loses an arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames. Note that the position of the ACK interrupt when operating as a receiver depends on whether hardware ACK generation is enabled. As a receiver, the interrupt for an ACK occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled. As a transmitter, interrupts occur **after** the ACK, regardless of whether hardware ACK generation is enabled or not.

### 22.5.1. Write Sequence (Master)

During a write sequence, an SMBus master writes data to a slave device. The master in this transfer will be a transmitter during the address byte, and a transmitter during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 0 (WRITE). The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. The transfer is ended when the STO bit is set and a STOP is generated. Note that the interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 22.5 shows a typical master write sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that all of the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

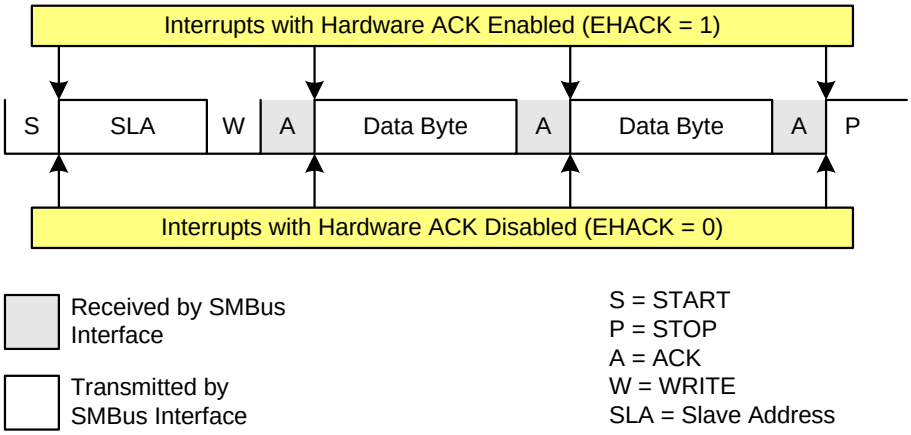



**Figure 22.5. Typical Master Write Sequence**

### 22.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

Writing a 1 to the ACK bit generates an ACK; writing a 0 generates a NACK. Software should write a 0 to the ACK bit for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 22.6 shows a typical master read sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

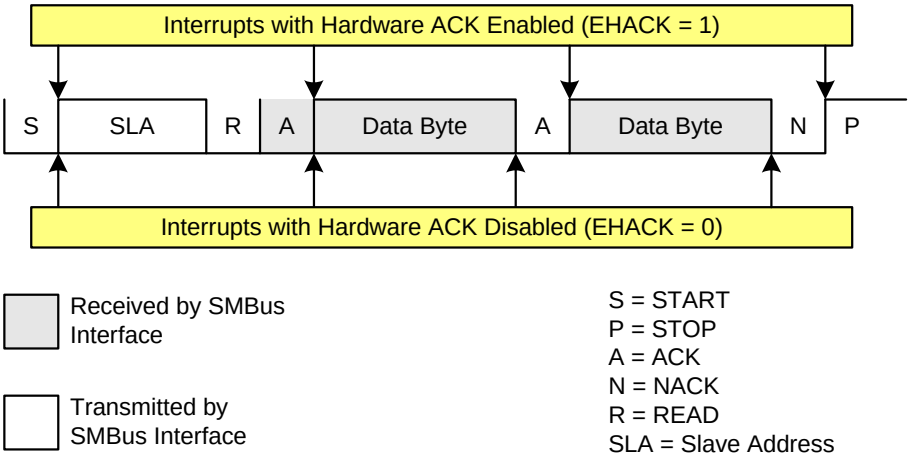



**Figure 22.6. Typical Master Read Sequence**

### 22.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

The interface exits Slave Receiver Mode after receiving a STOP. Note that the interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 22.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

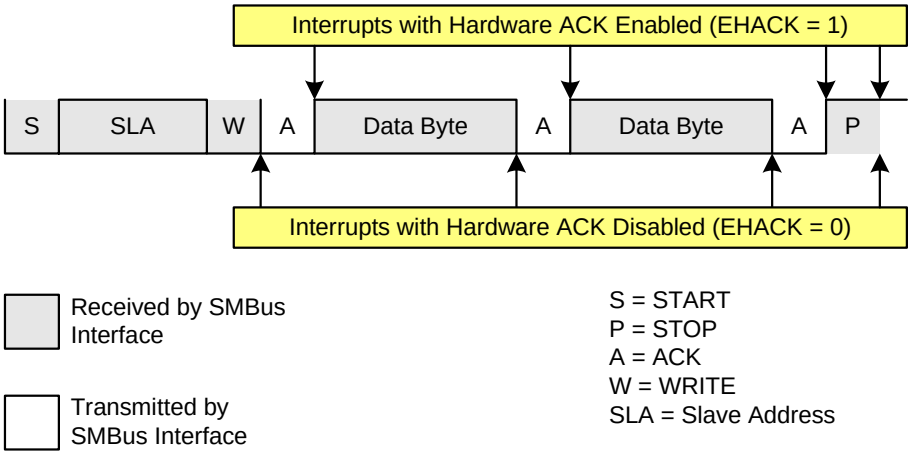



**Figure 22.7. Typical Slave Write Sequence**

### 22.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters Slave Transmitter Mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (Note: an error condition may be generated if SMB0DAT is written following a received NACK while in Slave Transmitter Mode). The interface exits Slave Transmitter Mode after receiving a STOP. Note that the interface will switch to Slave Receiver Mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 22.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the ‘data byte transferred’ interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

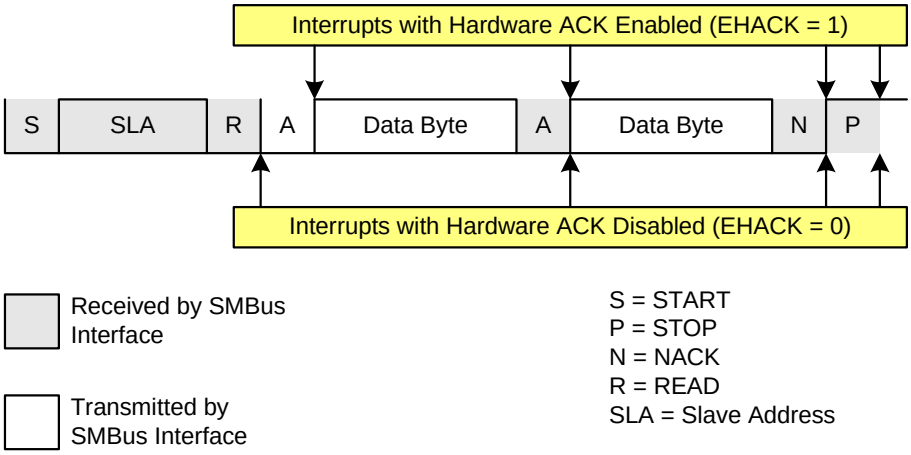



**Figure 22.8. Typical Slave Read Sequence**

## 22.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. The appropriate actions to take in response to an SMBus event depend on whether hardware slave address recognition and ACK generation is enabled or disabled. Table 22.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 22.6 describes the typical actions when hardware slave address recognition and ACK generation is enabled. In the tables, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

**Table 22.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Master Transmitter | 1110 | 0 | 0 | X | A master START was generated. | Load slave address + R/W into SMB0DAT. | 0 | 0 | X | 1100 |
| | 1100 | 0 | 0 | 0 | A master data or address byte was transmitted; NACK received. | Set STA to restart transfer. | 1 | 0 | X | 1110 |
| | | | | | | Abort transfer. | 0 | 1 | X | — |
| | | 0 | 0 | 1 | A master data or address byte was transmitted; ACK received. | Load next data byte into SMB0DAT. | 0 | 0 | X | 1100 |
| | | | | | | End transfer with STOP. | 0 | 1 | X | — |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X | — |
| | | | | | | Send repeated START. | 1 | 0 | X | 1110 |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). | 0 | 0 | X | 1000 |
| Master Receiver | 1000 | 1 | 0 | X | A master data byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 | 1000 |
| | | | | | | Send NACK to indicate last byte, and send STOP. | 0 | 1 | 0 | — |
| | | | | | | Send NACK to indicate last byte, and send STOP followed by START. | 1 | 1 | 0 | 1110 |
| | | | | | | Send ACK followed by repeated START. | 1 | 0 | 1 | 1110 |
| | | | | | | Send NACK to indicate last byte, and send repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 1 | 1100 |
| | | | | | | Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 0 | 1100 |

**Table 22.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) (Continued)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Slave Transmitter | 0100 | 0 | 0 | 0 | A slave byte was transmitted; NACK received. | No action required (expecting STOP condition). | 0 | 0 | X | 0001 |
| | | 0 | 0 | 1 | A slave byte was transmitted; ACK received. | Load SMB0DAT with next data byte to transmit. | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | A Slave byte was transmitted; error detected. | No action required (expecting Master to end transfer). | 0 | 0 | X | 0001 |
| | 0101 | 0 | X | X | An illegal STOP or bus error was detected while a Slave Transmission was in progress. | Clear STO. | 0 | 0 | X | — |
| Slave Receiver | 0010 | 1 | 0 | X | A slave address + R/W was received; ACK requested. | If Write, Acknowledge received address | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte; ACK received address | 0 | 0 | 1 | 0100 |
| | | | | | | NACK received address. | 0 | 0 | 0 | — |
| | | 1 | 1 | X | Lost arbitration as master; slave address + R/W received; ACK requested. | If Write, Acknowledge received address | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte; ACK received address | 0 | 0 | 1 | 0100 |
| | | | | | | NACK received address. | 0 | 0 | 0 | — |
| | | | | | | Reschedule failed transfer; NACK received address. | 1 | 0 | 0 | 1110 |
| | 0001 | 0 | 0 | X | A STOP was detected while addressed as a Slave Transmitter or Slave Receiver. | Clear STO. | 0 | 0 | X | — |
| | | 1 | 1 | X | Lost arbitration while attempting a STOP. | No action required (transfer complete/aborted). | 0 | 0 | 0 | — |
| | 0000 | 1 | 0 | X | A slave byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 | 0000 |
| | | | | | | NACK received byte. | 0 | 0 | 0 | — |
| Bus Error Condition | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0001 | 0 | 1 | X | Lost arbitration due to a detected STOP. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0000 | 1 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | 0 | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | 0 | 1110 |

**Table 22.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Master Transmitter | 1110 | 0 | 0 | X | A master START was generated. | Load slave address + R/W into SMB0DAT. | 0 | 0 | X | 1100 |
| | 1100 | 0 | 0 | 0 | A master data or address byte was transmitted; NACK received. | Set STA to restart transfer. | 1 | 0 | X | 1110 |
| | | | | | | Abort transfer. | 0 | 1 | X | — |
| | | 0 | 0 | 1 | A master data or address byte was transmitted; ACK received. | Load next data byte into SMB0DAT. | 0 | 0 | X | 1100 |
| | | | | | | End transfer with STOP. | 0 | 1 | X | — |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X | — |
| | | | | | | Send repeated START. | 1 | 0 | X | 1110 |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). Set ACK for initial data byte. | 0 | 0 | 1 | 1000 |
| Master Receiver | 1000 | 0 | 0 | 1 | A master data byte was received; ACK sent. | Set ACK for next data byte; Read SMB0DAT. | 0 | 0 | 1 | 1000 |
| | | | | | | Set NACK to indicate next data byte as the last data byte; Read SMB0DAT. | 0 | 0 | 0 | 1000 |
| | | | | | | Initiate repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | X | 1100 |
| | | 0 | 0 | 0 | A master data byte was received; NACK sent (last byte). | Read SMB0DAT; send STOP. | 0 | 1 | 0 | — |
| | | | | | | Read SMB0DAT; Send STOP followed by START. | 1 | 1 | 0 | 1110 |
| | | | | | | Initiate repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | X | 1100 |

**Table 22.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) (Continued)**

| Mode | Values Read | | | | Current SMbus State | Typical Response Options | Values to Write | | | Next Status Vector Expected |
|---|---|---|---|---|---|---|---|---|---|---|
| | Status Vector | ACKRQ | ARBLOST | ACK | | | STA | STO | ACK | |
| Slave Transmitter | 0100 | 0 | 0 | 0 | A slave byte was transmitted; NACK received. | No action required (expecting STOP condition). | 0 | 0 | X | 0001 |
| | | 0 | 0 | 1 | A slave byte was transmitted; ACK received. | Load SMB0DAT with next data byte to transmit. | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | A Slave byte was transmitted; error detected. | No action required (expecting Master to end transfer). | 0 | 0 | X | 0001 |
| | 0101 | 0 | X | X | An illegal STOP or bus error was detected while a Slave Transmission was in progress. | Clear STO. | 0 | 0 | X | — |
| Slave Receiver | 0010 | 0 | 0 | X | A slave address + R/W was received; ACK sent. | If Write, Set ACK for first data byte. | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte | 0 | 0 | X | 0100 |
| | | 0 | 1 | X | Lost arbitration as master; slave address + R/W received; ACK sent. | If Write, Set ACK for first data byte. | 0 | 0 | 1 | 0000 |
| | | | | | | If Read, Load SMB0DAT with data byte | 0 | 0 | X | 0100 |
| | | | | | | Reschedule failed transfer | 1 | 0 | X | 1110 |
| | 0001 | 0 | 0 | X | A STOP was detected while addressed as a Slave Transmitter or Slave Receiver. | Clear STO. | 0 | 0 | X | — |
| | | 0 | 1 | X | Lost arbitration while attempting a STOP. | No action required (transfer complete/aborted). | 0 | 0 | 0 | — |
| | 0000 | 0 | 0 | X | A slave byte was received. | Set ACK for next data byte; Read SMB0DAT. | 0 | 0 | 1 | 0000 |
| | | | | | | Set NACK for next data byte; Read SMB0DAT. | 0 | 0 | 0 | 0000 |
| Bus Error Condition | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0001 | 0 | 1 | X | Lost arbitration due to a detected STOP. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0000 | 0 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |

# 23. 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 “23.1. Enhanced Baud Rate Generation” on page 258). 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).

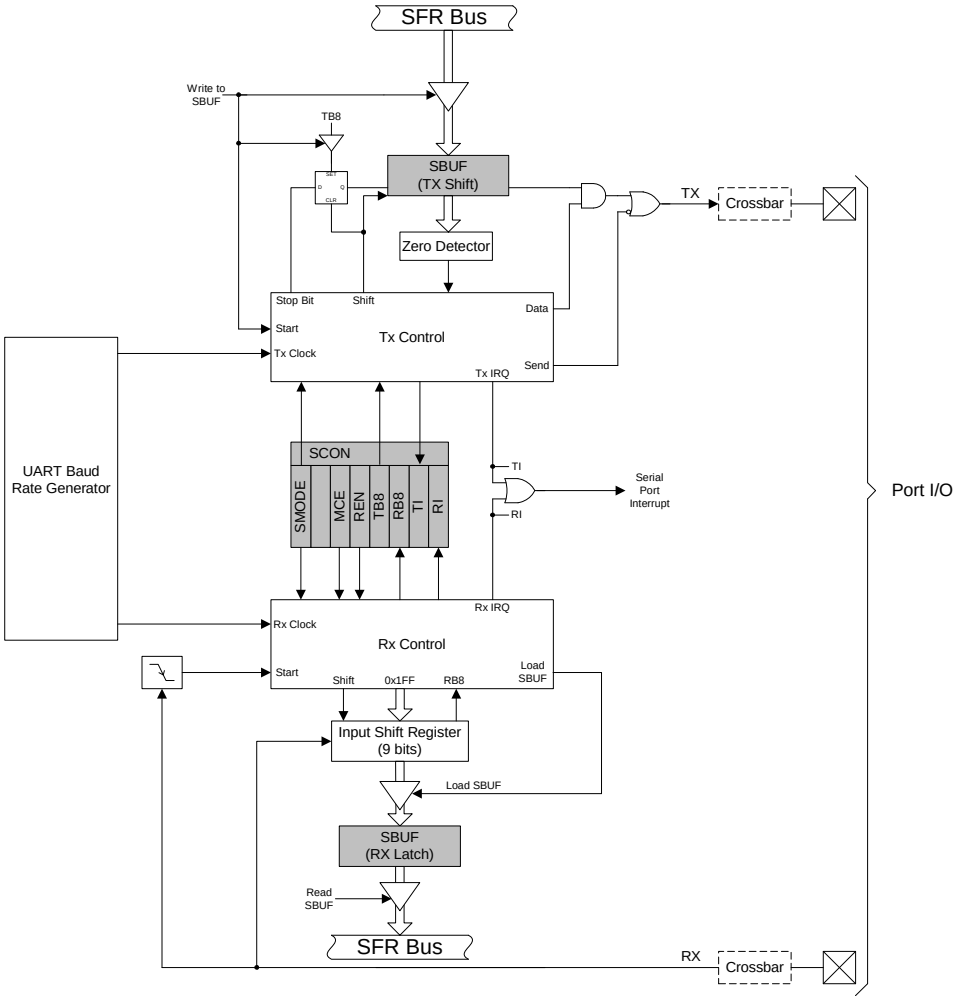



**Figure 23.1. UART0 Block Diagram**

## 23.1. 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 23.2), 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.

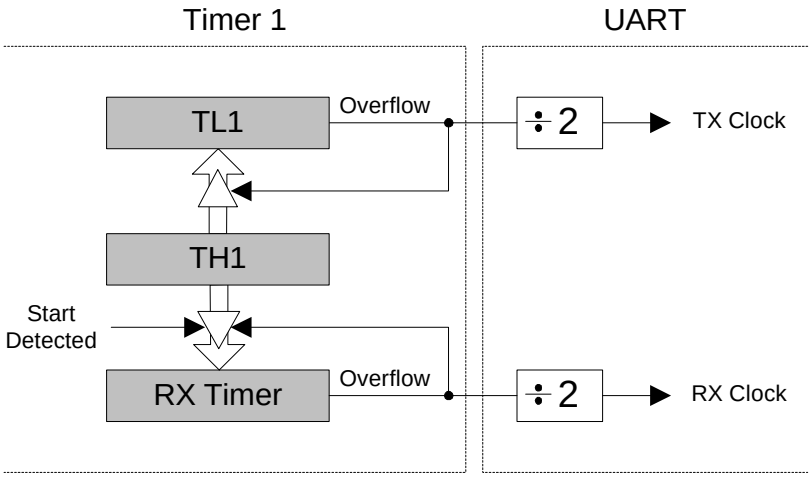



**Figure 23.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see Section "25.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload" on page 282). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Note that Timer 1 may be clocked by one of six sources: SYSCLK, SYSCLK / 4, SYSCLK / 12, SYSCLK / 48, the external oscillator clock / 8, or an external input T1. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 23.1-A and Equation 23.1-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 23.1. UART0 Baud Rate**

Where $T1_{CLK}$ is the frequency of the clock supplied to Timer 1, and *T1H* is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in Section "25.1. Timer 0 and Timer 1" on page 280. A quick reference for typical baud rates and system clock frequencies is given in Table 23.1 through Table 23.2. Note that the internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

## 23.2. 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.

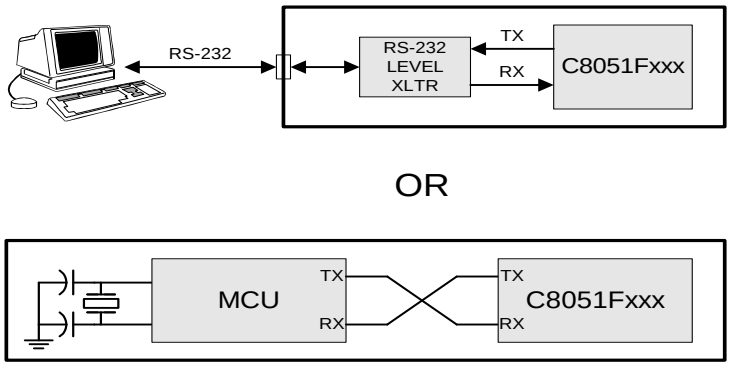



**Figure 23.3. UART Interconnect Diagram**

### 23.2.1. 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.

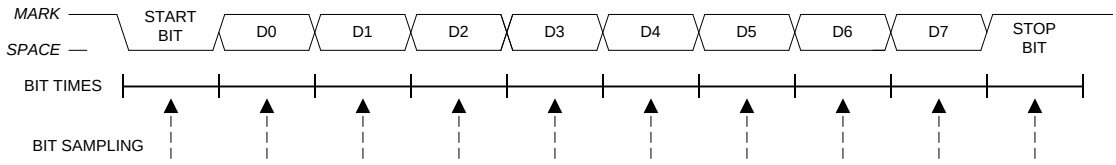



**Figure 23.4. 8-Bit UART Timing Diagram**

### 23.2.2. 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.

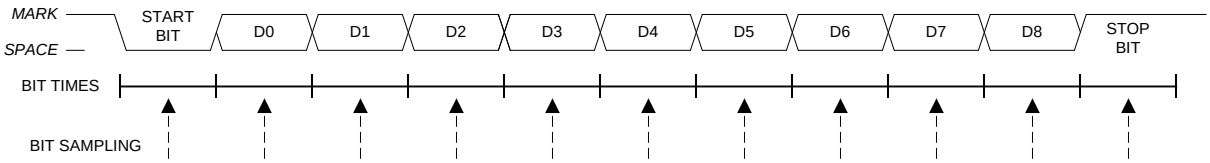



**Figure 23.5. 9-Bit UART Timing Diagram**

## 23.3. 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).

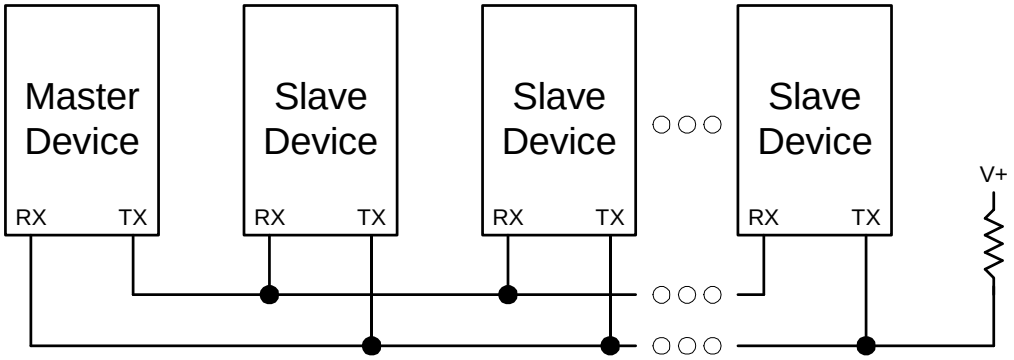



**Figure 23.6. UART Multi-Processor Mode Interconnect Diagram**

## SFR Definition 23.1. SCON0: Serial Port 0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | S0MODE | | MCE0 | REN0 | TB80 | RB80 | TI0 | RI0 |
| Type | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x98; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | S0MODE | **Serial Port 0 Operation Mode.**<br>Selects the UART0 Operation Mode.<br>0: 8-bit UART with Variable Baud Rate.<br>1: 9-bit UART with Variable Baud Rate. |
| 6 | Unused | Read = 1b. Write = Don't Care. |
| 5 | MCE0 | **Multiprocessor Communication Enable.**<br>**For Mode 0 (8-bit UART): Checks for valid stop bit.**<br>0: Logic level of stop bit is ignored.<br>1: RI0 will only be activated if stop bit is logic level 1.<br>**For Mode 1 (9-bit UART): Multiprocessor Communications Enable.**<br>0: Logic level of ninth bit is ignored.<br>1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1. |
| 4 | REN0 | **Receive Enable.**<br>0: UART0 reception disabled.<br>1: UART0 reception enabled. |
| 3 | TB80 | **Ninth Transmission Bit.**<br>The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0). |
| 2 | RB80 | **Ninth Receive Bit.**<br>RB80 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1. |
| 1 | TI0 | **Transmit Interrupt Flag.**<br>Set by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software. |
| 0 | RI0 | **Receive Interrupt Flag.**<br>Set to 1 by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). When the UART0 interrupt is enabled, setting this bit to 1 causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software. |

## SFR Definition 23.2. SBUF0: Serial (UART0) Port Data Buffer

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SBUF0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x99

| Bit | Name | Function |
|---|---|---|
| 7:0 | SBUF0 | **Serial Data Buffer Bits 7:0 (MSB–LSB).**<br>This SFR accesses two registers; a transmit shift register and a receive latch register. When data is written to SBUF0, it goes to the transmit shift register and is held for serial transmission. Writing a byte to SBUF0 initiates the transmission. A read of SBUF0 returns the contents of the receive latch. |

**Table 23.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator**

| | Frequency: 24.5 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscillator Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from Internal Osc. | 230400 | –0.32% | 106 | SYSCLK | XX[2] | 1 | 0xCB |
| | 115200 | –0.32% | 212 | SYSCLK | XX | 1 | 0x96 |
| | 57600 | 0.15% | 426 | SYSCLK | XX | 1 | 0x2B |
| | 28800 | –0.32% | 848 | SYSCLK/4 | 01 | 0 | 0x96 |
| | 14400 | 0.15% | 1704 | SYSCLK/12 | 00 | 0 | 0xB9 |
| | 9600 | –0.32% | 2544 | SYSCLK/12 | 00 | 0 | 0x96 |
| | 2400 | –0.32% | 10176 | SYSCLK/48 | 10 | 0 | 0x96 |
| | 1200 | 0.15% | 20448 | SYSCLK/48 | 10 | 0 | 0x2B |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in Section 25.1.
2. X = Don't care.

**Table 23.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator**

| | Frequency: 22.1184 MHz | | | | | | |
|---|---|---|---|---|---|---|---|
| | Target Baud Rate (bps) | Baud Rate % Error | Oscillator Divide Factor | Timer Clock Source | SCA1–SCA0 (pre-scale select)[1] | T1M[1] | Timer 1 Reload Value (hex) |
| SYSCLK from External Osc. | 230400 | 0.00% | 96 | SYSCLK | XX[2] | 1 | 0xD0 |
| | 115200 | 0.00% | 192 | SYSCLK | XX | 1 | 0xA0 |
| | 57600 | 0.00% | 384 | SYSCLK | XX | 1 | 0x40 |
| | 28800 | 0.00% | 768 | SYSCLK / 12 | 00 | 0 | 0xE0 |
| | 14400 | 0.00% | 1536 | SYSCLK / 12 | 00 | 0 | 0xC0 |
| | 9600 | 0.00% | 2304 | SYSCLK / 12 | 00 | 0 | 0xA0 |
| | 2400 | 0.00% | 9216 | SYSCLK / 48 | 10 | 0 | 0xA0 |
| | 1200 | 0.00% | 18432 | SYSCLK / 48 | 10 | 0 | 0x40 |
| SYSCLK from Internal Osc. | 230400 | 0.00% | 96 | EXTCLK / 8 | 11 | 0 | 0xFA |
| | 115200 | 0.00% | 192 | EXTCLK / 8 | 11 | 0 | 0xF4 |
| | 57600 | 0.00% | 384 | EXTCLK / 8 | 11 | 0 | 0xE8 |
| | 28800 | 0.00% | 768 | EXTCLK / 8 | 11 | 0 | 0xD0 |
| | 14400 | 0.00% | 1536 | EXTCLK / 8 | 11 | 0 | 0xA0 |
| | 9600 | 0.00% | 2304 | EXTCLK / 8 | 11 | 0 | 0x70 |

**Notes:**
1. SCA1–SCA0 and T1M bit definitions can be found in Section 25.1.
2. X = Don't care.

# 24. Enhanced Serial Peripheral Interface (SPI0)

The Enhanced Serial Peripheral Interface (SPI0) provides access to a flexible, full-duplex synchronous serial bus. SPI0 can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select SPI0 in slave mode, or to disable Master Mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a chip-select output in master mode, or disabled for 3-wire operation. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

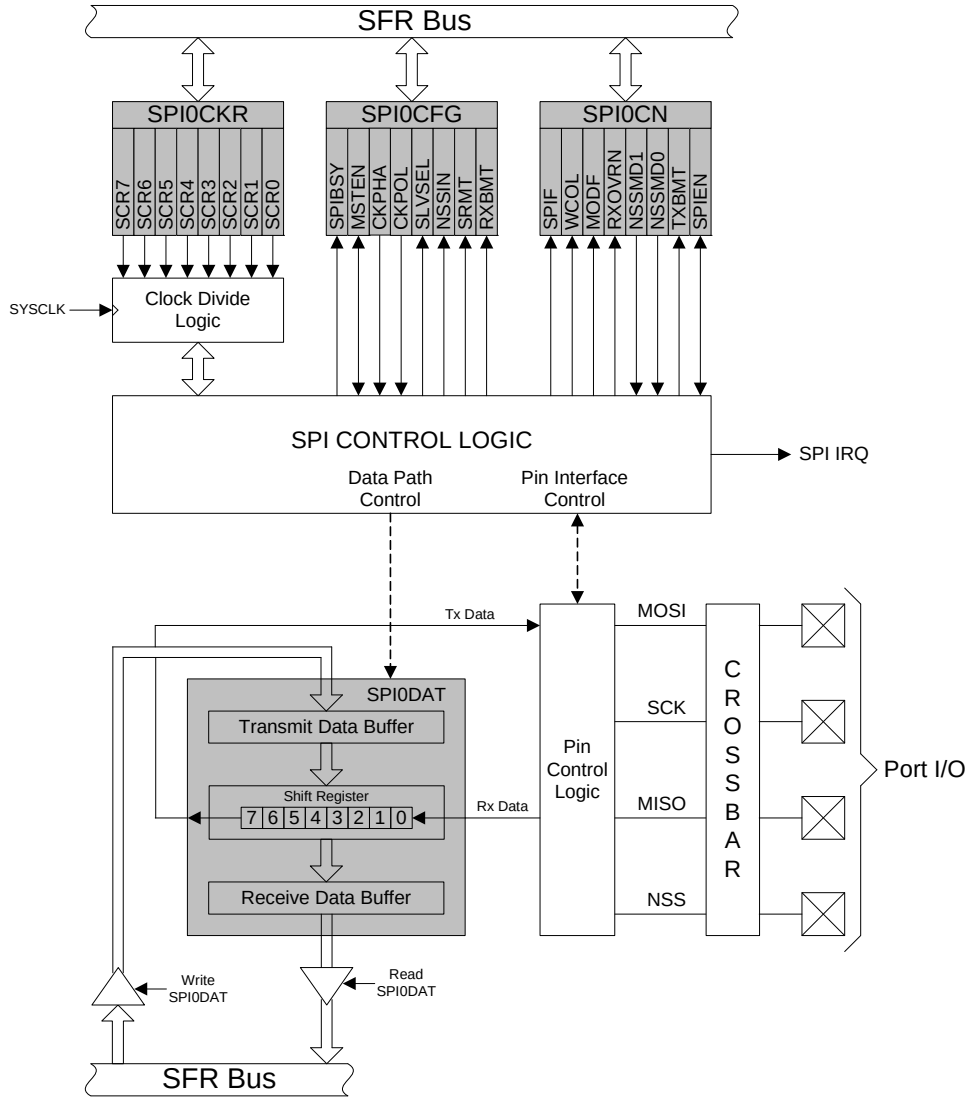



**Figure 24.1. SPI Block Diagram**

## 24.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

### 24.1.1. 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.

### 24.1.2. 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.

### 24.1.3. 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.

### 24.1.4. 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 24.2, Figure 24.3, and Figure 24.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “21. Port Input/Output” on page 215 for general purpose port I/O and crossbar information.

## 24.2. SPI0 Master Mode Operation

A SPI master device initiates all data transfers on a SPI bus. SPI0 is placed in master mode by setting the Master Enable flag (MSTEN, SPI0CN.6). Writing a byte of data to the SPI0 data register (SPI0DAT) when in master mode writes to the transmit buffer. If the SPI shift register is empty, the byte in the transmit buffer is moved to the shift register, and a data transfer begins. The SPI0 master immediately shifts out the data serially on the MOSI line while providing the serial clock on SCK. The SPIF (SPI0CN.7) flag is set to logic

1 at the end of the transfer. If interrupts are enabled, an interrupt request is generated when the SPIF flag is set. While the SPI0 master transfers data to a slave on the MOSI line, the addressed SPI slave device simultaneously transfers the contents of its shift register to the SPI master on the MISO line in a full-duplex operation. Therefore, the SPIF flag serves as both a transmit-complete and receive-data-ready flag. The data byte received from the slave is transferred MSB-first into the master's shift register. When a byte is fully shifted into the register, it is moved to the receive buffer where it can be read by the processor by reading SPI0DAT.

When configured as a master, SPI0 can operate in one of three different modes: multi-master mode, 3-wire single-master mode, and 4-wire single-master mode. The default, multi-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In this mode, NSS is an input to the device, and is used to disable the master SPI0 when another master is accessing the bus. When NSS is pulled low in this mode, MSTEN (SPI0CN.6) and SPIEN (SPI0CN.0) are set to 0 to disable the SPI master device, and a Mode Fault is generated (MODF, SPI0CN.5 = 1). Mode Fault will generate an interrupt if enabled. SPI0 must be manually re-enabled in software under these circumstances. In multi-master systems, devices will typically default to being slave devices while they are not acting as the system master device. In multi-master mode, slave devices can be addressed individually (if needed) using general-purpose I/O pins. Figure 24.2 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 24.3 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 24.4 shows a connection diagram for a master device in 4-wire master mode and two slave devices.

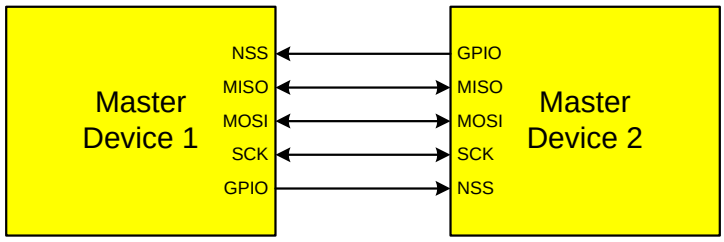



**Figure 24.2. Multiple-Master Mode Connection Diagram**

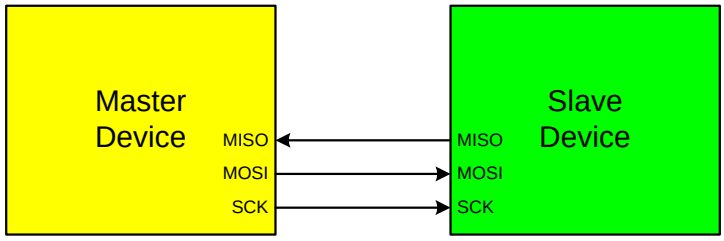



**Figure 24.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram**

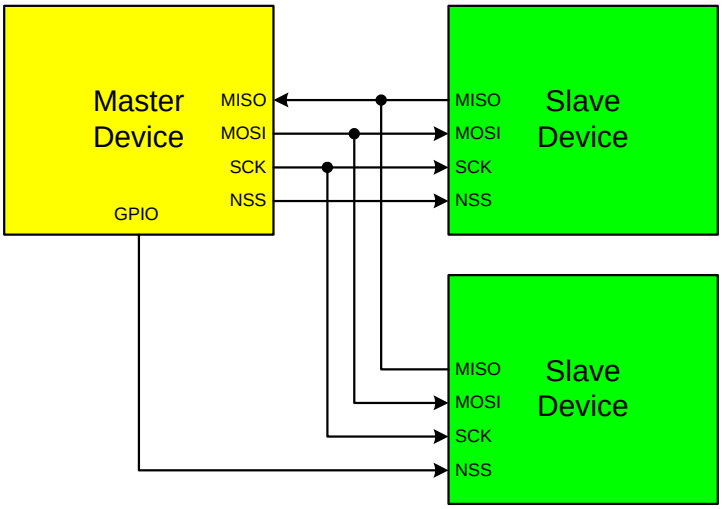



**Figure 24.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram**

## 24.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted through the shift register, the SPIF flag is set to logic 1, and the byte is copied into the receive buffer. Data is read from the receive buffer by reading SPI0DAT. A slave device cannot initiate transfers. Data to be transferred to the master device is pre-loaded into the shift register by writing to SPI0DAT. Writes to SPI0DAT are double-buffered, and are placed in the transmit buffer first. If the shift register is empty, the contents of the transmit buffer will immediately be transferred into the shift register. When the shift register already contains data, the SPI will load the shift register with the transmit buffer's contents after the last SCK edge of the next (or current) SPI transfer.

When configured as a slave, SPI0 can be configured for 4-wire or 3-wire operation. The default, 4-wire slave mode, is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 1. In 4-wire mode, the NSS signal is routed to a port pin and configured as a digital input. SPI0 is enabled when NSS is logic 0, and disabled when NSS is logic 1. The bit counter is reset on a falling edge of NSS. Note that the NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 24.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 24.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

## 24.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

All of the following bits must be cleared by software.

- The SPI Interrupt Flag, SPIF (SPI0CN.7) is set to logic 1 at the end of each byte transfer. This flag can occur in all SPI0 modes.
- The Write Collision Flag, WCOL (SPI0CN.6) is set to logic 1 if a write to SPI0DAT is attempted when the transmit buffer has not been emptied to the SPI shift register. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written.This flag can occur in all SPI0 modes.
- The Mode Fault Flag MODF (SPI0CN.5) is set to logic 1 when SPI0 is configured as a master, and for multi-master mode and the NSS pin is pulled low. When a Mode Fault occurs, the MSTEN and SPIEN bits in SPI0CN are set to logic 0 to disable SPI0 and allow another master device to access the bus.
- The Receive Overrun Flag RXOVRN (SPI0CN.4) is set to logic 1 when configured as a slave, and a transfer is completed and the receive buffer still holds an unread byte from a previous transfer. The new byte is not transferred to the receive buffer, allowing the previously received data byte to be read. The data byte which caused the overrun is lost.

## 24.5. Serial Clock Phase and Polarity

Four combinations of serial clock phase and polarity can be selected using the clock control bits in the SPI0 Configuration Register (SPI0CFG). The CKPHA bit (SPI0CFG.5) selects one of two clock phases (edge used to latch the data). The CKPOL bit (SPI0CFG.4) selects between an active-high or active-low clock. Both master and slave devices must be configured to use the same clock phase and polarity. SPI0 should be disabled (by clearing the SPIEN bit, SPI0CN.0) when changing the clock phase or polarity. The clock and data line relationships for master mode are shown in Figure 24.5. For slave mode, the clock and data relationships are shown in Figure 24.6 and Figure 24.7. Note that CKPHA should be set to 0 on both the master and slave SPI when communicating between two Silicon Labs C8051 devices.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 24.3 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.

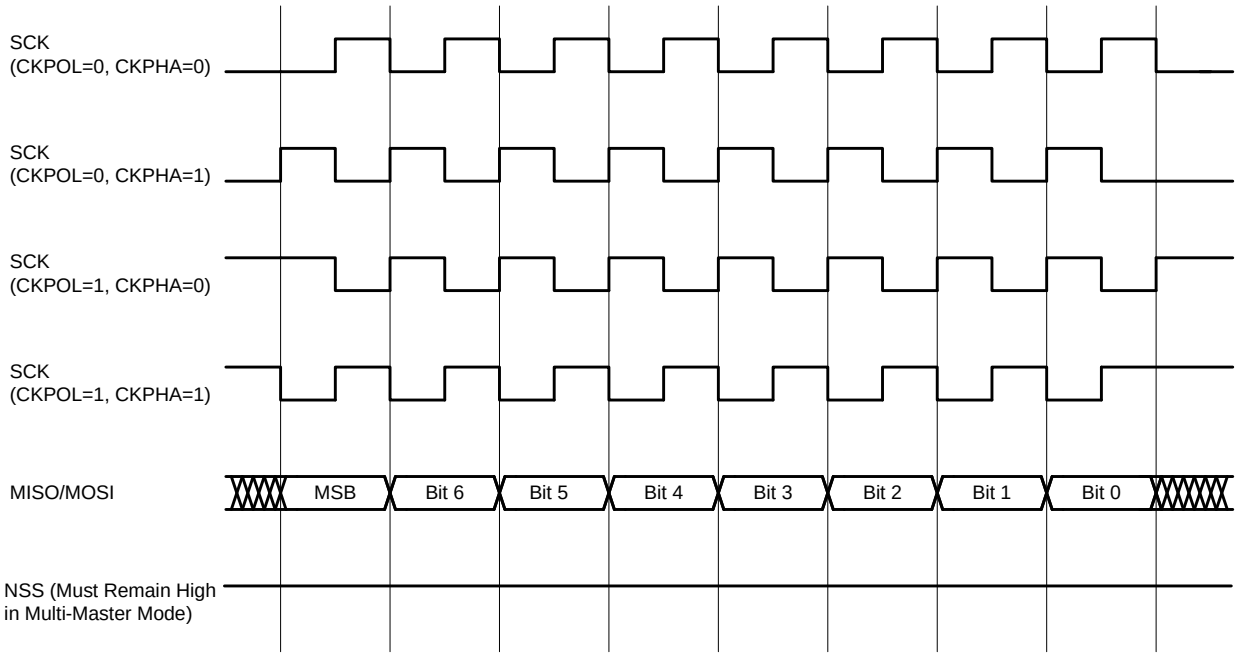



**Figure 24.5. Master Mode Data/Clock Timing**

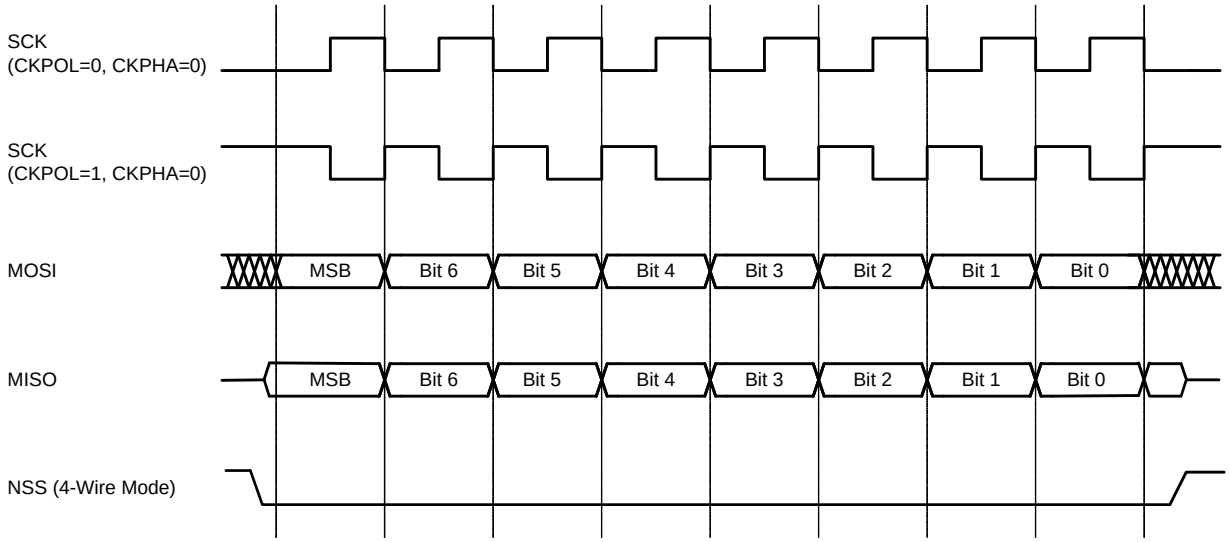



**Figure 24.6. Slave Mode Data/Clock Timing (CKPHA = 0)**

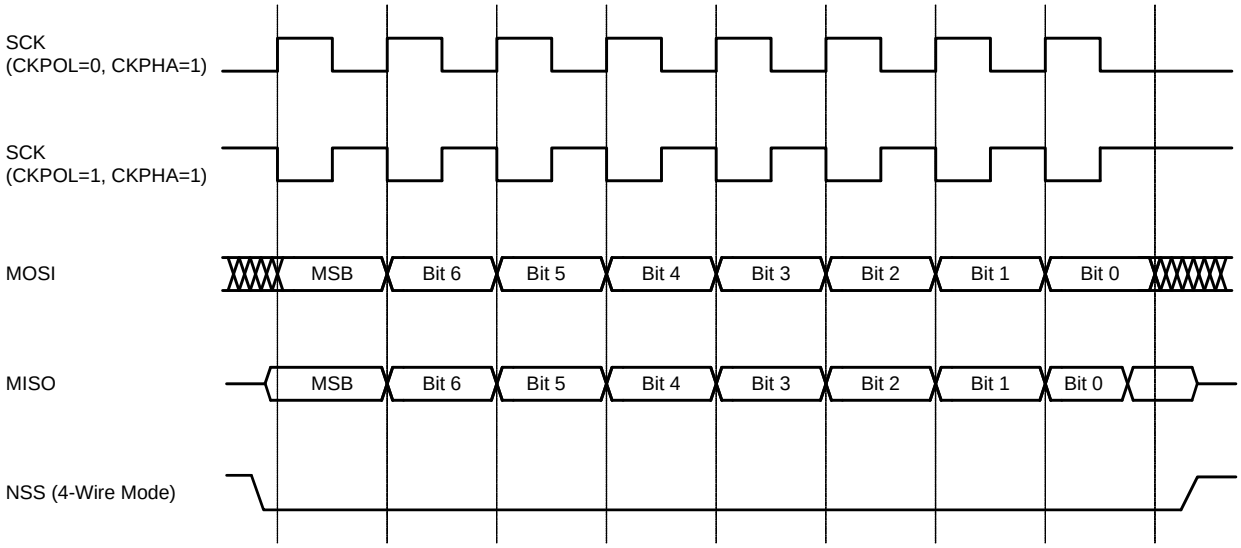



**Figure 24.7. Slave Mode Data/Clock Timing (CKPHA = 1)**

## 24.6. 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 24.1. SPI0CFG: SPI0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIBSY | MSTEN | CKPHA | CKPOL | SLVSEL | NSSIN | SRMT | RXBMT |
| Type | R | R/W | R/W | R/W | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

SFR Page = 0x0; SFR Address = 0xA1

| Bit | Name | Function |
|---|---|---|
| 7 | SPIBSY | **SPI Busy.**<br>This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode). |
| 6 | MSTEN | **Master Mode Enable.**<br>0: Disable master mode. Operate in slave mode.<br>1: Enable master mode. Operate as a master. |
| 5 | CKPHA | **SPI0 Clock Phase.**<br>0: Data centered on first edge of SCK period.*<br>1: Data centered on second edge of SCK period.* |
| 4 | CKPOL | **SPI0 Clock Polarity.**<br>0: SCK line low in idle state.<br>1: SCK line high in idle state. |
| 3 | SLVSEL | **Slave Selected Flag.**<br>This bit is set to logic 1 whenever the NSS pin is low indicating SPI0 is the selected slave. It is cleared to logic 0 when NSS is high (slave not selected). This bit does not indicate the instantaneous value at the NSS pin, but rather a de-glitched version of the pin input. |
| 2 | NSSIN | **NSS Instantaneous Pin Input.**<br>This bit mimics the instantaneous value that is present on the NSS port pin at the time that the register is read. This input is not de-glitched. |
| 1 | SRMT | **Shift Register Empty (valid in slave mode only).**<br>This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode. |
| 0 | RXBMT | **Receive Buffer Empty (valid in slave mode only).**<br>This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode. |

**Note:** In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 24.1 for timing parameters.

## SFR Definition 24.2. SPI0CN: SPI0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIF | WCOL | MODF | RXOVRN | NSSMD[1:0] | | TXBMT | SPIEN |
| Type | R/W | R/W | R/W | R/W | R/W | | R | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

SFR Page = 0x0; SFR Address = 0xF8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | SPIF | **SPI0 Interrupt Flag.**<br>This bit is set to logic 1 by hardware at the end of a data transfer. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 6 | WCOL | **Write Collision Flag.**<br>This bit is set to logic 1 if a write to SPI0DAT is attempted when TXBMT is 0. When this occurs, the write to SPI0DAT will be ignored, and the transmit buffer will not be written. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 5 | MODF | **Mode Fault Flag.**<br>This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 4 | RXOVRN | **Receive Overrun Flag (valid in slave mode only).**<br>This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software. |
| 3:2 | NSSMD[1:0] | **Slave Select Mode.**<br>Selects between the following NSS operation modes:<br>(See Section 24.2 and Section 24.3).<br>00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin.<br>01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device.<br>1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0. |
| 1 | TXBMT | **Transmit Buffer Empty.**<br>This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer. |
| 0 | SPIEN | **SPI0 Enable.**<br>0: SPI disabled.<br>1: SPI enabled. |

## SFR Definition 24.3. SPI0CKR: SPI0 Clock Rate

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

SFR Page = 0x0; SFR Address = 0xA2

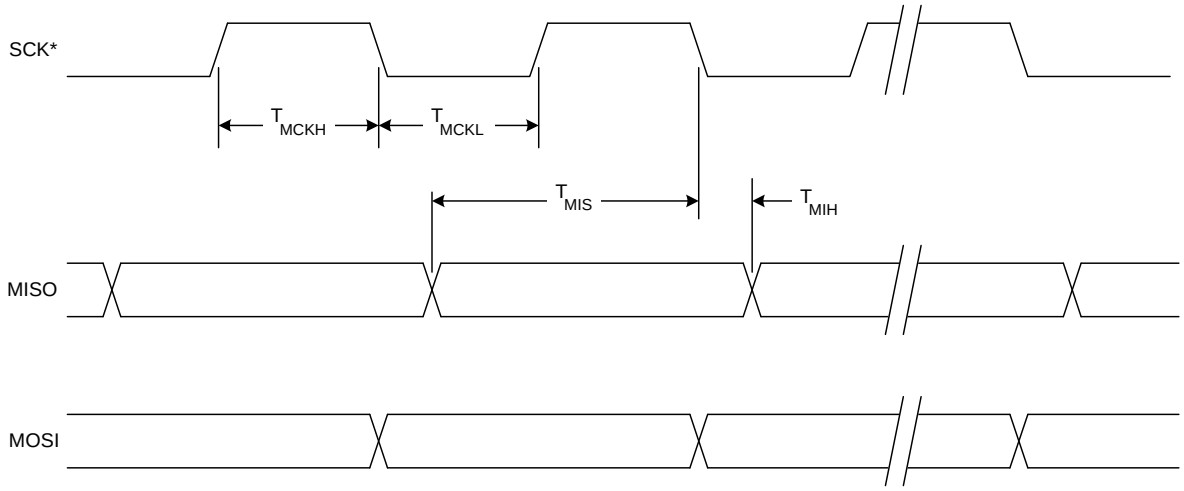
| Bit | Name | Function |
|---|---|---|
| 7:0 | SCR[7:0] | **SPI0 Clock Rate.**<br>These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI0CKR* is the 8-bit value held in the SPI0CKR register.<br>$f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR[7:0] + 1)}$<br>for 0 <= SPI0CKR <= 255<br><br>Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04,<br>$f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$<br>$f_{SCK} = 200kHz$ |

## SFR Definition 24.4. SPI0DAT: SPI0 Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPI0DAT[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

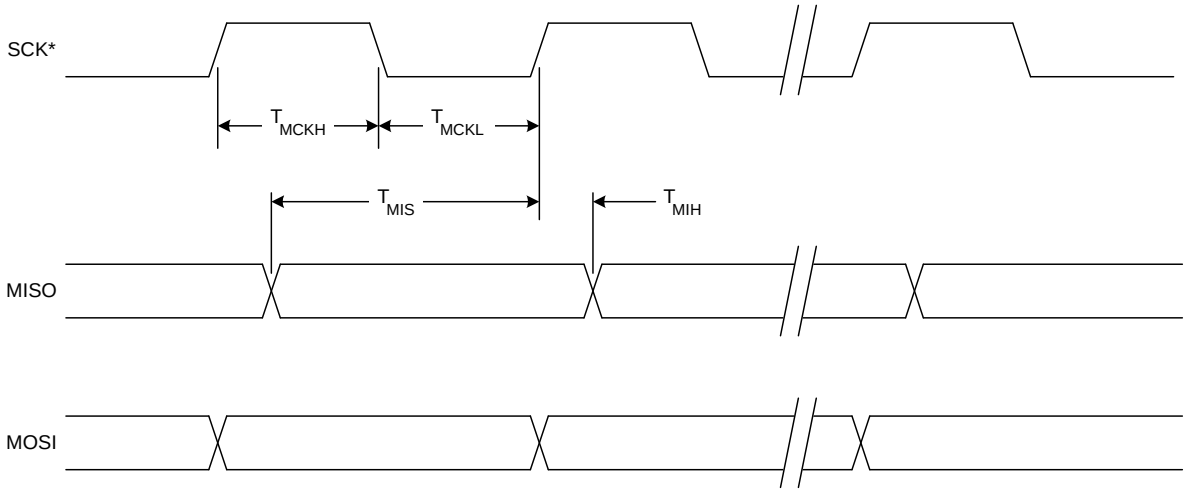
SFR Page = 0x0; SFR Address = 0xA3

| Bit | Name | Function |
|---|---|---|
| 7:0 | SPI0DAT[7:0] | **SPI0 Transmit and Receive Data.**<br>The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI0DAT returns the contents of the receive buffer. |



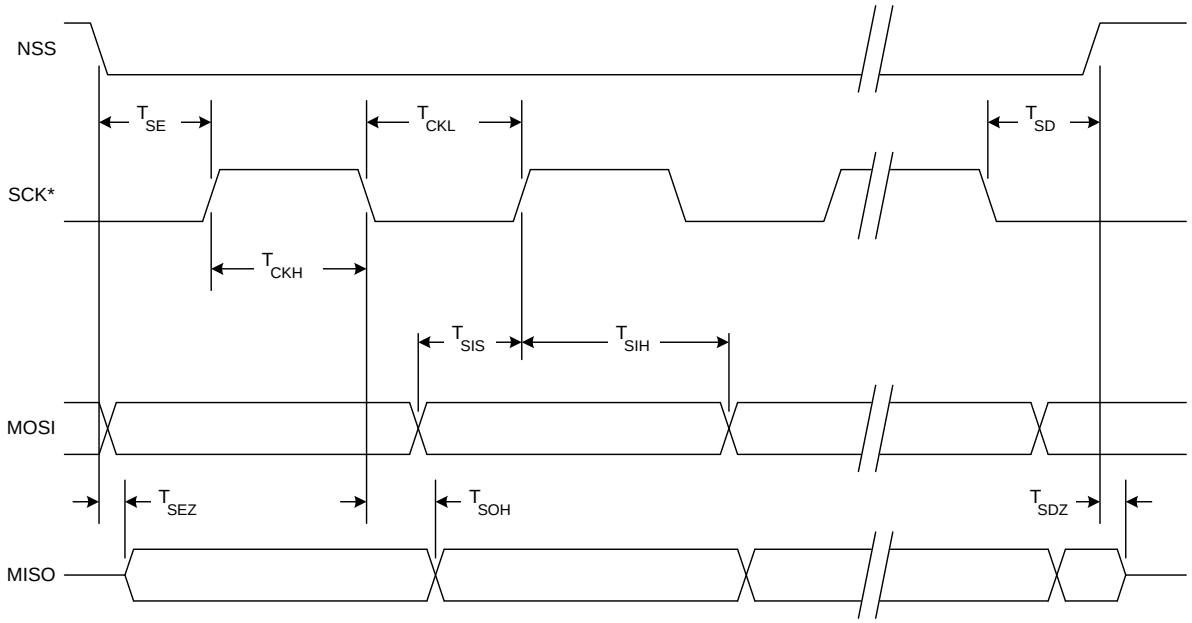

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 24.8. SPI Master Timing (CKPHA = 0)**



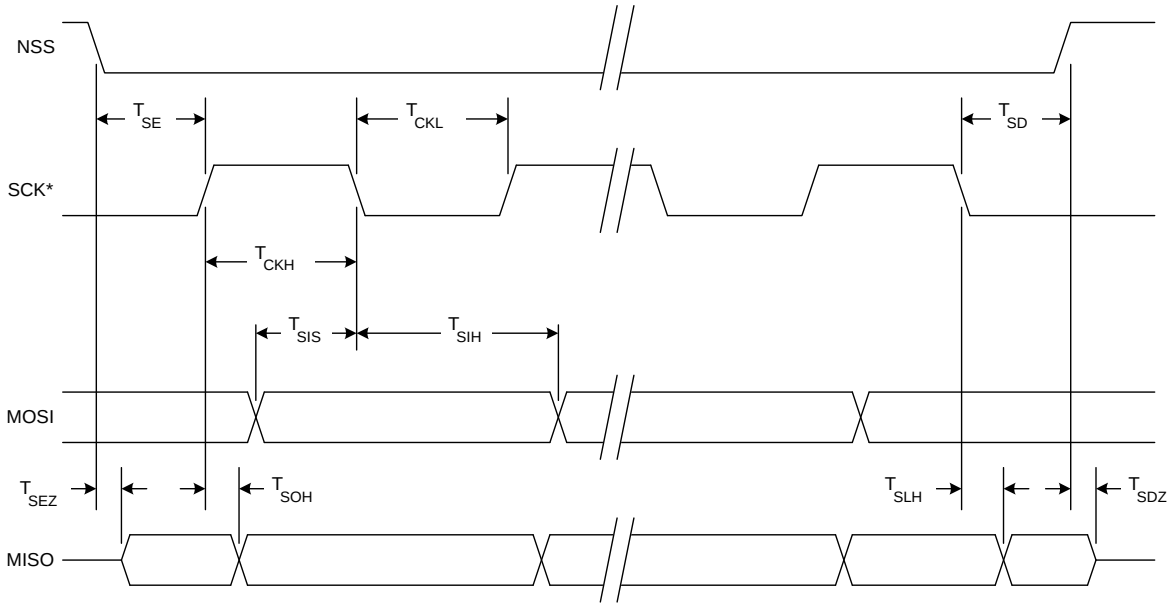

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 24.9. SPI Master Timing (CKPHA = 1)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 24.10. SPI Slave Timing (CKPHA = 0)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 24.11. SPI Slave Timing (CKPHA = 1)**

**Table 24.1. SPI Slave Timing Parameters**

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing** (See Figure 24.8 and Figure 24.9) | | | | |
| $T_{MCKH}$ | SCK High Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MCKL}$ | SCK Low Time | 1 x $T_{SYSCLK}$ | — | ns |
| $T_{MIS}$ | MISO Valid to SCK Shift Edge | 1 x $T_{SYSCLK}$ + 20 | — | ns |
| $T_{MIH}$ | SCK Shift Edge to MISO Change | 0 | — | ns |
| **Slave Mode Timing** (See Figure 24.10 and Figure 24.11) | | | | |
| $T_{SE}$ | NSS Falling to First SCK Edge | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SD}$ | Last SCK Edge to NSS Rising | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SEZ}$ | NSS Falling to MISO Valid | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{SDZ}$ | NSS Rising to MISO High-Z | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{CKH}$ | SCK High Time | 5 x $T_{SYSCLK}$ | — | ns |
| $T_{CKL}$ | SCK Low Time | 5 x $T_{SYSCLK}$ | — | ns |
| $T_{SIS}$ | MOSI Valid to SCK Sample Edge | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SIH}$ | SCK Sample Edge to MOSI Change | 2 x $T_{SYSCLK}$ | — | ns |
| $T_{SOH}$ | SCK Shift Edge to MISO Change | — | 4 x $T_{SYSCLK}$ | ns |
| $T_{SLH}$ | Last SCK Edge to MISO Change (CKPHA = 1 ONLY) | 6 x $T_{SYSCLK}$ | 8 x $T_{SYSCLK}$ | ns |
| **Note:** $T_{SYSCLK}$ is equal to one period of the device system clock (SYSCLK). | | | | |

## 25. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two are 16-bit auto-reload timer for use with the ADC, SMBus, or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload. Additionally, Timer 2 and Timer 3 have a Capture Mode that can be used to measure the SmaRTClock or a Comparator period with respect to another oscillator. This is particularly useful when using Capacitive Touch Switches.

| Timer 0 and Timer 1 Modes | Timer 2 Modes | Timer 3 Modes |
|---|---|---|
| 13-bit counter/timer | 16-bit timer with auto-reload | 16-bit timer with auto-reload |
| 16-bit counter/timer | | |
| 8-bit counter/timer with auto-reload | Two 8-bit timers with auto-reload | Two 8-bit timers with auto-reload |
| Two 8-bit counter/timers (Timer 0 only) | | |

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked (See SFR Definition 25.1 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12. Timer 2 may additionally be clocked by the SmaRTClock divided by 8 or the Comparator0 output. Timer 3 may additionally be clocked by the external oscillator clock source divided by 8 or the Comparator1 output.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin (T0 or T1). Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

## SFR Definition 25.1. CKCON: Clock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | T3MH | T3ML | T2MH | T2ML | T1M | T0M | SCA[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8E

| Bit | Name | Function |
|---|---|---|
| 7 | T3MH | **Timer 3 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only).<br>0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 high byte uses the system clock. |
| 6 | T3ML | **Timer 3 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode.<br>0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN.<br>1: Timer 3 low byte uses the system clock. |
| 5 | T2MH | **Timer 2 High Byte Clock Select.**<br>Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only).<br>0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 high byte uses the system clock. |
| 4 | T2ML | **Timer 2 Low Byte Clock Select.**<br>Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer.<br>0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN.<br>1: Timer 2 low byte uses the system clock. |
| 3 | T1M | **Timer 1 Clock Select.**<br>Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1.<br>0: Timer 1 uses the clock defined by the prescale bits SCA[1:0].<br>1: Timer 1 uses the system clock. |
| 2 | T0M | **Timer 0 Clock Select.**<br>Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1.<br>0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0].<br>1: Counter/Timer 0 uses the system clock. |
| 1:0 | SCA[1:0] | **Timer 0/1 Prescale Bits.**<br>These bits control the Timer 0/1 Clock Prescaler:<br>00: System clock divided by 12<br>01: System clock divided by 4<br>10: System clock divided by 48<br>11: External clock divided by 8 (synchronized with the system clock) |

## 25.1. 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 (“Interrupt Register Descriptions” on page 141); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (“Interrupt Register Descriptions” on page 141). 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.

### 25.1.1. Mode 0: 13-bit Counter/Timer

Timer 0 and Timer 1 operate as 13-bit counter/timers in Mode 0. The following describes the configuration and operation of Timer 0. However, both timers operate identically, and Timer 1 is configured in the same manner as described for Timer 0.

The TH0 register holds the eight MSBs of the 13-bit counter/timer. TL0 holds the five LSBs in bit positions TL0.4–TL0.0. The three upper bits of TL0 (TL0.7–TL0.5) are indeterminate and should be masked out or ignored when reading. As the 13-bit timer register increments and overflows from 0x1FFF (all ones) to 0x0000, the timer overflow flag TF0 (TCON.5) is set and an interrupt will occur if Timer 0 interrupts are enabled.

The C/T0 bit (TMOD.2) selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “21.3. Priority Crossbar Decoder” on page 219 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 25.1).

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 13.7). Setting GATE0 to 1 allows the timer to be controlled by the external input signal INT0 (see Section “13.5. Interrupt Register Descriptions” on page 141), facilitating pulse width measurements

**Table 25.1. Timer 0 Running Modes**

| TR0 | GATE0 | INT0 | Counter/Timer |
|---|---|---|---|
| 0 | X | X | Disabled |
| 1 | 0 | X | Enabled |
| 1 | 1 | 0 | Disabled |
| 1 | 1 | 1 | Enabled |
| **Note:** X = Don't Care | | | |

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal INT1 is used with Timer 1; the INT1 polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 13.7).

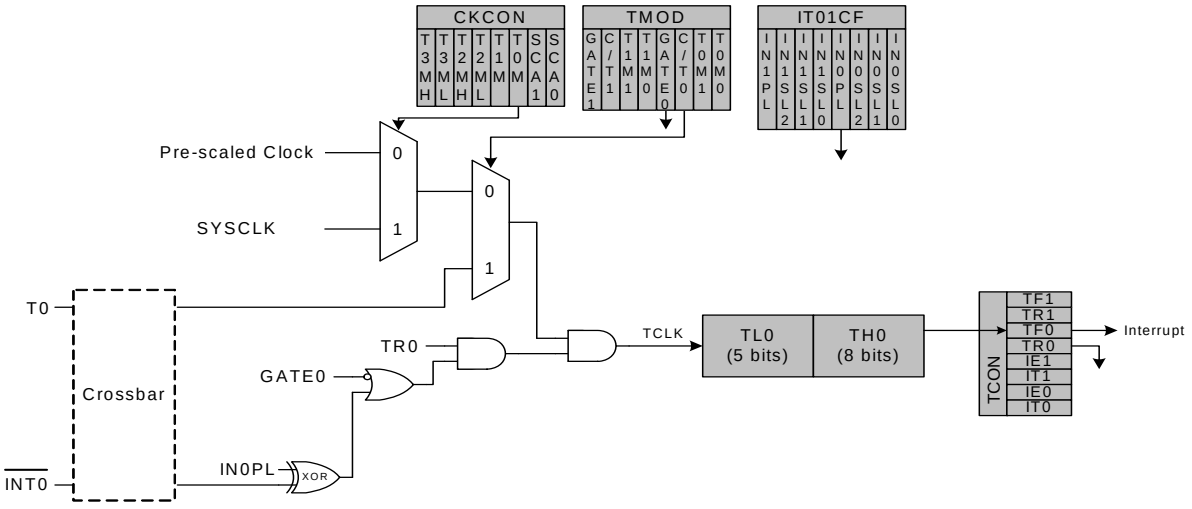



**Figure 25.1. T0 Mode 0 Block Diagram**

### 25.1.2. 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.

### 25.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload

Mode 2 configures Timer 0 and Timer 1 to operate as 8-bit counter/timers with automatic reload of the start value. TL0 holds the count and TH0 holds the reload value. When the counter in TL0 overflows from all ones to 0x00, the timer overflow flag TF0 (TCON.5) is set and the counter in TL0 is reloaded from TH0. If Timer 0 interrupts are enabled, an interrupt will occur when the TF0 flag is set. The reload value in TH0 is not changed. TL0 must be initialized to the desired value before enabling the timer for the first count to be correct. When in Mode 2, Timer 1 operates identically to Timer 0.

Both counter/timers are enabled and configured in Mode 2 in the same manner as Mode 0. Setting the TR0 bit (TCON.4) enables the timer when either GATE0 (TMOD.3) is logic 0 or when the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF (see Section “13.6. External Interrupts INT0 and INT1” on page 148 for details on the external input signals $\overline{INT0}$ and $\overline{INT1}$).

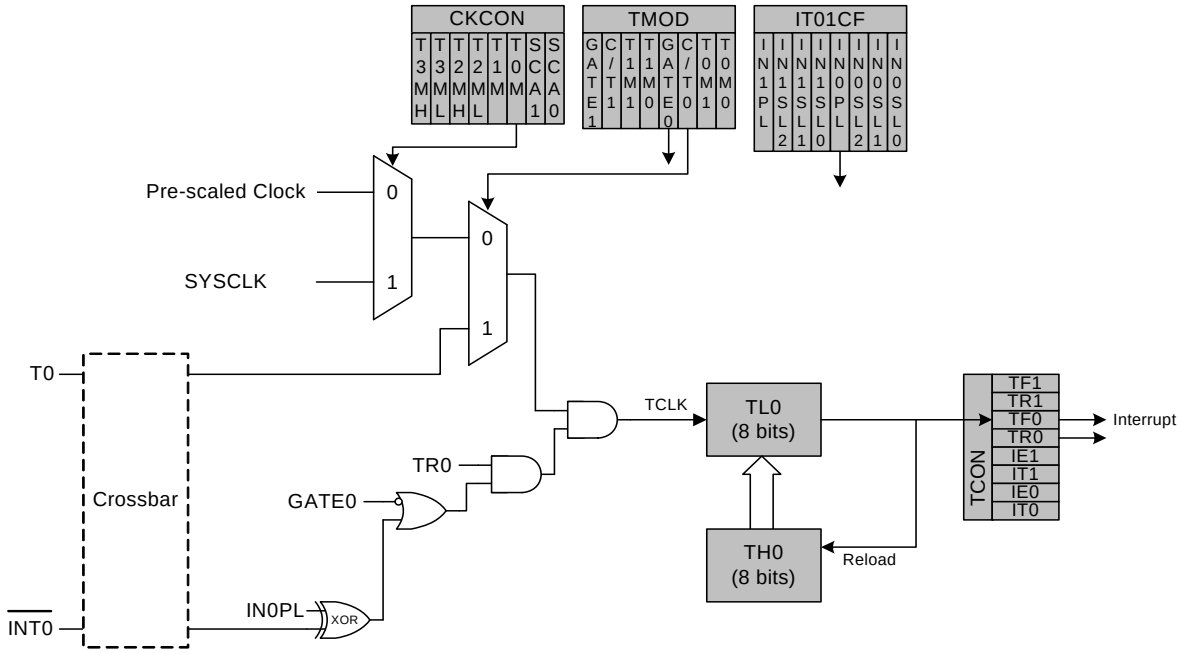



**Figure 25.2. T0 Mode 2 Block Diagram**

### 25.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

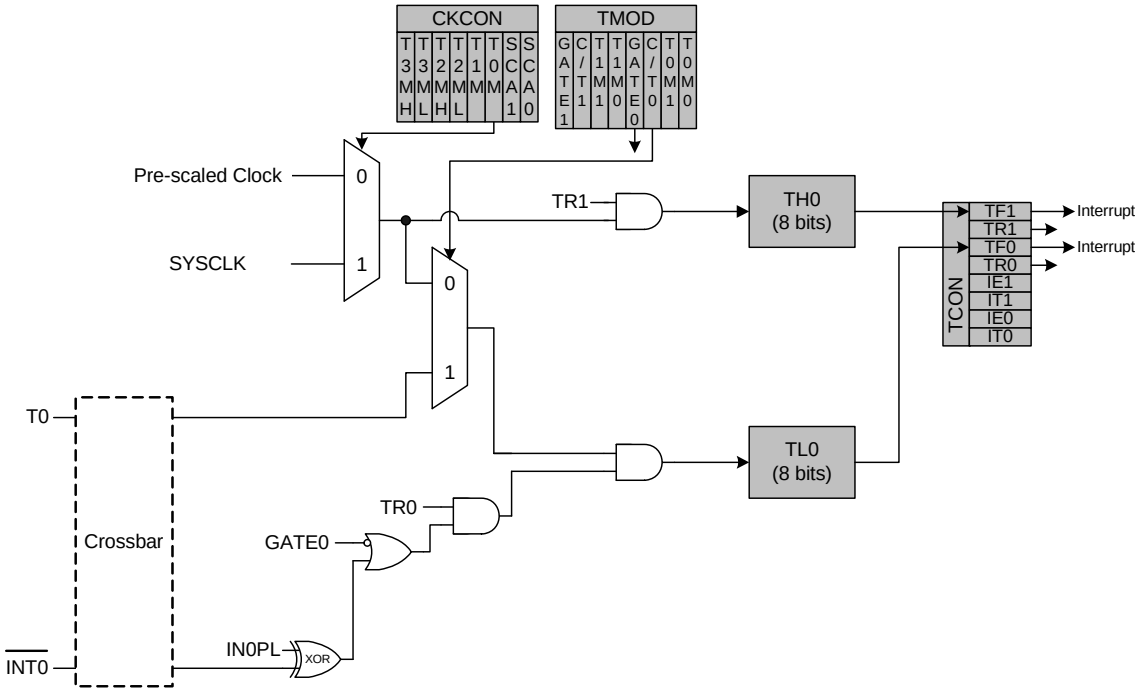



**Figure 25.3. T0 Mode 3 Block Diagram**

## SFR Definition 25.2. TCON: Timer Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x88; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | TF1 | **Timer 1 Overflow Flag.**<br>Set to 1 by hardware when Timer 1 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 1 interrupt service routine. |
| 6 | TR1 | **Timer 1 Run Control.**<br>Timer 1 is enabled by setting this bit to 1. |
| 5 | TF0 | **Timer 0 Overflow Flag.**<br>Set to 1 by hardware when Timer 0 overflows. This flag can be cleared by software but is automatically cleared when the CPU vectors to the Timer 0 interrupt service routine. |
| 4 | TR0 | **Timer 0 Run Control.**<br>Timer 0 is enabled by setting this bit to 1. |
| 3 | IE1 | **External Interrupt 1.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode. |
| 2 | IT1 | **Interrupt 1 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT1}}$ interrupt will be edge or level sensitive. $\overline{\text{INT1}}$ is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 13.7).<br>0: $\overline{\text{INT1}}$ is level triggered.<br>1: $\overline{\text{INT1}}$ is edge triggered. |
| 1 | IE0 | **External Interrupt 0.**<br>This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode. |
| 0 | IT0 | **Interrupt 0 Type Select.**<br>This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 13.7).<br>0: $\overline{\text{INT0}}$ is level triggered.<br>1: $\overline{\text{INT0}}$ is edge triggered. |

## SFR Definition 25.3. TMOD: Timer Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GATE1 | C/T1 | T1M[1:0] | | GATE0 | C/T0 | T0M[1:0] | |
| Type | R/W | R/W | R/W | | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x89

| Bit | Name | Function |
|---|---|---|
| 7 | GATE1 | **Timer 1 Gate Control.**<br>0: Timer 1 enabled when TR1 = 1 irrespective of $\overline{\text{INT1}}$ logic level.<br>1: Timer 1 enabled only when TR1 = 1 AND $\overline{\text{INT1}}$ is active as defined by bit IN1PL in register IT01CF (see SFR Definition 13.7). |
| 6 | C/T1 | **Counter/Timer 1 Select.**<br>0: Timer: Timer 1 incremented by clock defined by T1M bit in register CKCON.<br>1: Counter: Timer 1 incremented by high-to-low transitions on external pin (T1). |
| 5:4 | T1M[1:0] | **Timer 1 Mode Select.**<br>These bits select the Timer 1 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Timer 1 Inactive |
| 3 | GATE0 | **Timer 0 Gate Control.**<br>0: Timer 0 enabled when TR0 = 1 irrespective of $\overline{\text{INT0}}$ logic level.<br>1: Timer 0 enabled only when TR0 = 1 AND $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 13.7). |
| 2 | C/T0 | **Counter/Timer 0 Select.**<br>0: Timer: Timer 0 incremented by clock defined by T0M bit in register CKCON.<br>1: Counter: Timer 0 incremented by high-to-low transitions on external pin (T0). |
| 1:0 | T0M[1:0] | **Timer 0 Mode Select.**<br>These bits select the Timer 0 operation mode.<br>00: Mode 0, 13-bit Counter/Timer<br>01: Mode 1, 16-bit Counter/Timer<br>10: Mode 2, 8-bit Counter/Timer with Auto-Reload<br>11: Mode 3, Two 8-bit Counter/Timers |

## SFR Definition 25.4. TL0: Timer 0 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8A

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL0[7:0] | **Timer 0 Low Byte.**<br>The TL0 register is the low byte of the 16-bit Timer 0. |

## SFR Definition 25.5. TL1: Timer 1 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8B

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL1[7:0] | **Timer 1 Low Byte.**<br>The TL1 register is the low byte of the 16-bit Timer 1. |

## SFR Definition 25.6. TH0: Timer 0 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH0[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8C

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH0[7:0] | **Timer 0 High Byte.**<br>The TH0 register is the high byte of the 16-bit Timer 0. |

## SFR Definition 25.7. TH1: Timer 1 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH1[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x8D

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH1[7:0] | **Timer 1 High Byte.**<br>The TH1 register is the high byte of the 16-bit Timer 1. |

## 25.2. Timer 2

Timer 2 is a 16-bit timer formed by two 8-bit SFRs: TMR2L (low byte) and TMR2H (high byte). Timer 2 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T2SPLIT bit (TMR2CN.3) defines the Timer 2 operation mode. Timer 2 can also be used in Capture Mode to measure the SmaRTClock or the Comparator 0 period with respect to another oscillator. The ability to measure the Comparator 0 period with respect to the system clock is makes using Touch Sense Switches very easy.

Timer 2 may be clocked by the system clock, the system clock divided by 12, SmaRTClock divided by 8, or Comparator 0 output. Note that the SmaRTClock divided by 8 and Comparator 0 output is synchronized with the system clock.

### 25.2.1. 16-bit Timer with Auto-Reload

When T2SPLIT (TMR2CN.3) is zero, Timer 2 operates as a 16-bit timer with auto-reload. Timer 2 can be clocked by SYSCLK, SYSCLK divided by 12, SmaRTClock divided by 8, or Comparator 0 output. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 2 reload registers (TMR2RLH and TMR2RLL) is loaded into the Timer 2 register as shown in Figure 25.4, 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.

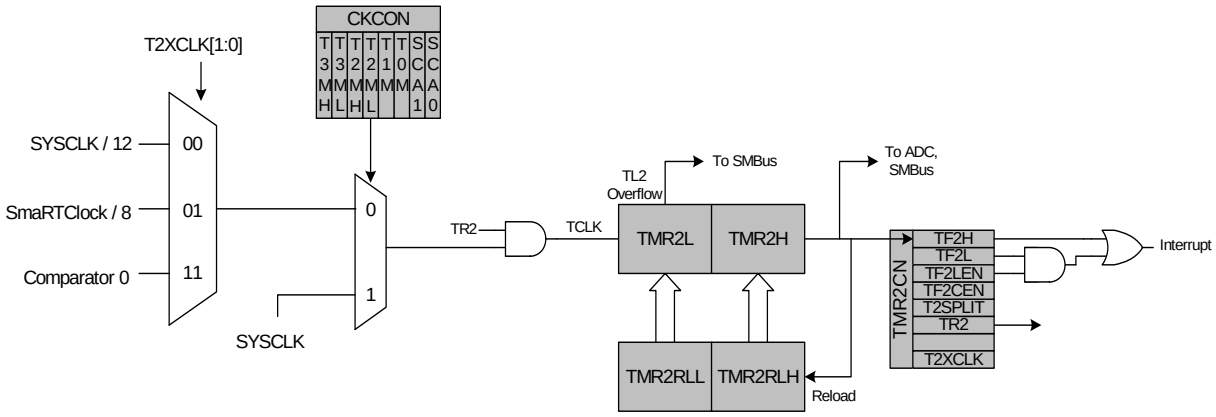



**Figure 25.4. Timer 2 16-Bit Mode Block Diagram**

### 25.2.2. 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 25.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, SmaRTClock divided by 8 or Comparator 0 output. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bits (T2XCLK[1:0] in TMR2CN), as follows:

| T2MH | T2XCLK[1:0] | TMR2H Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock / 8 |
| 0 | 10 | Reserved |
| 0 | 11 | Comparator 0 |
| 1 | X | SYSCLK |

| T2ML | T2XCLK[1:0] | TMR2L Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock / 8 |
| 0 | 10 | Reserved |
| 0 | 11 | Comparator 0 |
| 1 | X | SYSCLK |

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

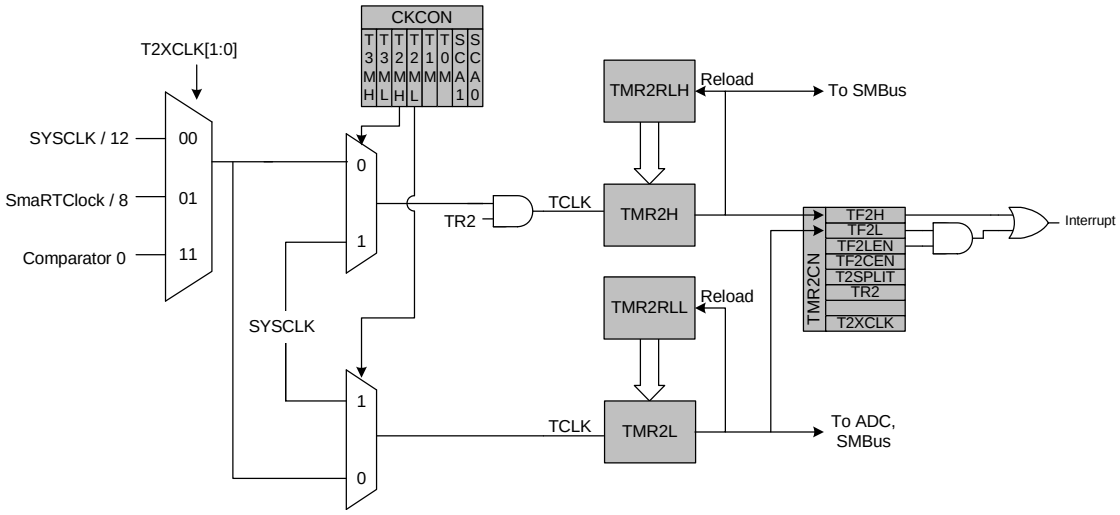



**Figure 25.5. Timer 2 8-Bit Mode Block Diagram**

### 25.2.3. Comparator 0/SmaRTClock Capture Mode

The Capture Mode in Timer 2 allows either Comparator 0 or the SmaRTClock period to be measured against the system clock or the system clock divided by 12. Comparator 0 and the SmaRTClock period can also be compared against each other. Timer 2 Capture Mode is enabled by setting TF2CEN to 1. Timer 2 should be in 16-bit auto-reload mode when using Capture Mode.

When Capture Mode is enabled, a capture event will be generated either every Comparator 0 rising edge or every 8 SmaRTClock clock cycles, depending on the T2XCLK1 setting. When the capture event occurs, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set (triggering an interrupt if Timer 2 interrupts are enabled). By recording the difference between two successive timer capture values, the Comparator 0 or SmaRTClock period can be determined with respect to the Timer 2 clock. The Timer 2 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T2ML = 1b, T2XCLK1 = 0b, and TF2CEN = 1b, Timer 2 will clock every SYSCLK and capture every SmaRTClock clock divided by 8. If the SYSCLK is 24.5 MHz and the difference between two successive captures is 5984, then the SmaRTClock clock is as follows:

24.5 MHz/(5984/8) = 0.032754 MHz or 32.754 kHz.

This mode allows software to determine the exact SmaRTClock frequency in self-oscillate mode and the time between consecutive Comparator 0 rising edges, which is useful for detecting changes in the capacitance of a Touch Sense Switch.

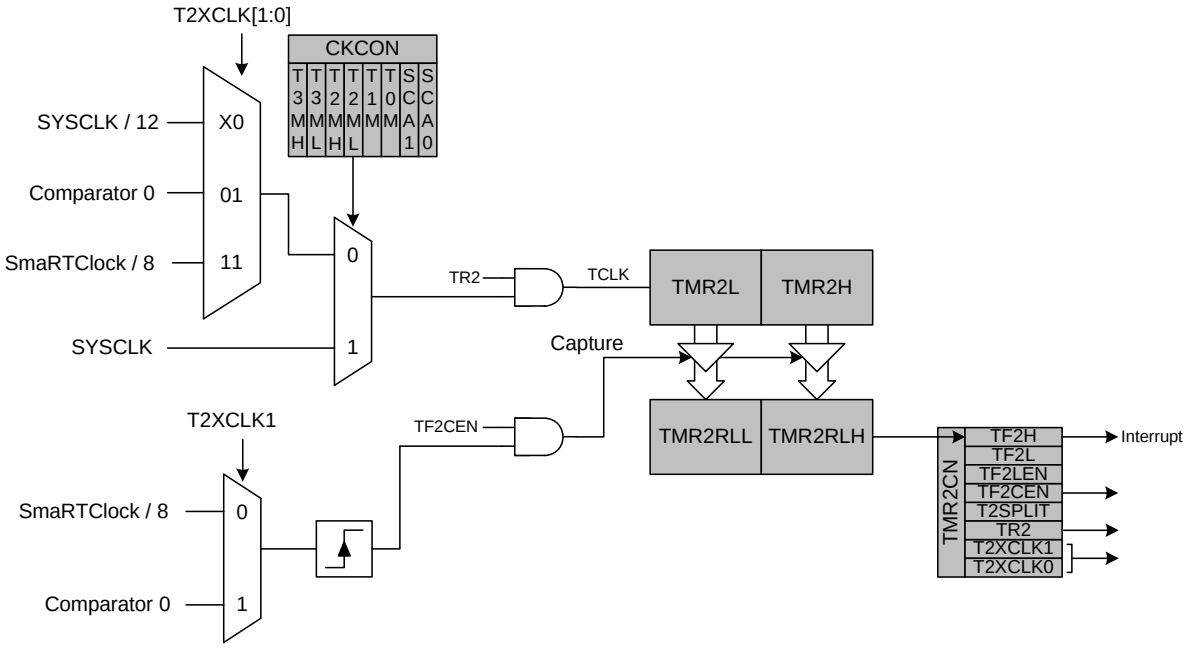



**Figure 25.6. Timer 2 Capture Mode Block Diagram**

## SFR Definition 25.8. TMR2CN: Timer 2 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF2H | TF2L | TF2LEN | TF2CEN | T2SPLIT | TR2 | T2XCLK[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xC8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | TF2H | **Timer 2 High Byte Overflow Flag.**<br>Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware. |
| 6 | TF2L | **Timer 2 Low Byte Overflow Flag.**<br>Set by hardware when the Timer 2 low byte overflows from 0xFF to 0x00. TF2L will be set when the low byte overflows regardless of the Timer 2 mode. This bit is not automatically cleared by hardware. |
| 5 | TF2LEN | **Timer 2 Low Byte Interrupt Enable.**<br>When set to 1, this bit enables Timer 2 Low Byte interrupts. If Timer 2 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 2 overflows. |
| 4 | TF2CEN | **Timer 2 Capture Enable.**<br>When set to 1, this bit enables Timer 2 Capture Mode. |
| 3 | T2SPLIT | **Timer 2 Split Mode Enable.**<br>When set to 1, Timer 2 operates as two 8-bit timers with auto-reload. Otherwise, Timer 2 operates in 16-bit auto-reload mode. |
| 2 | TR2 | **Timer 2 Run Control.**<br>Timer 2 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR2H only; TMR2L is always enabled in split mode. |
| 1:0 | T2XCLK[1:0] | **Timer 2 External Clock Select.**<br>This bit selects the “external” and “capture trigger” clock sources for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer.<br>Note: External clock sources are synchronized with the system clock.<br>00: External Clock is SYSCLK/12. Capture trigger is SmaRTClock/8.<br>01: External Clock is Comparator 0. Capture trigger is SmaRTClock/8.<br>10: External Clock is SYSCLK/12. Capture trigger is Comparator 0.<br>11: External Clock is SmaRTClock/8. Capture trigger is Comparator 0. |

## SFR Definition 25.9. TMR2RLL: Timer 2 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCA

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLL[7:0] | **Timer 2 Reload Register Low Byte.**<br>TMR2RLL holds the low byte of the reload value for Timer 2. |

## SFR Definition 25.10. TMR2RLH: Timer 2 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCB

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLH[7:0] | **Timer 2 Reload Register High Byte.**<br>TMR2RLH holds the high byte of the reload value for Timer 2. |

## SFR Definition 25.11. TMR2L: Timer 2 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCC

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2L[7:0] | **Timer 2 Low Byte.**<br>In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value. |

## SFR Definition 25.12. TMR2H Timer 2 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2H[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xCD

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2H[7:0] | **Timer 2 Low Byte.**<br>In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value. |

## 25.3. 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 (TMR2CN.3) defines the Timer 3 operation mode. Timer 3 can also be used in Capture Mode to measure the external oscillator source or the SmaRTClock oscillator period with respect to another oscillator.

Timer 3 may be clocked by the system clock, the system clock divided by 12, external oscillator source divided by 8, or the SmaRTClock oscillator. The external oscillator source divided by 8 and SmaRTClock oscillator is synchronized with the system clock.

### 25.3.1. 16-bit Timer with Auto-Reload

When T3SPLIT (TMR3CN.3) is zero, Timer 3 operates as a 16-bit timer with auto-reload. Timer 3 can be clocked by SYSCLK, SYSCLK divided by 12, external oscillator clock source divided by 8, or SmaRTClock oscillator. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 3 reload registers (TMR3RLH and TMR3RLL) is loaded into the Timer 3 register as shown in Figure 25.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 3 overflow. Additionally, if Timer 3 interrupts are enabled and the TF3LEN bit is set (TMR3CN.5), an interrupt will be generated each time the lower 8 bits (TMR3L) overflow from 0xFF to 0x00.

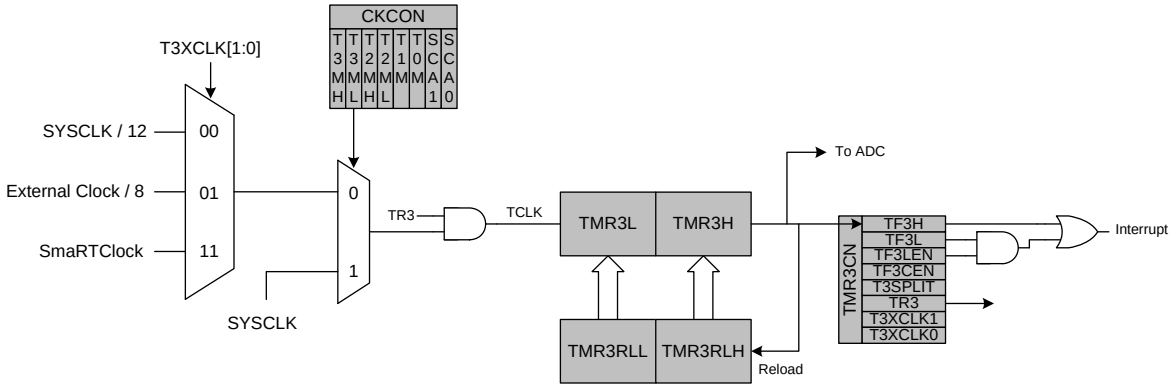



**Figure 25.7. Timer 3 16-Bit Mode Block Diagram**

### 25.3.2. 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 25.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, the external oscillator clock source divided by 8, or the SmaRTClock. The Timer 3 Clock Select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK[1:0] in TMR3CN), as follows:

| T3MH | T3XCLK[1:0] | TMR3H Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock |
| 0 | 10 | Reserved |
| 0 | 11 | External Clock / 8 |
| 1 | X | SYSCLK |

| T3ML | T3XCLK[1:0] | TMR3L Clock Source |
|---|---|---|
| 0 | 00 | SYSCLK / 12 |
| 0 | 01 | SmaRTClock |
| 0 | 10 | Reserved |
| 0 | 11 | 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.

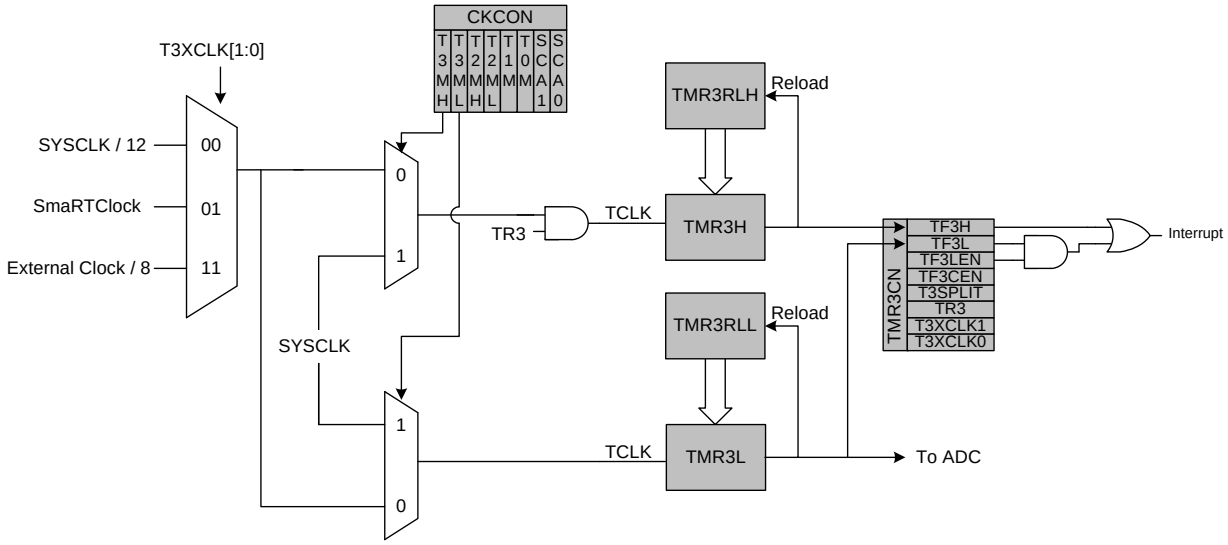



**Figure 25.8. Timer 3 8-Bit Mode Block Diagram**

### 25.3.3. SmaRTClock/External Oscillator Capture Mode

The Capture Mode in Timer 3 allows either SmaRTClock or the external oscillator period to be measured against the system clock or the system clock divided by 12. SmaRTClock and the external oscillator period can also be compared against each other.

Setting TF3CEN to 1 enables the SmaRTClock/External Oscillator Capture Mode for Timer 3. In this mode, T3SPLIT should be set to 0, as the full 16-bit timer is used.

When Capture Mode is enabled, a capture event will be generated either every SmaRTClock rising edge or every 8 external clock cycles, depending on the T3XCLK1 setting. When the capture event occurs, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set (triggering an interrupt if Timer 3 interrupts are enabled). By recording the difference between two successive timer capture values, the SmaRTClock or external clock period can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the capture clock to achieve an accurate reading.

For example, if T3ML = 1b, T3XCLK1 = 0b, and TF3CEN = 1b, Timer 3 will clock every SYSCLK and capture every SmaRTClock rising edge. If SYSCLK is 24.5 MHz and the difference between two successive captures is 350 counts, then the SmaRTClock period is as follows:

350 x (1 / 24.5 MHz) = 14.2 µs.

This mode allows software to determine the exact frequency of the external oscillator in C and RC mode or the time between consecutive SmaRTClock rising edges, which is useful for determining the SmaRTClock frequency.

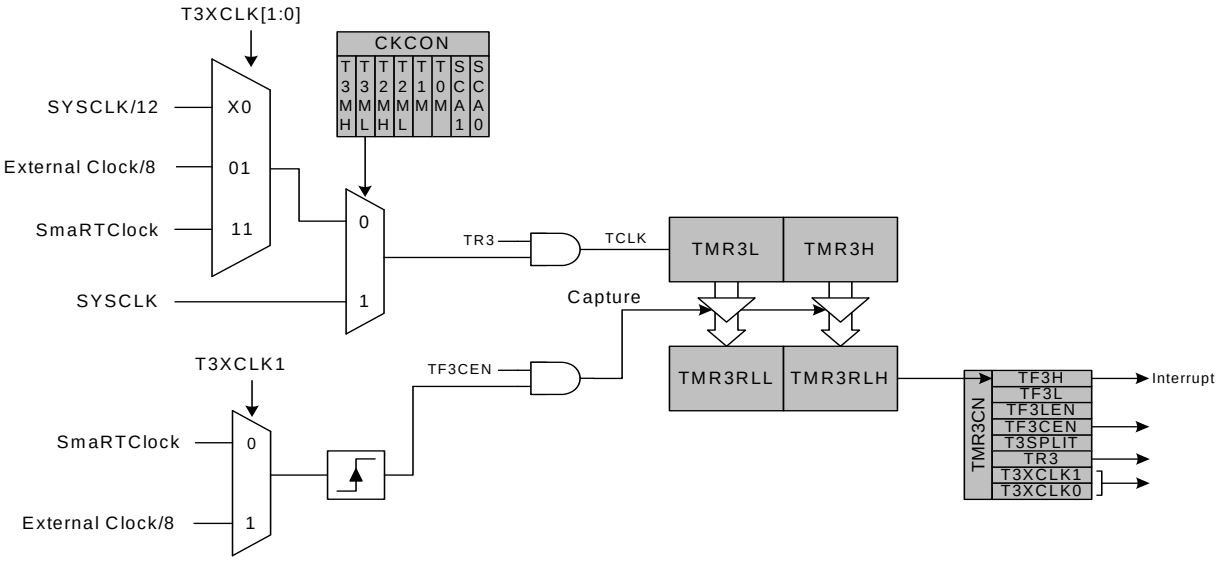



**Figure 25.9. Timer 3 Capture Mode Block Diagram**

## SFR Definition 25.13. TMR3CN: Timer 3 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF3H | TF3L | TF3LEN | TF3CEN | T3SPLIT | TR3 | T3XCLK[1:0] | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x91

| Bit | Name | Function |
|---|---|---|
| 7 | TF3H | **Timer 3 High Byte Overflow Flag.**<br>Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware. |
| 6 | TF3L | **Timer 3 Low Byte Overflow Flag.**<br>Set by hardware when the Timer 3 low byte overflows from 0xFF to 0x00. TF3L will be set when the low byte overflows regardless of the Timer 3 mode. This bit is not automatically cleared by hardware. |
| 5 | TF3LEN | **Timer 3 Low Byte Interrupt Enable.**<br>When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows. |
| 4 | TF3CEN | **Timer 3 SmaRTClock/External Oscillator Capture Enable.**<br>When set to 1, this bit enables Timer 3 Capture Mode. |
| 3 | T3SPLIT | **Timer 3 Split Mode Enable.**<br>When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload.<br>0: Timer 3 operates in 16-bit auto-reload mode.<br>1: Timer 3 operates as two 8-bit auto-reload timers. |
| 2 | TR3 | **Timer 3 Run Control.**<br>Timer 3 is enabled by setting this bit to 1. In 8-bit mode, this bit enables/disables TMR3H only; TMR3L is always enabled in split mode. |
| 1:0 | T3XCLK[1:0] | **Timer 3 External Clock Select.**<br>This bit selects the “external” and “capture trigger” clock sources for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the “external” clock source for both timer bytes. Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the “external” clock and the system clock for either timer.<br>Note: External clock sources are synchronized with the system clock.<br>00: External Clock is SYSCLK /12. Capture trigger is SmaRTClock.<br>01: External Clock is External Oscillator/8. Capture trigger is SmaRTClock.<br>10: External Clock is SYSCLK/12. Capture trigger is External Oscillator/8.<br>11: External Clock is SmaRTClock. Capture trigger is External Oscillator/8. |

## SFR Definition 25.14. TMR3RLL: Timer 3 Reload Register Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLL[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x92

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLL[7:0] | **Timer 3 Reload Register Low Byte.**<br>TMR3RLL holds the low byte of the reload value for Timer 3. |

## SFR Definition 25.15. TMR3RLH: Timer 3 Reload Register High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLH[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x93

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLH[7:0] | **Timer 3 Reload Register High Byte.**<br>TMR3RLH holds the high byte of the reload value for Timer 3. |

## SFR Definition 25.16. TMR3L: Timer 3 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3L[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x94

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3L[7:0] | **Timer 3 Low Byte.**<br>In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value. |

## SFR Definition 25.17. TMR3H Timer 3 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3H[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0x95

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3H[7:0] | **Timer 3 High Byte.**<br>In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value. |

## 26. 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 three 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between six sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, SmaRTClock divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section “26.3. Capture/Compare Modules” on page 303). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 26.1

**Important Note:** The PCA Module 2 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 26.4 for details.

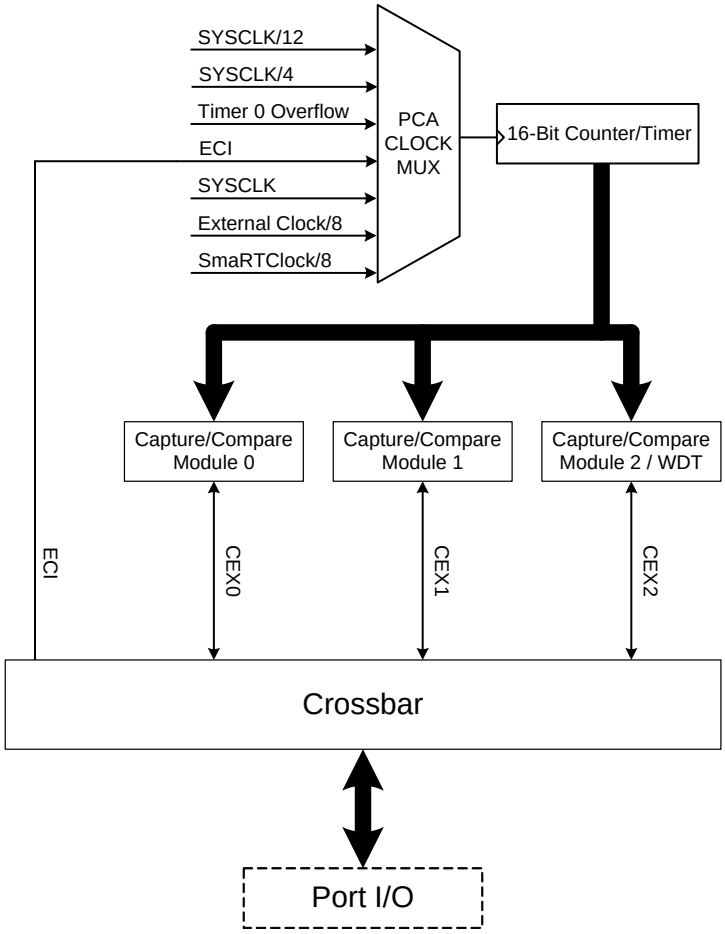



**Figure 26.1. PCA Block Diagram**

## 26.1. 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 26.1.

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 26.1. 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 | 1 | 0 | SmaRTClock oscillator source divided by 8[2] |
| 1 | 1 | 1 | Reserved |

**Notes:**
1. External oscillator source divided by 8 is synchronized with the system clock.
2. SmaRTClock oscillator source divided by 8 is synchronized with the system clock.

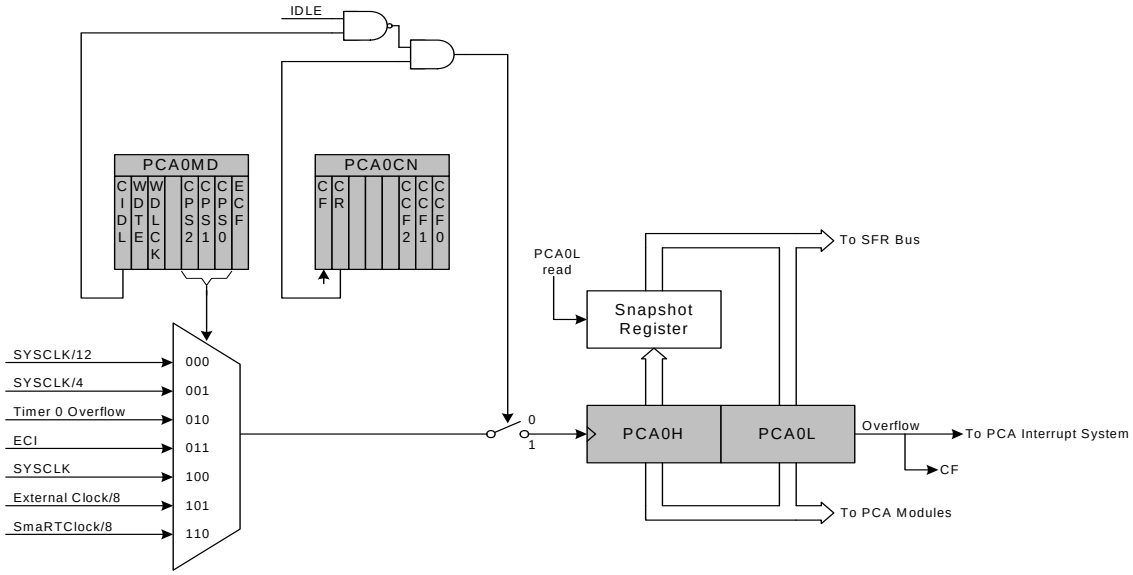



**Figure 26.2. PCA Counter/Timer Block Diagram**

## 26.2. PCA0 Interrupt Sources

Figure 26.3 shows a diagram of the PCA interrupt tree. There are five independent event flags that can be used to generate a PCA0 interrupt. They are: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter, an intermediate overflow flag (COVF), which can be set on an overflow from the 8th, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCF0, CCF1, and CCF2), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt, using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

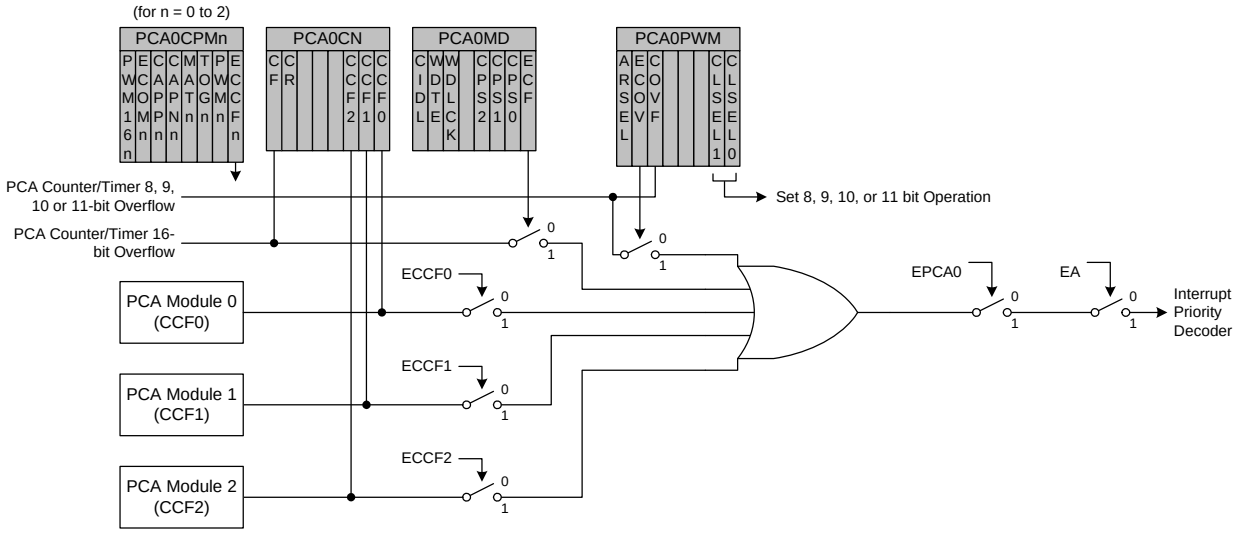



**Figure 26.3. PCA Interrupt Block Diagram**

## 26.3. Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: edge-triggered capture, software timer, high-speed output, frequency output, 8 to 11-bit pulse width modulator, or 16-bit pulse width modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation. Table 26.2 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module's operating mode. Note that all modules set to use 8, 9, 10, or 11-bit PWM mode must use the same cycle length (8–11 bits). Setting the ECCFn bit in a PCA0CPMn register enables the module's CCFn interrupt.

**Table 26.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules**

| **Operational Mode** | **PCA0CPMn** | | | | | | | | **PCA0PWM** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit Number | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4–2 | 1–0 |
| Capture triggered by positive edge on CEXn | X | X | 1 | 0 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by negative edge on CEXn | X | X | 0 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Capture triggered by any transition on CEXn | X | X | 1 | 1 | 0 | 0 | 0 | A | 0 | X | B | XXX | XX |
| Software Timer | X | C | 0 | 0 | 1 | 0 | 0 | A | 0 | X | B | XXX | XX |
| High Speed Output | X | C | 0 | 0 | 1 | 1 | 0 | A | 0 | X | B | XXX | XX |
| Frequency Output | X | C | 0 | 0 | 0 | 1 | 1 | A | 0 | X | B | XXX | XX |
| 8-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | 00 |
| 9-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 01 |
| 10-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 10 |
| 11-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XXX | 11 |
| 16-Bit Pulse Width Modulator | 1 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XXX | XX |

**Notes:**

1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th, 9th, 10th or 11th bit overflow interrupt (Depends on setting of CLSEL[1:0]).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

### 26.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPPn and CAPNn bits are set to logic 1, then the state of the Port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

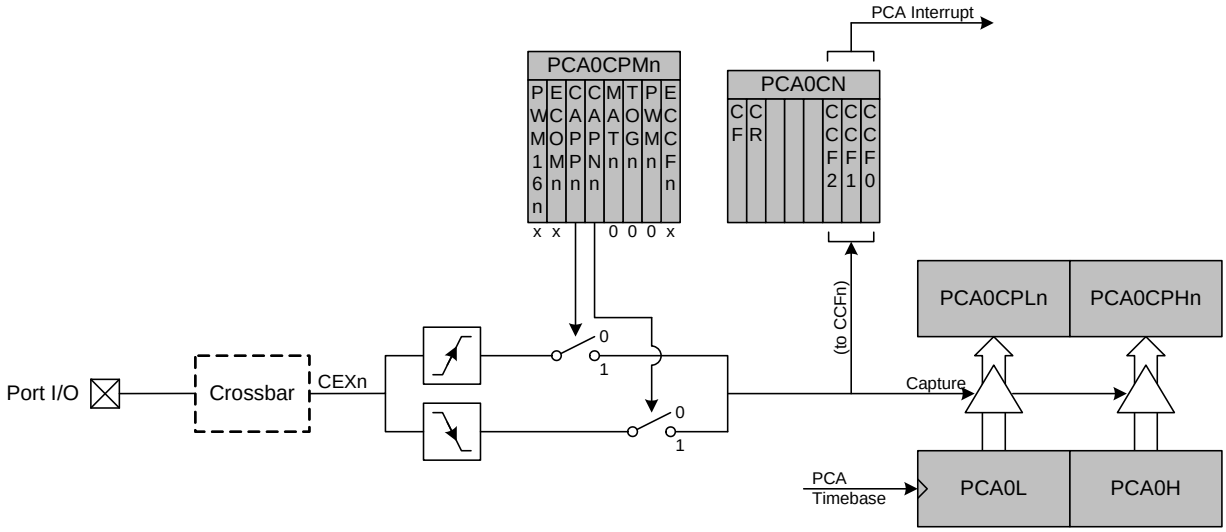



**Figure 26.4. PCA Capture Mode Diagram**

**Note:** The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

### 26.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

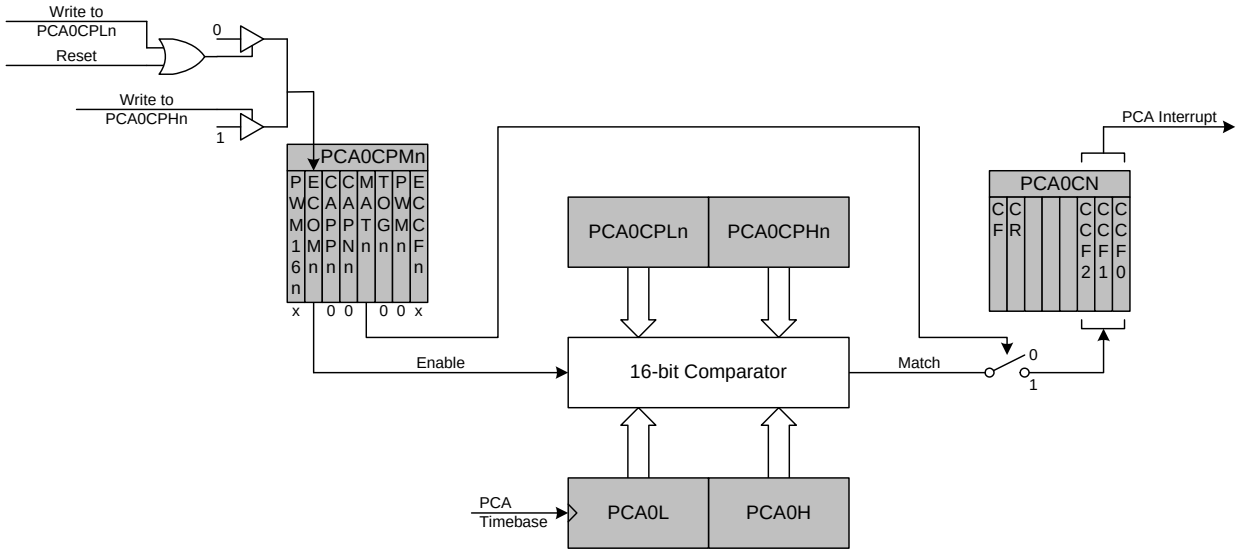



**Figure 26.5. PCA Software Timer Mode Diagram**

### 26.3.3. High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

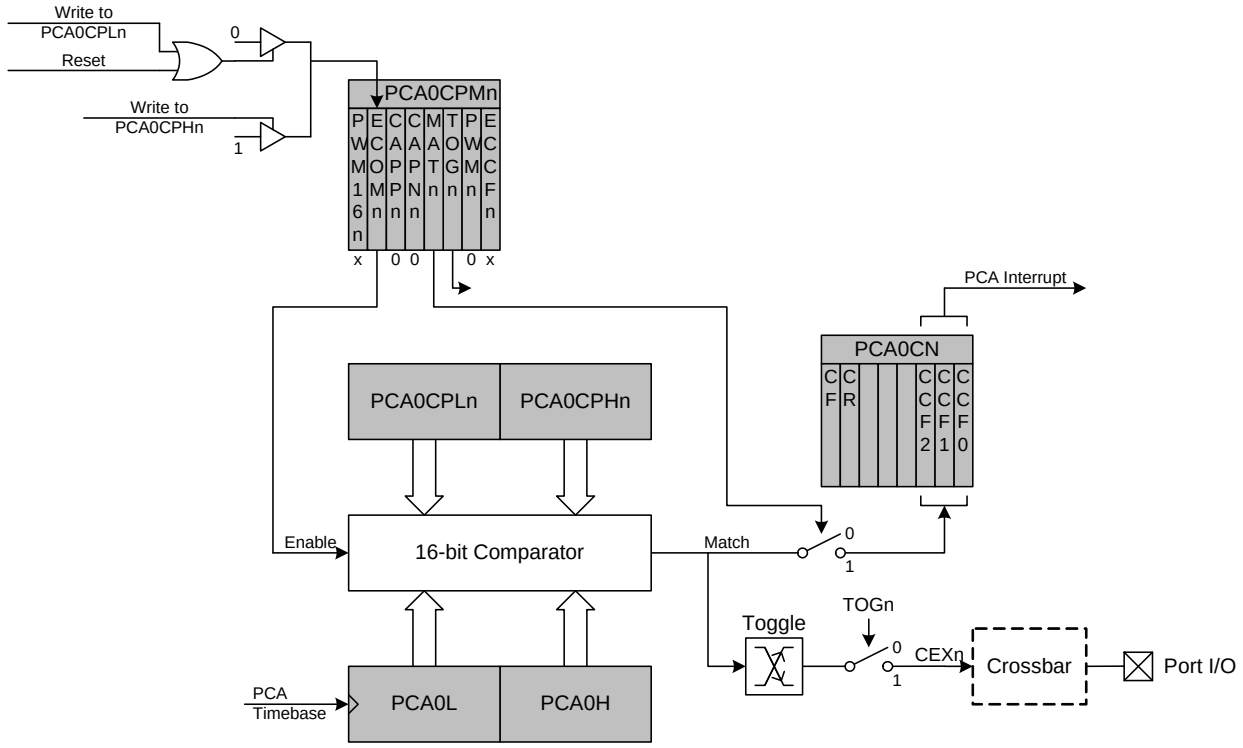



**Figure 26.6. PCA High-Speed Output Mode Diagram**

### 26.3.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 26.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

**Equation 26.1. Square Wave Frequency Output**

Where $F_{PCA}$ is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register. Note that the MATn bit should normally be set to 0 in this mode. If the MATn bit is set to 1, the CCFn flag for the channel will be set when the 16-bit PCA0 counter and the 16-bit capture/compare register for the channel are equal.

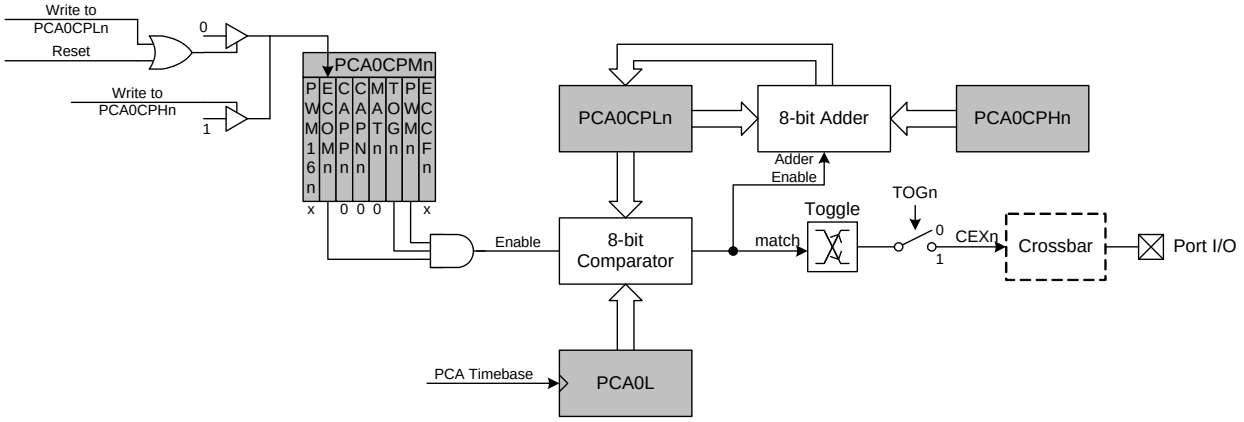



**Figure 26.7. PCA Frequency Output Mode**

### 26.3.5. 8-bit, 9-bit, 10-bit and 11-bit Pulse Width Modulator Modes

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer, and the setting of the PWM cycle length (8, 9, 10 or 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9, 10 and 11-bit PWM modes. **It is important to note that all channels configured for 8/9/10/11-bit PWM mode will use the same cycle length.** It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently.

#### 26.3.5.1. 8-Bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 8-bit PWM mode is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be set. When the count value in PCA0L overflows, the CEXn output will be reset (see Figure 26.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the module's capture/compare high byte (PCA0CPHn) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit Pulse Width Modulator mode. If the MATn bit is set to 1, the CCFn flag for the module will be set each time an 8-bit comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles. The duty cycle for 8-Bit PWM Mode is given in Equation 26.2.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

**Equation 26.2. 8-Bit PWM Duty Cycle**

Using Equation 26.2, 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.

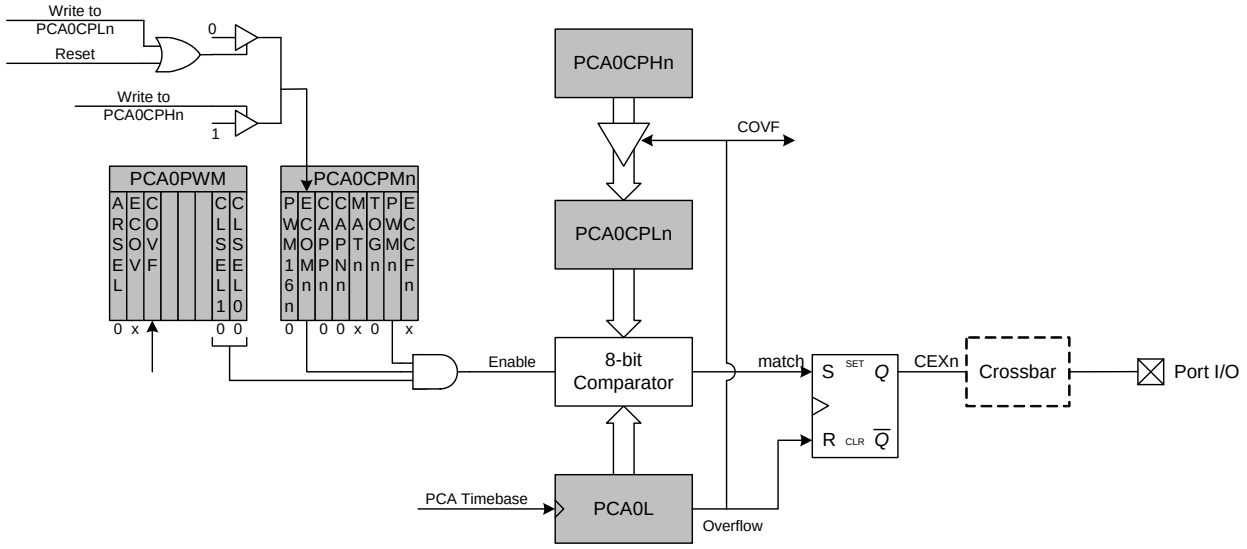



**Figure 26.8. PCA 8-Bit PWM Mode Diagram**

### 26.3.5.2. 9/10/11-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 9/10/11-bit PWM mode should be varied by writing to an “Auto-Reload” Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0.

When the least-significant N bits of the PCA0 counter match the value in the associated module’s capture/compare register (PCA0CPn), the output on CEXn is asserted high. When the counter overflows from the Nth bit, CEXn is asserted low (see Figure 26.9). Upon an overflow from the Nth bit, the COVF flag is set, and the value stored in the module’s auto-reload register is loaded into the capture/compare register. The value of N is determined by the CLSEL bits in register PCA0PWM.

The 9, 10 or 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module will be set each time a comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 512 (9-bit), 1024 (10-bit) or 2048 (11-bit) PCA clock cycles. The duty cycle for 9/10/11-Bit PWM Mode is given in Equation 26.2, where N is the number of bits in the PWM cycle.

**Important Note About PCA0CPHn and PCA0CPLn Registers**: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(2^N - PCA0CPn)}{2^N}$$

**Equation 26.3. 9, 10, and 11-Bit PWM Duty Cycle**

A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

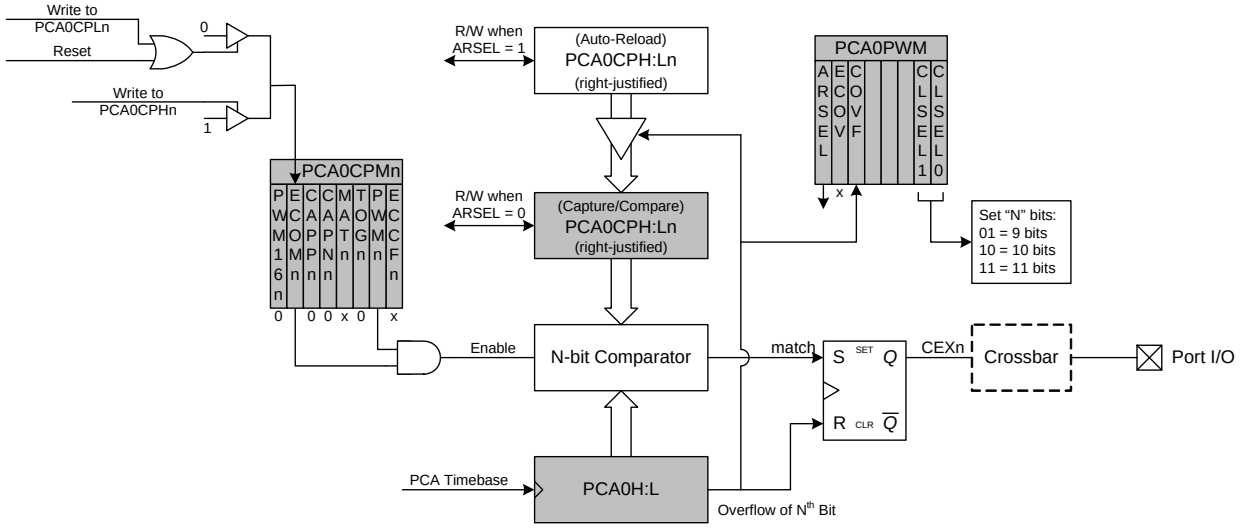



**Figure 26.9. PCA 9, 10 and 11-Bit PWM Mode Diagram**

### 26.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8/9/10/11-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. To output a varying duty cycle, new value writes should be synchronized with PCA CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 26.4.

**Important Note About Capture/Compare Registers**: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

**Equation 26.4. 16-Bit PWM Duty Cycle**

Using Equation 26.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

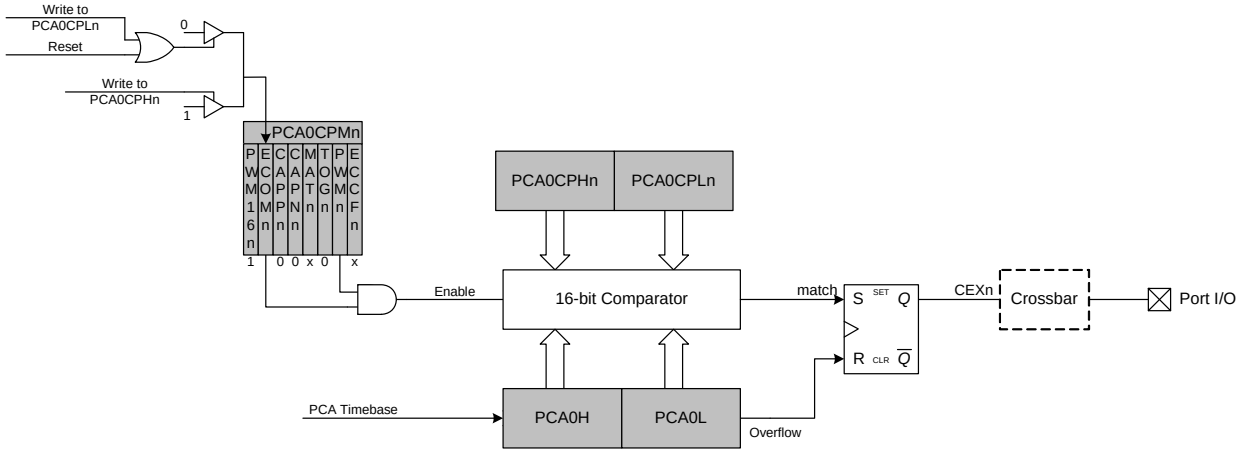



**Figure 26.10. PCA 16-Bit PWM Mode**

## 26.4. Watchdog Timer Mode

A programmable watchdog timer (WDT) function is available through the PCA Module 2. The WDT is used to generate a reset if the time between writes to the WDT update register (PCA0CPH2) 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 2 operates as a watchdog timer (WDT). The Module 2 high byte is compared to the PCA counter high byte; the Module 2 low byte holds the offset to be used when WDT updates are performed. **The Watchdog Timer is enabled on reset. Writes to some PCA registers are restricted while the Watchdog Timer is enabled.** The WDT will generate a reset shortly after code begins execution. To avoid this reset, the WDT should be explicitly disabled (and optionally re-configured and re-enabled if it is used in the system).

### 26.4.1. Watchdog Timer Operation

While the WDT is enabled:

- PCA counter is forced on.
- Writes to PCA0L and PCA0H are not allowed.
- PCA clock source bits (CPS2–CPS0) are frozen.
- PCA Idle control bit (CIDL) is frozen.
- Module 2 is forced into software timer mode.
- Writes to the Module 2 mode register (PCA0CPM2) are disabled.

While the WDT is enabled, writes to the CR bit will not change the PCA counter state; the counter will run until the WDT is disabled. The PCA counter run control bit (CR) will read zero if the WDT is enabled but user software has not enabled the PCA counter. If a match occurs between PCA0CPH2 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 PCA0CPH2. Upon a PCA0CPH2 write, PCA0H plus the offset held in PCA0CPL2 is loaded into PCA0CPH2 (See Figure 26.11).

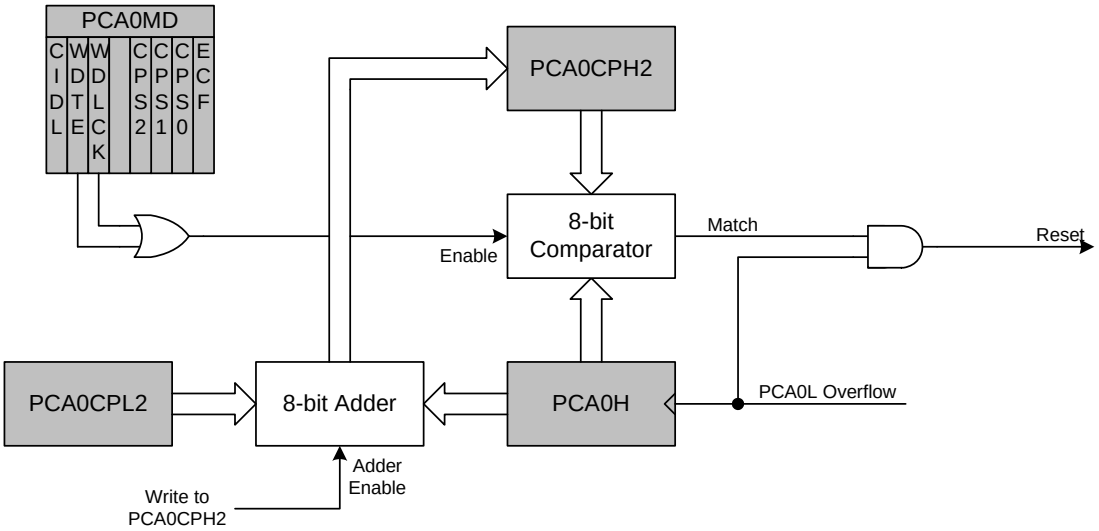



**Figure 26.11. PCA Module 2 with Watchdog Timer Enabled**

The 8-bit offset held in PCA0CPH2 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 26.5, where PCA0L is the value of the PCA0L register at the time of the update.

$$Offset = (256 \times PCA0CPL2) + (256 - PCA0L)$$

**Equation 26.5. Watchdog Timer Offset in PCA Clocks**

The WDT reset is generated when PCA0L overflows while there is a match between PCA0CPH2 and PCA0H. Software may force a WDT reset by writing a 1 to the CCF2 flag (PCA0CN.2) while the WDT is enabled.

### 26.4.2. Watchdog Timer Usage

To configure the WDT, perform the following tasks:

1. Disable the WDT by writing a 0 to the WDTE bit.
2. Select the desired PCA clock source (with the CPS2–CPS0 bits).
3. Load PCA0CPL2 with the desired WDT update offset value.
4. Configure the PCA Idle mode (set CIDL if the WDT should be suspended while the CPU is in Idle mode).
5. Enable the WDT by setting the WDTE bit to 1.
6. Reset the WDT timer by writing to PCA0CPH2.

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 PCA0CPL2 defaults to 0x00. Using Equation 26.5, this results in a WDT timeout interval of 256 PCA clock cycles, or 3072 system clock cycles. Table 26.3 lists some example timeout intervals for typical system clocks.

**Table 26.3. Watchdog Timer Timeout Intervals[1]**

| System Clock (Hz) | PCA0CPL2 | Timeout Interval (ms) |
|---|---|---|
| 24,500,000 | 255 | 32.1 |
| 24,500,000 | 128 | 16.2 |
| 24,500,000 | 32 | 4.1 |
| 3,062,500[2] | 255 | 257 |
| 3,062,500[2] | 128 | 129.5 |
| 3,062,500[2] | 32 | 33.1 |
| 32,000 | 255 | 24576 |
| 32,000 | 128 | 12384 |
| 32,000 | 32 | 3168 |

**Notes:**
1. Assumes SYSCLK/12 as the PCA clock source, and a PCA0L value of 0x00 at the update time.
2. Internal SYSCLK reset frequency = Internal Oscillator divided by 8.

## 26.5. Register Descriptions for PCA0

Following are detailed descriptions of the special function registers related to the operation of the PCA.

### SFR Definition 26.1. PCA0CN: PCA Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CF | CR | | | | CCF2 | CCF1 | CCF0 |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD8; Bit-Addressable

| Bit | Name | Function |
|---|---|---|
| 7 | CF | **PCA Counter/Timer Overflow Flag.**<br>Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |
| 6 | CR | **PCA Counter/Timer Run Control.**<br>This bit enables/disables the PCA Counter/Timer.<br>0: PCA Counter/Timer disabled.<br>1: PCA Counter/Timer enabled. |
| 5:3 | Unused | Read = 000b, Write = don't care. |
| 2:0 | CCF[2:0] | **PCA Module n Capture/Compare Flag.**<br>These bits are set by hardware when a match or capture occurs in the associated PCA Module n. When the CCFn interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |

## SFR Definition 26.2. PCA0MD: PCA Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CIDL | WDTE | WDLCK | | CPS2 | CPS1 | CPS0 | ECF |
| Type | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xD9

| Bit | Name | Function |
|---|---|---|
| 7 | CIDL | **PCA Counter/Timer Idle Control.**<br>Specifies PCA behavior when CPU is in Idle Mode.<br>0: PCA continues to function normally while the system controller is in Idle Mode.<br>1: PCA operation is suspended while the system controller is in Idle Mode. |
| 6 | WDTE | **Watchdog Timer Enable.**<br>If this bit is set, PCA Module 2 is used as the watchdog timer.<br>0: Watchdog Timer disabled.<br>1: PCA Module 2 enabled as Watchdog Timer. |
| 5 | WDLCK | **Watchdog Timer Lock.**<br>This bit locks/unlocks the Watchdog Timer Enable. When WDLCK is set, the Watchdog Timer may not be disabled until the next system reset.<br>0: Watchdog Timer Enable unlocked.<br>1: Watchdog Timer Enable locked. |
| 4 | Unused | Read = 0b, Write = don't care. |
| 3:1 | CPS[2:0] | **PCA Counter/Timer Pulse Select.**<br>These bits select the timebase source for the PCA counter<br>000: System clock divided by 12<br>001: System clock divided by 4<br>010: Timer 0 overflow<br>011: High-to-low transitions on ECI (max rate = system clock divided by 4)<br>100: System clock<br>101: External clock divided by 8 (synchronized with the system clock)<br>110: SmaRTClock divided by 8 (synchronized with the system clock)<br>111: Reserved |
| 0 | ECF | **PCA Counter/Timer Overflow Interrupt Enable.**<br>This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt.<br>0: Disable the CF interrupt.<br>1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set. |
| **Note:** When the WDTE bit is set to 1, the other bits in the PCA0MD register cannot be modified. To change the contents of the PCA0MD register, the Watchdog Timer must first be disabled. | | |

## SFR Definition 26.3. PCA0PWM: PCA PWM Configuration

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

SFR Page = 0x0; SFR Address = 0xDF

| Bit | Name | Function |
|---|---|---|
| 7 | ARSEL | **Auto-Reload Register Select.**<br>This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function.<br>0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn.<br>1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn. |
| 6 | ECOV | **Cycle Overflow Interrupt Enable.**<br>This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt.<br>0: COVF will not generate PCA interrupts.<br>1: A PCA interrupt will be generated when COVF is set. |
| 5 | COVF | **Cycle Overflow Flag.**<br>This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software.<br>0: No overflow has occurred since the last time this bit was cleared.<br>1: An overflow has occurred since the last time this bit was cleared. |
| 4:2 | Unused | Read = 000b; Write = don't care. |
| 1:0 | CLSEL[1:0] | **Cycle Length Select.**<br>When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode.<br>00: 8 bits.<br>01: 9 bits.<br>10: 10 bits.<br>11: 11 bits. |

## SFR Definition 26.4. PCA0CPMn: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16n | ECOMn | CAPPn | CAPNn | MATn | TOGn | PWMn | ECCFn |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Address, Page: PCA0CPM0 = 0xDA, 0x0; PCA0CPM1 = 0xDB, 0x0; PCA0CPM2 = 0xDC, 0x0

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16n | **16-bit Pulse Width Modulation Enable.**<br>This bit enables 16-bit mode when Pulse Width Modulation mode is enabled.<br>0: 8 to 11-bit PWM selected.<br>1: 16-bit PWM selected. |
| 6 | ECOMn | **Comparator Function Enable.**<br>This bit enables the comparator function for PCA module n when set to 1. |
| 5 | CAPPn | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture for PCA module n when set to 1. |
| 4 | CAPNn | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture for PCA module n when set to 1. |
| 3 | MATn | **Match Function Enable.**<br>This bit enables the match function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCFn bit in PCA0MD register to be set to logic 1. |
| 2 | TOGn | **Toggle Function Enable.**<br>This bit enables the toggle function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the logic level on the CEXn pin to toggle. If the PWMn bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWMn | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function for PCA module n when set to 1. When enabled, a pulse width modulated signal is output on the CEXn pin. 8 to 11-bit PWM is used if PWM16n is cleared; 16-bit mode is used if PWM16n is set to logic 1. If the TOGn bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCFn | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt.<br>0: Disable CCFn interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCFn is set. |
| **Note:** When the WDTE bit is set to 1, the PCA0CPM5 register cannot be modified, and module 5 acts as the watchdog timer. To change the contents of the PCA0CPM5 register or the function of module 5, the Watchdog Timer must be disabled. | | |

## SFR Definition 26.5. PCA0L: PCA Counter/Timer Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xF9

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[7:0] | **PCA Counter/Timer Low Byte.**<br>The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer. |

**Note:** When the WDTE bit is set to 1, the PCA0L register cannot be modified by software. To change the contents of the PCA0L register, the Watchdog Timer must first be disabled.

## SFR Definition 26.6. PCA0H: PCA Counter/Timer High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Page = 0x0; SFR Address = 0xFA

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0[15:8] | **PCA Counter/Timer High Byte.**<br>The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 26.1). |

**Note:** When the WDTE bit is set to 1, the PCA0H register cannot be modified by software. To change the contents of the PCA0H register, the Watchdog Timer must first be disabled.

## SFR Definition 26.7. PCA0CPLn: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[7:0] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPL0 = 0xFB, PCA0CPL1 = 0xE9, PCA0CPL2 = 0xEB

SFR Pages: PCA0CPL0 = 0x0, PCA0CPL1 = 0x0, PCA0CPL2 = 0x0,

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[7:0] | **PCA Capture Module Low Byte.**<br>The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will clear the module's ECOMn bit to a 0. | | |

## SFR Definition 26.8. PCA0CPHn: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPn[15:8] | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SFR Addresses: PCA0CPH0 = 0xFC, PCA0CPH1 = 0xEA, PCA0CPH2 = 0xEC

SFR Pages: PCA0CPH0 = 0x0, PCA0CPH1 = 0x0, PCA0CPH2 = 0x0,

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPn[15:8] | **PCA Capture Module High Byte.**<br>The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will set the module's ECOMn bit to a 1. | | |

# 27. C2 Interface

C8051F99x-C8051F98x devices include an on-chip Silicon Labs 2-Wire (C2) debug interface to allow Flash programming and in-system debugging with the production part installed in the end application. The C2 interface uses a clock signal (C2CK) and a bi-directional C2 data signal (C2D) to transfer information between the device and a host system. See the C2 Interface Specification for details on the C2 protocol.

## 27.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

### C2 Register Definition 27.1. C2ADD: C2 Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2ADD[7:0] | | | | | | | |
| Type | R/W | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | C2ADD[7:0] | **C2 Address.**<br>The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands. | |
| | | **Address** | **Description** |
| | | 0x00 | Selects the Device ID register for Data Read instructions |
| | | 0x01 | Selects the Revision ID register for Data Read instructions |
| | | 0x02 | Selects the C2 Flash Programming Control register for Data Read/Write instructions |
| | | 0xB4 | Selects the C2 Flash Programming Data register for Data Read/Write instructions |

## C2 Register Definition 27.2. DEVICEID: C2 Device ID

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

C2 Address: 0x00

| Bit | Name | Function |
|---|---|---|
| 7:0 | DEVICEID[7:0] | **Device ID.**<br>This read-only register returns the 8-bit device ID: 0x25. |

## C2 Register Definition 27.3. REVID: C2 Revision ID

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

C2 Address: 0x01

| Bit | Name | Function |
|---|---|---|
| 7:0 | REVID[7:0] | **Revision ID.**<br>This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A, 0x01 = Revision B, 0x02 = Revision C. |

## C2 Register Definition 27.4. FPCTL: C2 Flash Programming Control

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

C2 Address: 0x02

| Bit | Name | Function |
|---|---|---|
| 7:0 | FPCTL[7:0] | **Flash Programming Control Register.**<br>This register is used to enable Flash programming via the C2 interface. To enable C2 Flash programming, the following codes must be written in order: 0x02, 0x01. Note that once C2 Flash programming is enabled, a system reset must be issued to resume normal operation. |

## C2 Register Definition 27.5. FPDAT: C2 Flash Programming Data

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

C2 Address: 0xB4

| Bit | Name | Function | |
|---|---|---|---|
| 7:0 | FPDAT[7:0] | **C2 Flash Programming Data Register.**<br>This register is used to pass Flash commands, addresses, and data during C2 Flash accesses. Valid commands are listed below. | |
| | | **Code** | **Command** |
| | | 0x06 | Flash Block Read |
| | | 0x07 | Flash Block Write |
| | | 0x08 | Flash Page Erase |
| | | 0x03 | Device Erase |

## 27.2. C2 Pin Sharing

The C2 protocol allows the C2 pins to be shared with user functions so that in-system debugging and Flash programming may be performed. This is possible because C2 communication is typically performed when the device is in the halt state, where all on-chip peripherals and user software are stalled. In this halted state, the C2 interface can safely “borrow” the C2CK (RST) and C2D pins. In most applications, external resistors are required to isolate C2 interface traffic from the user application. A typical isolation configuration is shown in Figure 27.1.

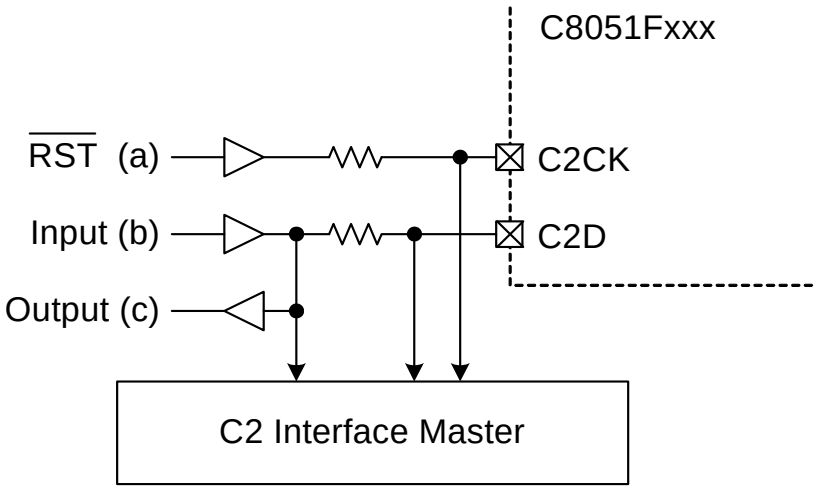



**Figure 27.1. Typical C2 Pin Sharing**

The configuration in Figure 27.1 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 0.3 to Revision 0.4

- QFN-20 package and landing diagram updated.

## Revision 0.4 to Revision 1.0

- IREF0CF register description updated.
- Updated ADC0 Chapter Text.
- Corrected an error in the Product Selector Guide.
- Updated SmaRTClock chapter to indicate how the Alarm value should be set when using Auto Reset and the LFO.
- Updated electrical specifications to fill TBDs and updated power specifications based on Rev B characterization data.
- Added a note to the OSCICL register description.
- Added a note to the CRC0CN register description.
- Updated equation in the CRC0CNT register description.
- Updated Power On Reset description.

## Revision 1.0 to Revision 1.1

- Removed references to AN338.

## Revision 1.1 to Revision 1.2

- Removed QuickSense references.
- Updated part numbers to Revision C in “Ordering Information” on page 31 and added Figure 3.4, Figure 3.5, and Figure 3.6 to identify the silicon revision.
- Updated REVID register (SFR Definition 14.2) and REVID C2 register (C2 Register Definition 27.3) with the 0x02 value for Revision C.
- Updated Figure “7.3 CP0 Multiplexer Block Diagram” on page 98 to remove the bar over the CPnOUT signals.
- Updated the “Reset Sources” on page 181 chapter to reflect the correct state of the $\overline{\text{RST}}$ pin during power-on reset.
- Updated Figure “1.14 Port I/O Functional Block Diagram” on page 26 and Figure “21.1 Port I/O Functional Block Diagram” on page 215 to mention P1.4 is not available on 20-pin devices.
- Removed references to the EMI0CN register, which does not exist.
- Updated Figure “8.2 Auto-Scan Example” on page 103 to refer to the correct pins.
- Updated POR Monitor Threshold ($V_{POR}$) Brownout Condition (VDD Falling) specification minimum, typical, and maximum values.
- Updated the reset value of the CLKSEL register (SFR Definition 19.1).
- Updated description of WEAKPUD in SFR Definition 21.3.
- Corrected SFR addresses for P0DRV (SFR Definition 21.12), P1DRV (SFR Definition 21.17), P2DRV (SFR Definition 21.20), PMU0MD (SFR Definition 15.3), FLSCL (SFR Definition 14.5), REF0CN (SFR Definition 5.15), CS0SCAN0 (SFR Definition 8.5), and CS0SCAN1 (SFR Definition 8.6).
- Replaced all instances of $V_{BAT}$ with $V_{DD}$.
- Added a note to “11.1. Accessing XRAM” , “15.5. Sleep Mode” , and “18. Reset Sources” regarding an issue with the first address of XRAM.
- Added a note to “15.5. Sleep Mode” and “19. Clocking Sources” regarding using the internal low power

oscillator while in Sleep mode.

- Added a note to “15.5. Sleep Mode” and SFR Definition “15.3. Stop Mode” regarding not disabling the POR Supply Monitor while operating above 2.4 V.
- Adjusted QFN20 c, D2, and E2 package specifications in Table 3.2.

## CONTACT INFORMATION

**Silicon Laboratories Inc.**

400 West Cesar Chavez
Austin, TX 78701

Please visit the Silicon Labs Technical Support web page:
https://www.silabs.com/support/pages/contacttechnicalsupport.aspx
and register to submit a technical support request.

The information in this document is believed to be accurate in all respects at the time of publication but is subject to change without notice. Silicon Laboratories assumes no responsibility for errors and omissions, and disclaims responsibility for any consequences resulting from the use of information included herein. Additionally, Silicon Laboratories assumes no responsibility for the functioning of undescribed features or parameters. Silicon Laboratories reserves the right to make changes without further notice. Silicon Laboratories makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Silicon Laboratories assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Silicon Laboratories products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Silicon Laboratories product could create a situation where personal injury or death may occur. Should Buyer purchase or use Silicon Laboratories products for any such unintended or unauthorized application, Buyer shall indemnify and hold Silicon Laboratories harmless against all claims and damages.

Silicon Laboratories and Silicon Labs are trademarks of Silicon Laboratories Inc.
Other products or brandnames mentioned herein are trademarks or registered trademarks of their respective holders

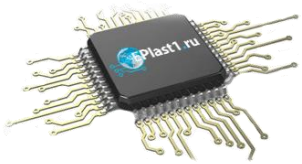
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Оперативные поставки широкого спектра электронных компонентов отечественного и импортного производства напрямую от производителей и с крупнейших мировых складов;
- Поставка более 17-ти миллионов наименований электронных компонентов;
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- Лицензия ФСБ на осуществление работ с использованием сведений, составляющих государственную тайну;
- Поставка специализированных компонентов (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Aeroflex, Peregrine, Syfer, Eurofarad, Texas Instrument, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits,General Dynamics и др.);

Помимо этого, одним из направлений компании «ЭлектроПласт» является направление «Источники питания». Мы предлагаем Вам помощь Конструкторского отдела:

- Подбор оптимального решения, техническое обоснование при выборе компонента;
- Подбор аналогов;
- Консультации по применению компонента;
- Поставка образцов и прототипов;
- Техническая поддержка проекта;
- Защита от снятия компонента с производства.




**Как с нами связаться**

**Телефон:** 8 (812) 309 58 32 (многоканальный)
**Факс:** 8 (812) 320-02-42
**Электронная почта:** org@eplast1.ru
**Адрес:** 198099, г. Санкт-Петербург, ул. Калинина, дом 2, корпус 4, литера А.